Lancaster University

Doctoral Thesis

Disruption Management in Vehicle Routing and Scheduling

A thesis submitted for the degree of
Doctor of Philosophy

by

Sofoclis P. Zambirinis
BSc (Hons), MPhil

Department of Management Science
Lancaster University Management School

October 2019

ABSTRACT

Traditionally, people in modern business environments have been focusing on planning: creating detailed and complete schemes for actions that will lead to gains of the highest value. There is no doubt that constructing a thorough plan before taking actions is extremely important and usually a prerequisite element of success. However, no matter how perfect or optimal a plan is, during the execution phase, several unanticipated events may disrupt the system and force the plan to deviate from its intended course, or even make it infeasible. How should we cope with disruptions in a timely manner? How can we reach the original goals and at the same time minimize the negative impact which was caused by the disruptions? These are amongst the essential topics examined by the field of *Disruption Management*.

Disruption Management has been applied by researchers to optimization problems arising in a wide range of applications, including airline scheduling and production management. In our research we focus on disruption management in vehicle routing and scheduling for road freight distribution, after having recognized several gaps in research in this specific domain. In this thesis we present the following three problems: (1) *the disrupted Vehicle Routing Problem with customer-specific orders and Vehicle Breakdown*, (2) *the Delayed Traveling Salesman Problem with Time Windows*, and (3) *the Single-Commodity Delayed Vehicle Routing Problem with Time Windows*. The second and third problems have never been studied before, to the best of our knowledge. The first one has been studied before under different assumptions (i.e. with non customer-specific orders), which differentiates substantially the problem from the one proposed here. For each problem we present at least one exact mixed-integer linear programming formulation (single-objective or multi-objective), which can be implemented in an optimization solver (e.g. Cplex or AIMMS) and solve small instances to optimality. Due to the fact that the problems under study are computationally hard, for each problem we also propose at least one heuristic algorithm, which is capable of solving larger instances in short time. The heuristics described in this thesis are all based on Tabu Search. We present several variants of problems 2 and 3, which are solved using both single-objective and multi-objective optimization approaches: the Weighting Method, the Lexicographic Approach, and the Epsilon Constraint Method. For each one of the three problems under study, we have constructed a dataset of test instances, which we solved using different approaches. Comparisons of the results of the exact and heuristic methods are provided for each problem.

ACKNOWLEDGEMENTS

First and foremost, I would like to express my sincerest gratitude to my supervisor, Professor Richard Eglese, for his guidance, patience, motivation, and continuous support during the long journey of my PhD studies. I cannot imagine having a better advisor and mentor.

Many thanks also go to Professors Konstantinos Zografos, John Boylan and Robert Fildes, for their kind advices and valuable feedback during my annual reviews. I wish to extend my thanks to the staff of the Department of Management Science of Lancaster University, for always being friendly and helpful.

Last but not least, I would like to express my gratitude to my family and to my close friends, for their unconditional love and continuous support throughout my life.

DEDICATION

This work is dedicated to my beloved son, Pantelis S. Zambirinis, from the very bottom of my heart.

Contents

1 INTRODUCTION 1
1.1 Introduction to Vehicle Routing Problems 2
1.2 Disruptions in vehicle routing and scheduling 3
1.2.1 Statistics about disruptions 4
1.3 Disruption Management in VRPs 7
1.4 Introduction to the three main problems addressed in this thesis, Research Summary & Contributions 8
1.5 System specifications 12
1.6 Thesis Outline 12
1.7 Publications 15

2 LITERATURE REVIEW 16
2.1 Introduction 16
2.2 Vehicle Routing Problems 17
2.3 Heuristics, Metaheuristics and hybrid methods for the Vehicle Routing Problem 19
2.3.1 Classical Heuristics for the VRP 19
2.3.2 Metaheuristics for the VRP 21
2.3.3 Hybridizations 25
2.4 Exact Approaches for the VRP 26
2.5 Literature Review on the TSPTW 26
2.5.1 Exact approaches for the TSPTW 27
2.5.2 Heuristic approaches for the TSPTW 28
2.6 Methods of dealing with uncertainties or disruptions in business operations 29
2.7 Disruption Management 31
2.8 Disruption Management in Vehicle Routing and Scheduling for freight distribution 34
2.8.1 Relevant objectives 34
2.8.2 Formulation and Solution Approaches 35
2.8.3 Discussion of relevant papers 37

2.9 Additional Literature on Disruption Management in Vehicle Routing and Scheduling 46
2.10 Multi-objective Optimization 48
2.10.1 Weighting Method 50
2.10.2 The ϵ-Constraint Method 51
2.10.3 A recent algorithm: The Quadrant Shrinking Method . 52
2.10.4 Performance indicators in Multi-Objective Optimization 52
2.11 Multi-objective Optimization for Vehicle Routing and Scheduling 53
2.12 Conclusions 53

3 Problem 1 - The disrupted VRP with Vehicle Breakdown: Formulation & Exact Approach 55
3.1 Introduction 55
3.2 Problem overview 56
3.3 Problem definition 57
3.4 A MILP formulation for the disrupted VRP with vehicle breakdown 62
3.5 Experimental results of the Exact Approach 65
3.5.1 Discussion about the instances of class T 75
3.6 The disrupted VRPTW with Vehicle Breakdown 76
3.7 Conclusions 78

4 Problem 1 - The disrupted VRP with Vehicle Breakdown: Heuristic Approach 79
4.1 Introduction 79
4.2 Heuristic overview 79
4.3 Notation 80
4.4 The easy plans 81
4.5 Diversification mechanisms and initial solutions 82
4.6 Neighborhood structure 83
4.7 Objective function 85
4.8 Search termination 86
4.9 Parameter values used in the heuristic 86
4.10 Experimental results of the heuristic approach 87
4.11 Conclusions 92

5 Problem 2 - The Delayed TSPTW: Formulations & Exact Approaches 94
5.1 Introduction 94
5.2 The easy plan 95

5.3 The Delayed TSPTW: Problem description and notation . . . 96
5.4 Objectives 99
5.4.1 Alternatives to objective 5 100
5.5 Preprocessing stage 100
5.6 General MOMILP Formulation for the Delayed TSPTW with 5 objectives 101
5.7 A MILP formulation with a single aggregated objective for the Delayed TSPTW-5 104
5.8 Notes 104
5.9 Exact Approach 1: Lexicographic Exact Approach for the Delayed TSPTW-5 106
5.10 Exact Approach 2: Weighted Sum Exact Approach for the 5-objective variant 107
5.11 Problem extension with customer priorities 108
5.12 Reduction to 3 objectives 109
5.13 Application: The disrupted VRPTW with customer-specific orders and vehicle delay 110
5.14 Other applications: other types of disruption 111
5.15 Other problem variants and solution methods, scope for further research and conclusions 112

6 Problem 2 - The Delayed TSPTW: Heuristic Approaches & Experimental Results 114
6.1 Introduction 114
6.2 Heuristic Approach 1: A Tabu Search heuristic for the Lexicographic Delayed TSPTW with 5 Objectives 115
6.2.1 Objective 116
6.2.2 Solution Notation 116
6.2.3 Description of the main algorithmic functions 117
6.2.4 Description of the Algorithm 119
6.2.5 Parameter values used in Heuristic Approach 1 124
6.2.6 Heuristic Approach 1b: A Tabu Search heuristic for the Multi-Objective Delayed TSPTW with 5 Objectives, using the Weighted-Sum method 124
6.3 Heuristic Approach 2: A Tabu Search heuristic for the Lexicographic Delayed TSPTW with 3 Objectives 125
6.3.1 Parameter values for Heuristic Approach 2 126
6.4 Heuristic Approach 3: A Tabu Search heuristic for the Delayed TSPTW with 3 objectives, using the ECM framework 127
6.4.1 ECM solutions table & ranges of the epsilons 128
6.4.2 Objectives 128

6.4.3 Solution Notation 130
6.4.4 Descriptions of the main algorithmic functions 130
6.4.5 Algorithm description 132
6.4.6 Parameter values used in Heuristic Approach 3 137
6.5 Experimental Results 137
6.5.1 Experimental Results of Exact Approach 1 139
6.5.2 Experimental Results of Heuristic Approach 1 140
6.5.3 Experimental Results of Heuristic Approach 3 148
6.6 Conclusions 150

7 Problem 3 - The Single-Commodity Delayed VRPTW: Formulations & Exact Approaches 152
7.1 Introduction 152
7.2 The easy plan 153
7.3 Problem description and notation 154
7.4 Objectives 158
7.5 Preprocessing stage 159
7.6 General MOMILP Formulation for the SCD-VRPTW with 5 objectives 159
7.7 A MILP formulation with a single aggregated objective for the SCD-VRPTW-5 162
7.8 Notes 163
7.9 Exact Approach 1: Lexicographic Exact Approach for the 5-objective variant 164
7.10 Exact Approach 2: Weighted Sum Exact Approach for the 5-objective variant 165
7.11 Problem extension with customer priorities 166
7.12 Reduction to 3 objectives 167
7.13 Applications: other types of disruption 169
7.14 Other problem variants and solution methods, scope for further research and conclusions 170

8 Problem 3 - The Single-Commodity Delayed VRPTW: Heuristic Approaches & Experimental Results 171
8.1 Introduction 171
8.2 Heuristic Approach 1: A Tabu Search heuristic for the Lexicographic SCD-VRPTW with 5 Objectives 172
8.2.1 Objective 173
8.2.2 Solution Notation 173
8.2.3 Description of the basic algorithmic functions 175
8.2.4 Description of the Algorithm 177

8.2.5 Heuristic Approach 1b: A Tabu Search heuristic for the Multi-Objective SCD-VRPTW with 5 Objectives, using the Weighted-Sum method 180
8.3 Heuristic Approach 2: A Tabu Search heuristic for the Lexicographic SCD-VRPTW with 3 Objectives 181
8.4 Heuristic Approach 3: A Tabu Search heuristic for the SCD-VRPTW with 3 objectives, using the ECM framework 183
8.4.1 Objectives 184
8.4.2 Solution Notation 186
8.4.3 Description of the main algorithmic functions 186
8.4.4 Algorithm description 188
8.4.5 Two procedures to speed up the ECM algorithm 193
8.5 Experimental Results 196
8.5.1 Discussion about the SCD-VRPTW instances 197
8.5.2 Discussion about the Experimental Results of Exact Approach 1 200
8.5.3 Discussion about the Experimental Results of Heuristic Approach 1 203
8.5.4 Parameter values used in Heuristic Approach 1 206
8.5.5 Discussion about the Experimental Results of Heuristic Approach 2 206
8.5.6 Parameter values used in Heuristic Approach 2 209
8.5.7 Parameter values used in Heuristic Approach 3 209
8.5.8 Experimental Results of Heuristic Approach 3 210
8.6 Conclusions 213

9 CONCLUSIONS 214
9.1 Research Summary & Scope for Further Research 214
9.1.1 Problem 1 - Summary 215
9.1.2 Problem 2 - Summary 216
9.1.3 Problem 3 - Summary 218
9.1.4 Additional scope for further research 221
9.2 Contributions 225

A Experimental Results of Heuristic Approach 3 for Problem 2 (Delayed-TSPTW-3) 240

B Experimental Results of Heuristic Approach 3 for Problem 3 (SCD-VRPTW-3) 259

List of Tables

2.1 Summary of papers on Disruption Management in Vehicle Routing and Scheduling for freight distribution 38

3.1 Experimental Results for Problem 1 solved using the Exact Approach (MILP, AIMMS) 66
3.2 Additional Experimental Results for Problem 1 - Exact Approach (AIMMS) 70

4.1 Experimental Results for Problem 1 solved using the Heuristic Approach (in MATLAB) - *Part 1 of 2* 88
4.2 Experimental Results for Problem 1 solved using the Heuristic Approach (in MATLAB) - *Part 2 of 2* 89

6.1 ECM solutions table 129
6.2 Experimental Results of *Exact Approach 1* for the *Lexicographic Delayed TSPTW with 5 objectives* (in AIMMS) . . . 141
6.3 Experimental Results of *Heuristic Approach 1* for the *Lexicographic Delayed TSPTW with 5 objectives* (in MATLAB) . . 146

8.1 ECM solutions table 194
8.2 First procedure to speed up the ECM Algorithm 195
8.3 Second procedure to speed up the ECM Algorithm 196
8.4 Additional information needed to construct the SCD-VRPTW instances 198
8.5 Experimental Results of Exact Approach 1 in AIMMS (for the Lexicographic SCD-VRPTW with 5 objectives) 201
8.6 Experimental Results of Heuristic Approach 1 in MATLAB (for the Lexicographic SCD-VRPTW with 5 objectives) 204
8.7 Experimental Results of Heuristic Approach 2 in MATLAB (for the Lexicographic SCD-VRPTW with 3 objectives) 207

A.1 Experimental results for instance 1-A (Heuristic Approach 3 - Delayed-TSPTW-3) 240

A.2 Experimental results for instance 1-B (Heuristic Approach 3 - Delayed-TSPTW-3) 241
A.3 Experimental results for instance 1-C (Heuristic Approach 3 - Delayed-TSPTW-3) 241
A.4 Experimental results for instance 2-A (Heuristic Approach 3 - Delayed-TSPTW-3) 241
A.5 Experimental results for instance 2-B (Heuristic Approach 3 - Delayed-TSPTW-3) 242
A.6 Experimental results for instance 2-C (Heuristic Approach 3 - Delayed-TSPTW-3) 243
A.7 Experimental results for instance 3-A (Heuristic Approach 3 - Delayed-TSPTW-3) 243
A.8 Experimental results for instance 3-B (Heuristic Approach 3 - Delayed-TSPTW-3) 244
A.9 Experimental results for instance 3-C (Heuristic Approach 3 - Delayed-TSPTW-3) 244
A.10 Experimental results for instance 4-A (Heuristic Approach 3 - Delayed-TSPTW-3) 244
A.11 Experimental results for instance 4-B (Heuristic Approach 3 - Delayed-TSPTW-3) 245
A.12 Experimental results for instance 4-C (Heuristic Approach 3 - Delayed-TSPTW-3) 245
A.13 Experimental results for instance 5-A (Heuristic Approach 3 - Delayed-TSPTW-3) 245
A.14 Experimental results for instance 5-B (Heuristic Approach 3 - Delayed-TSPTW-3) 246
A.15 Experimental results for instance 5-C (Heuristic Approach 3 - Delayed-TSPTW-3) 247
A.16 Experimental results for instance 6-A (Heuristic Approach 3 - Delayed-TSPTW-3) 247
A.17 Experimental results for instance 6-B (Heuristic Approach 3 - Delayed-TSPTW-3) 249
A.18 Experimental results for instance 6-C (Heuristic Approach 3 - Delayed-TSPTW-3) 250
A.19 Experimental results for instance 7-A (Heuristic Approach 3 - Delayed-TSPTW-3) 250
A.20 Experimental results for instance 7-B (Heuristic Approach 3 - Delayed-TSPTW-3) 252
A.21 Experimental results for instance 7-C (Heuristic Approach 3 - Delayed-TSPTW-3) 253

A.22 Experimental results for instance 8-A (Heuristic Approach 3 - Delayed-TSPTW-3) 253
A.23 Experimental results for instance 8-B (Heuristic Approach 3 - Delayed-TSPTW-3) 254
A.24 Experimental results for instance 8-C (Heuristic Approach 3 - Delayed-TSPTW-3) 255
A.25 Experimental results for instance 9-A (Heuristic Approach 3 - Delayed-TSPTW-3) 256
A.26 Experimental results for instance 9-B (Heuristic Approach 3 - Delayed-TSPTW-3) 257
A.27 Experimental results for instance 9-C (Heuristic Approach 3 - Delayed-TSPTW-3) 258

B.1 Experimental results for instance 1 - rc201-A1 (Heuristic Approach 3 for the SCD-VRPTW-3) 259
B.2 Experimental results for instance 2 - rc201-A2 (Heuristic Approach 3 for the SCD-VRPTW-3) 259
B.3 Experimental results for instance 3 - rc201-A3 (Heuristic Approach 3 for the SCD-VRPTW-3) 260
B.4 Experimental results for instance 4 - rc201-A4 (Heuristic Approach 3 for the SCD-VRPTW-3) 260
B.5 Experimental results for instance 5 - rc201-A5 (Heuristic Approach 3 for the SCD-VRPTW-3) 261
B.6 Experimental results for instance 6 - rc201-A6 (Heuristic Approach 3 for the SCD-VRPTW-3) 261
B.7 Experimental results for instance 7 - rc201-A7 (Heuristic Approach 3 for the SCD-VRPTW-3) 261
B.8 Experimental results for instance 8 - rc201-A8 (Heuristic Approach 3 for the SCD-VRPTW-3) 262
B.9 Experimental results for instance 9 - rc201-B1 (Heuristic Approach 3 for the SCD-VRPTW-3) 262
B.10 Experimental results for instance 10 - rc201-B2 (Heuristic Approach 3 for the SCD-VRPTW-3) 264
B.11 Experimental results for instance 11 - rc201-B3 (Heuristic Approach 3 for the SCD-VRPTW-3) 267
B.12 Experimental results for instance 12 - rc201-B4 (Heuristic Approach 3 for the SCD-VRPTW-3) 271
B.13 Experimental results for instance 13 - rc201-B5 (Heuristic Approach 3 for the SCD-VRPTW-3) 273
B.14 Experimental results for instance 14 - rc201-B6 (Heuristic Approach 3 for the SCD-VRPTW-3) 275

B.15 Experimental results for instance 15 - rc201-B7 (Heuristic Approach 3 for the SCD-VRPTW-3) 279
B.16 Experimental results for instance 16 - rc201-B8 (Heuristic Approach 3 for the SCD-VRPTW-3) 281
B.17 Experimental results for instance 17 - rc202-C1 (Heuristic Approach 3 for the SCD-VRPTW-3) 284
B.18 Experimental results for instance 18 - rc202-C2 (Heuristic Approach 3 for the SCD-VRPTW-3) 287
B.19 Experimental results for instance 19 - rc202-C3 (Heuristic Approach 3 for the SCD-VRPTW-3) 289
B.20 Experimental results for instance 20 - rc202-C4 (Heuristic Approach 3 for the SCD-VRPTW-3) 291
B.21 Experimental results for instance 21 - rc202-C5 (Heuristic Approach 3 for the SCD-VRPTW-3) 293
B.22 Experimental results for instance 22 - rc202-C6 (Heuristic Approach 3 for the SCD-VRPTW-3) 295
B.23 Experimental results for instance 23 - rc202-C7 (Heuristic Approach 3 for the SCD-VRPTW-3) 298
B.24 Experimental results for instance 24 - c203-D1 (Heuristic Approach 3 for the SCD-VRPTW-3) 300
B.25 Experimental results for instance 25 - c203-D2 (Heuristic Approach 3 for the SCD-VRPTW-3) 301
B.26 Experimental results for instance 26 - c203-D3 (Heuristic Approach 3 for the SCD-VRPTW-3) 302
B.27 Experimental results for instance 27 - c203-D4 (Heuristic Approach 3 for the SCD-VRPTW-3) 303
B.28 Experimental results for instance 28 - c203-D5 (Heuristic Approach 3 for the SCD-VRPTW-3) 305
B.29 Experimental results for instance 29 - c203-D6 (Heuristic Approach 3 for the SCD-VRPTW-3) 306
B.30 Experimental results for instance 30 - c203-D7 (Heuristic Approach 3 for the SCD-VRPTW-3) 311
B.31 Experimental results for instance 31 - c203-D8 (Heuristic Approach 3 for the SCD-VRPTW-3) 314
B.32 Experimental results for instance 32 - c101-E1 (Heuristic Approach 3 for the SCD-VRPTW-3) 316
B.33 Experimental results for instance 33 - c101-E2 (Heuristic Approach 3 for the SCD-VRPTW-3) 319
B.34 Experimental results for instance 34 - c101-E3 (Heuristic Approach 3 for the SCD-VRPTW-3) 323

B.35 Experimental results for instance 35 - c101-E4 (Heuristic Approach 3 for the SCD-VRPTW-3) 326
B.36 Experimental results for instance 36 - c101-E5 (Heuristic Approach 3 for the SCD-VRPTW-3) 330
B.37 Experimental results for instance 37 - c101-E6 (Heuristic Approach 3 for the SCD-VRPTW-3) 331

List of Figures

1.1 Percentage of total delay time attributable to different causes (all sectors) (McKinnon et al. (2008)) 4
1.2 Sectoral Variation in the Frequency of Total and Congestion-related Delays (McKinnon et al. (2008)) 5
1.3 Percentage of total delay time in the food supply chain attributable to different causes (McKinnon et al. (2008)) 6

Chapter 1

INTRODUCTION

Traditionally, people in modern businesses have been paying special attention to creating detailed and sound plans before taking actions. Evidently, planning is of great significance. However, no matter how perfect a plan might be, during the execution stage it is possible that an unanticipated event occurs, disrupting the normal execution of the plan. In such a case, the operational plan needs to be modified accordingly, so that the original goals are met, while at the same time the negative consequences of the disruption are reduced to the least possible degree. How can we handle disruptions promptly and effectively? This is one of the central topics investigated by the field of *Disruption Management*.

Disruption management has been used by researchers in a variety of applications. Yu & Qi (2004) present a general framework and a detailed study of disruption management, including applications in airline operations, machine scheduling, flight and crew scheduling, supply chain management etc. Visentini et al. (2013) review real-time vehicle schedule recovery problems in transportation services, where they classify these problems into three categories: vehicle rescheduling for road-based services, train-based rescheduling and airline schedule recovery problems.

There has been some research interest in disruption management in general in the past two decades. However, research in the application of the discipline in road-based services is somewhat scarce. In our research we focus on the application of disruption management in vehicle routing and scheduling, concentrating on road freight distribution, after having recognized several gaps in research in this specific domain. In fact, during our studies we published an article (Eglese & Zambirinis (2018a)) which reviews literature precisely in the area of disruption management in vehicle routing and scheduling for road freight transport. Before we introduce the three problems that we have been studying, we shall give a short introduction to vehicle routing, as well as an introduction to disruption management.

1.1 Introduction to Vehicle Routing Problems

The Vehicle Routing Problem (VRP) is a well-known combinatorial optimization and integer programming problem, which asks for the optimal set of routes to be traversed by a fleet of vehicles, in order to serve a given set of customers. It was first introduced in Dantzig & Ramser (1959) as the *Truck Dispatching Problem*, where the authors described a real-world application regarding the optimum routing of a fleet of trucks delivering gasoline between a bulk terminal and service stations.

Since then, hundreds of models and algorithms have been proposed for the optimal or approximate solution of the VRP and its numerous variants. A large number of packages for the solution of real-world VRPs are available on the market. The ongoing interest of researchers in the models and algorithms for the different versions of the VRP can be attributed to both its practical applications in logistics and distribution systems, as well as to its considerable difficulty. Examples of real-world applications of VRPs in transportation systems include the delivery or collection of goods, school bus routing, dial-a-ride systems etc. The considerable difficulty of VRPs originates from the fact that the basic variant of the VRP is a generalization of the well-known Traveling Salesman Problem (TSP) and therefore falls into the class of NP Hard problems. This means that the computational effort required for solving an instance of this problem increases exponentially with the instance size. In the following chapter we provide a more detailed discussion about the VRP and some of its basic variants that are relevant to our research, and provide a review of the main heuristics, metaheuristics and hybrid algorithms applied for the solution of such problems. For an extensive review and discussion of the various formulations and algorithms for the different VRP variants, one can refer to the book by Toth & Vigo (2014).

The Vehicle Routing Problem with Time Windows (VRPTW) is a VRP variant that asks for the determination of the optimal set of routes to be traversed by a fleet of identical vehicles of fixed capacity, starting from and finishing at a single central depot, in order to distribute goods to a set of geographically dispersed customers, while minimizing the total travel cost. Additionally, the service of each customer must start within an associated time interval, called a time window, and the vehicle must stop at the customer location for the duration of the service time of that customer. All the customers require deliveries and the customer demands are fixed, known in advance and cannot be split. All customers have to be served.

The *Traveling Salesman Problem with Time Windows (TSPTW)* is the problem of determining a minimum cost tour in which a set of nodes are visited exactly once within their requested time windows. Essentially, the TSPTW is a special case of the VRPTW where we have a single vehicle with infinite capacity.

1.2 Disruptions in vehicle routing and scheduling

No matter how perfect an operational plan may be, unexpected events originating from a variety of sources may occur from time to time, preventing the smooth execution of the plan. Specifically, during the execution of a VRP plan, disruptions of various forms may cause the current plan to be sub-optimal or even infeasible, under the new conditions of the evolved environment. Below we list some of the possible types of disruption that may occur during the execution of a VRP plan:

- Changes in the availability of resources: mechanical failure of a vehicle, driver sickness etc.
- Changes in customer requirements: cancellation of orders, new orders, demand change, delivery time change, delivery address change etc.
- Delays at various stages: delay in the arrival of supplied goods at the depot, delay due to longer service time than expected, delay due to longer travel times than expected (which may be caused by severe weather, traffic congestion, accidents, other uncertain factors, etc.).
- Change in the network travel times, e.g. due to traffic congestion.
- Change in the network structure: a single link blocked or a larger part of the network blocked. Again, this may have been caused by severe weather, by an accident, by the collapse of a bridge, etc.
- Disruption in the supply of goods. For example, the situation where the supply of the commodity does not arrive at the depot on time, so that not enough of the commodity is available to be loaded on all vehicles at the start of the delivery period.
- Combinations of the above.

1.2.1 Statistics about disruptions

McKinnon et al. (2008) report some statistics concerning deviations from schedule, that were derived from the UK Transport key performance measurement (KPI) surveys. Among others, they summarize and discuss data concerning road freight operations since 2002 across the following 7 sectors: drink, food, automotive, express parcels, non-food retailing, builder merchants and pallet-load networks. One of the key criteria of the surveys is the 'deviations from schedule'. Possible causes of delay that were listed include: Traffic congestion, problem at collection point, problem at delivery point, own company actions (internal problems), lack of driver, vehicle breakdown, and problems at the sortation hub (for the case of pallet-load networks).

By analyzing a sample of 55820 truck movements in these 7 sectors, it was found that in 26% of these journey legs there was a delay in delivery, whereas only 35% of the delays (i.e. 9% of the total legs) were mainly attributed to traffic congestion. McKinnon et al. (2008) also report an average duration of 24 minutes for delays that were due to traffic congestion, which is relatively low, compared to the average duration for all delays, which was 41 minutes. The longest delays are due to problems at sortation hubs for pallet-load distribution, or due to vehicle breakdowns, averaging 58 and 52 minutes of delay, respectively.

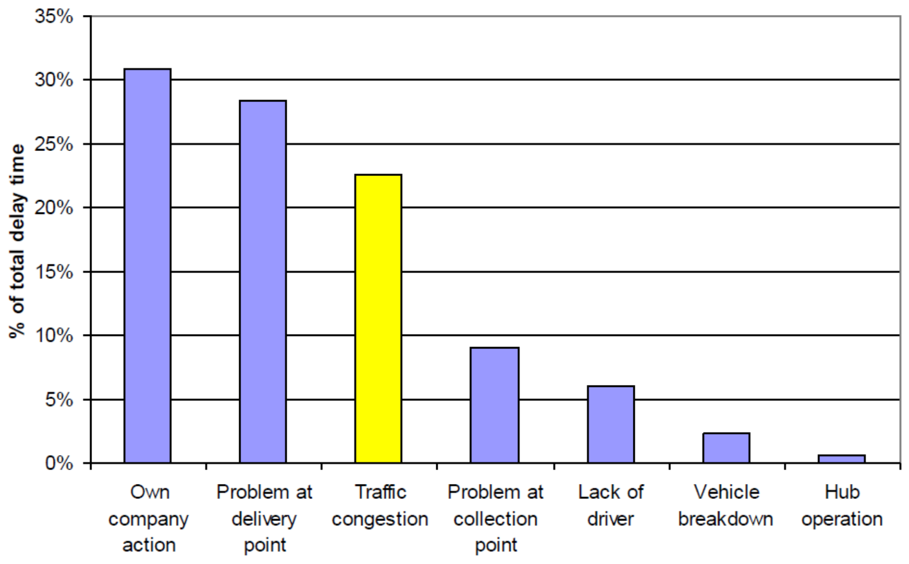
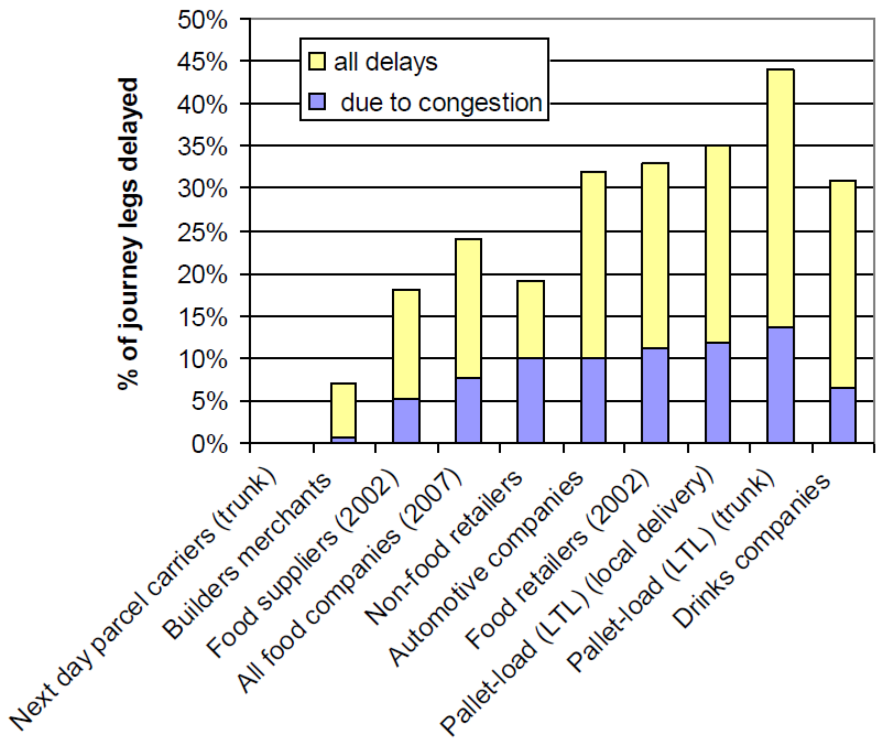


Figure 1.1: Percentage of total delay time attributable to different causes (all sectors) (McKinnon et al. (2008))

In figure 1.1 we can see the percentage of total delay time attributed to different causes, combined for all sectors. When the duration and frequency of delays are combined and the results are weighted by the number of journey legs that were surveyed in each sector, the following factors emerge as the main sources of delays, in descending order of importance:
(i) own company action
(ii) problem at delivery point
(iii) traffic congestion
(iv) problem at collection point
(v) lack of driver
(vi) vehicle breakdown
(vii) hub operation

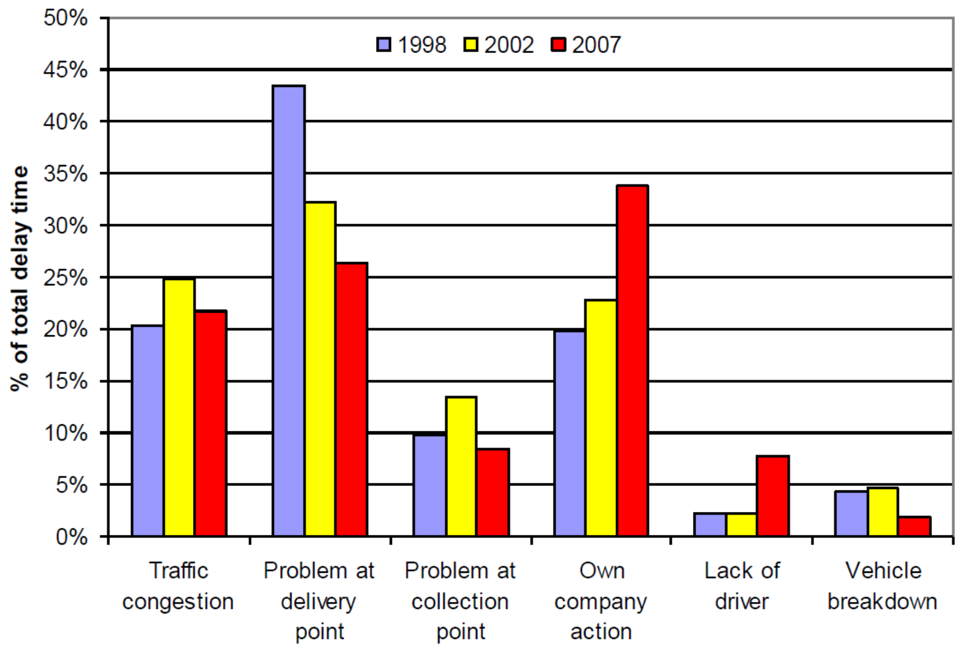
Figure 1.2: Sectoral Variation in the Frequency of Total and Congestion-related Delays (McKinnon et al. (2008))



There seems to be a wide variation in the frequency of delays among different sectors. In figure 1.2 we can see the percentage of journey legs delayed according to each sector, with additional references to the relevant percentages associated with congestion-related delays.

Furthermore, the time-series analysis of KPI data concerning the food sector, indicate that the relative importance of the various sources of delay is not constant over time. In figure 1.3 we can see the percentage of total delay time in the food supply chain that was attributed to the different causes, and how this evolved in time between 1998 and 2007.

Figure 1.3: Percentage of total delay time in the food supply chain attributable to different causes (McKinnon et al. (2008))



Note that all the statistics and figures included in this subsection up to this point, were taken from McKinnon et al. (2008).

McKinnon (2015) points out that the impact of congestion on logistics performance may be more a function of the variability around the average delay, instead of a function of the average delay itself. Where congestion is stable, regular and reasonably predictable, companies can prepare their delivery schedules accordingly, allowing extra slack in order to maintain high service standards - at a significant resource cost. However, when a highway network reaches close to full capacity, the vehicle flow becomes unstable and more vulnerable to accidents, breakdowns, bad weather and roadworks.

McKinnon (2015) also notes that there is usually a complex relationship between congestion and other causes of disruption that may occur either on the road or at points of origin or destination. Sankaran et al. (2005) mentions that congestion often amplifies delays and costs that are caused by such other factors. Finally, McKinnon (2015) mentions that the delays recorded in the UK road freight system may actually be very low, compared with the many days of delay experienced by, for example, trucks in India.

1.3 Disruption Management in VRPs

Suppose that, for a specific instance of a VRP variant, an optimal or near-optimal operational plan has been constructed. Assume that during the execution of this original plan, an unexpected disruption of any of the above types occurs, preventing the normal execution of the plan. Once a disruption occurs, the original plan may no longer be optimal; in fact, it might not even be feasible. Therefore, the original plan needs to be quickly modified, so that the remaining customers are served under the new conditions and the original objective is optimized, while at the same time the negative effect of the disruption is minimized. For instance, in the original problem the objective may be to minimize the total travel cost. This can be as simple as the total distance traveled by all the vehicles combined, the total travel time, or a more complex function that includes the costs incurred by the distance traveled, the number of vehicles and drivers involved, total travel time, extra costs occurring when a driver has to be paid overtime etc. After the disruption, the objective might be the same as in the original plan, i.e. minimizing the total travel cost function (a *pure rescheduling* approach), or it might be a more complex function which seeks to balance between optimizing the original objective(s), while at the same time minimize the negative effect of the disruption, or the deviation from the original plan (a *disruption management* approach).

In general, the problem of handling disruptions efficiently and in real time, is investigated by the field of Disruption Management. According to Yu & Qi (2004), disruption management is the process of dynamically revising an operational plan in real time, when various disruptions prevent the original plan from being executed smoothly, in order to obtain a new plan that reflects the constraints and objectives of the evolved environment while minimizing the negative impact of the disruption.

1.4 Introduction to the three main problems addressed in this thesis, Research Summary & Contributions

Throughout this research, we identified and have been studying three interesting problems that fall within the research area of disruption management in vehicle routing and scheduling for road freight transport: (1) *the disrupted Vehicle Routing Problem with customer-specific orders and Vehicle Breakdown*, (2) *the Delayed Traveling Salesman Problem with Time Windows*, and (3) *the Single-Commodity Delayed Vehicle Routing Problem with Time Windows*. These problems are introduced below.

Problem 1: The Disrupted VRP with customer-specific orders and Vehicle Breakdown

Assume that we have a VRP with vehicles of different capacities, in general, for which we have constructed a feasible and optimal or near-optimal solution plan. Suppose that during its execution, one of the vehicles breaks down and the damage is severe enough so that the driver alone is unable to restore it within a few minutes. If no further action is taken, the remaining customers of the disabled vehicle's route will remain unserved. The original plan is now infeasible under the new conditions of the evolved environment. Therefore, the plan needs to be quickly revised so that all customers are served by the remaining vehicles.

We assume that vehicles had departed from a single depot, where they should also return after the delivery of all their orders. We assume that there are no extra vehicles available at the depot, that there are no split deliveries, that all customers have to be served, that vehicles deliver customer-specific orders and that the parcels or packages may easily be transferred from one vehicle to another, if necessary, along with any accompanying documents. Therefore, each one of the unserved customers originally assigned to the broken-down vehicle, has to be served by one of the remaining vehicles. However, in order for this to happen, any such active vehicle must first visit the location of the disabled vehicle and transfer the corresponding packages to the active vehicle. We assume that in the original problem the objective was to minimize the total travel time.

The Disrupted VRP with customer-specific orders and Vehicle Breakdown asks for the determination of the optimal set of routes, sequences and times of visits of the active vehicles to customer nodes and to the location of the broken-down vehicle, so that all customers are served, including the ones assigned to the disabled vehicle, while all necessary constraints are satisfied. In this new problem that surfaces after the disruption, the main objective is the same as in the original VRP, i.e. minimize the total travel time. However, we include a secondary objective to minimize the sum of arrival times of the vehicles at the depot, which only comes into effect as a tie breaker, in cases where there are multiple solutions with the same optimal value for the primary objective.

We should mention that other researchers in the past have also studied the case of a VRP with vehicle breakdown, such as Mu et al. (2011). However, in other studies the problem was either studied under very different assumptions, which substantially distinguish their problem from ours, or the approaches that were proposed differ significantly from ours.

Problem 2: The Delayed Traveling Salesman Problem with Time Windows

Suppose that we have constructed a feasible and optimal or near-optimal solution plan for a given TSPTW instance. Assume that during the execution of this operational plan at some point there is a delay, and the vehicle or 'traveling salesman' has fallen behind schedule. Specifically, we assume that the delay is severe enough, so that if the original route is followed as planned, there will be a violation of the time window constraint for at least one of the remaining customers. The unfulfilled part of the original plan is no longer feasible with respect to time windows; the delay has caused a disruption to the original plan.

If we reschedule in an appropriate way, it is possible that fewer customers are served outside their time windows and that the total amount of time deviation outside the time windows is lower, compared to the case where no rescheduling takes place and the driver keeps following the original plan. Therefore, it is necessary to revise the original plan in such a way that the negative effect of the disruption on the enterprise is minimized; something that translates into minimizing both the number of customers served outside their time windows and the total amount of violation of their time windows.

Essentially, once this disruption occurs we are facing a new problem which is significantly different from the original TSPTW problem, with a different set of objectives, constraints and structure. We will be referring to this new problem as *the Delayed Traveling Salesman Problem with Time Windows (Delayed TSPTW)*, or simply as *Problem 2.* We define this problem as a multi-objective optimization problem with 5 objectives, where the first four component objectives seek to minimize the total amount of deviation from the original time windows, and the fifth objective to minimize the time of arrival at the endpoint node.

Problem 3: The Single-Commodity Delayed VRPTW

Suppose that we have created a feasible and optimal or near-optimal solution plan for a given instance of the single-commodity VRPTW with vehicles of different capacities, in general. Assume that, during the execution of this plan, there is a severe delay in one or more of the routes and the respective vehicles have fallen behind schedule, so that if the original routes are followed as planned, at least one customer in each one of the affected routes will be served with a delay (i.e. outside their promised time windows). This constitutes a disruption to the original plan, which is no longer feasible with respect to the time windows.

By rescheduling in an appropriate way, it is possible that fewer customers are served outside their time windows and that the total amount of violation of the time windows is lower, compared to the case where no rescheduling takes place and the drivers simply keep following the original plan. Clearly, we need to revise the current plan in order to minimize the negative effect of the disruption.

This gives rise to a new problem, which is substantially different from the original VRPTW, with different constraints, objectives and structure. We will name this problem as *the Single-Commodity Delayed VRPTW (SCD-VRPTW)*, or *the disrupted VRPTW with non-customer-specific orders and vehicle delay*, and we will be referring to it as *problem 3*. We define this problem as a multi-objective optimization problem with 5 objectives, where the first two component objectives seek to minimize the number of customers served outside their time windows, the following two objectives to minimize the total amount of violation of the time windows, and the final objective to minimize the sum of arrival times of the vehicles at the endpoint node.

An important characteristic of the SCD-VRPTW is the assumption that vehicles deliver a single-commodity, such as oil or gas, which means that the orders are non-customer-specific, or equivalently that the goods are 'transferable between customers'. Therefore, any vehicle can serve any customer, provided that it carries enough quantity of the commodity to fully serve him or her. This means that it is possible that in the revised plan, vehicles serve partially different or completely different sets of customers, compared to the original plan, and not just changing the order of visiting the customers within the same routes. It also means that vehicles not experiencing a delay may have to change their routes and schedules as well, if this would lead to a solution with a more desired set of objective values. We further assume that there are no extra vehicles available, and that all customers have to be served.

Problem 3 can be viewed as a generalization of problem 2 where we have multiple vehicles of fixed capacity carrying a single commodity, and having to reschedule after one or more vehicles experience a delay.

Note that there is another important generalization of problem 2, where we have multiple vehicles of fixed capacity carrying customer-specific orders (multi-commodity VRPTW), and having to reschedule after one or more vehicles experience a delay. This problem, which we will refer to as *the disrupted VRPTW with customer-specific orders and vehicle delay*, may seem quite similar but is substantially different from the SCD-VRPTW, and can be trivially solved as a direct application of the Delayed TSPTW. This is further discussed in chapter 5.

To the best of our knowledge, problems 2 and 3, along with any proposed generalizations, have never been studied before under similar assumptions and with the specific sets of objectives and constraints that we use. Therefore, from this point of view, they can be regarded as completely new problems.

Research Summary & Contributions

For each one of the three problems mentioned above, we present an exact mixed-integer linear programming (MILP) formulation which can be used to solve small instances to optimality. Due to the fact that the problems under study are computationally hard, heuristic algorithms are proposed capable of solving larger instances in real-time. Therefore, for each problem we also propose at least one appropriate heuristic. Furthermore, for each problem we have constructed a dataset and we have solved the respective instances with at least two of the different approaches proposed. Comparisons of the exact and heuristic approaches are provided for each problem, along with respective conclusions.

The first contribution of this dissertation is the introduction of two new problems, namely problems 2 and 3, that have never been studied before under the specific assumptions and from a similar point of view. We believe that it is very common to have delays during the delivery process, and thus these problems should occur frequently in the logistics sector, but we are not aware of an efficient way to deal with such delays in real time in a better way or in a comparable way to the approaches that we present. Therefore, we believe that the solution methods that we propose for problems 2 and 3 can be directly applied in practice in the case of a delay during the delivery process. The second contribution is a MILP formulation for each one of the three problems. Regarding problem 1, acknowledging that it may be beneficial to allow multiple visits to the broken-down vehicle and capturing this feature in the proposed formulation, is a significant point. The third contribution is the construction of at least one heuristic for each problem. A fourth contribution is the construction of one new dataset of test instances for each problem. Also, a very important contribution stems from the fact that we have conducted tests on these problem instances using at least an exact and a heuristic method for each problem. Important conclusions can be drawn from the results of these experiments. We give some idea about the problem sizes that can be solved to optimality using the exact approaches. For smaller instances, this provides a way to measure the quality of the heuristics. For larger instances, we use lower bounds provided by AIMMS from the use of the exact formulations (whose calculation involves, among others, the use of the LP relaxation and is provided automatically by the solver), or appropriate lower bounds that we found which are sometimes stronger than those provided by AIMMS and can be calculated by other procedures, in order to get proven optimality gaps and thus measure the quality of the heuristics. Furthermore, through the tests that we have conducted we can draw some conclusions about how the computational times relate to the size of the problems.

1.5 System specifications

The computational results reported in this thesis were derived from experiments that were performed on a personal computer with the following system specifications: Intel Core i5-3230M CPU running at 2.60 GHz, with 8 GB RAM, running on Windows 8, 64-bit Operating System.

1.6 Thesis Outline

The remainder of this thesis is organized as follows:

Chapter 2 provides an overview of disruption management and its application to vehicle routing problems, reviewing the existing literature. The chapter starts with an introduction to the family of VRPs, discussing some of the basic VRP variants that are particularly related to our research. A discussion of heuristics, metaheuristics and hybrid algorithms applied for the solution of VRPs is then presented, followed by a review of exact and heuristic approaches for the TSPTW. An overview of the five main approaches used by researchers to handle disruptions is then presented, with an emphasis on disruption management. This is followed by an extensive review of the literature that is concerned specifically with research on disruption management in the context of vehicle routing and scheduling for road freight distribution. At the end of the chapter we present a short introduction to multi-objective optimization and applications in vehicle routing and scheduling.

The main part of this thesis consists of chapters 3 to 8, where we devote two chapters to each one of the three problems under study: one chapter for MILP formulations and exact approaches, and one chapter for heuristic methods and experimental results.

Specifically, in chapter 3 we describe *the disrupted VRP with customer-specific orders and Vehicle Breakdown (d-VRP-cso-VB)*, to which we also refer as *problem 1*. After the problem definition, we present a mixed-integer linear programming (MILP) formulation, which can be used directly in standard commercial optimization software (e.g. Cplex or AIMMS) to find the exact optimal solution. This constitutes the *exact approach for problem 1*. The experimental results from using this approach to solve 101 instances that were created specifically for this problem are then presented. Then an extension of problem 1 is discussed, concerning *the disrupted VRPTW with Vehicle Breakdown*, under the assumptions of customer-specific orders and a heterogeneous fleet.

Chapter 4 describes a heuristic algorithm based on Tabu Search, as a second approach for solving problem 1. Experimental results of the heuristic approach are presented at the end of the chapter and compared with the results of the exact approach.

Then, in chapter 5 we describe *the Delayed Traveling Salesman Problem with Time Windows (Delayed TSPTW)*, to which we also refer as *problem 2*. This is defined as a multi-objective optimization (MOO) problem with 5 component objectives. After the problem definition, we establish notation and present a general multi-objective mixed-integer linear programming (MOMILP) formulation. We then present exact approach 1 for addressing this problem, which involves aggregating the 5 component objectives into a single objective, defined as the weighted sum of the 5 components, and using appropriate weights in order to enforce a lexicographic preference of the objectives. Therefore, under the additional assumptions regarding the lexicographic preference of the objectives, the more general MOMILP formulation is transformed into a single-objective MILP formulation. This can be used directly in a commercial solver (e.g. AIMMS) to find the exact optimal solution. After this, exact approach 2 for addressing the Delayed TSPTW is described. This is similar to exact approach 1, but instead of using a single set of weights, it involves systematically varying the weights of the 5 component objectives and solving a MILP for each set of weights; thus getting several non-dominated solutions to the MOO problem with 5 objectives. We then present an extension of problem 2 that involves customer priorities. Then we make some additional assumptions and reduce the number of objectives from 5 down to 3; thus differentiating between two variants of the problem, namely the *Delayed-TSPTW with 5 objectives* and the *Delayed-TSPTW with 3 objectives.* We then discuss a direct application of problem 2 to solve a more general problem: *the disrupted VRPTW with customer-specific orders and vehicle delay.* We also discuss other applications of problem 2 to handle other types of disruption, other variants, additional constraints that can be included etc.

In chapter 6, we describe three heuristic approaches for solving variants of problem 2, all of which employ Tabu Search. Heuristic approach 1 solves the Lexicographic Delayed TSPTW with 5 objectives. Heuristic approach 2 solves the Lexicographic variant with 3 objectives. Heuristic approach 3 uses the framework of the Epsilon Constraint Method to solve heuristically the Delayed TSPTW with 3 objectives, and finds a representative subset of the set of non-dominated solutions. A dataset of 27 instances was created specifically for testing the different solution methods proposed for the Delayed TSPTW. The chapter includes experimental results of exact approach 1, heuristic approach 1 and comparisons. The experimental results of heuristic approach 3 can be found in Appendix A.

In chapter 7 we describe *the Single-Commodity Delayed VRPTW (SCD-VRPTW)*, to which we also refer as *problem 3.* This is defined as a multi-objective optimization problem with 5 component objectives. It can be viewed as a generalization of the Delayed TSPTW with 5 objectives. After the problem definition, we establish notation and present a general multi-objective mixed-integer linear programming (MOMILP) formulation. We then present exact approaches 1 and 2, equivalent to those described for problem 2. Also, as in problem 2, we present an extension of problem 3 that involves customer priorities. Then we make some additional assumptions and reduce the number of objectives from 5 down to 3; thus differentiating between two variants of the problem; namely the *SCD-VRPTW with 5 objectives* and the *SCD-VRPTW with 3 objectives.* We also discuss other applications of problem 3 to handle other types of disruption, other variants, additional constraints that can be included etc.

In chapter 8 we describe three heuristic approaches based on Tabu Search for solving variants of problem 3, equivalent to the three heuristic approaches described in chapter 6 for problem 2. Heuristic approach 1 solves the Lexicographic SCD-VRPTW with 5 objectives. Heuristic approach 2 solves the Lexicographic SCD-VRPTW with 3 objectives. Heuristic approach 3 uses the framework of the Epsilon Constraint Method to solve heuristically the SCD-VRPTW with 3 objectives, and finds a representative subset of the set of non-dominated solutions. The different methods are tested on a dataset of

37 instances that were created for the SCD-VRPTW. The chapter includes the experimental results of exact approach 1, heuristic approach 1, heuristic approach 2 and comparisons. The experimental results of heuristic approach 3 are presented in Appendix B.

Finally, chapter 9 summarizes the research contained in this thesis, presents the main findings and conclusions, and discusses potential lines of further research in the area of disruption management in vehicle routing and scheduling.

1.7 Publications

The following articles were published based on material from chapter 2:

1. Eglese, R. W. & Zambirinis, S. (2018). Disruption management in vehicle routing and scheduling for road freight transport: a review. TOP, vol. 26, no. 1, pp. 1-17. DOI: 10.1007/s11750-018-0469-4

2. Eglese, R. W. & Zambirinis, S. (2018). Rejoinder on: Disruption management in vehicle routing and scheduling for road freight transport: a review. TOP, vol. 26, no.1, pp. 27-29. https://doi.org/10.1007/s11750-018-0470-y

We intend to submit three more articles for publication based on material from this thesis:

1. *The disrupted Vehicle Routing Problem with customer-specific orders and Vehicle Breakdown.* This concerns problem 1 and will be generated from chapters 3 and 4.

2. *The Delayed Traveling Salesman Problem with Time Windows.* This concerns problem 2 and will be based on the material contained in chapters 5 and 6.

3. *The Single-Commodity Delayed Vehicle Routing Problem with Time Windows.* This concerns problem 3 and will be generated from chapters 7 and 8.

A paper based on material from chapter 3 has been presented on the following occasion:

- Zambirinis, S. & Eglese, R. W. (2014). The Disrupted Vehicle Routing Problem with Vehicle Breakdown. *VEROLOG conference.* Oslo, Norway, 2014.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

The aim of this chapter is to provide a review of the existing literature relevant to disruption management in vehicle routing problems, as well as an overview of tools and solution methods that are relevant to our research, and in particular to the three problems studied in this thesis.

The chapter begins with a general overview of the family of Vehicle Routing Problems (VRPs). This is followed by a description of different heuristics, metaheuristics and hybrid algorithms that researchers have been applying for solving variants of the VRP over the past six decades. A survey of exact and heuristic approaches for the Traveling Salesman Problem with Time Windows (TSPTW) is then presented.

The focus is then shifted to methods of dealing with uncertainties or disruptions in business operations, and especially on Disruption Management. An introduction to the general field of Disruption Management is presented, along with a short review of applications of the discipline in business operations. This is followed by a detailed review of applications of Disruption Management in vehicle routing and scheduling, with an emphasis on the freight distribution scenario.

At the end of the chapter we present a brief introduction to the field of Multi-Objective Optimization. We establish some basic terminology and notation, and describe two basic methods for solving multi-objective optimization problems that we use widely in this thesis: the Weighting Method and the Epsilon-Constraint Method. We then briefly discuss a more recent state-of-the-art algorithm, and give some references about performance indicators that are used in multi-objective optimization, in order to measure the quality of approximation of the Pareto front. Finally, we present a short survey of applications of multi-objective optimization methods for modeling and solving vehicle routing and scheduling problems.

2.2 Vehicle Routing Problems

In this thesis we focus on disruption management in vehicle routing and scheduling for road freight transport. This means that the models used are closely related to the models that are used to address the Vehicle Routing Problem and its related variants.

The Vehicle Routing Problem (VRP) is the problem of determining the optimal set of routes to be traversed by a fleet of vehicles, in order to serve a given set of customers. It is considered to be one of the most studied and important combinatorial optimization problems. The problem was introduced in Dantzig & Ramser (1959). Since then, numerous papers and books have been published describing different models and algorithms for the optimal or approximate solution of the many different variants of the VRP. Toth & Vigo (2014) discuss and review formulations and algorithms used for the VRP and related problems. The models and algorithms proposed for the solution of the VRP and its variants are used for the solution of different real-world applications arising in transportation systems. Typical applications include the delivery or collection of goods, school bus routing, dial-a-ride systems, routing of salespeople, routing of maintenance units etc. The VRP and its variants are known to be NP-hard.

The basic version of the VRP is the Capacitated Vehicle Routing Problem (CVRP), which involves the determination of the optimal set of routes to be traversed by a fleet of vehicles based at a single central depot, in order to distribute goods to a set of geographically dispersed customers, while minimizing the total travel cost. Capacity constraints are imposed. All vehicles are identical, with a fixed capacity and must start and finish at the depot. All the customers require deliveries and the customer demands are fixed, known in advance and cannot be split. All customers have to be served.

The Vehicle Routing Problem with Time Windows (VRPTW) is an extension of the CVRP in which the service of each customer must start within an associated time interval, called a time window, and the vehicle must stop at the customer location for the duration of the service time of that customer. This may be particularly relevant to disruption management as the original plan may have created customer expectations about the time when service will be delivered. The special case of the VRPTW where we have a single vehicle with infinite capacity is called the Traveling Salesman Problem with Time Windows (TSPTW).

The Open Vehicle Routing Problem (OVRP) is another variant of the VRP, where the routes must end at a customer instead of the depot. More specifically, given a single depot, a homogeneous fleet of vehicles with given capacity, a set of customers with known demands and a (symmetric) matrix of the traveling cost between customers and depot, the OVRP seeks to find the set of routes, each starting at the depot and ending at a customer, so that all the customers are served and the total cost is minimized. One particular type of the *disrupted VRP with vehicle breakdown* that was studied by Mu et al. (2011), which assumes the delivery of a single commodity, has a network structure which shares many similarities with the OVRP.

There are many other variants of the VRP, such as the *VRP with Backhauls (VRPB)*, the *VRP with Pickup and Delivery (VRPPD)* etc. Additionally, one can expand such models by incorporating more constraints, such as adding a maximum length or maximum completion time to each route.

There is also a significant literature on dynamic or real-time vehicle routing and scheduling. These articles particularly deal with situations where customer demand may arise, or be canceled, during the operation and so plans must be constantly updated to react to the changing demand patterns. Other sources of dynamism may arise from changing travel times and vehicle availability. Some of the modeling and algorithmic approaches used to address a dynamic vehicle routing problem may provide useful insights for disruption management in vehicle routing and scheduling.

Pillac et al. (2013) present a review of the Dynamic Vehicle Routing Problems (DVRPs) with more than 150 references. That study classifies routing problems from the perspective of information quality (deterministic versus stochastic input) and evolution of information (whether or not the information available to the planner may change during the execution of the routes).

Psaraftis et al. (2016) also present a survey paper on Dynamic Vehicle Routing Problems and provide a taxonomy of DVRP papers according to 11 criteria: type of problem, logistical context, transportation mode, objective function, fleet size, time constraints, vehicle capacity constraints, the ability to reject customers, the nature of the dynamic element, the nature of the stochasticity (if any) and the solution method.

In Chapter 11 of the recent book edited by Toth & Vigo (2014), Bektaş et al. (2014) also present a survey of DVRPs. In this chapter, apart from providing an overview of the relevant literature, the authors present the state of the art in frameworks and strategies, providing a detailed analysis of the DVRP.

2.3 Heuristics, Metaheuristics and hybrid methods for the Vehicle Routing Problem

In the last six decades, hundreds of different heuristics have been proposed for the VRP and its variants. These can be broadly classified into the following classes: classical heuristics, which were mostly developed between 1960 and 1990, metaheuristics, whose growth mostly took place after 1990, and hybrid methods, mainly developed after 2000.

Classical heuristics are typically fast and often flexible enough in a sense that they can be easily extended to incorporate additional constraints encountered in practice, but they usually involve a relatively limited exploration of the search space. Metaheuristics on the other hand involve more advanced memory structures, sophisticated neighborhood search rules and recombinations of solutions; therefore they typically perform a deeper exploration of the most promising regions of the search space and produce higher quality solutions, but at the expense of increased computing time. Finally, hybrid methods try to combine several concepts borrowed from different metaheuristics, as well as from exact approaches.

2.3.1 Classical Heuristics for the VRP

In this subsection we provide a quick overview of the *Classical Heuristics for the VRP*, by following the description and categorization of Laporte & Semet (2002), who classify these into the following three categories:

1. *Constructive heuristics*, which gradually build a feasible solution, taking into account the solution cost, but without employing an improvement phase. Examples of constructive heuristics include, but not limited to:
    (a) the Savings Algorithm by Clarke & Wright (1964). Several enhancements of this algorithm have been proposed, such as by Gaskell (1967), Yellow (1970), Nelson et al. (1985) and Golden et al. (1977).
    (b) Matching-Based Savings Algorithms, such as in Desrochers & Verhoog (1989), Altinkemer & Gavish (1991) and Wark & Holt (1994).
    (c) Sequential Insertion Heuristics, such as the ones by Mole & Jameson (1976), and by Christofides et al. (1979).
2. *Two-phase heuristics*, where the problem is decomposed into two components: clustering of the vertices into feasible routes, and actually constructing the routes. Possible feedback loops between the two phases

are allowed. Two-phase heuristics can be further classified into two classes:

(a) *cluster-first, route-second* methods, where vertices are initially organized into feasible clusters, and for each cluster a route is then constructed. Algorithms of this class can be further categorized into:

- *Elementary Clustering Methods* which perform a single clustering of the vertex set first and then determine one route for each cluster, such as the Sweep Algorithm (see Gillett & Miller (1974), Wren & Carr (1971) and Wren & Holliday (1972)), the general-assignment-based algorithm of Fisher & Jaikumar (1981), and the two-phase location-based heuristic by Bramel & Simchi-Levi (1995).
- *Truncated Branch-and-Bound approach*, which was first proposed by Christofides et al. (1979).
- *Petal Algorithms*, which create a large family of overlapping clusters and associated routes, and then select a feasible set of routes among them (e.g. see Foster & Ryan (1976), Ryan et al. (1993), Renaud et al. (1996)).

(b) *route-first, cluster-second* methods, where initially a giant TSP tour containing all the vertices is built, disregarding side constraints, and then this tour is decomposed into feasible vehicle routes. This approach was first proposed by Beasley (1983).

3. *Improvement methods*, which attempt to upgrade feasible solutions found by performing a sequence of edge or vertex exchanges between or within routes. Improvement heuristics for the VRP may operate separately on each route, or on multiple routes each time:

   - *Single-Route Improvements:* For example, the λ-opt mechanism (e.g. 2-opt, 3-opt etc.) for the TSP, introduced by Lin (1965), where λ edges are first removed from the tour, and then the remaining λ segments are reconnected in all possible ways. If a profitable reconnection is identified (first or best), it is implemented. This procedure stops at a local minimum, i.e. once no further improvement can be made. Of course, several modifications to the previous basic scheme have been proposed since then, such as by Lin & Kernighan (1973), Or (1976), and Renaud et al. (1996).
   - *Multiroute Improvements:* Description of various multiroute edge exchanges for the VRP can be found, for example, in Van Breedam

(1994), Thompson & Psaraftis (1993), and Kindervater & Savelsbergh (1997). These cover a large number of edge exchange moves and schemes used by several authors.

Specifically, Van Breedam (1994) considers the following four improvement operations:

(a) *String Cross*, where two strings of vertices are exchanged by crossing two edges of two different routes,

(b) *String Exchange*, where two strings of at most k vertices are exchanged between two different routes,

(c) *String Relocation (or String Insertion)*, where a string of at most k vertices is removed from one route and inserted in another, and

(d) *String Mix*, which selects the best move between String Exchange and String Relocation.

Van Breedam considers two local improvement strategies to evaluate these moves, namely First Improvement and Best Improvement.

Also, Thompson & Psaraftis (1993) present a general b-cyclic, k-transfer scheme, where a circular permutation of b routes is considered, in which k customers from each one of those routes are shifted to the following route of the cyclic permutation. Therefore, each one of the four Van Breedam's operations described above, can be viewed as a special case of a 2-cyclic exchange.

Of course, the distinction between improvement and constructive methods may sometimes be obscure, since most constructive algorithms include improvement steps at various stages.

In general, classical heuristics involving only simple construction and local descent improvement methods, do not compete with more advanced metaheuristic or math-heuristic algorithms, in terms of solution quality. However, many of the ingredients of the classical heuristics are used as components within the more sophisticated schemes of metaheuristics and math-heuristics.

2.3.2 Metaheuristics for the VRP

In this subsection we present a very brief overview of several *Metaheuristics for the VRP*. Most of the material in this subsection is borrowed or adapted from Gendreau et al. (2002) and Laporte et al. (2014).

In the past decades, several *metaheuristics* were proposed for the VRP and its variants. These are general solution procedures which explore the solution space in order to identify good solutions, usually employing components of classical heuristics described in the previous subsection, for route construction and solution improvement. In general, the best known metaheuristics for the VRP are usually able to find better quality solutions than the earlier classical heuristics, but are typically more time consuming.

Metaheuristics for the VRP may be broadly classified into two major categories: local search algorithms and population-based heuristics. Below we briefly discuss each category and their most representative algorithms.

1. **Local Search Algorithms**:
   Local Search Algorithms start from an initial solution x_1 and at each iteration t, move from the current solution x_t to another solution x_{t+1} in the neighborhood $N(x_t)$ of solution x_t, until a stopping criterion is met. Suppose that the objective is to minimize $f(x)$, subject to $x \in S$. The value of the objective function (cost) $f(x_{t+1})$ of the newer solution x_{t+1} is not necessarily better (less) than the value of the objective function $f(x_t)$ of the current solution x_t. In general, metaheuristics may allow deteriorating and even infeasible intermediate solutions during the search process. Care must be taken to avoid cycling.

   Some of the most basic Local Search Methods applied to VRPs include:

   - *Simulated Annealing (SA)*: In Simulated Annealing, at each iteration t, a solution x is randomly selected in the neighborhood $N(x_t)$ of x_t. If $f(x) \leq f(x_t)$, then x_{t+1} is set to x; otherwise,

     $$x_{t+1} := \begin{cases} x & \text{with probability } p_t \\ x_t & \text{with probability } 1 - p_t \end{cases} \tag{2.3.1}$$

     where p_t is usually a decreasing function of both t and $f(x) - f(x_t)$. A common choice is $p_t = e^{-\frac{f(x)-f(x_t)}{\theta_t}}$, where θ_t represents the temperature at iteration t and is typically a decreasing step function of t: initially θ_t is set to a given positive value θ_1, and after every T iterations it is multiplied by a factor α $(0 \leq \alpha \leq 1)$, so that the probability of accepting a worse solution gradually decreases over time. The search is continued until the stopping criterion is met. Commonly used stopping criteria are: (i) the value f^* of the incumbent x^* has not decreased by at least $\pi_1\%$ for at least k_1 consecutive cycles of T iterations, (ii) the number of accepted moves has been less than $\pi_2\%$ of T for k_2 consecutive cycles of T iterations, or (iii) k_3 cycles of T iterations have been executed. Several researchers have applied SA for the solution of VRPs, such as Osman (1993).

- *Deterministic Annealing (DA)*: This method is similar to Simulated Annealing, with the difference that the rule used for the acceptance of a move is deterministic. Two standard implementations of DA are the threshold accepting algorithm (Dueck & Scheuer (1990)) and the record-to-record travel algorithm (Dueck (1993), F. Li et al. (2005)). In the threshold accepting algorithm, at each iteration t, a solution x_{t+1} is accepted if $f(x_{t+1}) < f(x_t) + \theta_1$, where θ_1 is a user-controlled parameter. In record-to-record travel, at each iteration t, a solution $x \in N(x_t)$ is selected and is accepted by setting $x_{t+1} := x$ if $f(x) \leq \theta_2 f(x^*)$; otherwise, $x_{t+1} := x_t$. Here, x^* is the best known solution so far, and θ_2 is a parameter usually slightly larger than 1 (e.g. $\theta_2 = 1.05$).
- *Tabu Search (TS)*: In this method the search moves from a solution x_t to the best solution $x_{t+1} \in N(x_t)$ which is not declared tabu. That is, in order to prevent cycling, solutions sharing some attributes with the current solution x_t are declared forbidden or *tabu* for a number of iterations. The tabu status of a candidate solution is revoked in case this solution improves the best known solution. Many implementations of TS have been proposed by researchers in the past decades. Several features of TS include rules to intensify the search in promising areas, or to diversify it. Various TS implementations for the VRP, such as in Gendreau et al. (1994), allow intermediate infeasible solutions during the search and penalize the objective function accordingly. This can be achieved by minimizing a penalized objective function $f'(x) := f(x) + \sum_k \alpha_k V_k(x)$, where $V_k(x)$ is the total violation of the constraints of type k in solution x, and α_k is either a constant parameter or a dynamically-adjusted parameter. For instance, in Cordeau et al. (2001), the parameters a_k are initially set equal to 1, and are self-adjusted throughout the search, so that at every iteration, if the current solution violates the constraint of type k, then a_k is multiplied by $1 + \delta$ (for some constant $\delta > 0$); otherwise, a_k is divided by $1 + \delta$.

  Tabu Search was first proposed by Glover (1986) as a general framework or metaheuristic that can be superimposed on another heuristic in order to solve difficult combinatorial optimization problems. The method was formalized in Glover (1989, 1990). Additionally, in these papers several applications of the method are reported, ranging from graph theory and matroid settings to general pure and mixed integer programming problems. Traveling salesman, graph coloring, integrated circuit design, job shop flow sequencing, time tabling and maximum satisfiability problems are only some of the problems where TS was applied with success. In the next decades, TS became one of the most widespread and most successful metaheuristics.

Cordeau & Laporte (2002) describe some of the basic features of Tabu Search for solving VRPs, including the following: neighborhood structures, short-term and long-term term memory structures, intensification and diversification mechanisms. They compare TS to simulated annealing, deterministic annealing, genetic algorithms, ant systems and neural networks, and conclude that TS outperforms other metaheuristics. They review ten of the top performing implementations of TS applied to solve the VRP, and conclude that the Adaptive Memory Procedure of Rochat & Taillard (1995) (a probabilistic technique to diversify, intensify and parallelize TS adapted for solving VRPs), outperforms the other implementations.

Basu (2012) presents a literature survey of 76 papers on the Tabu Search implementation on the TSP, the VRP and their variants. This paper compares and points out trends after classifying the literature based on problem size, initial solution generation, choice of moves, choice of short, medium and long-term memory structures, and aspiration criteria.

- *Iterated Local Search (ILS):* This is a simple algorithm which can be applied on top of any local search method, introduced by Baxter (1984). The idea is to apply an embedded local search mechanism (e.g. steepest descent, TS etc.) until it meets a stopping criterion which corresponds to a solution x; then perturb x to yield a new starting solution x', and then reapply the embedded local search. Repeat this procedure until a stopping criterion is reached (such as a predefined number of outer iterations, a number of consecutive solutions without an improvement, or a time limit). Applications of ILS for VRPs can be found in Chen et al. (2010), and in Subramanian et al. (2013).
- *Variable Neighborhood Search (VNS):* This method works with several neighborhoods N_1, N_2,..., N_p, which are usually of increasing complexity or even embedded (for example, 2-opt, 3-opt etc.). It starts with an initial solution and then iteratively applies these neighborhoods in a descent style, until no further improvement can be made. Once the last neighborhood is applied, a new cycle restarts. This procedure terminates when no further improvement

is possible, or after a preset number of cycles. Hansen et al. (2010) provides a survey for VNS, describing the basic schemes and extensions of VNS as a metaheuristic for solving optimization problems in general, and discusses several applications. Kytöjoki et al. (2007) presents an efficient VNS heuristic applied to the CVRP, which was able to provide high-quality solutions on instances with up to 20,000 customers within reasonable CPU times.

2. **Population-Based Algorithms:**
   Population-based heuristics keep track of a population of solutions that is continuously evolving, e.g. by combining good solutions in the current population, hoping to generate better ones. Examples of population-based algorithms include:

   - *Ant Colony Optimization (ACO)* (e.g. see Reimann et al. (2004)).
   - *Genetic Algorithms (GA)*, which typically examine a population of solutions at each step. Each population is derived from its previous generation, by combining its best elements while eliminating the worst. Description and applications of GAs to the VRP can be found in: Prins (2004), Nagata & Bräysy (2009), Vidal et al. (2012).
   - *Scatter Search and Path Relinking* (e.g. see Glover (1977) and Resende & Ribeiro (2010)).
   - *Learning Mechanisms*: These include, for example, the Neural Networks approach, which is a learning mechanism that involves a set of weights that are gradually adjusted until an acceptable solution is reached (e.g. see Gendreau et al. (2002)).

For an overview of metaheuristic principles and other search methodologies in optimization, from a general point of view, one can refer to the books by Burke et al. (2005) and Gendreau & Potvin (2010).

2.3.3 Hybridizations

As research for the family of VRPs progresses, a wider variety of hybrid methods emerge; methods which borrow concepts from various algorithms, including exact, heuristics and metaheuristics. Some important families of *hybridizations* include:

- Population-Based Search and Local Search.

- Meta-Meta Hybridizations.
- Hybridizations with Large Neighborhoods.
- Hybridizations with Mathematical Programming Solvers or other exact solutions (matheuristics).
- Hybrid heuristics that use parallel and cooperative search mechanisms.
- Metaheuristics complemented by decompositions or coarsening phases.

References and more details for the above hybridizations applied in VRPs can be found in Laporte et al. (2014).

2.4 Exact Approaches for the VRP

Extensive reviews of exact methods used for solving the Capacitated Vehicle Routing Problem (CVRP), can be found in Semet et al. (2014) and in Poggi & Uchoa (2014). More generally, mathematical formulations and exact solution methods for other variants of the VRP, including the VRPTW, can be found in the book by Toth & Vigo (2014).

2.5 Literature Review on the TSPTW

One of the problems that we study extensively in this thesis is the *Delayed TSPTW*. This is particularly relevant to the classical *Traveling Salesman Problem with Time Windows (TSPTW)*. For this, in this section we present a non-exhaustive review of papers that describe exact and heuristic solution methods for the TSPTW.

The TSPTW asks for the determination of a minimum-cost tour which starts from and returns to the same single depot, before it visits a predefined set of customer nodes exactly once, each of whom have to be visited within a specific time window. There are numerous applications of the TSPTW in business, including package delivery, bank couriers etc. The TSPTW is also mathematically equivalent to certain time-sensitive production scheduling problems. It can be viewed as a special case of the VRPTW with only one vehicle of infinite capacity.

2.5.1 Exact approaches for the TSPTW

There are many papers describing solution approaches for the TSPTW, ranging from exact mathematical programming techniques, to heuristic and sophisticated metaheuristic approaches. The earliest papers on the TSPTW involve exact approaches for the problem variant with makespan optimization. These include Christofides et al. (1981) and Baker (1983), that present branch-and-bound algorithms which solve instances with up to 50 nodes, but whose algorithms cannot effectively handle overlapping or wide time windows.

Langevin et al. (1993) present a two-commodity flow formulation within a branch-and-bound scheme for the TSPTW, capable of handling various objectives, and address both variants of the TSPTW with travel-time optimization (TSPTW-TT) and with makespan optimization (TSPTW-M). The latter is also referred to as *the makespan problem with time windows (MPTW)*. They were able to handle instances of up to 40 nodes.

Dumas et al. (1995) present a very effective optimal dynamic programming algorithm for the TSPTW with total travel cost minimization. Its effectiveness comes from the post-feasibility tests that exploit time window constraints in order to substantially reduce the state space and the number of state transitions. They report that their algorithm was successful in solving problems with up to 200 nodes and fairly wide time windows, as well as problems with up to 800 nodes within 650 seconds, provided that the density of the nodes in the geographical region was kept constant as the problem size was increased.

Pesant et al. (1998) propose a Constraint Logic Programming model for the TSPTW, which yields an exact branch and bound optimization algorithm. Ascheuer et al. (2001) solve the asymmetric TSPTW by branch-and-cut. Balas & Simonetti (2001) present an implementation of a linear-time dynamic programming algorithm for some variants of the TSP with precedence constraints, including the TSPTW. Focacci et al. (2002) describe a hybrid approach for solving the TSPTW which combines Constraint-Programming propagation algorithms with other optimization techniques.

Baldacci et al. (2012) introduce a new tour relaxation, the ngL-tour, to compute a valid lower bound on the TSPTW and solve the problem by column generation. They perform an extensive computational analysis on all the TSPTW instances of the literature (with up to 233 nodes), and their algorithm solves all but one instance, while outperforming all exact methods that were published in the literature up to that time.

Kara & Derya (2015) propose polynomial size integer linear programming formulations for the TSPTW with tour duration minimization, considering the symmetric and asymmetric cases separately. For the symmetric TSPTW, their proposed formulation succeeds in finding optimal solutions for all instances considered in a very short time with CPLEX 12.4, including some instances with up to 400 customers. For the asymmetric TSPTW, optimal solutions of most of the instances with up to 232 nodes are obtained within seconds.

Tilk & Irnich (2016) present a new effective approach based on dynamic programming, for the Minimum Tour Duration Problem (MTDP) (which is simply another term for the TSPTW with makespan optimization). Their approach generalizes the approach presented by Baldacci et al. (2012).

2.5.2 Heuristic approaches for the TSPTW

More than three decades ago, Savelsbergh (1985) showed that even finding a feasible solution to the TSPTW is an NP-complete problem. Therefore, a large number of researchers have focused on developing fast heuristics for the approximate solution of the TSPTW.

Carlton & Barnes (1996) solve the TSPTW using Tabu Search, considering infeasible solutions during the search and using a static penalty function. Gendreau et al. (1998) describe a generalized insertion heuristic for the TSPTW-TT. Calvo (2000) presents a heuristic for the TSPTW, which first solves an auxiliary assignment problem with an ad hoc objective function, in order to construct a solution which is close enough to a feasible solution of the original problem; then uses a greedy insertion procedure to insert all the subtours into the main tour and get a complete initial solution, and finally applies local search to further improve the current solution.

Cheng & Mao (2007) develop a modified ant algorithm, named ACS-TSPTW, which is based on Ant Colony Optimization (ACO) to solve the TSPTW. They test their algorithm on the 31 TSPTW benchmark problems proposed by Potvin & Bengio (1996) and conclude that its performance is comparable to that of ACS-Time, an existing ACO algorithm for solving the TSPTW that is used as a module of the multiple ant colony system (MACS), which was designed by Gambardella et al. (1999) for solving the VRPTW.

Ohlmann & Thomas (2007) solve the TSPTW using a Compressed Annealing heuristic, which is an extension of Simulated Annealing. Specifically, the Compressed Annealing method relaxes the time-window constraints and performs a stochastic search using a variable penalty function, generally outperforming Simulated Annealing with a suitable static penalty function. Computational results show that near-optimal solutions can be obtained at reasonable computational cost in most cases. Feasible solutions are found in every instance and best known results are obtained for numerous instances.

Da Silva & Urrutia (2010) propose a General Variable Neighborhood Search (GVNS) heuristic for the TSPTW, that is comprised of both constructive and optimization stages which take advantage of elimination tests, partial neighbor evaluation and neighborhood partitioning techniques. Computational results show that this approach is efficient, reducing the computational time by 80% on average, compared with the previously state-of-the-art heuristic by Ohlmann & Thomas (2007), and improving some best known results from the literature. A new set of instances with up to 400 customers is proposed and the algorithm is able to obtain feasible solutions for each one of these instances in a reasonable amount of time.

Mladenović et al. (2012) present another implementation of the GVNS for the TSPTW, which is faster than the GVNS implementation by Da Silva & Urrutia (2010), able to improve 14 large test instances from the literature.

Also, López-Ibáñez et al. (2013) adapt state-of-the-art algorithms for the TSPTW with travel-time minimization (TSPTW-TT), namely the Beam-ACO and compressed annealing algorithms, and apply them to solve the TSPTW with makespan minimization (TSPTW-M); thus improving on the existing state-of-the-art approaches for makespan minimization.

The list of papers for the TSPTW presented here is in no way exhaustive. More references can be found e.g. in Kona et al. (2015), that present a survey of the research on the TSPTW (and also include references for the TSP and the VRPTW).

2.6 Methods of dealing with uncertainties or disruptions in business operations

In the past years, researchers have been trying to deal with uncertainties occurring during business operations, using many different approaches. Following the description and categorization of Yu & Qi (2004), those approaches can be divided into two stages: the in-advance planning and the real-time re-planning. In the first stage the goal is to create a 'perfect' operational plan, based on estimates of the possible future uncertainties. In the second stage, the goal is to revise the original plan during its execution, if necessary.

The five main approaches used by researchers in order to cope with disruptions that may occur during the execution of an operational plan, are explained below:

1. *Contingency Planning.* This is a scenario-based approach. During the planning stage, the critical scenarios are identified, according to the severity of potential impact, the probability of occurrence or both. For

each scenario, all available options are identified, along with associated effectiveness and costs. Then, for each scenario, the best option is selected, based on available resources. In the final stage, extra resources that might be needed are identified, collected and placed in reserves. The formula and plan for dealing with each scenario are identified; these will be closely followed in the case of a disruption.

Obviously, contingency planning can only deal with a limited set of disruptions, where the exact characteristics of each disruption are known in advance and the associated costs and impacts can be calculated. However, in a large system, there is a huge number of possible future scenarios; so it is very difficult and often impossible to make a contingency plan for each scenario. If a disruption which was not included in the list of possible scenarios occurs, it may create havoc to the system. Some disruptions are unpredictable; some cannot even be identified ahead of time. So the limitations of this approach are obvious.

2. *Use of Stochastic models.* In a typical operational plan or policy that uses stochastic models, the first step is to build stochastic models, in order to describe future uncertainty. Then the models are analysed and the optimal policy is found so that the future output is optimized, with respect to the average output. When a disruption happens, the plan is closely followed.

   For the use of stochastic models, a complete list of all future uncertainties must be known, along with the associated probability of occurrence. Then, a contingency plan or policy is constructed which is optimal with respect to the average outcome. However, the in-advance knowledge of the precise probability distribution of future uncertainties is usually impossible. Therefore, stochastic models are usually not appropriate tools to deal with disruptions that might occur during the execution of an operational plan.

3. *Robust Optimization* is another approach that deals with uncertainty in the planning stage. Following this approach, a set of scenarios is used to model future uncertainties. An operational plan is generated, which is good for most scenarios and at the same time acceptable for the worst-case scenario. All possible disruptive scenarios must be specified, although the precise probability distribution associated is not required. Then, a robustness criterion appropriate is selected. A common criterion of this type is to minimize the maximum deviation from optimality under all possible scenarios. The advantage of such an approach is that it can guarantee the system's performance in any case

- even if the worst scenario happens. However, the solution might be too conservative, especially if the worst scenario has a very small probability of occurrence. A robust plan will sacrifice average performance to gain robustness against disruptions, especially if those disruptions are very unlikely to occur. Moreover, it is usually impossible to list all possible future disruptions in advance.

4. *Pure rescheduling.* People have been working for many years on the research of on-line scheduling problems, dynamic scheduling problems and rescheduling problems. Despite the sophisticated techniques used in most cases, many current rescheduling problems do not take into account the deviation costs involved. When the revision of a current schedule is costly, not considering the deviation costs doesn't provide a satisfactory solution in practice. Instead, we might end up with a sub-optimal solution.

5. *Disruption Management.* This approach is explained in detail in the following section.

2.7 Disruption Management

Disruption Management refers to dynamically revising an operational plan in real time, once a disruption occurs. It is very important in situations where the operational plan has been published in advance, and especially when its execution is subject to major disruptions. When the published operational plan is revised, some costs will occur that are associated with the transition from the original to the modified plan. The deviation cost might be a financial cost, caused, for example, by paying overtime to employees, or might be something not so easy to quantify, such as the dissatisfaction or loss of customers. It is essential to take those deviation costs into account when generating a new plan. A key feature of disruption management is the ability to deal with a disruption without knowing in advance that it is going happen and to provide good solutions in time for them to be implemented. Disruption Management is clearly the most promising approach among the five methods of dealing with disruptions that were described in the previous section.

A formal definition of disruption management can be found in Yu & Qi (2004): *"At the beginning of a business cycle, an optimal or near-optimal operational plan is obtained by using certain optimization models and solution schemes. When such an operational plan is executed, disruptions may occur from time to time caused by internal and external uncertain factors.*

As a result, the original operational plan may not remain optimal, or even feasible. Consequently, we need to dynamically revise the original plan and obtain a new one that reflects the constraints and objectives of the evolved environment while minimizing the negative impact of the disruption. This process is referred to as disruption management."

Disruptions may occur in many settings. In their book Yu & Qi (2004) provide a general framework for disruption management and a detailed study of the domain. They present an in-depth analysis of frameworks, models and applications of disruption management in each one of the following areas: airline operations, flight and crew scheduling, machine scheduling, logistics scheduling, discrete and continuous-time production and inventory management, supply chain management, project scheduling.

Clausen et al. (2010) provide a thorough review of models and methods used in disruption management in the airline industry, including aircraft, crew, passenger and integrated recovery. They also provide an overview of model formulations of the aircraft and crew scheduling problems, as well as an overview of research within schedule robustness in airline scheduling. In this article, a disruption is defined as *"a state during the execution of the current operation, where the deviation from plan is sufficiently large that the plan has to be changed substantially"*. Disruption management then refers to the replanning after a disruption has occurred.

Visentini et al. (2013) provide a review concerning the methods used for real-time vehicle schedule recovery in transportation services. They do not use the term "disruption management" in their title or abstract, preferring to refer to "real-time vehicle schedule recovery methods", but their review clearly covers disruption management methods as used in transportation services. They classify Real-Time Vehicle Schedule Recovery Problems (RTVSRP) into 3 categories: vehicle rescheduling for road-based services, train-based rescheduling and airline schedule recovery problems. They classify the models of each category according to the problem formulation and the solution strategy. The RTVSRP is usually modeled and solved using similar but not identical models as the respective off-line planning counterpart. A network structure is designed representing the new problem after disruption, which can describe how vehicles can be rescheduled. When designing such a network structure, one must take into account the current planned schedule, the current situation when the disruption occurred (for instance, the position of each vehicle at the time of the disruption), as well as other technical and timing constraints. Exact and heuristic optimization methods are designed based on this network representation. They report some interesting statistics which indicate the importance of the research on vehicle rescheduling for road-based services, including: (i) a delay in 29% of bus trips from 2007 to

2010, reported by the San Francisco Municipal Transportation Agency, and (ii) 82 road calls in a single month (July 2011) reported by SunTran transit agency in Tucson, Arizona.

Cacchiani et al. (2014) present an overview of recovery models and algorithms for real-time railway disturbance and disruption management. They consider algorithms for timetable, rolling stock and crew rescheduling, as well as algorithms that consider the integration of different rescheduling phases. They describe various approaches to solve these rescheduling problems in terms of the type and scale of the disturbances or disruptions, the network infrastructure and topology, the considered objective functions and constraints, and the utilized optimization methods.

The key factors that are involved in disruption management are as follows:

(i) The time for replanning may be limited. This will normally be the case and so limits the computation time available for any algorithm used to produce a revised plan for recovery. The time needed to receive information concerning the disruption and to communicate the revised plan to those who implement it must also be taken into consideration.

(ii) The original undisrupted plan may be a useful starting point for the new plan. When constructing the revised plan, there is no need to determine a complete plan from scratch. In disruption management, there will always be an original plan to consult.

(iii) It may be appropriate to include new costs relating to deviations from the original plan. Whereas the original plan may have been created with a single objective of minimizing relevant costs, the disruption management model may need to include other considerations, particularly the costs of deviating from the original plan. The disruption management model is therefore often a multi-objective model and the relevant objectives will be discussed later.

(iv) There may be constraints in the new plan that were not in the original. These constraints may be as a result of the disruption that has occurred, such as a blocked road due to an accident or the unavailability of a failed vehicle. Sometimes there will be additional constraints due to commitments that have been agreed after the original plan was published.

2.8 Disruption Management in Vehicle Routing and Scheduling for freight distribution

2.8.1 Relevant objectives

When determining the original plan for vehicle routing and scheduling in the context of road freight transport, it is usual to focus on an objective of minimizing the relevant costs subject to constraints on the operation. Sometimes fixed costs of using vehicles and their drivers are included in the objective to be minimized, while in other cases, the number of available vehicles may be regarded as a constraint and the objective to be minimized depends only on distance-related costs.

After a disruption has occurred, there will still normally be a focus on minimizing the costs of the revised plan that deals with the disruption, but the following additional objectives may also need to be taken into consideration that may not have been included in the original plans:

(i) The deviation from the original plan for customers. Customers may have made arrangements to receive deliveries at particular times according to the original schedule. For some sorts of deliveries, customers may be quite flexible about the timing of deliveries, but in other cases where members of staff need to be available to receive goods or when deliveries have been scheduled to avoid overlaps with other delivery operations, then the customer will experience a degree of dissatisfaction with a significant change in delivery time. The change may lead to increased costs for the customer, such as when additional overtime payments are incurred for members of staff who are waiting to help unload the delivery. Some estimate of this cost can be incorporated as an additional objective to be minimized in the disruption management model. An operator may also wish to give a higher priority to more important customers in terms of the size of the deviation from the original plan and this too may be incorporated in this objective. There may also be trade-offs to be considered in terms of the number of customers who will have a late delivery against the total deviation from the planned delivery schedule.

(ii) The deviation from the original plan for drivers. There may be direct extra costs involved for drivers if the revised plan involves the use of overtime or special payments. Additionally, there may be problems in requiring a driver to deliver to an unfamiliar customer. This could increase the service time if the driver assigned is unfamiliar with access arrangements at the customer. Also the customer may prefer to deal with a familiar driver. These are all factors which may be important in assessing the revised plan.

When the objectives for a particular application have been decided, it may be possible to convert them all to monetary values and so have a disruption management problem with a single objective of minimizing the sum of these relevant costs. Where there are more subjective issues to consider, then it may be preferable to adopt a multi-objective approach with a disruption management model that will present a set of Pareto optimal alternative solutions to the decision maker. However this approach generally requires more time to compute and select the revised plan and so may be more difficult to achieve in the limited time available before the revised plan is needed.

2.8.2 Formulation and Solution Approaches

There are various ways to formulate a disruption management problem, which depend on the details of the application and the amount of flexibility that is to be allowed in revising the original plan.

One important consideration is the type of commodity which is being distributed to customers and whether the goods loaded into vehicles at the start of each trip are only for the specific customers to be visited by that vehicle or whether the commodity is a general one where any customer can be served by any vehicle. In the case of delivering packages or parcels addressed to specific recipients, then the goods loaded must be regarded as customer-specific. If a commodity such as gas canisters or bottled water is to be distributed, then any customer can receive a delivery from any vehicle and the commodity can be regarded as non customer-specific. This distinction is important in terms of the feasibility of the revised plans. For example, following a vehicle breakdown, if the vehicle is carrying customer-specific commodities, then it must be visited by other vehicles to pick up the remaining goods it was carrying before taking them to other customers. But if the failed vehicle was carrying non customer-specific commodities, then there is no need for it to be visited by any of the other vehicles delivering to the remaining customers on its route.

Another issue is the degree of flexibility that is to be allowed in the revised plan. For example, some formulations assume that following a vehicle breakdown, the remaining customers on its route will be served by one of the other vehicles in the same order after completing one of their original trips. This simplifies the formulation of the disruption management problem. Such a formulation can be easily modified for situations involving non customer-specific and customer-specific commodities. However there are cases where less costly revised plans can be built if the remaining customers that were to have been visited by the failed vehicle are allowed to be served by two or more different vehicles.

For example, suppose that there are two customers remaining to be served on a trip affected by a vehicle breakdown and non customer-specific commodities are being distributed. If two of the other vehicles have enough capacity to add a delivery to one of those remaining customers, but not both, then revising the routes of these two vehicles to serve one of the remaining customers each is likely to be cheaper than a revised plan where a vehicle needs to reload at the depot, or another vehicle needs to be employed, in order to serve both the remaining customers on the affected trip.

A further detail of the formulation that should be specified is when vehicles can be re-routed after starting a trip in the original plan. Some formulations require that an existing trip must be completed by a vehicle before it can be used to respond to the disruption. In other cases, the formulation may allow a vehicle to divert from its original route during a trip. The change in route might only be allowed after the vehicle has finished serving the next customer on its route, which could aid implementation if the driver can only receive new instructions from the transport manager while at a customer location. If there are good communication facilities between the transport manager and the drivers, then it may be feasible to divert a vehicle during its journey between customers, but in this case, it is particularly important to allow for the time needed between the disruption being reported and the production of the revised plan when implementing the changes.

Particularly in the case of vehicle breakdown, an important consideration is whether another vehicle and driver are available at the depot or not. If one is available or can be hired, then any additional costs incurred should be taken into account in the formulation.

There are also differences in disruption management formulations regarding the service to customers included for service in the original plan. Some formulations assume that all original customers must receive their service, even if it is at a different time to the original plan, while other formulations allow for the possibility of canceling some customer orders. In this case, the loss in profit or a customer dissatisfaction cost is normally included in the objective.

The disruption management problem is normally some sort of optimization problem, subject to constraints. It may be expressed as an integer linear program. If this is done, there is then the question of whether to solve it exactly or whether it is more appropriate to devise a heuristic algorithm to obtain a good answer quickly. This will depend on how the problem has been formulated, the size of the problem instances and the computational time that is available to produce and communicate the revised plan. The vehicle routing problem itself is well known to be NP-Hard and so it might be expected that an associated disruption management problem may also be NP-Hard. This may not necessarily be the case, though some papers contain proofs that their formulation of the disruption management problem is NP-Hard. Even if the disruption management problem is NP-Hard, it may still be viable to make use of an exact approach in practice and an example is provided by J.-Q. Li et al. (2008a).

As has already been mentioned, in disruption management there are potentially several objectives that are relevant. It is therefore important to consider whether one of these objectives (e.g. the operational costs) is so important that only this objective needs to be taken into consideration in a single objective formulation. Alternatively, some form of multiple-objective approach could be employed. This could be a simple weighting system where a single objective to be optimized is formed by summing the values of the relevant objectives weighted by parameters that reflect their importance. A lexicographic approach could be employed where the relevant objectives are ordered in terms of importance and the revised disruption plan is made taking these priorities into account. As in other multi-criteria problems, methods are available to determine a set of non-dominated solutions forming a Pareto front from which the decision maker can select a preferred solution. However there may not be enough time available before the revised plan needs to be implemented for all these potential solutions to be calculated and one selected.

2.8.3 Discussion of relevant papers

In this subsection, a set of papers have been selected describing an approach to disruption management in the context of vehicle routing and scheduling for freight distribution. Some of the papers employ a methodology that can be applied more widely, but the focus in this subsection is on the freight distribution scenario. Papers by the same author or group of authors are presented consecutively, while the rest of the papers are listed according to the year of publication.

Table 2.1 provides a summary of the papers covered in this subsection. The first column references the paper. The second column indicates the main type or types of disruptions addressed in the paper where V denotes vehicle breakdown, L denotes a disrupted link in the road network, S denotes a disruption in the supply of goods and C denotes a disruption in customer demand. The third column lists the main objectives considered in the paper and the final column indicates the solution approach.

J.-Q. Li et al. (2007a) introduce the Vehicle Rescheduling Problem (VRSP), which arises in cases when a severe disruption such as an accident, a me-

Table 2.1: Summary of papers on Disruption Management in Vehicle Routing and Scheduling for freight distribution

Reference	Type of disruption	Objective	Solution approach
J.-Q. Li et al. (2007a)	V	Op. cost, delay	Auction-based alg.
J.-Q. Li et al. (2007b)	V	Op. cost, delay	Auction-based alg.
J.-Q. Li et al. (2008a)	V	Op. cost, delay	Exact alg. using CPLEX
J.-Q. Li et al. (2009a)	V	Op., fixed vehicle, service cancellation, route disruption costs	Lagrangian heuristic
J.-Q. Li et al. (2009b)	V	Schedule disruption, trip cancellation costs	Lagrangian heuristic
Ernst et al. (2007)	V, S, C	Lost sales, relocation, substitution and delay costs	Heuristic
Zhang & Tang (2007)	V, L	Min. distance, deviations from plan	ACO with Scatter Search
X. Wang & Cao (2008)	C	Op. cost, deviations	Local search
X. Wang et al. (2009a)	V	Customers serviced, deviations, op. cost	Genetic algorithm
X. Wang et al. (2009b)	C	Op. cost, deviations	Genetic algorithm
Z. Wang & Shi (2009)	L	Customers serviced, deviations, op. cost	Genetic algorithm
X. Wang et al. (2010)	V	Service times, deviation of routes, op. cost	Lagrangian relaxation
X. Wang et al. (2012)	V, S, C	Service times, deviation of routes, op. cost	Nested Partitions Method
Mu et al. (2011)	V	Op. cost	Tabu search
Mu & Eglese (2013)	S	Op. cost	Tabu search
Hu & Sun (2012)	V, L, C	Travel distance, time violations	Knowledge based approach, local search
Hu et al. (2013)	V, L, C	Op. cost, affected customers, time window violations, rejected demand	Knowledge based approach, local search
Minis et al. (2012)	V	Max. profit	Genetic algorithm
Mamasis et al. (2013)	V	Routing costs, customer priorities	Heuristic
Ngai et al. (2012)	V, L, C	Operating cost	Model management subsystem
Dhahri et al. (2013)	V	Distance and time window deviation	VNS
Jiang et al. (2013)	V, L	Min. delay, distribution cost, deviation of routes, customer priorities	Genetic Algorithm
Spliet et al. (2014)	C	Cost of deviation	Heuristic and MIP

chanical failure of a vehicle or traffic congestion, prevents the execution of the scheduled trips as planned. For instance, in the event of a vehicle breakdown, one or more vehicles must be rescheduled in order to serve the passengers/cargo of that trip. The VRSP seeks to serve the passengers/cargo on the affected trip and complete all remaining trips, while minimizing the operation and delay costs. The authors present a prototype decision support system which recommends solutions for the single-depot vehicle rescheduling problem (SDVRSP) and for the single-depot vehicle scheduling problem (SDVSP). They use a quasi-assignment formulation and a combined forward-backward auction algorithm, developed by Freling et al. (2001) to solve both the SDVSP and the SDVRSP, where the latter is treated as a sequence of SDVSP problems. The rescheduling decision support system is used in a real world problem involving the operational planning of solid waste collection for a city in Brazil.

J.-Q. Li et al. (2007b) also study the single-depot VRSP with focus on its modeling, algorithmic and computation aspects. A model formulation and several fast algorithms are presented, including parallel synchronous auction algorithms. They introduce the concept of the common feasible network (CFN). Among the algorithms they consider, parallel CFN-based auction algorithms have the best performance.

J.-Q. Li et al. (2008a) provide a further investigation of the case study involving waste collection in Porto Alegre, Brazil that was included in J.-Q. Li et al. (2007a). This study concentrates on the rescheduling problem where a trip that has been scheduled is cut due to the breakdown of one of the vehicles. The objective is taken to be minimizing the sum of the operational and delay costs. The formulation is presented as a non-linear program which is converted using a standard technique into an integer linear program. Additional constraints had to be taken into consideration, such as the need to obtain approximately balanced assignments of truck loads to different recycling facilities. Computational results are reported based on real-world data. The instances used were relatively small, consisting of 23 vehicles and 31 trips, which meant that the integer linear programs could be solved optimally using CPLEX in a short CPU time. The results showed reductions in the distances traveled and the time delays compared to the manual strategy that was employed.

J.-Q. Li et al. (2009a) introduce and study Real-Time Vehicle Rerouting Problems with Time Windows, applied in the case of a disruption to delivery and/or pickup services due to vehicle breakdowns. They define the Real-Time Vehicle Rerouting Problem with Time Windows as follows: "*Given a depot, a number of vehicles with the limited capacities, and a set of customer services having demands with time windows, given the travel time between all pairs of* *locations, given the vehicle routes originally planned, and given a breakdown vehicle with the breakdown time and position, find feasible route reassignments with the minimum weighted sum of operating, fixed vehicle, service cancellation and route disruption costs, in which (i) each vehicle performs a feasible sequence of services, satisfying constraints related to the service time windows and vehicle capacities; and (ii) a service is either satisfied, or cancelled with a large service cancellation cost when some constraints cannot be satisfied.*" The authors present a path-based formulation of the problem and a heuristic incorporating Lagrangian relaxation and an insertion based primal heuristic. They also use a dynamic programming based heuristic to solve the Lagrangian relaxation problem quickly, but not necessarily to optimality. Computational experiments are performed on benchmark problems taken from Solomon (1987) to show the effectiveness of the proposed algorithm. Additionally, the authors show that the Real-Time Vehicle Rerouting Problems with Time Windows, both in the case of a pickup and in the case of a delivery service, are NP-hard by association with the VRPTW and the TSPTW, respectively.

J.-Q. Li et al. (2009b) addresses a similar disruption management problem to the one studied in J.-Q. Li et al. (2009a). The approach is characterised by a formulation that seeks to minimize the total weighted sum of costs relating to operating costs, disruption costs and trip cancellation costs. In this paper, computational experiments are carried out using a range of randomly generated problems. The results are compared with a simple intuitive approach and it is found that the proposed algorithm is more beneficial compared to the intuitive approach when there is no back-up vehicle available at the depot or when breakdowns occur later rather than earlier in a trip.

Ernst et al. (2007) describe a case study where software has been developed to assist a recreational vehicle rental company based in New Zealand in its operations. The dynamic version of the software is referred to as the Dynamic Vehicle Assignment and Scheduling System (D-VASS). It is used to respond to availability queries from reservation staff, but can also be used to modify the schedule when there are disruptions such as vehicle breakdowns or delays in returning vehicles. The heuristic developed aims to minimize the sum of costs due to foregone profit if potential rentals are not included, relocation costs, substitution costs and costs due to delayed starts beyond requested commencement times, subject to a set of constraints. The heuristic is based on the successive shortest path method for solving the assignment problem.

Zhang & Tang (2007) present a rescheduling model for a Vehicle Routing Problem with Time Windows. The paper concentrates on vehicle disruptions, where a vehicle becomes unavailable due to a breakdown or a traffic incident. The objective is to find a new schedule that minimizes total distance and deviations from the original plan. A hybrid algorithm which combines ant colony optimization (ACO) with scatter search is proposed to solve the disruption problem. Computational results are reported to show the effectiveness of the hybrid algorithm.

X. Wang & Cao (2008) address the case where, during the execution of a solution plan for a vehicle routing problem with backhaul and time windows (VRPBTW), a disruption occurs in the form of a change in demand concerning the demand for backhaul services. A recovery model is proposed considering the disruption to customers from deviations to planned service times and the operating costs to the logistics service provider. Two strategies and a local search algorithm are designed to find an optimal or near-optimal solution in real time.

X. Wang et al. (2009a) present a disruption management model to solve the vehicle routing problem with vehicle breakdown. The problem under study and the proposed model are similar to the ones presented in X. Wang et al. (2010), which is discussed in more detail later. This paper uses a genetic algorithm for its solution approach.

In X. Wang et al. (2009b) the emphasis is on applying a disruption management approach when there are changes to the customers and their demand requests. A model is constructed which is compatible with the VRPTW. The solution approach is based on a genetic algorithm and computational results are presented to illustrate the method used.

Z. Wang & Shi (2009) address the case of a travel time delay of a route in a single-depot VRPTW. The delay could be caused by traffic congestion due to an accident or another unexpected event during a vehicle's route. A disruption measurement method is presented which takes into account the customer unsatisfied index, the deviation from the original transport network and the transport cost. A disruption recovery mathematical model and a genetic algorithm are constructed. The model and solution method are similar to those discussed in X. Wang et al. (2009a). Experimental results are provided for a small set of problems and show that the model's multiple objectives can be met simultaneously, according to the disruption evaluation criteria.

X. Wang et al. (2010) study the "Urgency Vehicle Routing Disruption Management Problem" where a disruption caused by a vehicle breakdown occurs in a logistics distribution system. They propose a mathematical model which is based on the theory of disruption management. A Lagrangian relaxation is used to simplify the model, decomposing it into two parts. An insertion algorithm is then used to obtain a feasible solution for the primal problem. When a disruption occurs, the authors assume that there are extra vehicles available at the depot. They also allow the cancellation of servicing of some customers. In order to quantify the magnitude of the disruption, the authors focus on three aspects of a solution: the service time of customers, the routes of service vehicles and the costs of the logistics provider. The model involves three objective functions. The first objective is the sum of the differences of the service time between the new plan and the original plan, for each customer. This is the total time deviation for the whole set of customers, which seeks to be minimized, and reflects the need that the service times in the new plan should be as close to the ones in the original plan as possible. The second objective seeks to minimize the deviation of routes between the original plan and the recovery plan. The third objective seeks to minimize other costs incurred by the deviation from the original plan, including the costs from deviation of routes, the cost of sending additional vehicles and the cost of canceling some customers. The Lexicographic approach is used to deal with the multiple objectives.

X. Wang et al. (2012) propose a recovery model for disruptions for the VRPTW, capable of handling a combination of disruptions. They try to follow closely the principles of Disruption Management and aim to reduce any deviations from the original plan. The authors report that, according to interviews with delivery drivers, drivers usually prefer routings they are familiar with, because they know the customers, the exact locations and the delivery processes. Customers also prefer familiar delivery staff. Therefore, they propose a mathematical model with the following three objectives. The first objective, which is a measurement of the disruption on customers, seeks to minimize the deviation of the time of start of service of customers between the original plan and the recovery plan. The second objective, which is a measurement of the disruption on drivers, seeks to minimize the deviation of routes between the original plan and the recovery plan. Apart from penalizing the change in travel distance, this objective penalizes any change which involves moving a customer from one route to another, or exchanging customers between routes. The third objective, which is a measurement of the disruption on providers, seeks to minimize the deviation of the total delivery costs between the original plan and the recovery plan (which is equivalent to minimizing the total delivery costs of the recovery plan, since the costs of the original plan are fixed). They define the delivery costs as a linear combination of the total travel distance and the number of vehicles used. In their study they consider the following types of disruption: vehicle breakdown, blocked vehicle, damaged cargo (fully or partially), change of customer's time windows, change of customer's delivery address, change of customer's demand amount, removal of customers. They also consider combinations of those disruptions occurring successively or simultaneously.

They apply the Nested Partitions Method (NPM) to solve the problem. The authors conclude by acknowledging the need for a more effective and efficient multi-objective optimization algorithm, which would aid in the development of a decision support system for different disruption events that occur during logistics deliveries.

Mu et al. (2011) study the disrupted VRP with vehicle breakdown during the execution of a vehicle routing plan. It is assumed that vehicles deliver a single commodity, which is transferable between customers (such as oil or gas), so that any customer can be served by any vehicle, as long as the vehicle carries enough quantity to serve him/her. The case of an extra vehicle being available at the depot is considered. All customers must be served and there is no upper bound on the total length or duration of the routes. The paper shows the link between the formulation adopted for the disruption management problem and an Open Vehicle Routing Problem (OVRP). Of course, this particular type of the disrupted VRP is not exactly the same as an OVRP, since in the disrupted VRP both ends of each route are fixed once a disruption occurs. Two Tabu Search heuristic algorithms are developed and tested over a set of test problems generated for the purpose. The computational results of the heuristics are then compared to an exact algorithm based on the method proposed by Letchford et al. (2007) for the OVRP.

Mu & Eglese (2013) study the situation when the supply of the commodity does not arrive at the depot on time, so that not enough of the commodity is available to be loaded on all vehicles at the start of the delivery period. The Disrupted Capacitated VRP with Order Release Delay (DCVRP-ORD) is introduced, which involves multiple trips and allows some vehicles to wait at the depot. Two Tabu Search heuristic algorithms are proposed. The authors assume that vehicles deliver a single commodity, which is transferable between customers (such as oil or gas), so that any customer can be served by any vehicle, as long as it carries enough quantity to serve him/her.

Hu & Sun (2012) present a knowledge-based modeling approach for disruption management in urban distribution. The knowledge of experienced schedulers is combined with operations research models and algorithms to revise the distribution plan and respond to a disruption in real-time. Policies, algorithms and models are represented by appropriate knowledge representation schemes, in order to support automated or semi-automated modeling by computers. The integration of the two kinds of knowledge combines the advantages of both and the approach is developed in Hu et al. (2013) which is described below.

Hu et al. (2013) develop a knowledge-based modeling approach, referred to as PAM (disruption-handling Policies, local search Algorithms and object-oriented Modeling), which can dynamically handle disruptions in Real-time

Vehicle Routing Problems. Their study is similar to the one presented in Hu & Sun (2012). This approach combines the scheduling knowledge of experienced schedulers with the optimization knowledge concerning OR models and algorithms, to obtain an effective solution in real time. Types of disruptions examined in this paper include: postponed delivery time, advanced delivery time, repairable vehicle breakdown, road construction, demand increasing, order canceling, disabled road, disabled vehicle. The first four types of disruption are handled using the same 'time violation' policy, whereas a separate policy is used for each of the remaining four types of disruption.

Minis et al. (2012) also study the case where, during the execution of the original VRP plan, a vehicle breakdown occurs. The problem is modeled as a variation of the Team Orienteering Problem (TOP), with a fixed upper bound on the length in time or distance of each route, and fixed vehicle capacity. A heuristic is proposed, which compares favorably to a computationally more expensive Genetic Algorithm. Specifically, for a computational time of 10 minutes, the heuristic results are only 3% away from those of the GA for the most complex problem set used (98 clients). The GA was used to set benchmark solutions for the heuristic, which is justified by the fact that the GA was used to solve TOP benchmark problems and gave results very close to the best published results. The authors assume that the orders are customer-specific, so each vehicle can only serve its own customers and the customers of the failed vehicle. There is no extra vehicle available at the depot. The problem is modeled as a profit maximization problem, in which a set of vehicles are routed to maximize the total reward accumulated by serving clients, within a predefined time horizon, where some of the customers might not be served. Priority is given to the most important customers, where the importance is considered equivalent to the reward of the TOP. In order to serve clients of the failed vehicle, an active vehicle must first visit the location of the failed vehicle and load the relevant orders from the failed vehicle to the active one. An active vehicle is allowed to visit the failed vehicle more than once. The proposed heuristic has been incorporated in a real-time fleet management system of a food company operating in the region of Attica, Greece, and tested in practical breakdown cases with success.

Mamasis et al. (2013) also address the case where, during the distribution of a single product to a set of customers by a fleet of vehicles, a single vehicle is immobilized (vehicle breakdown). Some active vehicles are then rerouted to serve selected clients of the immobilized vehicle. This re-planning problem is modeled as a variation of the Team Orienteering Problem. All vehicle routes are constrained to an upper time or distance limit. A vehicle capacity constraint is also present. There are no extra or back-up vehicles available at the depot. Active vehicles are allowed to visit the depot and/or the immobilized vehicle (more than once, if needed), for replenishment purposes. Clients are served based on their importance, which is a rating assigned a priori to each client. Some clients may remain unserved. Reallocation of clients among vehicles is feasible, given that the respective vehicles carry enough quantity of the product to satisfy clients' demand, since the product to be delivered is common to all clients. The rerouting decisions in the modified plan are based on client importance, routing costs (times) and capacity restrictions. A heuristic is proposed which can provide solutions in almost real-time. The effectiveness of the heuristic is tested by comparing its solutions with those obtained by an appropriate Genetic Algorithm, which yields high quality results but is computationally expensive. Some applications of the problem mentioned in this paper include: Distribution systems dealing with critical commodities, such as vital supplies in relief logistics, cash in money transfer operations, transfer of staff or ammunition from incapacitated military vehicles (in battlefield operations), or in general distribution of any single product (e.g. bottled water).

Ngai et al. (2012) is an example of a paper where the emphasis is on the design of a complete decision support system that can help to reschedule a distribution plan following an accident or other incident which disrupts the original plan. The system includes a model management subsystem which aims to minimize the total operating costs of the rescheduled plan. A prototype system was built and evaluated for a case study concerning a supplier of portable toilets based in Hong Kong.

Dhahri et al. (2013) present a rescheduling model based on Variable Neighborhood Search (VNS), for the VRPTW when one or more vehicles require maintenance activities after the supply of a set of customers. In this paper only one type of disruption is considered, namely vehicle breakdown.

Jiang et al. (2013) study two variants of the disrupted VRP: A single-vehicle delivery disruption management recovery model with service priority, and a more generalized multi-vehicle version of it. The customers are divided into high-priority and low-priority customers. When a delay happens, priority is given to serving all the key customers in their original time windows; this is imposed as a hard constraint. A genetic algorithm is proposed to solve the problems.

Spliet et al. (2014) study the Vehicle Rescheduling Problem (VRSP). In the classical CVRP the demand is deterministic and known before planning. This paper considers the situation where the demand becomes known at a late moment. For example, in the retail industry a common situation occurs when individual stores place their orders a few days, or sometimes just one day, before delivery. In such a case, it is beneficial for operational processes to prepare the delivery plan before the orders are placed. A common practice in this case is to have a long term schedule or master schedule prepared, serving as a guiding plan over a certain period of time during which multiple deliveries are made. For example, a master schedule could describe the weekly or daily deliveries for the next 6 months. Such a master plan is usually created by solving deterministic CVRP instances based on the average customer demands and demand predictions for the upcoming period. Since the master schedule is prepared before demand realizations become apparent, when the demand becomes known, this schedule may no longer be optimal, or even feasible. In these cases, which occur frequently in practice, the master schedule has to be revisited. A new schedule has to be constructed, once demand realizations become known, which will typically deviate from the master plan. Given the master schedule and a demand realization, the VRSP seeks to find a new schedule which minimizes the total travel cost and the cost of deviating from the master plan, while respecting the capacity constraints. The authors present a MIP formulation for the rescheduling problem, based on a CVRP formulation by Baldacci et al. (2004), along with a two-phase heuristic solution method.

2.9 Additional Literature on Disruption Management in Vehicle Routing and Scheduling

In this section we review a number of additional papers relevant to disruption management in vehicle routing and scheduling, which do not necessarily refer to freight transportation.

J.-Q. Li et al. (2008b) consider the case when a bus on a scheduled trip breaks down and in response, one or more buses need to be rescheduled in real-time, in order to serve the customers on the affected trip with minimum operating and delay costs. This problem is referred to as the bus rescheduling problem (BRP). Modeling, algorithmic and computational aspects of the single-depot BRP are considered in the paper. Sequential and parallel auction algorithms are developed and experimental results show that the proposed approaches are able to produce solutions quickly.

Mirchandani et al. (2010) describe a macroscopic model for integrating Bus Signal Priority with Vehicle Rescheduling. In case of a bus breakdown on a scheduled trip, one or more vehicles need to be rescheduled to serve that trip and other scheduled trips. The vehicle rescheduling problem (VRSP) seeks to reassign and reschedule the bus fleet in order to minimize the sum of operating costs, delay costs, schedule disruption costs and trip cancellation costs. Bus signal priority (BSP) is an effective strategy to facilitate bus movement through signalized intersections. If BSP is integrated with the vehicle rescheduling operations, the travel time of the backup bus to the stranded passengers can be reduced. Therefore, the combination of BSP and VRSP effectively reduces the delay and the associated costs, which can be verified from the experimental results.

Carosi et al. (2015) propose a decision support tool for the real-time delay management in the context of public transportation. A simulation based optimization system is proposed, which takes into account both vehicle and driver shifts. They describe a tabu-search procedure for the online vehicle scheduling, optimizing the regularity of the service, and a column generation approach for the consequential crew rescheduling to minimize the drivers' extra time. As a case study, the authors analyze the management of urban surface lines of a public transportation company operating in Milan.

Sun et al. (2016) propose an ontology-based model for typical context awareness in the oil products distribution system. The model consists of the disruption ontology, the distribution-state ontology and the SWRL rules. During the distribution process of oil products, a range of different types of disruptions can occur at different distribution states. Examples of such disruptions include gas stations changing their demands, vehicle breakdowns, blocked links induced by accidents and traffic jams. Vehicle routes must be readjusted in real time in order to respond to the disruptions; otherwise the oil products distribution process will be interrupted, which can negatively affect the social production and citizens' living.

Uçar et al. (2017) present a recovery method to handle two types of disruptions arising in the multi-depot vehicle scheduling problem: delays and extra trips. During the execution of a vehicle (or bus) scheduling problem (VSP), a delay may be caused by adverse weather conditions or by traffic congestion. The second type of disruption may be caused by the excess demand for a particular origin-destination pair that mandates the planners to insert an extra trip into the schedule at the operation phase. Both types of disruption are quite common in countries with a large bus transportation network (e.g. Turkey). A trivial recovery option for both types of disruption is to use an extra vehicle to cover the trips succeeding the delayed trip, or to cover the extra trip itself. However, this option would involve additional costs and may cause issues in crew scheduling. Hence, a much more preferable option would be to recover the delay or cover the extra trips by rescheduling the routes of the active vehicles. The disruptions are indirectly incorporated into the planned schedule by anticipating their likely occurrence times. The proposed disruption recovery method is based on partially swapping two planned routes, so that if these disruptions actually occur, the effect on the planned schedule would be minimal. A mathematical programming model incorporating these robustness considerations is proposed. An exact simultaneous column-and-row generation algorithm is designed, which can be used to find a valid lower bound. Finally, a heuristic is designed, capable of yielding small optimality gaps in a short computation time and present recovery solutions that can actually be operated when the disruptions are realized.

Ng et al. (2017) present a multiple colonies artificial bee colony algorithm for a Capacitated VRP and re-routing strategies under time-dependent traffic congestion. A flexible delivery rerouting strategy is proposed, which aims at reducing the risk of late delivery.

2.10 Multi-objective Optimization

We will now present some basic concepts, notation and a general formulation of multi-objective optimization problems, following closely the book by Branke et al. (2008), which constitutes a comprehensive guide to this field. A *multi-objective optimization problem* is typically a problem of the form

$$\begin{aligned} &\text{minimize} \;\; \{f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})\} \\ &\text{subject to} \;\; \mathbf{x} \in S \end{aligned} \tag{2.10.1}$$

that involves k ($k \geq 2$) conflicting *objective functions* $f_i : \mathbb{R}^n \rightarrow \mathbb{R}$ which we wish to minimize simultaneously. The decision vectors $\mathbf{x} = (x_1, x_2, ..., x_n)^T$ belong to the *feasible region* $S \subset \mathbb{R}^n$, where $S \neq \emptyset$. The images of the decision vectors are called *objective vectors*, $\mathbf{z} = \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x}))^T$, and consist of *objective values* $f_1(\mathbf{x}), f_2(\mathbf{x}), ..., f_k(\mathbf{x})$. The image of the feasible region in the objective space is called the *feasible objective region* $Z = \mathbf{f}(S)$.

In multi-objective optimization, objective vectors are considered to be optimal if it is impossible to improve any one of their components, without deteriorating at least one of the remaining components. Specifically, a decision vector $\mathbf{x}' \in S$ is termed *Pareto optimal* if there does not exist another decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) \leq f_i(\mathbf{x}')$ for all $i = 1, ..., k$ and $f_j(\mathbf{x}) < f_j(\mathbf{x}')$ for at least one index j. The set of Pareto optimal decision vectors is usually denoted by $P(S)$. An objective vector is Pareto optimal if the corresponding decision vector is Pareto optimal. The set of Pareto optimal objective vectors is usually denoted by $P(Z)$.

The set of Pareto optimal solutions is a subset of the set of weakly Pareto optimal solutions. A decision vector $\mathbf{x}' \in S$ is called *weakly Pareto optimal* if there does not exist another decision vector $\mathbf{x} \in S$ such that $f_i(\mathbf{x}) < f_i(\mathbf{x}')$ for all $i = 1, ..., k$.

Because vectors cannot be ordered completely, all the Pareto optimal solutions (or *non-dominated solutions*) can be regarded as equally desirable in the mathematical sense, and we need a decision maker to identify the most preferred one among them. This is because, despite the fact that typically multiple non-dominated solutions exist, in practice, usually only one of them is eventually chosen. Therefore, there are at least two important tasks involved in multi-objective optimization: (i) an optimization task for finding the set of Pareto optimal solutions, and (ii) a decision-making task for choosing the single most preferred solution (which requires preference information from a decision maker).

Multi-objective optimization methods are often classified into the four following classes, according to the role of the decision maker in the solution process: (i) no-preference methods, where there is no preference information available from the decision maker, (ii) a priori methods, where the decision maker first provides preference information and aspirations and then the solution process tries to find a Pareto optimal solution satisfying those as well as possible, (iii) a posteriori methods, where a representation of the set of Pareto optimal solutions is generated first, and then the decision maker is asked to select the most preferred one among them, and (iv) interactive methods, where an iterative solution algorithm is formed and repeated several times, and after each iteration, some information is given to the decision maker and he/she is asked to specify preference information. Some of the basic methods of non-interactive multi-objective optimization are: weighting method, ϵ-constraint method, method of global criterion, neutral compromise solution, method of weighted metrics, achievement scalarizing function approach, value function method, lexicographic ordering, and goal programming.

Marler & Arora (2004) present a survey of continuous non-linear multi-objective optimization (MOO) concepts and methods. Also, Alves & Climaco (2007) provide a review of interactive methods devoted to multi-objective integer and mixed-integer (MOIP/MOMIP) programming problems.

Below we describe two important methods for solving multi-objective optimization problems, which we use extensively in our research (following Branke et al. (2008)): the Weighting Method and the Epsilon-Constraint Method. We also mention the more recent Quadrant Shrinking Method, before we present a short discussion on performance metrics in multi-objective optimization.

2.10.1 Weighting Method

One of the most well-known and widely used multi-objective optimization methods is the weighting method. This method refers to solving the following problem:

$$\begin{aligned} &\text{minimize} \quad \sum_{i=1}^{k} w_i f_i(\mathbf{x}) \\ &\text{subject to} \quad \mathbf{x} \in S \end{aligned} \tag{2.10.2}$$

where $w_i \geq 0 \;\; \forall \;\; i = 1, ..., k$ and, typically, $\sum_{i=1}^{k} w_i = 1$.

The solution of this problem can be shown to be Pareto optimal if $w_i > 0 \;\; \forall \;\; i = 1, ..., k$ or if the solution is unique (e.g. see Miettinen (1999)).

The weighting method can be used both as an a posteriori method, and as an a priori method. In the first case, different weights are used to generate several different Pareto optimal solutions, and afterwards the decision maker is asked to choose the single, most satisfactory solution among these. In the second case, the decision maker is initially asked to specify the weights, according to his or her relevant preference of the objectives, and then the problem is solved.

In general, in multi-objective optimization it is important that the solutions generated are Pareto optimal, and that any solution that belongs to the set of Pareto optimal solutions can be found. From this point of view, the weighting method has some serious limitations. It can be shown that any Pareto optimal solution can be reached by varying the weights only if the problem is convex. So, it is possible that some non-dominated solutions of non-convex problems cannot be detected using this method, no matter how the weights are selected. Therefore, the weighting method does not work correctly for non-convex problems.

Suppose that we are interested in using the weighting method as an a posteriori method, so that we vary systematically the weights w_i, $i = 1, ..., k$, and for each choice of weights we solve a mathematical program of the form (2.10.2). It is generally advisable that the weights are normalized with some scaling. Systematic ways of perturbing the weights in order to reach different Pareto optimal solutions can be found in Chankong & Haimes (1983). However, as was shown by Das & Dennis (1997), an evenly distributed set of weights may not necessarily produce an evenly distributed representative subset of the Pareto optimal set, even in the cases where the problem is convex. In other words, even by normalizing the objectives with some scaling and then systematically varying the weights, there is no guarantee that the subset of non-dominated solutions found will be a good representation of the Pareto front. Instead, for a good representation of the Pareto front, other multi-objective optimization methods may be more appropriate, such as the Epsilon-Constraint Method.

2.10.2 The ϵ-Constraint Method

In the ϵ-Constraint Method (or Epsilon-Constraint Method (ECM)), we select one of the objective functions as the main function to be optimized, while the others are converted into constraints. The problem therefore gets the following form:

$$\begin{aligned} \text{minimize} \quad & f_i(\mathbf{x}) \\ \text{subject to} \quad & f_j(\mathbf{x}) \leq \epsilon_j \quad \forall \;\; j = 1, ..., k, \;\; j \neq i \\ & \mathbf{x} \in S \end{aligned} \tag{2.10.3}$$

Here $i \in \{1, ..., k\}$, whereas ϵ_j is an upper bound for the objective f_j, for all $j = 1, ..., k$ with $j \neq i$. This method was introduced in Haimes et al. (1971) and discussed extensively in Chankong & Haimes (1983).

It can be proven that the solution of the above problem is always weakly Pareto optimal. Moreover, $\mathbf{x}^* \in S$ can be shown to be Pareto optimal if and only if this decision vector solves (2.10.3) for every $i = 1, ..., k$, where $\epsilon_j = f_j(\mathbf{x}^*)$ for $j = 1, ..., k$, $j \neq i$. Additionally, a unique solution of (2.10.3) can be shown to be Pareto optimal for any choice of upper bounds.

Therefore, to ensure Pareto optimality, one must either solve k different problems (which increases computational cost), or obtain a unique solution (something which is not necessarily trivial to verify). However, an advantage of the ECM over the weighting method, is that finding any Pareto optimal solution does not require convexity. Therefore, the ECM works for both convex and non-convex problems.

Chankong & Haimes (1983) suggests systematic ways of perturbing the upper bounds for obtaining different Pareto optimal solutions. This way, the ECM can be used as an a posteriori method. Alternatively, the ECM can also be used as an a priori method, by asking the decision maker to specify the function to be optimized and the upper bounds, before solving the mathematical program.

2.10.3 A recent algorithm: The Quadrant Shrinking Method

A recent method that has been developed for solving tri-objective integer programs (TOIPs) is the Quadrant Shrinking Method (QSM), introduced by Boland et al. (2017). This is a very efficient exact algorithm for generating the complete non-dominated frontier of a multi-objective integer program with 3 objectives. It is a variant of the full 2 split method for TOIPs. The QSM solves at most $3|y_N| + 1$ single-objective integer programs, where y_N is the set of all non-dominated points. Computational results verify that QSM outperforms existing algorithms for TOIPs.

2.10.4 Performance indicators in Multi-Objective Optimization

Generally speaking, in single-objective optimization it is trivial to quantify the quality of a solution: for minimization problems, the smaller the objective value, the better. However, in multi-objective optimization the task of evaluating the quality of a Pareto front approximation is more complicated. For this, a large number of performance indicators have been developed in the past decades. These indicators have been introduced in order to measure the quality of approximation of the Pareto front by different algorithms.

Audet et al. (2018) review a total of 57 such performance indicators, which are categorized into four groups according to the following properties: cardinality, convergence, distribution and spread. The main purposes of using performance metrics are: (i) to allow the comparison of different algorithms, (ii) to be used as stopping criteria at various stages of multi-objective optimization algorithms, and (iii) to identify promising performance indicators for evaluating and improving the distribution of points for a given solution. For the purpose of comparing algorithms, Audet et al. (2018) consider the hyper-volume indicator to be the most relevant metric, and report that its efficiency compared to other performance metrics is the main reason why it is widely used in the evolutionary community.

Further discussion and review of methods for comparing and assessing Pareto set approximations can be found e.g. in Zitzler et al. (2008).

2.11 Multi-objective Optimization for Vehicle Routing and Scheduling

Multi-objective optimization methods have been applied extensively to model and solve problems in the area of vehicle routing and scheduling.

Rancourt & Paquette (2014) introduce and solve the US MOVRTDSP, a multi-objective vehicle routing and truck driver scheduling problem which satisfies the legislative requirements on work and rest hours in the US. They present a Tabu Search algorithm which solves the problem and provides a heuristic non-dominated solution set, from which trade-offs between operating costs and driver inconvenience are evaluated. Trade-offs between the number of vehicles used and the operating costs are also investigated.

Ombuki et al. (2006) represent the VRPTW as a multi-objective problem with two objectives to be minimized, namely the number of vehicles used and the total cost, and present a genetic algorithm solution using the Pareto ranking technique.

Jozefowiez et al. (2008) provide a survey of multi-objective optimization in routing problems. They examine routing problems in terms of their definitions, their objectives and the multi-objective algorithms proposed for solving them. They report that the two main strategies most widely used for solving multi-objective routing problems are weighted aggregation and multi-objective evolutionary algorithms.

Lust & Teghem (2010) present a survey on the multi-objective traveling salesman problem, along with a new approach for the problem, which they call two-phase Pareto local search.

2.12 Conclusions

Having looked at the types of problems that have already been studied in the area of disruption management in vehicle routing and scheduling, we have decided to focus our research on a set of three new problems related to the literature. These problems were introduced in chapter 1 and are examined thoroughly in the following chapters. To the best of our knowledge, these problems have never been studied before under the same assumptions.

The editor of TOP invited the contribution of a survey paper on disruption management in vehicle routing, Eglese & Zambirinis (2018a), which was based on part of this literature review. As is the custom in TOP, the paper was reviewed by other researchers and their comments have been published in Gendreau (2018), Doerner & Hartl (2018), and Vigo (2018). The reviewers have agreed that our survey paper is a useful summary, with the potential to encourage further research in this area, and also shows how these problems relate to other dynamic or real-time problems in the transport area. We had a chance to publish our response to the reviewers' comments in our rejoinder, Eglese & Zambirinis (2018b).

Chapter 3

Problem 1 - The disrupted VRP with Vehicle Breakdown: Formulation & Exact Approach

3.1 Introduction

In this chapter we present *the disrupted Vehicle Routing Problem with customer-specific orders and Vehicle Breakdown (d-VRP-cso-VB).* In short, we will also be referring to this problem as *the disrupted VRP with Vehicle Breakdown (d-VRP-VB)*, or simply as *Problem 1.* We provide the problem definition and a mixed-integer linear programming (MILP) formulation, which can be used directly in standard commercial optimization software (e.g. Cplex or AIMMS) to find the exact optimal solution. We refer to this approach as *the exact approach for Problem 1.* We then present the experimental results from using this approach to solve 101 different instances which we created specifically for this problem. Most of the instances of this dataset were created by modifying some standard benchmark VRP instances taken from the literature, and specifically from the Augerat et al. (1995) dataset.

The exact approach succeeds in providing optimal or very good quality solutions in many cases within a short time frame. However, due to the fact that the problem is NP-hard, for larger instances this approach may not be suitable for finding a good-quality solution within a few minutes. Therefore, a heuristic may be more suitable in these cases. Such a heuristic is presented in the following chapter, along with comparisons between the two methods.

Of course, even in the cases where the exact approach provides an inferior solution within a short time frame, when compared to the heuristic, the exact approach may still contribute by providing a useful lower bound. This lower bound can be compared to the solution derived by the heuristic to calculate a bound on the optimality gap; thus providing a guarantee on the quality of the heuristic solution.

3.2 Problem overview

Assume that we have a Capacitated Vehicle Routing Problem (CVRP) with a heterogeneous fleet in general (i.e. where vehicles may have different capacities) for which we have constructed a feasible solution plan, which would typically be optimal or near-optimal. We will refer to this heterogeneous-fleet CVRP as the *original problem* or *underlying problem*, and to the already derived solution plan as the *original* or *undisrupted plan*.

Suppose that during the execution stage of this original plan, one of the vehicles breaks down. Assume that the damage is severe enough so that it cannot be restored within a few minutes by the driver alone. This causes a disruption to the original plan, which in effect becomes infeasible. If no further action is taken, the remaining customers of the disabled vehicle's route will never be served. Clearly the plan needs to be revised, so that all the customers - including those that were to be served by the disabled vehicle - are served by the remaining active vehicles.

We assume that in the original plan, $d + 1$ vehicles had departed from a single depot (or, more generally, from a different position each), in order to serve a given set of customers and return to the depot; but during the execution of the plan one of the vehicles unexpectedly broke down. Therefore, in the *new* or *disrupted problem*, we have a set of d active vehicles of different capacities, in general, where each vehicle starts from a different position and has to visit a preassigned set of customers before finishing at the depot. We assume that this assignment of customers to vehicles had taken place during the development of the original plan.

We also assume that the vehicles deliver *customer-unique* or *customer-specific orders* and that there is a one-to-one match between orders and customers. Each order may be unique (e.g. groceries), or may contain unique packages or items, for which no copy or replacement exist. Therefore, goods are non-transferable between customers. This is unlike some other vehicle routing applications where vehicles may carry a single commodity (such as oil or gas).

We further assume that there are no extra vehicles available, that there are no split deliveries, that any parcel of the broken-down vehicle can easily be transferred to any one of the active vehicles if necessary (along with the accompanying documents), and that all customers have to be served. Therefore, each customer that was originally assigned to the broken-down vehicle has to be served by one of the remaining d vehicles. In this case, any vehicle that serves any subset of the disabled vehicle's original customers, must visit the location of the broken-down vehicle first, in order to load the corresponding packages from the disabled vehicle to the active one. Note that, since we assume that the packages being delivered are unique, the scenario of an active vehicle returning to the depot to load a copy of the order of a customer who was supposed to be served by the disabled vehicle, is not an option.

In the new problem we wish to construct a set of d routes, one for each active vehicle, so that each customer is visited exactly once, while the relevant capacity and precedence constraints are satisfied. We assume that in the original problem the objective was to minimize the total travel time (which can trivially be replaced by total distance or total travel cost). In the new problem the main objective is also to minimize the total travel time; but for practical reasons we also add a secondary objective, which is to minimize the sum of arrival times of active vehicles at the depot. We assign an extremely marginal weight to the second objective, so that it mainly serves as a tie-breaker, in cases where multiple solutions with the exact same total travel time exist.

At this point, it is worthwhile to mention that Mu et al. (2011) has studied a similar problem, which was also called 'the disrupted VRP with vehicle breakdown'. However, in their study they were assuming that vehicles were delivering a single commodity such as oil or gas, so that in case of a vehicle breakdown, any active vehicle could serve any customer without the need to visit the disabled vehicle beforehand (provided that it carried the necessary quantity to satisfy the demand, of course). Furthermore, they allowed the use of extra vehicles.

In contrast, in our study we do not allow the use of any extra vehicles and, more importantly, we assume that we have customer-specific and customer-unique orders. These assumptions distinguish significantly the structure, the solution space and the problem itself, compared to the one that was previously studied.

In what follows in this chapter, we will present a MILP formulation for this problem, which is based on the formulations of both the VRPTW and the Pickup-and-Delivery VRPTW.

3.3 Problem definition

Assume that we have a feasible and optimal or near-optimal plan for a multi-commodity Capacitated VRP with a heterogeneous fleet of $d + 1$ vehicles.

Suppose that the vehicles have departed, either from a single depot, or more generally each from a different position, in order to deliver goods to a number of customers. Note that the term *multi-commodity* here means that we assume that the orders are customer-specific; i.e. each customer corresponds to a unique set of items. So, once the orders are loaded to the vehicles and the vehicles start their routes, only a single vehicle can serve any specific customer, the one that carries his or her order. Now, suppose that after the vehicles have departed from their starting points, one of the vehicles breaks down. The original operational plan has to be quickly modified, in such a way that all the customers whose items were loaded on the immobilized vehicle, are now served by one of the remaining active vehicles of the fleet. In order for this to happen, any active vehicle that visits a customer of the immobilized vehicle, must visit the broken-down vehicle beforehand, to transfer the relevant items from the disabled vehicle to the active one. Essentially, revising the plan after such a disruption is a new problem. In what follows, we describe the resulting problem, we establish the notation and propose a mixed-integer linear programming formulation, which can be used in a commercial solver to find the exact optimal solution.

Suppose that, once a disruption occurs, it takes τ time units for the operations team to revise the plan and communicate this to the drivers, and that the operations team can accurately estimate the position of the vehicles τ time units later. Suppose that at time T_0 the new plan is communicated to the drivers. Thus, T_0 denotes the time of start of the new plan, whereas $T_0 - \tau$ denotes the time when rescheduling started. For the sake of simplicity and without loss of generality, from now on we assume that $T_0 = 0$. This means that rescheduling started at time $-\tau$, and at time 0 the new plan starts.

This is the setting of the new problem, once the disruption occurs: Let d be the number of active vehicles and n be the number of customers that are or will still be unserved at time T_0. From now on n will simply be referred to as 'the number of customers'.

Let $K = \{1, 2, \ldots, d\}$ be the set of active vehicles and let $K_0 = K \cup \{d+1\}$ be the set of all vehicles of the fleet, including the broken-down vehicle, which is represented by vehicle $d + 1$. This means that we assume that there are no extra vehicles available at the depot, or anywhere else.

Let $I_C = \{1, 2, \ldots, n\}$ be the set of customers and let $I_D = \{n + 1, n + 2, \ldots, n + d\}$ be the set containing the starting points of the active vehicles, at the time instant T_0. Node 0 represents the depot, which is the end-point of each route.

Let S_k be the subset of customers originally assigned to vehicle k that are still unserved at time T_0, for all $k \in K_0$. These sets are of course assumed to be known. The set $\{S_1, S_2, \ldots, S_d, S_{d+1}\}$ should form a partition of the set of customers I_C; i.e. $I_C = S_1 \cup S_2 \cup \ldots \cup S_d \cup S_{d+1}$ and $S_i \cap S_j = \emptyset \;\; \forall\, i, j \in \{1, ..., d+1\},\; i \neq j$.

In general, there are cases where the optimal solution may involve multiple visits to the broken-down vehicle by the same active vehicle, and/or visits by more than one active vehicles. One of the novelties of the proposed formulation is that it successfully models the feature of multiple visits to the broken-down vehicle, either by the same vehicle or by different vehicles.

Let $S_{d+1} = \{\gamma_1, \gamma_2, ..., \gamma_u\} \subseteq I_C$ be the set containing the customers that were supposed to be served by the broken-down vehicle. The known parameter $u = |S_{d+1}|$ is the number of customers that were originally assigned to the immobilized vehicle and were still unserved at time T_0. Obviously, u is a natural upper bound for the number of times that any specific vehicle needs to visit the immobilized vehicle, as well as an upper bound for the total number of times that the immobilized vehicle needs to be visited by all the active vehicles combined. Assuming that the set S_{d+1} is not empty[1], we have $1 \leq u \leq n$. Note that we do not allow split deliveries.

At this point we should clarify that in general, the serial numbers of vehicles and customers in the revised plan may differ from the ones in the original plan. In fact, in order to simplify the notation used, we assume that when the plan is revised, the vehicles and customer nodes are relabeled. Specifically, we assume that when the plan is revised, the serial numbers of the vehicles are changed so that vehicle with serial number $d+1$ represents the broken-down vehicle. Additionally, the serial numbers of the active customers in the revised plan are changed, so that the u customers that were originally assigned to the broken-down vehicle and are still unserved at time T_0, are now relabeled as the first u active customers $1, 2, ..., u$ of the set I_C, and the remaining $n - u$ active customers are relabeled as customers $u+1, u+2, ..., n$. Therefore, in the revised plan we have $S_{d+1} = \{1, 2, ..., u\} \subseteq I_C = \{1, 2, ..., u, u+1, ..., n\}$.

In order for an active vehicle to serve some customers who were originally assigned to the immobilized vehicle, we assume that the active vehicle must visit the immobilized one first, load the unique orders of those customer, and then depart to serve the customers whose orders it carries. If necessary, it may visit the broken-down vehicle more than once. The capacity constraint must be satisfied at all times. In order to capture the precedence constraint described above, together with the feature of allowing multiple visits to the broken-down vehicle, we create u copies of the node that corresponds to the position of the broken-down vehicle at the moment of breakdown, which

[1]otherwise, in the trivial case when $S_{d+1} = \emptyset$, rescheduling is unnecessary.

are represented by nodes $n + d + 1, n + d + 2, ..., n + d + u$, or in short by $B_1, B_2, ..., B_u$ respectively.

Let $B = \{n + d + 1, n + d + 2, ..., n + d + u\} = \{B_1, B_2, ..., B_u\}$, be the set containing the u copies of the node that represents the position of the broken-down vehicle at the moment of breakdown. We assume that node $B_r = n + d + r$, which represents the r^{th} copy of the broken-down vehicle, 'holds' the goods of customer $\gamma_r = r$, for all $r = 1, 2, ..., u$. Therefore, before visiting a customer node $r \in S_{d+1}$ (for $r = 1, 2, ..., u$), an active vehicle must visit node B_r to load the order of that particular customer.

Let I be the set of all nodes, defined as: $I = \{0\} \cup I_C \cup I_D \cup B = \{0, 1, \ldots, n, n + 1, \ldots, n + d, n + d + 1, ..., n + d + u\}$.

Define the set of arcs A as: $A = A_1 \cup A_2 \cup A_3$, where $A_1 = \{(n + k, i) : k \in K,\ i \in S_k \cup B \cup \{0\}\}$, $A_2 = \{(i, 0) : i \in I \setminus \{0\}\}$, and $A_3 = \bigcup_{k \in K}\{(i, j) : i, j \in B \cup S_{d+1} \cup S_k,\ i \neq j\}$.

The *disrupted VRP with customer-specific orders and Vehicle Breakdown (d-VRP-cso-VB)*, or in short the *disrupted VRP with Vehicle Breakdown (d-VRP-VB)* is formulated on the directed graph $\Gamma = (I, A)$. Note that $A \subseteq I \times I$ and that the set A will, in general, be substantially smaller than the set of arcs $I \times I$ of a complete graph.

Each node $i \in I_C \cup B$ must have exactly 1 vehicle entering and the same vehicle leaving that node. Each node $i \in I_D$ must have no vehicles entering and exactly 1 vehicle leaving the node. The endpoint node 0 (depot) should have exactly d vehicles entering and no vehicles leaving. No vehicle can visit the same node in I more than once.

Each active vehicle $k \in K$ will start from node $n+k$, which is the position of vehicle k at time T_0 when the revised plan starts. It will serve all the customers of the set S_k, it might also serve some customers of the set S_{d+1}, provided that it visits node B_r to pick up the goods of customer r before it visits any customer $r \in S_{d+1}$, and it will finish at the endpoint node 0 (the depot).

Let x_{ijk} be a binary variable representing the number of times that vehicle k traverses arc (i, j), for all $(i, j) \in A,\ k \in K$. Let Q_{ik} be the load of vehicle k immediately after visiting node i, $\forall\ i \in I,\ k \in K$. Let w_{ik} be the time of start of service of node i by vehicle k, where $w_{ik} \geq 0,\ \forall\ i \in I,\ k \in K$. If vehicle k does not visit node i, or if vehicle k starts servicing node i at time 0, then $w_{ik} = 0$.

Let t_{ij} be the travel time from node i to node j, $\forall\ (i, j) \in A$. We assume that $t_{ij} \geq 0,\ \forall\ (i, j) \in A$.

Define $\Delta^+(i) = \{j \in I : (i, j) \in A\}$ to be the set of nodes that are directly reachable from node i, $\forall\ i \in I$. Also define $\Delta^-(i) = \{j \in I : (j, i) \in A\}$ to be the set of nodes from which node i is directly reachable, $\forall\ i \in I$.

Let Q_k be the capacity of vehicle k, for all $k \in K$, where we assume that $Q_k \geq 0 \;\; \forall\, k \in K$. Let M be a large positive constant (say, $M = 10^{10}$) and let ϵ be a very small positive constant (say, $\epsilon = 10^{-7}$).

Let q_i be the demand at node i, $\forall\, i \in I$. We assume that q_i is a known parameter, with $q_i < 0$ for nodes that require a pickup, and $q_i > 0$ for nodes that require a delivery. Therefore, we assume that $q_i > 0$ for all $i \in I_C$, $q_i < 0$ for all $i \in B$, and $q_i = 0$ for all $i \in I_D \cup \{0\}$. We further assume that $q_{n+i} = -q_i$ for all $i \in S_{d+1} = \{1, 2, ..., u\}$, which means that the load which is to be picked up at node $n+i$ matches the load to be delivered to customer $i \in S_{d+1}$.

Let s_i be the service time at node i; i.e. how long it takes to serve node i, $\forall\, i \in I$. For nodes $i \in B$, s_i represents the time needed to load goods from the broken-down vehicle to another vehicle. We assume that s_i are known parameters, with $s_i \geq 0$ for all $i \in I$.

We use time windows to model the disrupted problem, even if time windows did not actually appear in the original problem. For this, let $[a_i, b_i]$ be the time window associated with node i, where $a_i \geq 0$ and $b_i \geq 0$ are assumed to be known parameters, $\forall\, i \in I$. To be more specific, we assume that $[a_i, b_i]$ are the time windows associated with node i which may have been updated accordingly after the disruption happened (which means that for a certain node i, these may be the same as the original time windows before disruption, or they may have been extended or changed after the disruption and just before rescheduling, for instance, after communicating with the customer and informing them of a potential delay due to the unexpected vehicle breakdown). We also assume that $b_i \leq b_0 \;\; \forall\, i \in I$. The time of start of service s_i of each node i in the disrupted problem, must lie within the associated time window $[a_i, b_i]$, $\forall\, i \in I$. This means that the time when the service finishes can lie inside or outside the time window.

Since the notation and formulation presented here involve time windows, they can also be used to solve a more generalized variant of the problem under study, namely *the disrupted VRPTW with Vehicle Breakdown* (with heterogeneous fleet and customer-specific orders). We will refer to this more generalized problem as *the VRPTW extension of problem 1*, which is discussed in section 3.6.

Now, returning to our main problem, where time windows were not actually involved in the original problem, we can simply set $[a_i, b_i] = [0, b_0] \;\; \forall\, i \in I$, and use the same model as a special case of the VRPTW extension, where $b_0 > 0$ here represents the maximum allowed time for a vehicle to return to the depot. For instance, for simplicity we can set b_0 equal to a large enough positive number (e.g. $b_0 = 10^5$).

Therefore, our formulation can be used to solve both the *disrupted VRP* *with vehicle breakdown*, and the *disrupted VRPTW with vehicle breakdown* (with heterogeneous fleet and customer-specific orders).

Note that x_{ijk}'s, w_{ik}'s and Q_{ik}'s are the decision variables, whereas d, n, u, M, ϵ, Q_k's, t_{ij}'s, S_k's, a_i's, b_i's, s_i's and q_i's are parameters, which are assumed to be known.

3.4 A MILP formulation for the disrupted VRP with vehicle breakdown

The *disrupted VRP with Vehicle Breakdown*, as well as its VRPTW extension (in the first two out of three cases, as described later in section 3.6), can both be formulated as a Mixed Integer Linear Program (MILP), composed of equations (3.4.1) - (3.4.25) as follows:

$$\text{minimize} \quad f := \sum_{k\in K} \sum_{(i,j)\in A} t_{ij} x_{ijk} + \epsilon \sum_{k\in K} w_{0k} \tag{3.4.1}$$

subject to:

$$\sum_{k\in K} \sum_{j\in\Delta^+(i)} x_{ijk} = 1 \qquad \forall\, i \in I \setminus \{0\} \tag{3.4.2}$$

$$\sum_{j\in\Delta^+(i)} x_{ijk} - \sum_{j\in\Delta^-(i)} x_{jik} = 0 \qquad \forall\, i \in I_C \cup B, \quad k \in K \tag{3.4.3}$$

$$\sum_{k\in K} \sum_{i\in\Delta^-(j)} x_{ijk} = 0 \qquad \forall\, j \in I_D \tag{3.4.4}$$

$$\sum_{i\in\Delta^-(0)} x_{i0k} = 1 \qquad \forall\, k \in K \tag{3.4.5}$$

$$\sum_{k\in K} \sum_{i\in\Delta^+(0)} x_{0ik} = 0 \tag{3.4.6}$$

$$\sum_{j\in\Delta^+(i)} x_{ijk} - \sum_{j\in\Delta^+(B_i)} x_{B_i,j,k} = 0 \qquad \forall\, i \in S_{d+1}, \quad k \in K \tag{3.4.7}$$

$$\sum_{j\in\Delta^+(i)} x_{ijk} = 1 \qquad \forall\, i \in S_k, \quad k \in K \tag{3.4.8}$$

$$\sum_{j\in\Delta^+(i)} x_{ijk} = 0 \qquad \forall\ k \in K,\ i \in I_C \setminus (S_k \cup S_{d+1}) \tag{3.4.9}$$

$$\sum_{j\in\Delta^+(n+k)} x_{n+k,j,k} = 1 \qquad \forall\ k \in K \tag{3.4.10}$$

$$w_{ik} + s_i + t_{ij} - w_{jk} \le (1 - x_{ijk})M \qquad \forall\ k \in K, (i,j) \in A \tag{3.4.11}$$

$$Q_{ik} - q_j - Q_{jk} \le (1 - x_{ijk})M \qquad \forall\ k \in K, (i,j) \in A \tag{3.4.12}$$

$$w_{B_i,k} + s_{B_i} + t_{B_i,i} - w_{i,k} \le M \cdot \Big(1 - \sum_{j\in\Delta^+(i)} x_{i,j,k}\Big) \qquad \forall\ i \in S_{d+1},\ k \in K \tag{3.4.13}$$

$$max\{0, -q_i\} - Q_{ik} \le M \cdot \Big(1 - \sum_{j\in\Delta^+(i)} x_{i,j,k}\Big) \qquad \forall\ i \in I \setminus \{0\},\ k \in K \tag{3.4.14}$$

$$max\{0, -q_i\} - Q_{ik} \le M \cdot \Big(1 - \sum_{j\in\Delta^-(i)} x_{j,i,k}\Big) \qquad \forall\ i \in \{0\},\ k \in K \tag{3.4.15}$$

$$Q_{ik} - min\{Q_k, Q_k - q_i\} \le M \cdot \Big(1 - \sum_{j\in\Delta^+(i)} x_{i,j,k}\Big) \qquad \forall\ i \in I \setminus \{0\},\ k \in K \tag{3.4.16}$$

$$Q_{ik} - min\{Q_k, Q_k - q_i\} \le M \cdot \Big(1 - \sum_{j\in\Delta^-(i)} x_{j,i,k}\Big) \qquad \forall\ i \in \{0\},\ k \in K \tag{3.4.17}$$

$$Q_{n+k,k} = \sum_{j\in S_k} q_j \qquad \forall\ k \in K \tag{3.4.18}$$

$$a_i \sum_{j\in\Delta^+(i)} x_{ijk} \le w_{ik} \le b_i \sum_{j\in\Delta^+(i)} x_{ijk} \qquad \forall\ i \in I \setminus \{0\},\ k \in K \tag{3.4.19}$$

$$a_i \sum_{j \in \Delta^-(i)} x_{jik} \leq w_{ik} \leq b_i \sum_{j \in \Delta^-(i)} x_{jik} \qquad \forall\, i \in \{0\},\ k \in K \tag{3.4.20}$$

$$w_{n+k,k} = 0 \quad \forall\, k \in K \tag{3.4.21}$$

$$Q_{0k} = 0 \quad \forall\, k \in K \tag{3.4.22}$$

$$x_{ijk} \in \{0,1\} \qquad \forall\, (i,j) \in A,\ k \in K \tag{3.4.23}$$

$$w_{ik} \geq 0 \quad \forall\, i \in I,\ k \in K \tag{3.4.24}$$

$$Q_{ik} \geq 0 \quad \forall\, i \in I,\ k \in K \tag{3.4.25}$$

The objective is to minimize the total travel time combined for the whole fleet; but among those solutions with the same minimum total travel time, we choose the solution which also minimizes the sum of arrival times at the depot. This is reflected by the two components of equation (3.4.1), which transforms this multi-criteria optimization problem with two objectives into a MILP with a single objective, in a lexicographic manner, by the use of ϵ, a very small positive constant.

Equation (3.4.2) ensures that exactly 1 vehicle exits each node $i \in I \backslash \{0\}$. Equation (3.4.3) ensures that any vehicle k enters and exits each node $i \in I_C \cup B$ the same number of times. More specifically, when combined with equation (3.4.2), equation (3.4.3) guarantees that any vehicle k enters and exits each node $i \in I_C \cup B$ the same number of times, which is either 0 or 1 times. Equation (3.4.4) ensures that no vehicles enter any node $j \in I_D$. Equation (3.4.5) ensures that each vehicle enters the endpoint node 0 exactly once. This also implies that exactly d vehicles enter node 0 (i.e. that $\sum_{k \in K} \sum_{i \in \Delta^-(0)} x_{i0k} = d$). Equation (3.4.6) ensures that no vehicles exit node 0. Equation (3.4.7) guarantees that the same vehicle that visits node i for $i \in S_{d+1} = \{1, 2, ..., u\}$, also visits node B_i. Equation (3.4.8) guarantees that vehicle k visits all customers in the set S_k, for all $k \in K$. Equation (3.4.9) ensures that for all $k \in K$, vehicle k does not visit any customer nodes that do not belong to one of the sets S_k or S_{d+1}. Equation (3.4.10) forces vehicle k to be the one that departs from node $n + k$, for all $k \in K$. Consistency of the time and load variables is ensured by constraints (3.4.11) and (3.4.12), respectively. Constraint (3.4.13) makes sure that, if vehicle

$k \in K$ visits node $i \in S_{d+1}$, it must have visited the i-th copy of the broken-down vehicle (i.e. node $B_i = n + d + i$) beforehand. Constraints (3.4.14) - (3.4.17) ensure that, if vehicle $k \in K$ visits node $i \in I$, then the inequality $max\{0, -q_i\} \leq Q_{ik} \leq min\{Q_k, Q_k - q_i\}$ must hold. Constraint (3.4.18) guarantees that the load of vehicle $k \in K$ after departing from node $n + k$, which is the starting point of vehicle k, is equal to the sum of demands of all the customers in the set S_k. Constraints (3.4.19) and (3.4.20) ensure that, if a vehicle $k \in K$ visits node $i \in I$, then the time of start of service w_{ik} at node i should lie within the respective time window $[a_i, b_i]$; otherwise, if vehicle k does not visit node i, then w_{ik} is set to zero. Constraint (3.4.21) forces the time of departure of vehicle $k \in K$ from node $n + k$ to be zero. Constraint (3.4.22) guarantees that vehicles arrive empty at the endpoint node 0; in other words, it ensures that they have delivered everything before they finish their routes. Finally, constraints (3.4.23)-(3.4.25) give the ranges of the decision variables.

The formulation described in this section is based on the formulations of both the regular VRPTW and the Pickup-and-Delivery VRPTW, as described in chapters 5 and 6 respectively of Toth & Vigo (2014). Using this formulation, theoretically we can solve any instance to optimality (i.e. exactly). In practice, due to the fact that the VRP and its variants are NP-hard problems, we can use a solver such as Cplex to implement the formulation and solve small-sized instances only. For larger instances a heuristic is needed. Such a heuristic is described in the following chapter.

3.5 Experimental results of the Exact Approach

The MILP formulation that was presented in section 3.4 was implemented in AIMMS to solve several instances of the VRP with Vehicle Breakdown. We refer to this approach as *the exact approach for Problem 1.* The experimental results and relevant conclusions that can be drawn from these results are presented in this section.

In order to test the exact approach for problem 1, which is presented in this chapter, we have created a dataset of 101 instances for the VRP with Vehicle Breakdown. These are divided into 6 classes. The first class with label 'class T' contains 8 very small instances that we created specifically to test some aspects of the formulation and the problem itself, such as the effect of changing the demands and capacities on the number of rescue vehicles needed and on the number of visits of each rescue vehicle to the immobilized

Table 3.1: Experimental Results for Problem 1 solved using the Exact Approach (MILP, AIMMS)

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_E = n+}$ $\mathbf{+d+u+1}$	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f^E := f_1 + \epsilon f_2}$	$\mathbf{f^L = LB}$ **(AIMMS)**	**% of $\mathbf{f^E}$ from LB**	**O/F/I**	**runtime (sec)**	**RV**	**NVBV**
Class T													
1 - T1	6	2	4	13	110.64	120.64	110.640012064	110.640012064	0.00%	O	0.1	2	4
2 - T2	6	2	4	13	70.1	80.1	70.10000801	70.10000801	0.00%	O	8	1	1
3 - T3	6	2	4	13	75.92	85.92	75.920008592	75.920008592	0.00%	O	1	1	4
4 - T4	6	2	4	13	72.32	82.32	72.320008232	72.320008232	0.00%	O	4	1	2
5 - T5	6	2	4	13	71.89	81.89	71.890008189	71.890008189	0.00%	O	4	1	2
6 - T6	6	2	4	13	107.04	117.04	107.040011704	107.040011704	0.00%	O	0.6	2	2
7 - T7	6	2	4	13	104.82	114.82	104.820011482	104.820011482	0.00%	O	1.6	2	1
8 - T8	6	2	4	13	106.61	116.61	106.610011661	106.610011661	0.00%	O	1.8	2	2
Class A-n32-k5													
9 - A-n32-k5(1,20)	27	4	7	39	569.29	569.29	569.290056929	544.5300443	4.35%	F	300	4	1
10 - A-n32-k5(2,20)	27	4	4	36	568.21	568.21	568.210056821	568.210056821	0.00%	O	199.5	1	1
11 - A-n32-k5(3,20)	27	4	2	34	561.95	561.95	561.950056195	561.950056195	0.00%	O	2.0	4	1
12 - A-n32-k5(4,20)	27	4	10	42	553.78	553.78	553.780055378	482.7600473	12.82%	F	300.31	4	1
13 - A-n32-k5(5,20)	27	4	8	40	558.43	558.43	558.430055843	482.2137705	13.65%	F	300.02	3	1
14 - A-n32-k5(1,30)	24	4	7	36	508.31	508.31	508.310050831	477.1510456	6.13%	F	300.02	4	1
15 - A-n32-k5(2,30)	24	4	3	32	498.79	498.79	498.790049879	498.790049879	0.00%	O	1.38	1	1
16 - A-n32-k5(3,30)	24	4	1	30	492.26	492.26	492.260049226	492.260049226	0.00%	O	0.39	4	1
17 - A-n32-k5(4,30)	24	4	10	39	527.73	527.73	527.730052773	473.2800473	10.32%	F	300.03	4	1
18 - A-n32-k5(5,30)	24	4	7	36	540.14	540.14	540.140054014	467.0921215	13.52%	F	300.03	3	1
Class A-n45-k6													
19 - A-n45-k6(1,20)	39	5	8	53	777.82	777.82	777.820077782	650.1828592	16.41%	F	300.14	2,4	1,1
20 - A-n45-k6(2,20)	39	5	6	51	703.02	703.02	703.020070302	658.0842417	6.39%	F	300.11	2	1
21 - A-n45-k6(3,20)	39	5	9	54	740.53	740.53	740.530074053	652.2401926	11.92%	F	300.06	1	1
22 - A-n45-k6(4,20)	39	5	8	53	743.38	743.38	743.380074338	684.9760773	7.86%	F	300.05	3	1
23 - A-n45-k6(5,20)	39	5	6	51	715.76	715.76	715.760071576	651.4502106	8.98%	F	300.05	1	1

(Continued on the next page)

Table 3.1 – *Continued from the previous page*

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_E = n + d + u + 1}$	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f^E := f_1 + \epsilon f_2}$	$\mathbf{f^L = LB}$ **(AIMMS)**	**% of $\mathbf{f^E}$ from LB**	**O/F/I**	**runtime (sec)**	**RV**	**NVBV**
24 - A-n45-k6(6,20)	39	5	7	52	711.09	711.09	711.090071109	711.090071109	0.00%	O	93.03	3	1
25 - A-n45-k6(1,40)	35	5	8	49	744.42	744.42	744.420074442	576.5200815	22.55%	F	300.14	1,3	1,2
26 - A-n45-k6(2,40)	35	5	5	46	627.89	627.89	627.890062789	627.890062789	0.00%	O	54.30	2	1
27 - A-n45-k6(3,40)	35	5	8	49	645.21	645.21	645.210064521	607.6412513	5.82%	F	300.08	2	1
28 - A-n45-k6(4,40)	35	5	8	49	660.80	660.80	660.800066080	612.9704023	7.24%	F	300.05	3	1
29 - A-n45-k6(5,40)	35	5	5	46	616.74	616.74	616.740061674	616.740061674	0.00%	O	132.88	1	2
30 - A-n45-k6(6,40)	35	5	6	47	654.25	654.25	654.250065425	654.250065425	0.00%	O	25.38	3	1
31 - A-n45-k6(1,60)	26	5	6	38	502.77	502.77	502.770050277	502.770050277	0.00%	O	216.38	2	1
32 - A-n45-k6(2,60)	26	5	4	36	470.47	470.47	470.470047047	470.470047047	0.00%	O	0.80	2	1
33 - A-n45-k6(3,60)	26	5	7	39	524.77	524.77	524.770052477	463.0900373	11.75%	F	300.03	3	1
34 - A-n45-k6(4,60)	26	5	7	39	509.51	509.51	509.510050951	460.6100429	9.60%	F	300.03	3	1
35 - A-n45-k6(5,60)	26	5	3	35	455.29	455.29	455.290045529	455.290045529	0.00%	O	0.42	1	1
36 - A-n45-k6(6,60)	26	5	4	36	524.28	524.28	524.280052428	524.280052428	0.00%	O	8.09	3	1
37 - A-n45-k6(1,80)	21	5	6	33	462.86	462.86	462.860046286	413.3428062	10.70%	F	300.01	4	1
38 - A-n45-k6(2,80)	21	5	3	30	403.29	403.29	403.290040329	403.290040329	0.00%	O	0.31	2	1
39 - A-n45-k6(3,80)	21	5	7	34	437.38	437.38	437.380043738	374.6400306	14.34%	F	300.02	3	1
40 - A-n45-k6(4,80)	21	5	5	32	440.1	440.1	440.10004401	440.10004401	0.00%	O	16.59	3	1
41 - A-n45-k6(5,80)	21	5	1	28	411.33	411.33	411.330041133	411.330041133	0.00%	O	0.00	1	1
42 - A-n45-k6(6,80)	21	5	4	31	442.93	442.93	442.930044293	442.930044293	0.00%	O	12.47	4	1
Class A-n61-k9													
43 - A-n61-k9(1,10)	51	8	4	64	815.13	815.13	815.130081513	815.130081513	0.00%	O	258	8	1
44 - A-n61-k9(2,10)	51	8	8	68	839.94	839.94	839.940083994	792.1000792	5.70%	F	300.14	8	1
45 - A-n61-k9(3,10)	51	8	7	67	845.51	845.51	845.510084551	754.6373367	10.75%	F	300.13	3,8	1,1
46 - A-n61-k9(4,10)	51	8	6	66	842.56	842.56	842.560084256	730.2250487	13.33%	F	300.13	3,4	2,1
47 - A-n61-k9(5,10)	51	8	10	70	1064.15	1064.15	1064.150106415	725.9932583	31.78%	F	300.27	4,5,7	2,1,1
48 - A-n61-k9(6,10)	51	8	7	67	814.11	814.11	814.110081411	786.0073779	3.45%	F	300.13	7	2
49 - A-n61-k9(7,10)	51	8	3	63	795.53	795.53	795.530079553	795.530079553	0.00%	O	34.06	8	1
50 - A-n61-k9(8,10)	51	8	8	68	852.76	852.76	852.760085276	789.8864281	7.37%	F	300.13	5	1

(Continued on the next page)

Table 3.1 – *Continued from the previous page*

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_E = n+}$ $\mathbf{+d+u+1}$	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f^E := f_1 + \epsilon f_2}$	$\mathbf{f^L = LB}$ **(AIMMS)**	**% of $\mathbf{f^E}$ from LB**	**O/F/I**	**runtime (sec)**	**RV**	**NVBV**
51 - A-n61-k9(9,10)	51	8	6	66	791.81	791.81	791.810079181	770.7036443	2.67%	F	300.11	7	2
52 - A-n61-k9(1,20)	46	8	2	57	727.76	727.76	727.760072776	727.760072776	0.00%	O	1.13	1	1
53 - A-n61-k9(2,20)	46	8	8	63	745.20	745.20	745.200074520	705.2900703	5.36%	F	300.19	8	1
54 - A-n61-k9(3,20)	46	8	7	62	709.59	709.59	709.590070959	678.7839041	4.34%	F	300.11	8	2
55 - A-n61-k9(4,20)	46	8	5	60	749.11	749.11	749.110074911	695.2007494	7.20%	F	300.09	3	1
56 - A-n61-k9(5,20)	46	8	9	64	824.99	824.99	824.990082499	675.646932	18.10%	F	300.14	1,5	1,1
57 - A-n61-k9(6,20)	46	8	7	62	756.27	756.27	756.270075627	708.3504756	6.34%	F	300.11	5	1
58 - A-n61-k9(7,20)	46	8	3	58	702.69	702.69	702.690070269	702.690070269	0.00%	O	7.91	8	1
59 - A-n61-k9(8,20)	46	8	7	62	765.13	765.13	765.130076513	723.0012527	5.51%	F	300.11	6	1
60 - A-n61-k9(9,20)	46	8	6	61	706.10	706.10	706.100070610	686.8448633	2.73%	F	300.11	7	2
61 - A-n61-k9(1,35)	39	8	1	49	662.47	662.47	662.470066247	662.470066247	0.00%	O	0.84	1	1
62 - A-n61-k9(2,35)	39	8	6	54	672.20	672.20	672.200067220	644.0030121	4.19%	F	300.09	8	1
63 - A-n61-k9(3,35)	39	8	7	55	691.31	691.31	691.310069131	599.86812	13.23%	F	300.09	3	2
64 - A-n61-k9(4,35)	39	8	5	53	647.84	647.84	647.840064784	620.6300609	4.20%	F	300.24	3	1
65 - A-n61-k9(5,35)	39	8	9	57	718.38	718.38	718.380071838	606.076703	15.63%	F	300.30	7	1
66 - A-n61-k9(6,35)	39	8	5	53	660.92	660.92	660.920066092	660.920066092	0.00%	O	81.84	7	1
67 - A-n61-k9(7,35)	39	8	3	51	628.39	628.39	628.390062839	628.390062839	0.00%	O	3.83	8	1
68 - A-n61-k9(8,35)	39	8	6	54	677.38	677.38	677.380067738	659.3900771	2.66%	F	300.08	6	1
69 - A-n61-k9(9,35)	39	8	5	53	638.66	638.66	638.660063866	638.660063866	0.00%	O	68.78	7	1
Class A-n65-k9													
70 - A-n65-k9(1,20)	54	8	6	69	832.24	832.24	832.240083224	823.0649518	1.10%	F	300.09	7	1
71 - A-n65-k9(2,20)	54	8	9	72	880.61	880.61	880.610088061	799.176345	9.25%	F	300.13	3,6	1,1
72 - A-n65-k9(3,20)	54	8	8	71	833.65	833.65	833.650083365	795.8900747	4.53%	F	300.13	7	2
73 - A-n65-k9(4,20)	54	8	6	69	857.84	857.84	857.840085784	815.3364065	4.95%	F	300.13	2	1
74 - A-n65-k9(5,20)	54	8	7	70	847.62	847.62	847.620084762	828.9910426	2.20%	F	300.13	8	1
75 - A-n65-k9(6,20)	54	8	8	71	830.42	830.42	830.420083042	824.8500818	0.67%	F	300.30	3	1
76 - A-n65-k9(7,20)	54	8	6	69	939.69	939.69	939.690093969	803.1003917	14.54%	F	300.23	2	1
77 - A-n65-k9(8,20)	54	8	7	70	898.18	898.18	898.180089818	765.39721	14.78%	F	300.25	3	2

(Continued on the next page)

Table 3.1 – *Continued from the previous page*

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_E = n+ +d+u+1}$	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f^E := f_1 + \epsilon f_2}$	$\mathbf{f^L = LB}$ **(AIMMS)**	**% of** $\mathbf{f^E}$ **from LB**	**O/F/I**	**runtime (sec)**	**RV**	**NVBV**
78 - A-n65-k9(9,20)	54	8	5	68	856.99	856.99	856.990085699	856.990085699	0.00%	O	287.11	4	1
79 - A-n65-k9(1,40)	42	8	4	55	689.75	689.75	689.750068975	689.750068975	0.00%	O	17.34	7	1
80 - A-n65-k9(2,40)	42	8	8	59	696.17	696.17	696.170069617	665.9100666	4.35%	F	300.36	6	1
81 - A-n65-k9(3,40)	42	8	7	58	678.85	678.85	678.850067885	640.896293	5.59%	F	300.08	7	1
82 - A-n65-k9(4,40)	42	8	5	56	682.62	682.62	682.620068262	682.620068262	0.00%	O	111.16	2	1
83 - A-n65-k9(5,40)	42	8	4	55	691.41	691.41	691.410069141	691.410069141	0.00%	O	17.13	8	1
84 - A-n65-k9(6,40)	42	8	6	57	680.98	680.98	680.980068098	680.980068098	0.00%	O	23.75	3	1
85 - A-n65-k9(7,40)	42	8	6	57	718.54	718.54	718.540071854	651.2143404	9.37%	F	300.25	7	1
86 - A-n65-k9(8,40)	42	8	6	57	717.46	717.46	717.460071746	646.0845104	9.95%	F	300.13	3	1
87 - A-n65-k9(9,40)	42	8	4	55	693.70	693.70	693.700069370	693.700069370	0.00%	O	30.25	4	1
Class E-n76-k7													
88 - E-n76-k7(1,10)	64	6	1	72	591.81	591.81	591.810059181	591.810059181	0.00%	O	1.88	6	1
89 - E-n76-k7(2,10)	64	6	13	84	NA	NA	NA	560.8306415	NA	I	300.20	NA	NA
90 - E-n76-k7(3,10)	64	6	9	80	646.10	646.10	646.100064610	567.2700505	12.20%	F	300.16	1,3	1,1
91 - E-n76-k7(4,10)	64	6	13	84	811.03	811.03	811.030081103	569.1675307	29.82%	F	300.16	1,3,5,6	1,1,1,1
92 - E-n76-k7(5,10)	64	6	10	81	633.48	633.48	633.480063348	564.2100564	10.93%	F	300.14	2,3	1,1
93 - E-n76-k7(6,10)	64	6	11	82	691.11	691.11	691.110069111	571.220338	17.35%	F	300.14	2,3,4	1,1,1
94 - E-n76-k7(7,10)	64	6	13	84	NA	NA	NA	555.1100425	NA	I	300.23	NA	NA
95 - E-n76-k7(1,15)	58	6	1	66	544.00	544.00	544.000054400	544.000054400	0.00%	O	0.73	6	1
96 - E-n76-k7(2,15)	58	6	12	77	629.69	629.69	629.690062969	531.1205454	15.65%	F	300.14	4,5	1,2
97 - E-n76-k7(3,15)	58	6	8	73	555.50	555.50	555.500055550	533.3533242	3.99%	F	300.13	6	1
98 - E-n76-k7(4,15)	58	6	12	77	589.66	589.66	589.660058966	527.8897469	10.48%	F	300.14	5,6	1,1
99 - E-n76-k7(5,15)	58	6	9	74	557.17	557.17	557.170055717	532.1500491	4.49%	F	300.17	3	1
100 - E-n76-k7(6,15)	58	6	11	76	552.69	552.69	552.690055269	529.2900533	4.23%	F	300.13	4	1
101 - E-n76-k7(7,15)	58	6	11	76	562.00	562.00	562.000056200	531.9104956	5.35%	F	300.11	3	1

Table 3.2: Additional Experimental Results for Problem 1 - Exact Approach (AIMMS)

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_E = n+ +d+u+1}$	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f^E := f_1 + \epsilon f_2}$	$\mathbf{f^L = LB}$ **(AIMMS)**	**% of** $\mathbf{f^E}$ **from LB**	**O/F/I**	**runtime (sec)**	**RV**	**NVBV**
29b - A-n45-k6(5,40)	35	5	5	46	616.74	616.74	616.740061674	616.740061674	0.00%	O	81.27	1	2 (ϵ=0.001, translated)
51b - A-n61-k9(9,10)	51	8	6	66	791.81	791.81	791.810079181	778.8200779	1.64%	F	900 (15 min)	7	2
51c - A-n61-k9(9,10)	51	8	6	66	791.81	791.81	791.810079181	791.810079181	0.00%	O	3809.61 (63.5 min)	7	2
89b - E-n76-k7(2,10)	64	6	13	84	609.53	609.53	609.530060953	560.8306415	7.99%	F	1800.22 (30 min)	1	1
94b - E-n76-k7(7,10)	64	6	13	84	NA	NA	NA	555.1100425	NA	I	1800.16	NA	NA

one.

The other five classes have the following code-names: A-n32-k5, A-n45-k6, A-n61-k9, A-n65-k9, and E-n76-k7. They contain 10, 24, 27, 18 and 14 instances respectively. Each one of those five classes contains different instances that were created based on the five Augerat's benchmark VRP instances with the corresponding names. In each 'disrupted' instance of our dataset we assume that the best solution reported in the literature for the original Augerat's VRP instance was followed, and that a vehicle breaks down at some time after the vehicles have departed from the depot. Starting from one standard VRP instance, by choosing different vehicles to be immobilized at different times, we have created several different instances for the disrupted problem. The two numbers in the brackets that follow the instance code-name, indicate the serial number of the vehicle that broke down and the time when the disruption occurred, respectively. Note that the serial number of the broken-down vehicle refers to the label used in the original VRP, and is in general different than its serial number in the disrupted one, since vehicles are relabeled in the disrupted problem. Also, the time when the disruption occurs refers to the time as measured in the original VRP. In the new problem after disruption, when the rescheduled plan starts, time is reset to zero.

Let us explain, for example, how the 17-th instance with code-name A-n32-k5(4,30) was created: We assume that the underlying VRP problem was Augerat's VRP instance A-n32-k5, with 31 customers and 5 vehicles, and that the best solution reported in the literature was followed, with all the vehicles leaving the depot at zero time. We assume that the travel time is equal to the Euclidean distance (which is a very common assumption in such problems), and that the service time and waiting times are all zero in the underlying problem. Then, at time 30, vehicle 4 breaks down, at the exact location that this vehicle should be by that time if following the original plan and assuming that it moves with a constant speed in a straight line between two points on the xy-plane; therefore it is trivial to calculate its xy-coordinates mathematically, using the formula that is explained below:

Given two points $A(x_1, y_1)$ and $B(x_2, y_2)$ on the xy-plane, we can calculate the coordinates of the point P that lies on the (interior of the) straight segment AB and divides it internally in the ratio $m : n$ (i.e. so that $AP : PB = m : n$), as: $P(\frac{nx_1+mx_2}{m+n}, \frac{ny_1+my_2}{m+n})$.

Continuing the explanation of how instance A-n32-k5(4,30) was derived, we use the above formula to calculate the coordinates of the broken-down vehicle. We assume that at the moment of the breakdown all other vehicles are also at the exact location that they should be by that time, if following the original plan and assuming that they move with a constant speed in a straight line between two points on the xy-plane. Therefore, their coordinates can also calculated with the formula described above. Then, by removing all served customers, relabeling the vehicles and the remaining customers, we get the new instance, with fewer customers, with 4 active vehicles and 1 disabled vehicle. This concludes the discussion about the construction of instance A-n32-k5(4,30).

In general, the following parameters are given as an input separately for each instance: n, d, u, Q_k (for $k = 1, ..., d$), sets $S_1, S_2, ..., S_{d+1}$, as well as a_i, b_i, s_i, and q_i (for $i = 0, 1, ..., n + d + u$). The set $S_{d+1} = \{1, 2, ..., u\}$ is assumed to contain exactly the first u customers of the set I_C. The Cartesian coordinates (X_i, Y_i) of each node $i \in I$ are also given as an input, which are used to calculate the travel time $t_{i,j}$ between each pair of nodes i and j as the Euclidean distance between the nodes, i.e. as $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$, rounded to 2 decimal places.

Recall that the objective is to minimize the function f, which is defined by equation (3.4.1) as: $f := \sum\limits_{k \in K} \sum\limits_{(i,j) \in A} t_{ij} x_{ijk} + \epsilon \sum\limits_{k \in K} w_{0k}$. We can also write this as $f := f_1 + \epsilon \cdot f_2$, where $f_1 := \sum\limits_{k \in K} \sum\limits_{(i,j) \in A} t_{ij} x_{ijk}$ and $f_2 := \sum\limits_{k \in K} w_{0k}$ are the two component objectives. Note that, in the tables with the experimental results for problem 1 following the exact approach (tables 3.1 and 3.2), instead of f, the objective is indicated by f^E to denote that this is the value of the objective found by the exact approach (i.e. in f^E, the letter E stands for the exact approach). In the following chapter, in the respective tables with the heuristic results, we will use f^H to denote the value of the objective found by the heuristic approach (i.e. in f^H, the letter H stands for the heuristic approach).

Table 3.1 summarizes the experimental results of the exact approach for Problem 1. The first 10 columns of this table show the following information for each instance: the serial number and code-name of the instance, the number of customer nodes n, the number of active vehicles d, the number of unserved customers u that were originally assigned to broken-down vehicle, the total number of nodes ($N_E := n + d + u + 1$), the values of the two component objectives f_1 and f_2, the value of the actual objective f^E (defined as $f^E := f_1 + \epsilon \cdot f_2$), the lower bound that was found by AIMMS (referred to as f^L or as LB), and the optimality gap between the best solution found and the lower bound provided by AIMMS (calculated as: *optimality gap* $:= \frac{f^E - f^L}{f^E} \cdot 100\%$). The 11-th column with label O/F/I indicates whether the solution was optimal (O), feasible (F) or infeasible (I). The last 3 columns show, respectively, the runtime in seconds, the serial number of the rescue vehicle(s) (RV) involved in the best solution found, and the number of visits to the broken-down vehicle (NVBV) by each one of the rescue vehicles.

Regarding the experiments of the exact approach for Problem 1, the MILP formulation was implemented in AIMMS and a cutoff time of 5 minutes was applied. In the instances considered, the number of active customers (n) varies from 6 to 64, the number of active vehicles (d) varies from 2 to 8, there is just 1 disabled vehicle, the number of nodes originally assigned to the broken-down vehicle (u) varies from 1 to 13, and the total number of nodes[2] ($N_E := n + d + u + 1$) varies from 13 to 84.

From table 3.1 we can see that within the allowed time of 5 minutes (300 seconds), AIMMS was able to find the optimal solution and prove its optimality in 40 out of the 101 instances considered (i.e. in 39.6% of the cases). In 59 other instances, AIMMS provided a feasible solution within the time limit of 5 minutes, whereas in the remaining 2 instances no feasible solution was reached within 5 minutes. AIMMS provided a lower bound in each one of the 101 instances.

Among the 40 instances for which the optimal solution was found in less than 300 seconds, in 22 cases the optimal was found within 10 seconds, in 12 other cases the optimal needed more than 10 but less than 100 seconds, and in the remaining 6 instances the optimal needed between 100 and 300 seconds.

Among the 59 instances for which a feasible but not necessarily optimal solution was reached within 300 seconds, the optimality gap varied between 0.5% and 32%. Specifically, in 35 out of those 59 instances the optimality gap was less than 10%, in 21 other cases the optimality gap was between 10% and 20%, whereas in the remaining 3 instances the optimality gap was between 20% and 32%.

In 72 out of the 101 instances, the best solution which was found by AIMMS within the available time of 5 minutes, involved only 1 visit to the broken-down vehicle, by a single active vehicle. In 12 other cases, the best solution reported involved 2 visits to the broken-down vehicle, by a single active vehicle. In 2 other instances, the reported solution involved 4 visits to the broken-down vehicle, by a single active vehicle. Note that both of these instances, namely instances T1 and T3, were intentionally created so that the optimal solution involves 4 visits to the disabled vehicle, for testing our formulation. In 7 other instances, the best solution reported involved 2 different active vehicles, each visiting the disabled vehicle once. In 3 instances, 2 different active vehicles visit the disabled vehicle once and twice, respectively. In 1 instance, 3 different active vehicles each visit the disabled vehicle once.

[2]Note that the number of nodes in the exact approach for problem 1 is denoted by N_E, to be distinguished from the number of nodes in the heuristic approach for problem 1, which is denoted by N_H. Generally, N_H differs from N_E, because of the different modeling that was employed in the heuristic approach.

In 1 instance, 3 different active vehicles visit the disabled vehicle once, once and twice, respectively. In 1 instance, 4 different active vehicles each visit the disabled vehicle once. In the remaining 2 instances, no feasible solution was reported.

Of course, in some of the instances where the best solution found by AIMMS involves multiple visits to the broken-down vehicle, either by a single or by more than one vehicle, it may be the case that a better feasible solution exists with fewer visits to the broken-down vehicle and/or less active vehicles being involved, but AIMMS couldn't find it within the available time.

Finally, from table 3.1 we observe that the time needed to reach the optimal solution highly depends on the value of u, which is the number of customers originally assigned to the disabled vehicle.

For instances 1-28 and 43-69, we used the following parameter values in our experiments: $M = 200000$, $\epsilon = 10^{-7}$. For the remaining instances, we used $M = 99999$ and $\epsilon = 10^{-7}$.

We also present table 3.2 with some additional results of the exact approach for problem 1. In this table, instance 29b represents a second run of instance 29, where a different value for the ϵ parameter was used, namely $\epsilon = 0.001$, instead of $\epsilon = 10^{-7}$ that was used in other instances. Using the value of $\epsilon = 0.001$, AIMMS was able to find the optimal solution within 81 seconds, compared to the runtime of 133 seconds needed when using the default value of $\epsilon = 10^{-7}$. In the last column of instance 29b, we write "$\epsilon = 0.001$, translated" to denote that, although a different value was used for the ϵ parameter, the value of the objective f^E provided in the 8-th column was calculated using the default value of $\epsilon = 10^{-7}$ that was used in all other examples and thus 'translated', for comparison reasons (and so was the value in the 9-th column, which is equal to f^E in this case). This result may suggest that, although the value of ϵ should be small enough so that the optimal solution to f always uses the optimal solution to f_1, making ϵ too small means that the solver needs more time to confirm the optimal solution. However, further experimentation is necessary to verify or disprove the above statement. This is left open for further research.

In instance 51 we tried two more runs, marked as instances 51b and 51c, with a cutoff time of 15 minutes for the first one, and without a cutoff time for the second one. In these instances, AIMMS found the same solution as the one reported with a 5-minute cutoff time; however the optimality gaps differ, because the longer the available time AIMMS had, the better the lower bound was. Therefore, although in all 3 runs (51, 51b, 51c) the same solution was found, a different optimality gap was provided and only in the latter case AIMMS was able to prove the optimality of the solution found, after 63.5 minutes.

In instances 89 and 94 no feasible solution was found within 5 minutes; therefore we performed one additional run for each instance with 30-minute cutoff time (results shown as instances 89b and 94b, respectively). In instance 89b, AIMMS was able to find a feasible solution within the 30 minutes cutoff time, whereas in instance 94b no feasible solution was found.

At this point we should clarify that, in instances 89 and 94 of table 3.1, as well as in instance 94b of table 3.2 where the entry of the 11-th column is "I" (infeasible), no feasible solution was found within the available time, which does not necessarily mean that the problem is infeasible.

3.5.1 Discussion about the instances of class T

We will now examine class T, which contains 8 instances (T1,...,T8) of very small test problems that we created in order to test our formulation. Instances T2, T3, T4 and T5 of Class T, are variations of the same instance with different capacities assigned to vehicles. These very small instances were designed specifically to show the possibility of requiring an active vehicle visiting the broken-down vehicle multiple times. To understand the differences between these instances we must go into the details of the specific input parameters of each instance.

Specifically, in instance T2 we have two active vehicles (vehicles 1 and 2) of capacities $Q_1 = Q_2 = 40$ and 6 customers (customers 1,2,...,6), each of demand $q_i = 10$ (for $i = 1, 2, ..., 6$). Customers $1, 2, 3$ and 4 were supposed to be served by the broken-down vehicle. The optimal solution in the disrupted problem requires 1 visit to the disabled vehicle, by vehicle 1. In instance T3, we have the same setting as in instance T2, but with $Q_1 = Q_2 = 10$; in which case the optimal solution involves 4 visits to the disabled vehicle, by vehicle 1. In instance T4, we have the same setting as in instance T2, but with $Q_1 = Q_2 = 20$; in which case the optimal solution involves 2 visits to the disabled vehicle, by vehicle 1. In instance T5, we have the same setting as in instance T2, but with $Q_1 = 32$ and $Q_2 = 26$; in which case the optimal solution involves 2 visits to the disabled vehicle, by vehicle 1.

In instance T1, we have the same setting as in instance T2, but with $Q_1 = 14$, $Q_2 = 23$, $q_i = 20$ for $i = 1, 2, 3, 4$, and $q_i = 10$ for $i = 5, 6$; in which case the optimal solution involves 4 visits to the disabled vehicle, by vehicle 2. In instance T6, we have the same setting as in instance T1, but with $Q_2 = 42$ (and $b_i = 200$, instead of 5000, for all $i \in I$; although this specific change of the b_i's doesn't have any effect here); in which case the optimal solution involves 2 visits to the disabled vehicle, by vehicle 2. In instance T7, we have the same setting as in instance T6, but with $Q_2 = 83$; in which case the optimal solution involves only 1 visit to the disabled vehicle, by vehicle

2. In instance T8, we have the same setting as in instance T6, but with $Q_2 = 62$; in which case the optimal solution involves 2 visits to the disabled vehicle, by vehicle 2.

3.6 The disrupted VRPTW with Vehicle Breakdown

As mentioned earlier in section 3.3, since the notation and formulation presented in this chapter involve time windows, they may also be used for solving a more generalized variant of the problem under study, namely *the disrupted VRPTW with Vehicle Breakdown (d-VRPTW-VB)*, under the assumptions of a heterogeneous fleet and customer-specific orders. In short, we also refer to this problem as *the VRPTW extension of problem 1.*

This problem refers to the direct generalization of problem 1, where additionally each customer $i \in I_C$ is associated with a time window $[a_i, b_i]$ and the time of start of service of this customer must lie within this time window, where now the a_i's and b_i's do not necessarily have to equal 0 and b_0, respectively (which was the case in problem 1). Note that the limits a_i and b_i of the time windows $[a_i, b_i]$ of any customer $i \in I_C$ refer to the problem after the disruption has occurred, are measured in a different time scale and may be completely different, compared to the time windows $[a_i^0, b_i^0]$ of that customer in the original problem.

To be more specific, in the VRPTW extension of problem 1 we further assume that each customer i of the set $I_C = \{1, 2, ..., n\}$, who was still unserved at time instant T_0, was associated with a time window $[a_i^0, b_i^0]$ in the original plan. Define T^* as the time when the original plan was supposed to start (whether this was set originally to 0 or to the actual time), plus the time between the start of the original plan and the time when the new plan starts, after rescheduling. The time windows $[a_i^0, b_i^0]$ of the underlying problem can trivially be converted into the time windows for the disrupted problem $[a_i, b_i]$ by letting $a_i := a_i^0 - T^*$ and $b_i := b_i^0 - T^*$ (i.e. by shifting the time windows by T^* time units) for all $i \in I_C$.

There are three possible scenarios that may occur when solving an instance of the disrupted VRPTW with Vehicle Breakdown. In the first case, where the time windows are wide enough so that a feasible solution that respects all time windows exists, then the approach described in this chapter is perfectly capable of modeling and solving the VRPTW extension of the problem to optimality.

The second case occurs if after the disruption the problem becomes infeasible because of the time windows, but it is possible to extend the time windows $[a_i, b_i]$ of some customers, in such a way that a feasible solution exists. If we assume that the original plan was feasible and that there was no time lost once the disruption occurred for any active vehicles, meaning that they continued following their routes as scheduled, then after the disruption it should be enough to extend the time windows of some, or at worst all of the customers of the disabled vehicle, in order for the disrupted problem to become feasible. For instance, the company's department of customer support may contact the affected customers, explain the situation and, upon agreement of the customers, their time windows may be updated so that a feasible solution now exists. Therefore, in this second scenario it should be enough to update or widen the time windows $[a_i, b_i]$ for each customer $i = 1, ..., u$ of the broken-down vehicle accordingly (for instance, one can simply set $a_i = 0$ and $b_i = M$ for $i = 1, ..., u$, where M is a large positive constant). If indeed the time windows of some customers are changed in such a way that a feasible solution now exists, then again, we can use the approach described in this chapter to model and solve to optimality the VRPTW extension of the problem.

Finally, the third case occurs if after the disruption the problem becomes infeasible because of the time windows, but it is not possible to extend all necessary time windows of customers, in such a way that a feasible solution exists. Therefore, in this scenario, even if we are allowed to change some of the time windows, the problem is still infeasible with respect to the time windows. In other words, it is now impossible to serve all of the remaining customers within their requested time windows. In this case we cannot use the approach described in this chapter; instead, this case should be further investigated by researchers. More specifically, proper disruption management techniques should be employed, in order to construct a plan that minimizes the negative impact of the disruption, while also minimizing the original objective of minimum total travel time (or minimum distance or travel cost, accordingly). For instance, the new plan may seek to minimize the deviation from the time windows, minimize the number of customers that are served outside their time windows and minimize the total travel time. Other options may include hiring other vehicles, canceling some customers etc. There may be several different ways to define, model and solve the VRPTW extension in this third case, depending on the assumptions, constraints and objectives that one decides to employ. One possibility may be to modify and combine the models used for problem 1 with those of problem 3 *(the Single-Commodity Delayed VRPTW)*, which is described in chapters 7 and 8. We leave the study of this third case of the VRPTW extension to future researchers.

3.7 Conclusions

To summarize, in this chapter we presented *the disrupted Vehicle Routing Problem with Vehicle Breakdown*, under the assumptions of customer-specific and customer-unique orders (multi-commodity VRP), heterogeneous fleet, single-depot, no extra vehicles available, no split deliveries, and that no customer remains unserved. We presented a precise definition of the problem, established notation and presented a mixed-integer linear programming formulation that can be directly embedded in a commercial solver, such as Cplex or AIMMS, to solve instances to optimality in an exact way that provides proven optimality.

We created a dataset of 101 instances for this problem and used the above exact approach to solve these in AIMMS. For small instances and some medium-sized instances with only a small number of customers originally assigned to the disabled vehicle, this approach succeeds in finding the proven optimal solution within reasonable time. In some other cases, even if it fails to provide the optimal solution or to prove its optimality, this approach may provide a feasible solution which sometimes is quite close to the optimal. However, for larger instances a faster approach is needed; one that can provide a good quality solution in real time, even if this is not always the exact optimal solution. In the following chapter we present such a heuristic approach. Of course, even in cases where the exact approach fails to provide a good quality solution, the lower bound that it provides can always be used to establish a guarantee on the optimality gap of any solution that is found by a different method.

Chapter 4

Problem 1 - The disrupted VRP with Vehicle Breakdown: Heuristic Approach

4.1 Introduction

In this chapter we present a heuristic algorithm based on the Tabu Search metaheuristic, as a second approach for solving the *disrupted VRP with Vehicle Breakdown (problem 1)*, which was described in the previous chapter. The heuristic is used to solve the 101 instances that were constructed for the problem. The experimental results are presented at the end of the chapter, along with comparisons with the results from the exact approach which were presented in the previous chapter.

4.2 Heuristic overview

We employ Tabu Search with the standard aspiration criteria, multiple restarts, short-term and long-term memory structures, best improvement feature and 4 types of neighborhood moves: intra-route exchange, intra-route insertion, double insertion, and double exchange. The last two types of moves mentioned were created specifically for the problem under study.

At each restart, we begin from a different initial solution and apply several local search iterations to improve the current solution. At each iteration, all possible combinations of the 4 neighborhood moves are tried, and the one that gives the best improvement in the objective function of the current solution is selected (provided that, either the move strictly improves the best solution found so far, or the move does not involve nodes that are in the tabu list or nodes that have been involved too frequently in the moves performed so far). Each restart terminates when there is no improvement in the last J_1 iterations.

After some experimentation, we decided not to consider moves that have a zero change in the objective function, to avoid getting stuck in a plateau.

4.3 Notation

In general, throughout this chapter we use the notation described in chapter 3, unless it is explicitly stated otherwise. Therefore, as in the previous chapter, $K = \{1, 2, ..., d\}$ is the set containing the d active vehicles, whereas $d+1$ is the serial number of the broken-down vehicle (BV). Also, $I_C = \{1, 2, ..., n\}$ is the set of customers and $I_D = \{n+1, n+2, ..., n+d\}$ is the set containing the starting positions of the active vehicles at the time of start of the new plan T_0 (where $T_0 = 0$).

However, instead of denoting the endpoint node by node 0 (as in the previous chapter), in the heuristic for technical reasons we use d copies of the depot, one for each route. Therefore, let $I_E = \{n+d+u+1,\ n+d+u+2, ...,\ n+d+u+d\}$ be the set containing the endpoint nodes of vehicles $1, 2, ..., d$ respectively. This notation is general enough to allow different endpoint nodes for different vehicles. For the problem under study, however, since the endpoint node of each route is the depot, all of those d endpoint nodes correspond to the same geographic location.

For each $k = 1, 2, ..., d$, the route of the k-th active vehicle is denoted by R_k and is a vector of variable size. Initially, the vector R_k starts with node $n+k$, followed by all nodes of the set S_k, and ends with node $n+d+u+k$, which in the heuristic denotes the endpoint node of vehicle k (for all $k = 1, 2, ..., d$).

We also make use of another vector R_{d+1} of variable size, which is of different structure compared to the vectors $R_1, R_2, ..., R_d$. Specifically, vector R_{d+1} initially contains the u copies of the broken-down vehicle, i.e. nodes $n+d+1, n+d+2, ..., n+d+u$, followed by all nodes $1, 2, ..., u$ of the set S_{d+1} of customers that were originally assigned to the broken-down vehicle.

As the heuristic progresses, each customer of the set S_{d+1} is moved from route R_{d+1} to one of the active routes R_k ($k = 1, 2, ..., d$), with the convention that whenever you move such a customer $i \in S_{d+1}$ from one route R_{k_1} ($k_1 = 1, 2, ..., d+1$) to another route R_{k_2} ($k_2 = 1, 2, ..., d,\ k_2 \neq k_1$), you also move node $n+d+i$ from its current position and place it in the same route R_{k_2} in which customer i is placed, at the position which is just before the new position of customer i in route R_{k_2}.

By the time when the heuristic terminates, the vector R_{d+1} should be an empty vector, and each one of the customers originally assigned to the BV, along with respective BV copies, should be included in one of the vectors $R_1, R_2, ..., R_d$. So, eventually, for each $k = 1, 2, ..., d$, the vector R_k should start with node $n + k$, followed by all nodes of the set S_k and perhaps some nodes of the set S_{d+1}, provided that it visits node $n + d + r$ to pick up the goods of customer r before it visits any customer $r \in S_{d+1}$, and ends with node $n + d + u + k$ (which is the endpoint node of vehicle k).

Finally, let $R = (R_1, R_2, ..., R_{d+1})$ be the ordered collection of the $d + 1$ variable-sized vectors. From now on, R will be referred to as *the collection of routes R*. Also, let $R_k = (R_k^1, R_k^2, ..., R_k^{l_k})$ be the vector of variable length l_k which corresponds to the route of the k-th vehicle, for $k = 1, 2, ..., d+1$. Using this notation, R_k^j denotes the j-th node of the k-th route, for $k = 1, 2, ..., d+1$ and $j = 1, 2, ..., l_k$. For each $k = 1, 2, ..., d$, the first node R_k^1 of the k-th route is fixed and denotes the starting position $n + d + k$ of vehicle k. The last node $R_k^{l_k}$ is also fixed and denotes the endpoint node $n + d + u + k$ of vehicle k.

Note that, if vector R_k ($k = 1, 2, ..., d$) contains several consecutive copies of the BV node, this corresponds to a single actual visit to the BV. After all, all copies of the BV node correspond to the same geographical location and both the distance and travel times between two such nodes is zero.

Finally, let us mention that in this section what was described as the initial solution and initial setup of the vectors $R_1, R_2, ..., R_{d+1}$, are actually the initial solution and setup of the $(d + 1)$-th restart alone. In all the other restarts, the initial setup and solutions are described in section 4.5.

4.4 The easy plans

In general, there is an obvious and very simple way to construct d different solutions to the problem, following the so-called *i-th easy plan*, for $i = 1, 2, ..., d$. These solutions are always feasible in the case of a homogeneous fleet, i.e. if all $d + 1$ vehicles have the same capacity. However, for the more general case that we study where the fleet is heterogeneous, an additional capacity constraint needs to be satisfied in order for each one of these plans to be feasible.

The i-th easy plan (for $i = 1, 2, ..., d$):
Once the disruption occurs, all active vehicles apart from vehicle i will continue with their routes, as scheduled in the original plan, serve all their remaining customers and finish at the depot. Vehicle i will also continue with its route, as scheduled in the original plan, and serve all remaining customers

that were originally assigned to it, until it reaches its last customer. Then, it will be diverted to visit the broken-down vehicle, pick up all the orders of the BV's unserved customers, and serve all those unserved customers of the BV, in the order that they were originally supposed to be served by the BV. Vehicle i will then return to the depot.

The i-th easy plan is feasible if and only if the capacity of vehicle i is no less than the sum of demand of the BV's unserved customers (i.e. iff $Q_i \geq \sum_{i \in S_{d+1}} q_i$). For instance, in case of a homogeneous fleet, or more generally in case where vehicles i and $d+1$ have the same capacity, then the i-th easy plan is always feasible - assuming, of course, that the original solution for the undisrupted problem was a feasible one.

Considering, however, the VRPTW extension of problem 1 (see section 3.6), i.e. in the presence of time windows, the above condition is still necessary for the i-th easy plan to be feasible, but not necessarily sufficient.

4.5 Diversification mechanisms and initial solutions

We use two diversification mechanisms. The first one involves the use of multiple restarts. Specifically, we employ $d+1$ restarts of the algorithm, each time starting from a different initial solution, as described below.

For $i = 1, 2, ..., d$, the i-th restart of the algorithm uses the *i-th easy plan* as the starting solution. In this initial solution, for all $k = 1, 2, ..., d$ with $k \neq i$, vector R_k starts with node $n+k$, contains all nodes of the set S_k in the same sequence as in the solution of the original undisrupted problem, and ends with node $n+d+u+k$. On the other hand, vector R_i starts with node $n+i$, contains all nodes of the set S_i in the same sequence as in the original solution plan of the underlying problem, then contains all the copies of the BV node (i.e. nodes $n+d+1, n+d+2, ..., n+d+u$), then all nodes of the set S_{d+1} in the same sequence as in the solution of the original undisrupted problem, and ends with node $n+d+u+i$. Also, in this initial solution, vector R_{d+1} is empty. In fact, in this case vector R_{d+1} remains empty until the end of this restart.

For the last restart, where $i = d+1$, the algorithm uses an initial solution which resembles the unfulfilled part of the original solution plan of the undisrupted problem, and was also discussed in section 4.3. In this initial solution, for all $k = 1, 2, ..., d$, vector R_k starts with node $n+k$, contains all nodes of the set S_k in the same sequence as in the original solution plan of the undisrupted problem, and ends with node $n+d+u+k$. Also, in this initial solution, vector R_{d+1} contains all the copies of the BV node (i.e. nodes $n+d+1, n+d+2, ..., n+d+u$), and then all nodes of the set S_{d+1} in the order $1, 2, ..., d$ (which is not necessarily the same order as in the undisrupted plan). Then, as the algorithm progresses, the nodes of the last vector R_{d+1} are sequentially inserted into the other vectors R_k for $k = 1, 2, ..., d$, which represent the routes of the active vehicles. Eventually, all nodes that were initially contained in the vector R_{d+1} must be placed in the other routes, so at the end of the restart the vector R_{d+1} must be empty.

As a second diversification mechanism, we try to prevent the scenario where some nodes are used much more frequently than others in neighborhood moves. For this, we do not allow moves which involve a node that has been used more than $\lceil \theta_1 + \theta_2 \cdot \bar{x} \rceil$ times so far, unless these moves lead to solutions that improve the best solution found so far. Note that $\bar{x}$ here denotes the average number of times an active node has been used in a move so far, where by the term *active nodes* we refer to nodes $1, 2, ..., n$ and $n+d+1, n+d+2, ..., n+d+u$, which are the nodes that can actually be involved in a move (since the position of the remaining nodes remains unchanged). Also, θ_1 and θ_2 are known, fixed parameters.[1]

4.6 Neighborhood structure

The following 4 types of neighborhood moves are considered:

- **Exchange:**
  This move involves swapping two nodes with one another. Specifically, $exchange(R_{k_1}^i, R_{k_2}^j)$ swaps node i from route k_1, with node j from route k_2. Here we only consider *intra-route moves* of this type, i.e. $k_1 = k_2$, with $k_1 \in \{1, 2, ..., d\}$ and with $i, j \in \{2, 3, ..., l_{k_1} - 1\}$. The initial and final nodes of routes $R_1, R_2, ..., R_d$ are not allowed to be moved by the exchange operator, nor is any node of the vector R_{d+1}.

  As mentioned before, if vector R_k ($k = 1, 2, ..., d$) contains several consecutive copies of the BV node, this corresponds to a single actual visit to the BV. Specifically, suppose that vector R_k is of the form $R_k = (R_k^1, R_k^2, ..., R_k^{i_1}, R_k^{i_1+1}, ..., R_k^{i_1+j}, ..., R_k^{l_k})$ where the nodes $R_k^{i_1}, R_k^{i_1+1}, ..., R_k^{i_1+j}$ are all copies of the BV node (for instance, they may be equal to $n+d+1, n+d+2, ..., n+d+j+1$), where $j+1 \leq u$. Any permutation of the sequence $R_k^{i_1}, R_k^{i_1+1}, ..., R_k^{i_1+j}$ of any number of consecutive copies of the BV will have no actual effect on the route. Therefore, we do

[1]Throughout the experiments, we used $\theta_1 = d + 3$ and $\theta_2 = 1.7$.

not consider moves of the form *exchange*(R_k^i, R_k^j) in the case where all nodes $R_k^i, R_k^{i+1}, ..., R_k^j$ are copies of the BV.

- **Insertion:**
  In general, this move involves removing a node from its current position in a route and placing it into a different position, either in the same route or in a different route.

  Here we only consider *intra-route insertion*, which takes one node from a single route and places it at a different position in the same route. Specifically, *insertion*(R_i^k, R_j^k) with $i \neq j$ takes the i-th node of the k-th route, removes it from there and inserts it at the position where the j-th node of the same route was before the move. Additionally, if $i > j$ then the nodes that before the move were at positions $j, j+1, ..., i-1$ are shifted by one position to the right, whereas the remaining nodes and the size of the route remain unchanged. Alternatively, if $i < j$ then the nodes that before the move were at positions $i+1, i+2, ..., j$ are shifted by one position to the left, whereas the remaining nodes and the size of the route remain unchanged.

  We only consider intra-route *insertion*(R_i^k, R_j^k) moves with $k \in \{1, 2, ..., d\}$, $i, j \in \{2, 3, ..., l_k - 1\}$ and $i \neq j$. The initial and final nodes of routes $R_1, R_2, ..., R_d$ are not allowed to be moved by the insertion move, nor is any node of the vector R_{d+1}.

  Furthermore, we do not consider moves of the form *insertion*(R_k^i, R_k^j) with $i > j$, in the case where all nodes $R_k^j, R_k^{j+1}, ..., R_k^{i-1}$ are copies of the BV, because such a move will have no actual effect on the route. Similarly, we do not consider the move *insertion*(R_k^i, R_k^j) with $i < j$, in the case where all nodes $R_k^{i+1}, R_k^{i+2}, ..., R_k^j$ are copies of the BV.

- **Double Insertion:**
  This is a special type of move that was designed specifically for this problem and for the specific notation and approach. *Double insertion* $(R_{k_1}, p_1, l_1, R_{k_2}, p_2, l_2)$ takes a BV copy $n+d+i$ from position p_2 of route R_{k_2} and the corresponding customer i $(i = 1, 2, ..., u)$ from position l_2 of route R_{k_2}, removes both from there, and inserts them at positions p_1 and l_1, respectively, of route R_{k_1}.

- **Double Exchange:**
  This is also a special type of move designed for this problem and for the specific notation and approach. *Double exchange*$(R_{k_1}, p_1, l_1, R_{k_2}, p_2, l_2)$ takes a BV copy $n + d + i$ from position p_1 of route R_{k_1} and the corresponding customer i $(i = 1, 2, ..., u)$ from position l_1 of route R_{k_1}, and

exchanges them both with another BV copy $n+d+j$ at position p_2 of route R_{k_2} and the corresponding customer j ($j = 1, 2, ..., u$, $j \neq i$) from position l_2 of route R_{k_2}, respectively.

4.7 Objective function

The objective function of the heuristic is:

$$\begin{aligned} \text{minimize} \quad G := & \sum_{k \in K} \sum_{(i,j) \in A} t_{ij} x_{ijk} + \epsilon \sum_{k=1}^{d} w_{n+d+u+k} + \\ & + \sum_{k=1}^{d} (P_1 \xi_k + P_2 \phi_k) + \sum_{i=1}^{u} P_3 \psi_i + P_4 |R_{d+1}| \end{aligned} \tag{4.7.1}$$

where for all $k = 1, 2, ...d$, the variables ξ_k and ϕ_k are defined as:

$$\xi_k := \begin{cases} 1, & \text{if } Q_{ik} > Q_k \\ 0, & \text{otherwise} \end{cases} \tag{4.7.2}$$

$$\phi_k := \begin{cases} Q_{ik} - Q_k, & \text{if } Q_{ik} > Q_k \\ 0, & \text{otherwise} \end{cases} \tag{4.7.3}$$

Also, for all $i = 1, 2, ...u$ the variable ψ_i is set to 1 if nodes i and $n+d+i$ are not in the same route, or if they are in the same route and node i comes before node $n+d+i$. Otherwise, ψ_i is set to zero.

Note that $|R_{d+1}|$ is the cardinality of the vector R_{d+1}, if seen as a set; i.e. it denotes the number of nodes contained in the vector R_{d+1}.

The objective function G defined by equation (4.7.1) is composed of five components. The first summation term is the total travel time $f_1 := \sum_{k \in K} \sum_{(i,j) \in A} t_{ij} x_{ijk}$, as in the previous chapter. The second summation term represents the sum of arrival times of the active vehicles at the endpoint nodes, which in the heuristic is defined as $f_2^H := \sum_{k=1}^{d} w_{n+d+u+k}$. Note that, using the notation of the exact approach, this is equivalent to the term $\sum_{k \in K} w_{0k}$ that was used in the previous chapter as f_2. The third summation term is the penalty term for capacity violation, where ξ_k is equal to 1 if the capacity constraint is violated in the k-th route, or 0 otherwise. Also, ϕ_k is the amount of capacity violation in the k-th route, if any (ϕ_k is zero if there is no capacity violation in the k-th route). The fourth summation term is the penalty function for violating the precedence constraints. Finally, the fifth component of G, i.e. $P_4|R_{d+1}|$, is a penalty function proportional to the number of nodes that have not been allocated to an active vehicle so far. The parameters P_1, P_2, P_3 and P_4 are fixed penalty coefficients.[2]

Essentially we use the function G defined above as the objective and allow intermediate solutions which are infeasible. However, the heuristic will always seek to minimize the penalties added, since this will decrease and thus improve the value of the objective function. In general, we should end up with a final solution with zero penalties and thus feasible with respect to the capacity and precedence constraints, where all nodes are allocated to the active vehicles. In such a case, the third, fourth and fifth components of G will all be zero; therefore, G will be equal to $f^H := f_1 + \epsilon \cdot f_2^H$, which is the equivalent of the objective function (3.4.1) using the heuristic notation.

4.8 Search termination

Each one of the $d+1$ different restarts terminates when there is no improvement in the last J_1 iterations.[3]

4.9 Parameter values used in the heuristic

After some experimentation, we decided to use the following parameter values in our experiments with the heuristic:

P_1	P_2	P_3	P_4	θ_1	θ_2	J_1	ϵ
10^{10}	10^9	10^{15}	10^{20}	$d+3$	1.7	40	10^{-7}

The *tabu length (TL)* is defined as a function of the *number of active nodes* $(n+u)$, as shown below:

$$TL = \begin{cases} \lfloor \frac{n+u}{3} \rfloor & \text{if } n+u \leq 11 \\ \lfloor \frac{n+u}{4} \rfloor & \text{if } 12 \leq n+u \leq 19 \\ \lfloor \frac{n+u}{5} \rfloor & \text{if } 20 \leq n+u \leq 39 \\ 7 & \text{if } 40 \leq n+u \leq 69 \\ \lfloor \frac{n+u}{10} \rfloor & \text{if } 70 \leq n+u \leq 119 \\ \lfloor \frac{n+u}{12} \rfloor & \text{if } n+u \geq 120 \end{cases}$$

[2]After some experimentation, we decided to use the following parameter values in our experiments: $P_1 = 10^{10}$, $P_2 = 10^9$, $P_3 = 10^{15}$ and $P_4 = 10^{20}$.

[3]In our experiments, we used the parameter value $J_1 = 40$.

Additionally, the following parameters and sets should be given as an input separately for each instance: n, d, S_k's, x_i's, y_i's, a_i's, b_i's, s_i's, q_i's, Q_k's. Note that the value of u is implied from the set S_{d+1}, since $u = |S_{d+1}|$.

Finally, the travel times $t_{i,j}$ in the heuristic approach are calculated as the Euclidean distances between each pair of nodes (i, j), rounded to 2 decimal places; i.e. the same way as in the exact approach (described in section 3.5).

4.10 Experimental results of the heuristic approach

The heuristic described in this chapter was implemented in MATLAB and tested on the 101 instances that were presented in the previous chapter. The computational experiments were performed on a personal computer with the following system specifications: Intel Core i5-3230M CPU running at 2.60 GHz, with 8 GB RAM, running on Windows 8, 64-bit Operating System. The results from the heuristic approach are compared with the respective results from the exact approach, which were presented in table 3.1 of section 3.5. Tables 4.1 and 4.2 that are presented in this section, summarize the experimental results for problem 1 solved using the heuristic approach.

Specifically, the first 7 columns of tables 4.1 and 4.2, show the following information for each instance: the serial number and code-name of the instance, the number of customer nodes n, the number of active vehicles d, the number of unserved customers u that were originally assigned to broken-down vehicle, the total number of nodes in the heuristic ($N_H := n + u + 2d$), and the values of the two component objectives f_1 (total travel time) and f_2^H (sum of arrival times at the endpoint nodes, which includes travel times, service times any idle waiting times).

The 8th column (with label $f^H := f_1 + \epsilon \cdot f_2^H$) actually shows the value of the objective G of the heuristic, as defined by equation (4.7.1). However, in all 101 instances the heuristic solution found is feasible, and therefore G simplifies to $f^H := f_1 + \epsilon \cdot f_2^H$ in every instance examined. Therefore, regarding the experimental results of this section, from now on we will refer to the objective of the heuristic as f^H, without further distinction from G, in order to make the comparison between the exact and heuristic approaches more direct and easier to understand.

The 9th column of tables 4.1 and 4.2, shows the percentage gap between the objective value f^H in the heuristic solution, and the objective value f^E found by AIMMS when following the exact approach, calculated as: $\frac{f^H - f^E}{f^H} \cdot 100\%$. The 10th column shows the percentage gap between the

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_H}$ **= n+u +2d**	$\mathbf{f_1}$	$\mathbf{f_2^H}$	$\mathbf{f^H := f_1 + \epsilon f_2^H}$ **(same as G)**	**% of** $\mathbf{f^H}$ **from best exact** $\mathbf{f^E}$	**% of** $\mathbf{f^H}$ **from LB,** $\mathbf{f^L}$	**O/ F/ I**	**run-time (sec)**	**RV**	**NV BV**
Class T													
1 - T1	6	2	4	14	110.64	120.64	110.640012064	0.00%	0.00%	O	4.9	2	4
2 - T2	6	2	4	14	70.1	80.1	70.10000801	0.00%	0.00%	O	5.1	1	1
3 - T3	6	2	4	14	75.92	85.92	75.920008592	0.00%	0.00%	O	4.9	1	4
4 - T4	6	2	4	14	72.32	82.32	72.320008232	0.00%	0.00%	O	5	1	2
5 - T5	6	2	4	14	71.89	81.89	71.890008189	0.00%	0.00%	O	5	1	2
6 - T6	6	2	4	14	107.04	117.04	107.040011704	0.00%	0.00%	O	5.2	2	2
7 - T7	6	2	4	14	104.82	114.82	104.820011482	0.00%	0.00%	O	5	2	1
8 - T8	6	2	4	14	106.61	116.61	106.610011661	0.00%	0.00%	O	5	2	2
Class A-n32-k5													
9 - A-n32-k5(1,20)	27	4	7	42	569.29	569.29	569.290056929	0.00%	4.35%	F	36.9	4	1
10 - A-n32-k5(2,20)	27	4	4	39	568.21	568.21	568.210056821	0.00%	0.00%	O	19.7	1	1
11 - A-n32-k5(3,20)	27	4	2	37	566.79	566.79	566.790056679	0.85%	0.85%	F	12.1	4	1
12 - A-n32-k5(4,20)	27	4	10	45	596.22	596.22	596.220059622	7.12%	19.03%	F	70.1	4	1
13 - A-n32-k5(5,20)	27	4	8	43	558.43	558.43	558.430055843	0.00%	13.65%	F	46.4	3	1
14 - A-n32-k5(1,30)	24	4	7	39	508.31	508.31	508.310050831	0.00%	6.13%	F	33.5	4	1
15 - A-n32-k5(2,30)	24	4	3	35	498.79	498.79	498.790049879	0.00%	0.00%	O	14.4	1	1
16 - A-n32-k5(3,30)	24	4	1	33	492.26	492.26	492.260049226	0.00%	0.00%	O	8.9	4	1
17 - A-n32-k5(4,30)	24	4	10	42	528.02	528.02	528.020052802	0.05%	10.37%	F	61.7	4	1
18 - A-n32-k5(5,30)	24	4	7	39	540.14	540.14	540.140054014	0.00%	13.52%	F	33.6	3	1
Class A-n45-k6													
19 - A-n45-k6(1,20)	39	5	8	57	712.14	712.14	712.140071214	-9.22%	8.70%	F	76.1	2	1
20 - A-n45-k6(2,20)	39	5	6	55	710.13	710.13	710.130071013	1.00%	7.33%	F	48.6	2	1
21 - A-n45-k6(3,20)	39	5	9	58	724.48	724.48	724.480072448	-2.22%	9.97%	F	82	2,3	1,1
22 - A-n45-k6(4,20)	39	5	8	57	743.38	743.38	743.380074338	0.00%	7.86%	F	71.5	3	1
23 - A-n45-k6(5,20)	39	5	6	55	711.76	711.76	711.760071176	-0.56%	8.47%	F	49.9	3	1
24 - A-n45-k6(6,20)	39	5	7	56	711.09	711.09	711.090071109	0.00%	0.00%	O	61.9	3	1
25 - A-n45-k6(1,40)	35	5	8	53	636.26	636.26	636.260063626	-17.00%	9.39%	F	65.3	2	1
26 - A-n45-k6(2,40)	35	5	5	50	637.55	637.55	637.550063755	1.52%	1.52%	F	34	2	1
27 - A-n45-k6(3,40)	35	5	8	53	648.04	648.04	648.040064804	0.44%	6.23%	F	67.8	2	1
28 - A-n45-k6(4,40)	35	5	8	53	660.8	660.8	660.80006608	0.00%	7.24%	F	69.9	3	1
29 - A-n45-k6(5,40)	35	5	5	50	616.74	616.74	616.740061674	0.00%	0.00%	O	38.2	1	2
30 - A-n45-k6(6,40)	35	5	6	51	654.25	654.25	654.250065425	0.00%	0.00%	O	42.1	3	1
31 - A-n45-k6(1,60)	26	5	6	42	502.77	502.77	502.770050277	0.00%	0.00%	O	30.6	2	1
32 - A-n45-k6(2,60)	26	5	4	40	470.47	470.47	470.470047047	0.00%	0.00%	O	19.9	2	1
33 - A-n45-k6(3,60)	26	5	7	43	545.95	545.95	545.950054595	3.88%	15.18%	F	38.1	3	1
34 - A-n45-k6(4,60)	26	5	7	43	509.51	509.51	509.510050951	0.00%	9.60%	F	44.1	3	1
35 - A-n45-k6(5,60)	26	5	3	39	455.29	455.29	455.290045529	0.00%	0.00%	O	20.8	1	1
36 - A-n45-k6(6,60)	26	5	4	40	524.28	524.28	524.280052428	0.00%	0.00%	O	20.9	3	1
37 - A-n45-k6(1,80)	21	5	6	37	435.16	435.16	435.160043516	-6.37%	5.01%	F	26	2	1
38 - A-n45-k6(2,80)	21	5	3	34	403.29	403.29	403.290040329	0.00%	0.00%	O	13.2	2	1
39 - A-n45-k6(3,80)	21	5	7	38	478.59	478.59	478.590047859	8.61%	21.72%	F	33	3	1
40 - A-n45-k6(4,80)	21	5	5	36	440.1	440.1	440.10004401	0.00%	0.00%	O	20.8	3	1
41 - A-n45-k6(5,80)	21	5	1	32	411.33	411.33	411.330041133	0.00%	0.00%	O	9	1	1
42 - A-n45-k6(6,80)	21	5	4	35	442.93	442.93	442.930044293	0.00%	0.00%	O	17.6	4	1
Class A-n61-k9													
43 - A-n61-k9(1,10)	51	8	4	71	815.13	815.13	815.130081513	0.00%	0.00%	O	61.4	8	1
44 - A-n61-k9(2,10)	51	8	8	75	822.7	822.7	822.70008227	-2.10%	3.72%	F	144.2	1	1
45 - A-n61-k9(3,10)	51	8	7	74	805.13	805.13	805.130080513	-5.02%	6.27%	F	120.6	8	1
46 - A-n61-k9(4,10)	51	8	6	73	804.81	804.81	804.810080481	-4.69%	9.27%	F	97.1	6	1
47 - A-n61-k9(5,10)	51	8	10	77	810.51	810.51	810.510081051	-31.29%	10.43%	F	218.1	5	2
48 - A-n61-k9(6,10)	51	8	7	74	814.11	814.11	814.110081411	0.00%	3.45%	F	130.5	7	2
49 - A-n61-k9(7,10)	51	8	3	70	795.53	795.53	795.530079553	0.00%	0.00%	O	48.4	8	1

Table 4.1: Experimental Results for Problem 1 solved using the Heuristic Approach (in MATLAB) - *Part 1 of 2*

Serial no. & Instance name	**n**	**d**	**u**	$\mathbf{N_H =}$ **n+u +2d**	$\mathbf{f_1}$	$\mathbf{f_2^H}$	$\mathbf{f^H := f_1 + \epsilon f_2^H}$ **(same as G)**	**% of** $\mathbf{f^H}$ **from best exact** $\mathbf{f^E}$	**% of** $\mathbf{f^H}$ **from LB,** $\mathbf{f^L}$	**O/ F/ I**	**run-time (sec)**	**RV**	**NV BV**
50 - A-n61-k9(8,10)	51	8	8	75	810.85	810.85	810.850081085	-5.17%	2.59%	F	147.5	1	1
51 - A-n61-k9(9,10)	51	8	6	73	799.42	799.42	799.420079942	0.95%	3.59%	F	97.3	7	1
52 - A-n61-k9(1,20)	46	8	2	64	727.76	727.76	727.760072776	0.00%	0.00%	O	30.9	1	1
53 - A-n61-k9(2,20)	46	8	8	70	731.46	731.46	731.460073146	-1.88%	3.58%	F	130.6	1	1
54 - A-n61-k9(3,20)	46	8	7	69	709.59	709.59	709.590070959	0.00%	4.34%	F	106.3	8	2
55 - A-n61-k9(4,20)	46	8	5	67	749.11	749.11	749.110074911	0.00%	7.20%	F	67.4	3	1
56 - A-n61-k9(5,20)	46	8	9	71	740.98	740.98	740.980074098	-11.34%	8.82%	F	152	4	1
57 - A-n61-k9(6,20)	46	8	7	69	723.97	723.97	723.970072397	-4.46%	2.16%	F	106.9	7	2
58 - A-n61-k9(7,20)	46	8	3	65	702.69	702.69	702.690070269	0.00%	0.00%	O	42.1	8	1
59 - A-n61-k9(8,20)	46	8	7	69	735.63	735.63	735.630073563	-4.01%	1.72%	F	105.2	1	1
60 - A-n61-k9(9,20)	46	8	6	68	710.82	710.82	710.820071082	0.66%	3.37%	F	88	7	1
61 - A-n61-k9(1,35)	39	8	1	56	662.47	662.47	662.470066247	0.00%	0.00%	O	20.2	1	1
62 - A-n61-k9(2,35)	39	8	6	61	669.42	669.42	669.420066942	-0.42%	3.80%	F	78.2	1	1
63 - A-n61-k9(3,35)	39	8	7	62	632.93	632.93	632.930063293	-9.22%	5.22%	F	94.1	6	1
64 - A-n61-k9(4,35)	39	8	5	60	647.84	647.84	647.840064784	0.00%	4.20%	F	58.8	3	1
65 - A-n61-k9(5,35)	39	8	9	64	655.89	655.89	655.890065589	-9.53%	7.59%	F	156.6	4	1
66 - A-n61-k9(6,35)	39	8	5	60	660.92	660.92	660.920066092	0.00%	0.00%	O	60.2	7	1
67 - A-n61-k9(7,35)	39	8	3	58	628.39	628.39	628.390062839	0.00%	0.00%	O	36.1	8	1
68 - A-n61-k9(8,35)	39	8	6	61	670.4	670.4	670.40006704	-1.04%	1.64%	F	76	1	1
69 - A-n61-k9(9,35)	39	8	5	60	644.79	644.79	644.790064479	0.95%	0.95%	F	58.6	7	1
Class A-n65-k9													
70 - A-n65-k9(1,20)	54	8	6	76	832.24	832.24	832.240083224	0.00%	1.10%	F	105.9	7	1
71 - A-n65-k9(2,20)	54	8	9	79	830.59	830.59	830.590083059	-6.02%	3.78%	F	177.6	6	1
72 - A-n65-k9(3,20)	54	8	8	78	833.63	833.63	833.630083363	0.00%	4.53%	F	158.6	7	2
73 - A-n65-k9(4,20)	54	8	6	76	857.84	857.84	857.840085784	0.00%	4.95%	F	99.2	2	1
74 - A-n65-k9(5,20)	54	8	7	77	846.07	846.07	846.070084607	-0.18%	2.02%	F	125.9	8	1
75 - A-n65-k9(6,20)	54	8	8	78	830.42	830.42	830.420083042	0.00%	0.67%	F	149.4	3	1
76 - A-n65-k9(7,20)	54	8	6	76	908.97	908.97	908.970090897	-3.38%	11.65%	F	104.3	2	1
77 - A-n65-k9(8,20)	54	8	7	77	833.19	833.19	833.190083319	-7.80%	8.14%	F	122.7	7	1
78 - A-n65-k9(9,20)	54	8	5	75	857.91	857.91	857.910085791	0.11%	0.11%	F	82.2	4	1
79 - A-n65-k9(1,40)	42	8	4	62	689.75	689.75	689.750068975	0.00%	0.00%	O	45.9	7	1
80 - A-n65-k9(2,40)	42	8	8	66	696.17	696.17	696.170069617	0.00%	4.35%	F	110.4	6	1
81 - A-n65-k9(3,40)	42	8	7	65	676.57	676.57	676.570067657	-0.34%	5.27%	F	89.3	7	1
82 - A-n65-k9(4,40)	42	8	5	63	682.62	682.62	682.620068262	0.00%	0.00%	O	57.4	2	1
83 - A-n65-k9(5,40)	42	8	4	62	691.41	691.41	691.410069141	0.00%	0.00%	O	45.4	8	1
84 - A-n65-k9(6,40)	42	8	6	64	680.98	680.98	680.980068098	0.00%	0.00%	O	72.2	3	1
85 - A-n65-k9(7,40)	42	8	6	64	718.54	718.54	718.540071854	0.00%	9.37%	F	73.8	7	1
86 - A-n65-k9(8,40)	42	8	6	64	685.41	685.41	685.410068541	-4.68%	5.74%	F	73.1	7	1
87 - A-n65-k9(9,40)	42	8	4	62	693.7	693.7	693.70006937	0.00%	0.00%	O	45.2	4	1
Class E-n76-k7													
88 - E-n76-k7(1,10)	64	6	1	77	592.59	592.59	592.590059259	0.13%	0.13%	F	34	3	1
89 - E-n76-k7(2,10)	64	6	13	89	586.17	586.17	586.170058617	NA	4.32%	F	339.5	5	1
90 - E-n76-k7(3,10)	64	6	9	85	584.97	584.97	584.970058497	-10.45%	3.03%	F	289	6	2
91 - E-n76-k7(4,10)	64	6	13	89	585.57	585.57	585.570058557	-38.50%	2.80%	F	476.6	5	1
92 - E-n76-k7(5,10)	64	6	10	86	593.28	593.28	593.280059328	-6.78%	4.90%	F	333.5	2	1
93 - E-n76-k7(6,10)	64	6	11	87	597.27	597.27	597.270059727	-15.71%	4.36%	F	289.6	4	1
94 - E-n76-k7(7,10)	64	6	13	89	583.87	583.87	583.870058387	NA	4.93%	F	396.7	3	1
95 - E-n76-k7(1,15)	58	6	1	71	544	544	544.0000544	0.00%	0.00%	O	31	6	1
96 - E-n76-k7(2,15)	58	6	12	82	547.3	547.3	547.30005473	-15.05%	2.96%	F	253.5	5	1
97 - E-n76-k7(3,15)	58	6	8	78	554.05	554.05	554.050055405	-0.26%	3.74%	F	142.2	6	1
98 - E-n76-k7(4,15)	58	6	12	82	541.59	541.59	541.590054159	-8.88%	2.53%	F	257.3	5	1
99 - E-n76-k7(5,15)	58	6	9	79	556.96	556.96	556.960055696	-0.04%	4.45%	F	185.9	2	1
100 - E-n76-k7(6,15)	58	6	11	81	552.69	552.69	552.690055269	0.00%	4.23%	F	231.1	4	1
101 - E-n76-k7(7,15)	58	6	11	81	546.89	546.89	546.890054689	-2.76%	2.74%	F	250.8	3	1

Table 4.2: Experimental Results for Problem 1 solved using the Heuristic Approach (in MATLAB) - *Part 2 of 2*

objective value f^H in the heuristic solution, and the lower bound f^L that was found by AIMMS when following the exact approach, calculated as: $\frac{f^H - f^L}{f^H} \cdot 100\%$. The 11-th column with label O/F/I indicates whether the solution found is optimal (O), feasible (F) or infeasible (I). The last 3 columns show, respectively, the runtime in seconds, the serial number of the rescue vehicle(s) (RV) involved in the best solution found, and the number of visits to the broken-down vehicle (NVBV) by each one of the rescue vehicles. Note that the values of f^E and f^L that are used to calculate the percentage gaps in columns 9 and 10, were taken from table 3.1 of section 3.5 that summarizes the experimental results of the exact approach for Problem 1.

As discussed earlier in this chapter, the stopping criterion for the heuristic algorithm was to reach a fixed number of iterations without an improvement (in each one of the $d + 1$ different restarts). Therefore, no cutoff time was applied (in contrast to the exact approach where a 5 minutes cutoff time was applied). However, from the results tables we can see that the heuristic terminated within 300 seconds (5 minutes) in 97 out of 101 instances, whereas in the remaining 4 instances it needed between 300 and 480 seconds (5-8 minutes) to terminate. Therefore, valid comparisons between the two methods can still be made, since in 97 out of the 101 instances (i.e. in 96% of the cases) the heuristic results would have been the same even if a cutoff time of 5 minutes was applied in the heuristic method, as was the case in the exact approach. Furthermore, 2 out of the 4 remaining instances where we allowed more than 5 minutes in the heuristic, namely instances 89 and 94 (with runtimes 340 and 397 seconds, respectively), are the two instances where the exact approach didn't provide any feasible solution within 5 minutes.[4] [5]

From tables 4.1 and 4.2, we can see that in all 101 instances (i.e. in 100% of the cases), the solution provided by the heuristic was feasible. Among those, in 35 instances (i.e. in 34.7% of the cases) the solution found by the heuristic method matches precisely the lower bound f^L that was found by AIMMS in the exact approach, and is therefore definitely optimal. Of course, this does not necessarily imply that the solution provided by the heuristic in the remaining 66 instances was not optimal, since we had found the proven optimal in just 40 out of the 101 instances by following the exact approach.

[4]In fact, in instance 94, AIMMS wasn't able to provide a feasible solution even when we used a cutoff time of 30 minutes, as shown in table 3.2.

[5]Note that, considering the 4 instances 89, 91, 92 and 94 for which the runtime reported by the heuristic was more than 5 minutes, we tried solving them again after adding a cutoff time of 5 minutes and with the same set of parameters. In 3 out of those 4 instances (apart from instance 91), we ended up with the same solution as the one reported in tables 4.1 and 4.2. Therefore, if we actually had a cutoff time of 5 minutes, the results would have been the same in 100 out of the 101 instances.

Therefore, since we do not know the optimal solutions in 61 instances, we can make comparisons with the objective value f^E of the solution found by the exact approach, as well as with the lower bound f^L found by AIMMS.

Considering the remaining 66 instances for which the heuristic solution may not necessarily be optimal, the percentage gap between the objective value f^H in the heuristic solution and the lower bound f^L found by AIMMS, varied between 0% and 22%, with an average of 5.86%. More specifically, this percentage gap varied between 0% and 5% in 37 cases, between 5% and 10% in 21 other cases, between 10% and 15.2% in 6 other cases, and between 19% and 22% in the remaining 2 cases.

A direct comparison between the heuristic and the exact approaches can be made by examining the 9th column of tables 4.1 and 4.2, which shows the percentage gap between the objective values f^H and f^E of the solutions found by the heuristic and exact approaches, respectively. In this column, a negative percentage value indicates that the solution found by the heuristic outperforms the one found by the exact approach. Therefore, we can see that out of the 101 instances considered, the heuristic solution was better than the one found by the exact approach in 35 instances (including the two instances where the exact approach didn't provide a feasible solution), was exactly the same in 53 instances, and was worse in 13 other instances. In the cases where the heuristic approach outperformed the exact approach, the percentage gap varied between -38.5% and -0.04%, with an average of -7.47% (considering only the 33 out of 35 such cases for which an exact solution was available). On the other hand, in the 13 cases where the heuristic approach was outperformed by the exact approach, the percentage gap varied between 0.05% and 8.61%, with an average of 2.02%.

In 87 out of the 101 instances, the best solution found by the heuristic involved only 1 visit to the broken-down vehicle, by a single active vehicle. In 11 other cases, the heuristic solution reported involved 2 visits to the broken-down vehicle, by a single active vehicle. In 1 other instance, the heuristic solution involved 2 different active vehicles, each visiting the disabled vehicle once. In the remaining 2 instances, the heuristic solution involved 4 visits to the broken-down vehicle, by a single active vehicle.

Considering the runtime, the heuristic terminated in less than 100 seconds in 70 out of the 101 instances, between 100 and 200 seconds in 20 other instances, between 200 and 300 seconds in 7 other cases, and finally between 300 and 480 seconds in the remaining 4 instances. The average runtime for the whole dataset of the 101 instances, was 90.7 seconds.

At this point, we should mention that it may be possible to significantly reduce the runtime of the heuristic down to around, say, $\frac{1}{d+1}$ or $\frac{2}{d+1}$ of the time provided in these tables, by not exhaustively considering all $d+1$ possible restarts. Instead, we can have some preprocessing to decide on the most promising rescue vehicles for the initial solution, based for example on the proximity of rescue vehicles to the broken-down vehicle, and then actually employing the heuristic with just 1 or 2 different restarts. Furthermore, the runtime highly depends on the parameter J_1 of the number of iterations without an improvement before termination in each restart. By proper calibration of this parameter, the runtime may be further reduced. Improving the runtime and, perhaps, other aspects of the heuristic, is left open for future research.

4.11 Conclusions

To summarize, in this chapter we presented a heuristic algorithm based on Tabu Search, as a second approach for solving *the disrupted VRP with Vehicle Breakdown*. The heuristic was used to solve the 101 instances that were constructed for the purpose, and the experimental results presented were compared to the results produced by the exact approach which were presented in the previous chapter.

In general, the results show that the heuristic is capable of finding optimal or near-optimal solutions within a few minutes. In many occasions the heuristic solution matched exactly the optimal solution found by AIMMS within the 5 minutes cutoff time, and in many other cases the heuristic solution outperformed the one found by AIMMS. Of course, whenever AIMMS was able to terminate before the 5 minutes cutoff time, it produced a proven optimal solution which no other method can improve. Hence, in some cases the heuristic found a worse solution than the one found by the exact approach.

In general, both methods are important and useful. In practice, we recommend to have both approaches run simultaneously, and take the best solution found within the available time by either one of the two methods (which will differ from case to case). We expect that for small instances and some medium-sized instances with only a few customers originally assigned to the disabled vehicle, the exact approach should be able to provide the proven optimal solution within a few minutes. In some other cases, even if it does not provide the optimal solution or prove its optimality, the exact approach may still provide a feasible solution which sometimes outperforms the heuristic one. However, for larger instances we expect that the heuristic should provide a good quality solution within a few minutes, whereas the exact approach may either provide a much worse solution, or it may not be able to provide a feasible solution at all. Of course, even in cases where the exact approach fails to provide a good quality solution, it may still be very useful, since the lower bound that it provides can be used to establish a guarantee on the optimality gap of the heuristic solution.

Chapter 5

Problem 2 - The Delayed TSPTW: Formulations & Exact Approaches

5.1 Introduction

This chapter introduces a new problem, *the Delayed Traveling Salesman Problem with Time Windows (Delayed TSPTW)* and presents detailed description, notation, formulations and exact approaches for the problem.

Suppose that we have created a feasible and optimal or near-optimal solution plan for a given instance of the Traveling Salesman Problem with Time Windows (TSPTW). Assume that during the execution of this plan, for some reason there is a major delay and the vehicle (i.e. the 'traveling salesman') has fallen behind schedule, so that if the original route is followed as planned, one or more customers will be served with a delay. This causes a disruption to the original plan, which is no longer feasible, since the proposed solution leads to the violation of some time windows.

Obviously, one option is to keep following the original plan and visit the remaining customers in the order defined by the original schedule. However, this could cause a delay in serving too many of the remaining customers. In fact, if no rescheduling takes place, then under certain conditions we may experience an extreme 'domino' effect where, once a customer is served with a delay, then all of the following customers will also be served with a delay. Is it possible to revise the plan and have fewer customers served outside their promised time windows, or perhaps have shorter amounts of deviation from their time windows?

In general, servicing customers outside their promised time windows, may cause substantial direct or indirect costs for the enterprise. Such costs may be incurred, for example, by losing some current customers, or by having some of the dissatisfied customers negatively influencing the company's reputation. It is clear that the original schedule has to be revised in such a way so that the negative effect of the disruption on the enterprise is minimized. In fact, our experiments verify that in many cases we can find a better plan where fewer customers are served with a delay and/or the total amount of violation of their time windows is lower, compared to following the original plan.

Essentially, once such a disruption occurs we are facing a new problem which we will call *the Delayed TSPTW*, that is substantially different from the original TSPTW, with different structure, constraints and objectives. In this chapter we will define this problem and propose mathematical formulations and exact approaches which theoretically can be used to solve the problem to optimality. Heuristic approaches and experimental results will be presented in the following chapter.

5.2 The easy plan

There is a trivial way to construct a solution to the disrupted problem, following the so-called *easy plan* that is described below.

Once a delay occurs, the operational plan may be rescheduled as follows: The vehicle will skip the first few customers and be diverted to visit next that specific customer which, if visited first after rescheduling, then all of the following customers in the order of the original plan will be served within their requested time windows. Then the vehicle will visit the first few customers that were skipped in their original order, and finish at the endpoint node. This way, only the customers that were at the first part of the original route and were skipped, may be served with delay.

In more detail, suppose that the traveling salesman has fallen behind schedule, and assume that $R = (R_1, R_2, ..., R_l)$ is the remaining part of the route at the time of rescheduling. Here, R_1 denotes the starting node (or current position) of the vehicle at the time of rescheduling, and R_l denotes the endpoint node, whereas nodes $R_2, R_3, ..., R_{l-1}$ represent customer nodes. With trivial calculations we can find the smallest index j such that, if the vehicle skips the first j customers and goes directly from its starting node R_1 to its $(j + 1)$-th customer R_{j+2}, then all of the following customers will be served within their promised time windows. In this case, the vehicle can follow the revised route $R' = (R_1, R_{j+2}, R_{j+3}, ..., R_{l-1}, R_2, R_3, ..., R_{j+1}, R_l)$. In other words, it will be redirected from its starting position R_1 to serve the $(j + 1)$-th customer R_{j+2} and all of its following customers $R_{j+3}, ..., R_{l-1}$

within their time windows, in the same order as scheduled in the original plan. Then it will go back to serve the first customer R_2 and all of its following customers $R_3, ..., R_{j+1}$, possibly with some delay, in the same order as scheduled in the original plan, before finishing its route at the endpoint node R_l. This way, at most j customers will be served outside their requested time windows.

In general, following the easy plan should be better than not rescheduling and staying with the original plan. However, rescheduling in a more sophisticated way, may often lead to a new plan where fewer customers are served outside their time windows, and where the total amount of violation of the time windows is lower, compared to following either the original plan or the easy plan. Therefore, in the remaining part of this chapter, as well as in the following chapter, we describe more appropriate methods to solve this rescheduling problem.

5.3 The Delayed TSPTW: Problem description and notation

In this section we will present a detailed description and establish notation for the problem under study, namely *the Delayed Traveling Salesman Problem with Time Windows (Delayed TSPTW)*, or in short *problem 2*.

Assume that we have created a feasible and optimal or near-optimal solution plan for a given TSPTW instance. Suppose that, during the execution stage of this plan, at some time instant T the operations team is informed that there has been a severe delay and that the vehicle has fallen behind schedule. Note that by the term *severe delay* we refer to a delay which causes a violation of the time-window constraint for at least one of the remaining customers of the original route, if this is followed as planned.[1] Therefore, the remaining part of the original plan is no longer feasible with respect to the time windows. Consequently, the plan has to be revised in such a way that the negative effect of the disruption is minimized. The new problem that arises is *the Delayed TSPTW*. The operations team can use the methods proposed in the current chapter and those of the following chapter, in order to construct a sufficiently good revised schedule in real time.

[1]In contrast, it is possible that the vehicle experiences just a *minor delay*. This refers to the case where the vehicle falls slightly behind schedule, but if it continues to follow the original plan, then it can still serve all of the remaining customers within their requested time windows and without any delays. In this case, the driver can simply continue following the route as originally planned, without the need for rescheduling.

The original problem is assumed to be either a regular TSPTW or an open-TSPTW, in which a single vehicle departs from a fixed starting location and is required to serve a given set of customers within their requested time windows, before it returns to a fixed endpoint location. In the first case where the original problem is assumed to be a regular TSPTW, the starting point and endpoint coincide and are referred to as *the depot*. In the second case where the original problem is assumed to be an open-TSPTW, the starting point and endpoint may differ. Since the second case generalizes the first one, we define the Delayed TSPTW in the slightly more general context of the second case; i.e. assuming that in the original problem the starting point and endpoint are both fixed but may differ. Note that the vehicle may be delivering either a single commodity, such as oil or gas, or unique parcels for each customer (customer-specific orders). We also assume that there is no extra vehicle available and that all customers have to be served.

As mentioned above, we assume that during the execution stage of the original plan, at time instant T the operations team is informed that a severe delay has occurred during delivery. Suppose that T_0 is the estimated time needed for the operations team to construct a new plan and communicate this to the driver. The position of the vehicle at time $T + T_0$ can then trivially be estimated. Let N_1 denote the node which represents the expected or estimated position of the vehicle at time $T + T_0$. We wish to define T_1 as the time when the vehicle will be ready to depart from node N_1. There are two possible cases to consider.

In the first case, at time $T+T_0$ the vehicle is expected to be at a customer node. In this case, by time $T + T_0$ the vehicle may be expected to have just arrived at the customer, to have just finished servicing the customer, or to have already started but not finished servicing the customer. In all three possible scenarios of the first case, we define T_1 to be the expected time by which the vehicle will have finished servicing the particular customer (where $T_1 \geq T + T_0$). We also define N_1 to be the node corresponding to the geographic location of that customer.

In the second case, at time $T + T_0$ the vehicle is projected to be on the road, traveling from one node to another. In this case, we can define node N_1 to be a new node that corresponds to the expected geographic location of the vehicle at time $T + T_0$ (if following the route of the original plan), and we also define $T_1 := T + T_0$.

Thus, in any case, the vehicle is expected to be ready to depart from node N_1 at time T_1, where $T_1 \geq T + T_0$. Note that node N_1 and time T_1 can be uniquely defined. In the case where node N_1 represents a customer location, since by time T_1 the particular customer will have already been served, he or she is removed from the set containing the customers that are still unserved by time $T + T_0$ when the revised plan starts. Note that in all cases node N_1 will have zero service time. Also, we assume that $T \geq 0$ and $T_0 \geq 0$; therefore $T_1 \geq 0$ (since $T_1 \geq T + T_0$).

Once the starting node N_1 and the corresponding time T_1 when the vehicle is ready to depart from node N_1 are defined, we can define the set of customers I_C for the revised plan. Specifically, this set contains exclusively all customers that are still unserved by time T_1. Suppose that there are n such customers. We relabel the nodes corresponding to these customers as nodes $1, 2, ..., n$. From now on, we will refer to the set $I_C = \{1, 2, ..., n\}$ as 'the set of customers that are unserved at time T_1', or simply as 'the set of customers' (in the revised plan). Furthermore, from now on, the starting node N_1 will also be denoted as node $n + 1$. Note that in the case where the starting node N_1 corresponds to a customer location in the original plan, then this customer is not included in I_C.

Let node 0 denote the endpoint of the route (depot). Let $I = I_C \cup \{0, n+1\}$ be the set of all nodes, which can be written explicitly as $I = \{0, 1, \ldots, n, n+1\}$. Let $A = \{(i, j) : i, j \in I, i \neq 0, j \neq n+1, i \neq j\}$ be the set of arcs. Clearly $A \subseteq I \times I$. Let $\Delta^+(i) = \{j \in I : (i, j) \in A\}$ be the set of nodes that are directly reachable from node i, for all $i \in I$. Let $\Delta^-(i) = \{j \in I : (j, i) \in A\}$ be the set of nodes from which node i is directly reachable, for all $i \in I$. The Delayed TSPTW is formulated on the directed graph $\Gamma = (I, A)$.

A feasible route for the Delayed TSPTW is any route which starts from node $n + 1$ at time T_1 or later, visits all customers of the set I_C in the appropriate order, and then finishes at the endpoint node 0. The time of start of service of each customer $i \in I_C$ may or may not lie within the associated time window; but in the latter case, the objective is penalized. Therefore, all time windows are treated as soft constraints.

Each node in I_C must have exactly 1 vehicle entering and the same vehicle leaving the node. The starting node $n+1$ must have no vehicles entering and exactly 1 vehicle leaving the node. The endpoint node 0 should have exactly 1 vehicle entering and no vehicles leaving.

Let x_{ij} be a binary variable representing the number of times that the vehicle traverses the arc (i, j), for all $(i, j) \in A$. Let w_i be a nonnegative variable representing the time of start of service of node i, for all $i \in I$. Specifically, w_{n+1} represents the time of departure of the vehicle from node $n+1$, whereas w_0 represents the time of arrival of the vehicle at the endpoint node 0. Obviously, $w_i \geq T_1$ for all $i \in I$.

Let t_{ij} be the travel time from node i to node j, for all $(i, j) \in A$. Assume that $t_{ij} \geq 0 \;\; \forall \; (i, j) \in A$. Let $[a_i, b_i]$ and s_i be the time window and service time associated with node i, respectively, for all $i \in I$. We set $s_i = 0$ for nodes $i \in \{0, n + 1\}$; i.e. the starting node and endpoint node have zero service time.

Let M be a large positive constant (e.g. $M = 10^{10}$) and let ϵ be a small positive constant (e.g. $\epsilon = 10^{-8}$). The quantities n, M, ϵ, T, T_0, T_1, t_{ij}'s, s_i's, a_i's and b_i's are assumed to be known parameters.

For all $i \in I_C$, we introduce the variables μ_i, λ_i, ρ_i and σ_i as follows:
(i) The variable μ_i is a binary variable indicating whether or not customer i is served with a delay. For this, μ_i is equal to 1 if and only if $w_i > b_i$ (i.e. if customer i is served with a delay, that is, later than b_i); otherwise, $\mu_i = 0$ (i.e. $\mu_i = 0$ if and only if $w_i \leq b_i$).
(ii) The variable λ_i is a binary variable indicating whether or not customer i is served earlier than promised. For this, λ_i is equal to 1 if and only if $w_i < a_i$ (i.e. if customer i is served earlier than a_i, that is, earlier than promised); otherwise, $\lambda_i = 0$ (i.e. $\lambda_i = 0$ if and only if $w_i \geq a_i$).
(iii) The variable ρ_i represents the amount of lateness of customer i, if this customer is served with a delay (later than b_i); otherwise, $\rho_i = 0$. For this, ρ_i is equal to $w_i - b_i$ if $w_i > b_i$; otherwise, ρ_i is equal to 0 if $w_i \leq b_i$. Mathematically, this can be expressed as:

$$\rho_i = (w_i - b_i)\mu_i = \begin{cases} 0 & \text{if } w_i \leq b_i \qquad (\Leftrightarrow \mu_i = 0) \\ w_i - b_i & \text{if } w_i > b_i \qquad (\Leftrightarrow \mu_i = 1) \end{cases}$$

(iv) The variable σ_i represents the amount of earliness of customer i, if this customer is served early (earlier than a_i); otherwise, $\sigma_i = 0$. For this, σ_i is equal to $a_i - w_i$ if $w_i < a_i$; otherwise, σ_i is equal to 0 if $w_i \geq a_i$. Mathematically, this can be expressed as:

$$\sigma_i = (a_i - w_i)\lambda_i = \begin{cases} 0 & \text{if } w_i \geq a_i \qquad (\Leftrightarrow \lambda_i = 0) \\ a_i - w_i & \text{if } w_i < a_i \qquad (\Leftrightarrow \lambda_i = 1) \end{cases}$$

5.4 Objectives

The *Delayed TSPTW with 5 objectives* (or in short, the *Delayed TSPTW-5*) is the problem of determining the appropriate sequence in which the customers of the set $I_C = \{1, 2, ..., n\}$ have to be visited by the vehicle, after it departs from the starting position $n + 1$ and before it arrives at the endpoint node 0 (depot), as well as the appropriate times of visit, so that the following 5 goals are achieved: 1) have as few late customers as possible, 2) have as few early customers as possible, 3) have as low amount of total lateness as possible, 4) have as low amount of total earliness as possible, and 5) finish the route as early as possible.

More specifically, we define the *Delayed TSPTW-5* as a Multi-Objective Optimization (MOO) problem with the following 5 objectives:

1. Minimize $F_1 := \sum_{i \in I_C} \mu_i$, i.e. minimize the number of customers served with a delay (later than b_i).
2. Minimize $F_2 := \sum_{i \in I_C} \lambda_i$, i.e. minimize the number of customers served earlier than promised (earlier than a_i).
3. Minimize $F_3 := \sum_{i \in I_C} \rho_i$, i.e. minimize the total amount of lateness (combined for all customers being served with a delay).
4. Minimize $F_4 := \sum_{i \in I_C} \sigma_i$, i.e. minimize the total amount of earliness (combined for all customers being served earlier than promised).
5. Minimize $F_5 := w_0$, i.e. minimize the time of arrival of the vehicle at the endpoint node (the time when the route ends).

5.4.1 Alternatives to objective 5

Note that, when solving other variants of the Delayed TSPTW, as an alternative to objective 5, one may use any one of the following objectives accordingly:

- Minimize $\sum_{(i,j) \in A} t_{ij} x_{ij}$, i.e. minimize the total travel time of the vehicle (that is, not including waiting time).
- Minimize $\sum_{(i,j) \in A} c_{ij} x_{ij}$, i.e. minimize the total travel cost of the route.
- Minimize $(w_0 - w_{n+1})$, i.e. minimize the makespan (the total journey time, including waiting time).

In our study we consider $F_1, F_2, ..., F_5$ as the component objectives. Experimentation with alternative component objectives is left open for future research.

5.5 Preprocessing stage

Before solving the mathematical program proposed below, a preprocessing stage takes place, which identifies some customers that will definitely be served with a delay, as well as some other customers that will definitely not be served earlier than promised.

For this, we define *the set of definitely late customers*, $I_{DL} = \big\{ i \in I_C \,\big|\, T_1 + s_{n+1} + t_{n+1,i} > b_i \big\}$. Its cardinality $|I_{DL}|$ is the number of definitely late customers, which serves as a lower bound for F_1; that is, $LB_{F_1} = |I_{DL}|$. Similarly, we define *the set of definitely not early customers*, $I_{DNE} = \big\{ i \in I_C \,\big|\, T_1 + s_{n+1} + t_{n+1,i} \geq a_i \big\}$. Its cardinality $|I_{DNE}|$ is the number of definitely not early customers.

Additionally, the *minimum amount of lateness*, which serves as a lower bound for F_3, is defined as: $LB_{F_3} := \sum\limits_{i \in I_{DL}} (T_1 + s_{n+1} + t_{n+1,i} - b_i)$.

5.6 General MOMILP Formulation for the Delayed TSPTW with 5 objectives

The more general variant of problem 2, namely *the Delayed TSPTW with 5 objectives (Delayed TSPTW-5)*, can be formulated as the Multi-Objective Mixed Integer Linear Program *(MOMILP 2A)*, composed of equations (5.6.1)-(5.6.24), as follows:

$$\min \;\; F_1 := \sum_{i \in I_C} \mu_i \tag{5.6.1}$$

$$\min \;\; F_2 := \sum_{i \in I_C} \lambda_i \tag{5.6.2}$$

$$\min \;\; F_3 := \sum_{i \in I_C} \rho_i \tag{5.6.3}$$

$$\min \;\; F_4 := \sum_{i \in I_C} \sigma_i \tag{5.6.4}$$

$$\min \;\; F_5 := w_0 \tag{5.6.5}$$

subject to:

$$\sum_{j \in \Delta^+(i)} x_{ij} = 1 \qquad \forall\, i \in I \setminus \{0\} \tag{5.6.6}$$

$$\sum_{i \in \Delta^-(j)} x_{ij} = 1 \qquad \forall\, j \in I \setminus \{n+1\} \tag{5.6.7}$$

$$w_{n+1} \geq T_1 \tag{5.6.8}$$

$$w_i + s_i + t_{ij} - w_j \leq (1 - x_{ij})M \qquad \forall \ (i,j) \in A \tag{5.6.9}$$

$$w_i - b_i \leq M\mu_i \qquad \forall \ i \in I_C \tag{5.6.10}$$

$$b_i - w_i + \epsilon \leq M(1 - \mu_i) \qquad \forall \ i \in I_C \tag{5.6.11}$$

$$\rho_i \leq M\mu_i \qquad \forall \ i \in I_C \tag{5.6.12}$$

$$\rho_i \leq (w_i - b_i) - (1 - \mu_i)(-M) \qquad \forall \ i \in I_C \tag{5.6.13}$$

$$\rho_i \geq (w_i - b_i) - (1 - \mu_i)M \qquad \forall \ i \in I_C \tag{5.6.14}$$

$$a_i - w_i \leq M\lambda_i \qquad \forall \ i \in I_C \tag{5.6.15}$$

$$w_i - a_i + \epsilon \leq M(1 - \lambda_i) \qquad \forall \ i \in I_C \tag{5.6.16}$$

$$\sigma_i \leq M\lambda_i \qquad \forall \ i \in I_C \tag{5.6.17}$$

$$\sigma_i \leq (a_i - w_i) - (1 - \lambda_i)(-M) \qquad \forall \ i \in I_C \tag{5.6.18}$$

$$\sigma_i \geq (a_i - w_i) - (1 - \lambda_i)M \qquad \forall \ i \in I_C \tag{5.6.19}$$

$$\mu_i + \lambda_i \leq 1 \quad \forall \ i \in I_C \tag{5.6.20}$$

$$x_{ij} \in \{0, 1\} \qquad \forall \ (i,j) \in A \tag{5.6.21}$$

$$w_i \geq T_1 \qquad \forall \ i \in I \setminus \{n+1\} \tag{5.6.22}$$

$$\rho_i, \sigma_i \geq 0 \qquad \forall \ i \in I_C \tag{5.6.23}$$

$$\lambda_i, \mu_i \in \{0, 1\} \quad \forall \ i \in I_C \tag{5.6.24}$$

Equations (5.6.1)-(5.6.5) define the 5 component objectives, as described in section 5.4. Equation (5.6.6) ensures that the vehicle exits each node exactly once, apart from the endpoint node (node 0). Equation (5.6.7) ensures that the vehicle enters each node exactly once, apart from the starting node (node $n + 1$). Constraint (5.6.8) states that the time of departure of the vehicle from the starting node $n+1$ is not before time T_1. Constraint (5.6.9) guarantees schedule feasibility with respect to time considerations. Constraints (5.6.10) - (5.6.14) relate any potential violation of the upper limits of the time windows (due dates) with the variables μ_i and ρ_i. Constraints (5.6.15) - (5.6.19) relate any potential violation of the lower limits of the time windows (ready times) with the variables λ_i and σ_i. Constraint (5.6.20) states that no customer can violate both the lower limit and the upper limit of the time windows; i.e. no customer can be both an early and a late customer. Note that constraint (5.6.20) is not necessary for the model; however, it is a valid cut which reduces the feasible region and is therefore included. Finally, constraints (5.6.21) - (5.6.24) give the ranges of the variables.

Note that we allow the vehicle to depart from its starting point at any time on or after time T_1. In practice, in the presence of a delay, it will usually be the case that the vehicle will depart from its starting position precisely at time T_1, without any further delay. Therefore, an alternative but slightly less general way to define the problem, is to make sure that the vehicle departs from node $n + 1$ precisely at time T_1, which can simply be achieved by replacing constraint (5.6.8) with the following equation:

$$w_{n+1} = T_1 \tag{5.6.25}$$

After all, the vehicle can wait upon arrival at its first customer, before the service actually begins. Therefore, these two constraints may produce similar results.

We recognize that, since one of the objectives is to minimize $\sum_{i \in I_C} \mu_i$, constraint (5.6.11) is not needed. Similarly, since one of the objectives is to minimize $\sum_{i \in I_C} \lambda_i$, constraint (5.6.16) is not needed. The same argument applies when minimizing a weighted sum of the five component objectives, with positive weights. However, even if these constraints may be considered as redundant, they are valid cuts. Therefore, we decided to include them as part of the main formulation. Also, they were included when performing our experiments. The task of experimenting both with and without these cuts and assessing the performance, in order to decide which version of the formulation produces results faster, is left open for future researchers.

5.7 A MILP formulation with a single aggregated objective for the Delayed TSPTW-5

One way to address the Delayed TSPTW with 5 objectives, is to aggregate the component objectives into a single objective function, defined as the weighted sum of the 5 component objectives.

More specifically, if in the formulation (MOMILP 2A) of section 5.6 we replace the 5 component objectives (5.6.1)-(5.6.5) with the single objective function defined by equation (5.7.1) shown below, and keep all other equations (5.6.6)-(5.6.24), then we get the Mixed Integer Linear Programming formulation *(MILP 2B)* for the Delayed TSPTW-5, with the following aggregated objective function:

$$\text{minimize} \quad \beta_1 \sum_{i \in I_C} \mu_i + \beta_2 \sum_{i \in I_C} \lambda_i + \beta_3 \sum_{i \in I_C} \rho_i + \beta_4 \sum_{i \in I_C} \sigma_i + \beta_5 w_0 \tag{5.7.1}$$

Of course, equation (5.7.1) can also be written more concisely, either as equation (5.7.2) or as equation (5.7.3):

$$\text{minimize} \quad \sum_{i \in I_C} (\beta_1 \mu_i + \beta_2 \lambda_i + \beta_3 \rho_i + \beta_4 \sigma_i) + \beta_5 w_0 \tag{5.7.2}$$

$$\text{minimize} \quad \sum_{i=1}^{5} \beta_i F_i \tag{5.7.3}$$

The weights β_i are assumed to be known parameters, not necessarily normalized, tuned according to the decision maker's guidelines, where $\beta_i \in [0, +\infty)\ \forall\ i = 1, 2, ..., 5$.

By solving (MILP 2B) with $\beta_l = 1$ and $\beta_r = 0\ \ \forall\ \ r = 1, ..., 5,\ r \neq l$, we can get a lower bound for F_l. By repeating this process for each $l = 1, ..., 5$, we can get lower bounds for each one of the 5 component objectives $F_1, ..., F_5$.

More importantly, by varying the values of the weights β_i's, with $\beta_i > 0$ for all $i = 1, \ldots, 5$ and $\sum_{i=1}^{5} \beta_i = 1$, and solving (MILP 2B) defined before, we can get different non-dominated solutions to the *Multi-Objective Delayed TSPTW with 5 objectives* (Delayed TSPTW-5). This will be further discussed in section 5.10 when describing *Exact Approach 2*.

5.8 Notes

1. The Delayed TSPTW (problem 2) can be considered to be a generalization of the TSPTW. Indeed, if we let $T = T_0 = T_1 = 0$ and node

$n+1$ to represent the same geographic location as node 0 (i.e. copy of the depot), then problem 2 reduces to a TSPTW with the objective to minimize the time of arrival at the depot. Although in problem 2 the time windows are treated as soft constraints, in the presence of a solution that respects all time windows, the last reduction should make the problem equivalent to a TSPTW with hard time windows. Therefore, the Delayed TSPTW is NP-hard.

2. If we let $T = T_0 = T_1 = 0$, without requiring that nodes 0 and $n+1$ represent the same geographic location, then the Delayed TSPTW reduces to an Open-TSPTW with the objective to minimize the time of arrival at the endpoint node.[2]

3. We are assuming that, in the original plan, the time of start of service of customer i was supposed to lie within the time window $[a_i, b_i]$. This time window was a hard constraint in the original problem, but in the new problem after disruption it is treated as a soft constraint.

4. If in the original problem customer i was promised to be served at a specific time instant τ_i^*, instead of a time window, we can define the respective time window as $[a_i, b_i] = [\tau_i^*, \tau_i^*]$. In fact, if in the original problem customer i was supposed to be served at a specific time instant τ_i representing his or her *estimated time of start of service*, and if ξ_1 and ξ_2 are non-negative numbers representing the amount of time allowed to deviate below and above the estimated time of start of service without being considered an early arrival or a delay, respectively, then this can be converted into a time window in the new problem, by setting $[a_i, b_i] = [\tau_i - \xi_1, \tau_i + \xi_2]$.

5. Suppose that the operations team decides that customer i would not mind having a delivery time outside their promised time window, either earlier or later by some specific amount of time. For instance, this decision may be taken after communicating and reaching an agreement with the customer. Then the operations team can expand or update their time window $[a_i, b_i]$ accordingly, and use the updated time window when constructing the revised plan.

6. In our experiments, for the sake of simplicity, we assume that the Cartesian coordinates (X_i, Y_i) of each node $i \in I$ are also given as an input, which are used to calculate the travel time $t_{i,j}$ between each pair of

[2]Note that here, by the term *Open-TSPTW* we refer to the variant of the TSPTW where the starting and endpoint nodes are both fixed but may differ.

nodes i and j as the Euclidean distance between the nodes, i.e. as $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$, rounded to 2 decimal places.

5.9 Exact Approach 1: Lexicographic Exact Approach for the Delayed TSPTW-5

A relatively simple approach to solve the multi-objective optimization problem under study, is to use the Lexicographic Preference method. For this, if we further assume that there is a lexicographic preference of the five component objectives, so that objectives $F_1, F_2, ..., F_5$ are in strict descending order of importance, then the more complex multi-objective problem simplifies to a multi-criteria problem. The objective vectors $(F_1, F_2, ..., F_5)$ are first compared on the most important component, namely F_1, and only in case of equality the next most important component (F_2) is considered, and so on.

Thus, we can solve the *Delayed TSPTW with 5 objectives of Lexicographic Preference*[3] *(Delayed-TSPTW-Lex5)* by using formulation (MILP 2B) of section 5.7 with a single aggregated objective (equation (5.7.1)) and with a single choice of weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ so that $\beta_1 >> \beta_2 >> \beta_3 >> \beta_4 >> \beta_5 > 0$. For instance, one can use the following set of weights: $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (10^{10}, 10^8, 10^4, 1, 10^{-4})$. This can be implemented in an optimization solver (e.g. Cplex or AIMMS) to directly solve different problem instances. We will call this method *Exact Approach 1* for solving Problem 2.

We implemented formulation (MILP 2B) in AIMMS and used Exact Approach 1 to solve a number of instances which were created for the problem under study. Specifically, we created a dataset of 27 instances for the Delayed TSPTW, with a number of active customers ranging from 5 to 84. The experimental results of Exact Approach 1 are presented in table 6.2 and discussed in detail in subsection 6.5.1 of the following chapter.

In short, Exact Approach 1 was able to provide the optimal solution within 10 minutes in just 6 out of 27 instances, where the number of active customers was very small (up to 15 customers). Also, AIMMS was unable to produce a solution with an optimality gap of less than 10% in any of the instances that involved more than 40 active customers, within 10 minutes. The results imply that this method cannot be used in practice for solving instances of more than 40 customers within a few minutes. Instead, a heuristic method may be more appropriate in such cases. More details on the dataset, on the results of Exact Approach 1, as well as comparisons between this approach and the heuristic approaches can be found in section 6.5 of the

[3]or the *Lexicographic Delayed TSPTW with 5 objectives*

following chapter.

5.10 Exact Approach 2: Weighted Sum Exact Approach for the 5-objective variant

A second and more general exact approach to solve the problem under study, is to use the Weighted Sum Method, without making any assumption about any preference regarding the 5 component objectives.

Thus, we can solve the *Multi-Objective Delayed TSPTW with 5 objectives (Delayed-TSPTW-MOO-5*, or simply *Delayed-TSPTW-5)* by using the weighted-sum method and formulation (MILP 2B) of section 5.7. As before, a single aggregated objective (equation (5.7.1)) is involved. However, instead of simply using a single choice of weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ as done in Exact Approach 1, in this approach we consider many different sets of weights. More specifically, we vary systematically the values of the weights $\beta_1, \beta_2, ..., \beta_5$ with $\beta_i > 0$ for all $i = 1, \ldots, 5$, and for each set of weights we solve (MILP 2B); thus getting several different solutions to the MOO problem, all of which will belong to the Pareto front. This can be implemented in an optimization solver to directly solve different instances. We will be referring to this method as *Exact Approach 2* for solving Problem 2. Note that the weights may be normalized (i.e. to have $\sum_{i=1}^{5} \beta_i = 1$), or not.

Note that, in order to solve a single instance using Exact Approach 2, we must solve an adequate number of programs of the form (MILP 2B), each time with a different combination of weights. This way, we can get a subset of the set of non-dominated solutions. However, in practice we would probably need to solve too many MILPs, choosing some of the too many possible combinations, and even then there is no guarantee that we will get a representative subset of the Pareto front.

Furthermore, since Exact Approach 1 was unable to provide an acceptable solution for instances with more than 40 active customers in short time, it is reasonable to expect similar or even worse results with the more computationally expensive Exact Approach 2. For this reason we decided not to perform any experiments with exact approach 2, nor with any other exact methods. Instead, heuristic approaches seem more promising for solving instances with more than 40 customers. We will explore the use of different heuristic approaches for this problem in the chapter that follows.

5.11 Problem extension with customer priorities

The Delayed TSPTW can be extended to include different priorities for different customers. This additional feature may be used in cases where the company has some customers who are considered to be more important than others, and would prefer that they are affected the least in case of a disruption.

For this, we define $p_i > 0$ to be a known parameter representing the priority of customer i, for all $i \in I_C$. The more important a customer is, the larger the value of the associated parameter p_i should be, and thus the more a potential violation of his or her time window should be penalized in the objective(s). For example, a simple priority assignment can be achieved by dividing up the customers into high-priority customers with p_i=2, and low-priority customers with p_i=1, as done in Jiang et al. (2013).

In order to incorporate customer priorities into our models, in the more general formulation *(MOMILP 2A)* of section 5.6, we can replace the 5 component objectives (5.6.1) - (5.6.5) with the following 5 equations:

$$\min \; F_1^p := \sum_{i \in I_C} p_i \mu_i \tag{5.11.1}$$

$$\min \; F_2^p := \sum_{i \in I_C} p_i \lambda_i \tag{5.11.2}$$

$$\min \; F_3^p := \sum_{i \in I_C} p_i \rho_i \tag{5.11.3}$$

$$\min \; F_4^p := \sum_{i \in I_C} p_i \sigma_i \tag{5.11.4}$$

$$\min \; F_5^p := w_0 \tag{5.11.5}$$

Likewise, if we wish to incorporate customer priorities into formulation *(MILP 2B)* of section 5.7, we can replace the single aggregated objective function (5.7.1), with the following more general objective: [4]

$$\text{minimize} \;\; \beta_1 \sum_{i \in I_C} p_i \mu_i + \beta_2 \sum_{i \in I_C} p_i \lambda_i + \beta_3 \sum_{i \in I_C} p_i \rho_i + \beta_4 \sum_{i \in I_C} p_i \sigma_i + \beta_5 w_0 \tag{5.11.8}$$

[4]Obviously, equation (5.11.8) can be written more concisely as either one of the following two equations:

$$\text{minimize} \;\; \sum_{i \in I_C} p_i(\beta_1 \mu_i + \beta_2 \lambda_i + \beta_3 \rho_i + \beta_4 \sigma_i) + \beta_5 w_0 \tag{5.11.6}$$

Note that, starting from the more general models presented above that involve customer priorities, if we simply set $p_i = 1$ for all $i \in I_C$, then these reduce down to models *(MOMILP 2A)* and *(MILP 2B)* without priorities.

No experiments were performed using these generalized formulations which include customer priorities. This is left open for future research.

5.12 Reduction to 3 objectives

In general, for a multi-objective optimization (MOO) problem, the greater the number of objectives, the more complex and time-consuming it is to find the set of non-dominated solutions, to visualize the Pareto front and to decide amongst the (possibly many) alternatives. For instance, finding the set of non-dominated solutions of a MOO problem with 5 objectives, is significantly more complex than for a MOO problem with 3 objectives.

For this, the 5 objectives of this MOO problem can be reduced down to 3 objectives, if we make the following two additional assumptions: (i) that having one early customer is equally as bad as having one late customer, and (ii) that having a delay in service of χ time units is equally as bad as having an early start of service by χ time units.

Under the above additional assumptions, the Delayed TSPTW with 5 objectives (Delayed-TSPTW-5) reduces to *the Delayed TSPTW with 3 objectives (Delayed-TSPTW-3)*. The latter is a MOO problem which is very similar to the former one, but instead of having 5 objectives, it only involves the following 3 objectives:

1. Minimize $f_1 := F_1 + F_2 = \sum_{i \in I_C} (\mu_i + \lambda_i)$, i.e. minimize the number of customers served outside their promised time windows.
2. Minimize $f_2 := F_3 + F_4 = \sum_{i \in I_C} (\rho_i + \sigma_i)$, i.e. minimize the total amount of time by which the time windows are violated.
3. Minimize $f_3 := F_5 = w_0$, i.e. minimize the time of arrival of the vehicle at the endpoint node (i.e. minimize the time when the route finishes).

The Delayed-TSPTW-3 can therefore be formulated as the Multi-Objective Mixed Integer Linear Program *(MOMILP 2C)*, composed of all constraints (5.6.6)-(5.6.24) of (MOMILP 2A) presented in section 5.6, but with

$$\text{minimize} \quad \sum_{i=1}^{5} \beta_i F_i^p \qquad (5.11.7)$$

the following three objectives:

$$\min \ f_1 := \sum_{i \in I_C} (\mu_i + \lambda_i) \tag{5.12.1}$$

$$\min \ f_2 := \sum_{i \in I_C} (\rho_i + \sigma_i) \tag{5.12.2}$$

$$\min \ f_3 := w_0 \tag{5.12.3}$$

Furthermore, the Delayed-TSPTW-3 can also be solved using the weighted-sum method, the same way that Delayed-TSPTW-5 can be solved, if in equation (5.7.1) we let $\beta_1 = \beta_2$, $\beta_3 = \beta_4$, and define $(\beta'_1, \beta'_2, \beta'_3) = (\beta_1, \beta_3, \beta_5)$, which will transform it into the following equation:

$$\text{minimize} \ \ \beta'_1 \sum_{i \in I_C} (\mu_i + \lambda_i) + \beta'_2 \sum_{i \in I_C} (\rho_i + \sigma_i) + \beta'_3 w_0 \tag{5.12.4}$$

or equivalently

$$\text{minimize} \ \ \sum_{i=1}^{3} \beta'_i f_i \tag{5.12.5}$$

For this, starting from formulation (MOMILP 2C) presented above, if we replace the three objectives (5.12.1) - (5.12.3) with equation (5.12.5) (or its equivalent (5.12.4)) and keep all other equations (5.6.6)-(5.6.24), we get the single-objective formulation *(MILP 2D)* for the Delayed-TSPTW-3.

Then, formulation (MILP 2D) can be used to solve the 3-objective variant of the problem, i.e. the Delayed-TSPTW-3, either assuming a Lexicographic Preference (i.e. as in Exact Approach 1 of section 5.9), or as a MOO problem using the Weighted Sum approach and varying systematically the set of weights (i.e. as in Exact Approach 2 of section 5.10).

5.13 Application: The disrupted VRPTW with customer-specific orders and vehicle delay

A direct application of the Delayed TSPTW is to solve *the disrupted VRPTW with customer-specific orders and vehicle delay*, which is described below.

Suppose that we have created a feasible and optimal or near-optimal solution plan for a given instance of the multi-commodity VRPTW with heterogeneous fleet (i.e. with vehicles of different capacities, in general).

Assume that, during the execution of this original solution plan, a severe delay occurs in one or more of the routes, and if the original plan is not revised then there will be at least one customer served with a delay in each one of the affected routes. This disruption causes the remaining part of the original plan to be infeasible with respect to the time windows. Clearly the operational plan has to be revised appropriately, so that the negative effect of the disruption is minimized.

We assume that the goods delivered are non-transferable between customers, that the orders are customer-specific, that there are no split deliveries, that there are no extra vehicles available, that vehicles cannot meet and exchange goods between each other and that vehicles cannot return to the depot to reload goods and leave the depot again. In fact, we assume that once the vehicles depart from the depot, even after a severe delay has occurred in some of the routes, each customer can only be served by the vehicle that was originally planned to serve him or her, which already carries his or her order.

In this problem, if a severe delay occurs in ν routes, each one of these routes can be revised independently from one another. Each one of those routes will be a path that starts from the current position of the vehicle, visits the remaining customers of the route (perhaps in a different order than originally scheduled) and finishes at the depot. Specifically, assume that at time τ_1 the operations team is informed that there is a severe delay in route r_1, at time τ_2 that there is a severe delay in route r_2,..., and at time τ_ν that there is a severe delay in route r_ν. Solving *the disrupted VRPTW with customer-specific orders and vehicle delay* is equivalent to solving ν different *Delayed-TSPTW* problems, one for each route r_i, where each time we set $T = \tau_i$, for all $i = 1, \ldots, \nu$. Obviously, any route which does not experience a delay will not be affected.

Therefore, once we have completed the study of problem 2, i.e. the Delayed TSPTW, this will imply that the study of *the disrupted VRPTW with customer-specific orders and vehicle delay* has also been concluded.

5.14 Other applications: other types of disruption

Apart from delays, the models described in this chapter may be used to handle other forms of disruption which may occur, either in a TSPTW, or in a multi-commodity VRPTW with customer-specific orders. Examples include the following types of disruption:

- one or multiple roads/arcs blocked, or more generally parts of the network blocked, affecting the routes of some delivery vehicles. For instance, this may occur due to an accident which does not involve a delivery vehicle, due to a landslide etc.
- traffic congestion in some parts of the network, which was not taken into account or was not known when constructing the original plan.

In each one of the above cases, the formulations and models of this chapter can be used, with the only difference that the operations team has to update the travel times accordingly before rescheduling, as follows: If arc (i, j) is blocked, then we should set $t_{i,j} = \infty$. If there is traffic congestion and the travel time $t_{i,j}$ has increased, then $t_{i,j}$ should be updated using a new estimated travel time $t'_{i,j}$, or an overestimate if it is impossible to predict the travel jam effect on arc (i, j). Historical data of travel times under traffic congestion may be used, if available and if this is the type of disruption experienced.

5.15 Other problem variants and solution methods, scope for further research and conclusions

It is possible to extend the formulations described in this chapter to include other constraints, such as certain customers being served by particular vehicles, if this is relevant for a particular application. Additionally, other variants of problem 2 can arise if we consider a different set of objectives.

For instance, there are no restrictions on the amounts of delay or earliness in the models described in this chapter. However, if desired, such restrictions can be easily incorporated to the current models as follows:
(a) If we want to have a maximum delay of ρ_{max} per customer, we can add the constraint $\rho_i \leq \rho_{max} \;\; \forall\, i \in I_C$ (or for a specific subset of I_C).
(b) Similarly, if we want to have a maximum earliness of σ_{max} per customer, we can include the constraint $\sigma_i \leq \sigma_{max} \;\; \forall\, i \in I_C$ (or for a subset of I_C).
(c) If we want to have a maximum delay of ρ'_{max} combined for all customers, we can add the constraint $\sum_{i \in I_C} \rho_i \leq \rho'_{max}$.
(d) Likewise, if we want to have a maximum earliness of σ'_{max} combined for all customers, we can include the constraint $\sum_{i \in I_C} \sigma_i \leq \sigma'_{max}$.

Obviously there are many different variants that can occur from the problem under study, by considering different combinations of constraints and objectives. Additionally, each one of these variants may be solved using different methods. In this chapter we only considered the lexicographic approach and the weighted-sum approach as exact methods to address the 5-objective variant of problem. Apart from these methods, of course, one can apply other multi-objective optimization methods to address each variant of the problem, such as the Epsilon Constraint Method.

Since the experimental results of Exact Approach 1 indicate that exact approaches based on the formulations presented in this chapter are not promising for quickly solving instances with more than 40 customers, we decided not to perform any experiments using other exact approaches. Therefore, we leave open for further research the task of experimenting with other multi-objective optimization methods to find the set of non-dominated solutions in an exact approach. Instead, in the following chapter we focus on heuristic approaches that may be more appropriate for solving such problems in practice. Note also that in the following chapter we do consider another MOO method, namely the Epsilon Constraint Method, to solve the 3-objective variant of problem 2; however, we use this method in a heuristic framework.

Chapter 6

Problem 2 - The Delayed TSPTW: Heuristic Approaches & Experimental Results

6.1 Introduction

In chapter 5 we introduced *the Delayed Traveling Salesman Problem with Time Windows* (*Delayed TSPTW*), also referred to as *Problem 2*, and presented a detailed description, notation, formulations and exact approaches for the problem.

In this chapter we describe three heuristic approaches for the Delayed TSPTW. The first heuristic approach solves the Lexicographic variant of the problem with 5 objectives (the *Delayed-TSPTW-Lex5*). The second one solves the Lexicographic variant with 3 objectives, which is a slight simplification of the first variant. Both approaches 1 and 2 assume a lexicographic preference in the component objectives, transforming the problem into one with a single aggregated objective, and eventually find a single optimal or near-optimal solution. Finally, the third heuristic approach uses the framework of the ϵ-Constraint Method to solve heuristically the 3-objective version of the problem, and finds a representative subset of the set of non-dominated solutions, as a more general multi-objective optimization (MOO) approach and without assuming any preference between the component objectives. All three approaches employ the Tabu Search metaheuristic.

Although the second heuristic approach can be thought of as a special case of the first one, we mainly describe it separately for two reasons: Firstly, it serves as a bridge so that valid comparisons between the first and third heuristic approach can be made. Secondly, we need a separate statement of the algorithm corresponding to the second heuristic approach, since this is called several times by the algorithm of the third heuristic approach; otherwise, it would be even more complicated to describe the already complex algorithm of heuristic approach 3.

In the following three sections of this chapter we describe the three heuristic approaches. Then, in section 6.5 we present the experimental results for heuristic approaches 1 and 3, as well as the results derived from exact approach 1, which was described in the previous chapter. Discussion of the results and comparisons between the different methods can be found in the same section.

Throughout this chapter we use the notation established in the previous chapter, unless it is explicitly stated otherwise.

6.2 Heuristic Approach 1: A Tabu Search heuristic for the Lexicographic Delayed TSPTW with 5 Objectives

In this section we describe a heuristic based on Tabu Search, to solve the *Lexicographic Delayed TSPTW with 5 objectives (Delayed-TSPTW-Lex5)*. We will be referring to this approach as *Heuristic Approach 1* for problem 2.

The Delayed-TSPTW-Lex5 was described in section 5.9. In this variant of problem 2, the five component objectives F_1, F_2,..., F_5 which were defined in section 5.4, are transformed into a single aggregated objective, namely $\sum_{i=1}^{5} \beta_i F_i$, which is to be minimized. Essentially we employ the weighted-sum method to transform the MOO problem into a single-objective optimization problem. Moreover, we choose a set of weights $(\beta_1, \beta_2, ..., \beta_5)$ with $\beta_1 >> \beta_2 >> ... >> \beta_5 > 0$. This way we enforce the lexicographic preference of the five component objectives, where the most important component F_1 is minimized first, then among the solutions which involve the minimum value of F_1, the one(s) which also minimize F_2 are chosen, and so on.

Recall that in the Delayed TSPTW, we have n unserved customers waiting to be served by a single vehicle, the so-called 'traveling salesman'. The vehicle must depart from node $n + 1$ (initial position) at a time not earlier than T_1, then visit all n customers, corresponding to nodes $1, 2, ..., n$ in the appropriate order and at the appropriate times, and finish at the endpoint node $n + 2$ (depot). Note that for technical reasons, the endpoint node in this chapter is denoted by node $n + 2$ (whereas in the previous chapter, this was denoted by node 0). All nodes must be visited exactly once. Each node $i = 1, 2, ..., n + 2$ is associated with its Cartesian coordinates (X_i, Y_i), a service time s_i and a time window $[a_i, b_i]$. All customers must be served, but not necessarily within their time windows. Instead, customers' time windows are treated as soft constraints, so any intermediate or final feasible solutions may violate some time windows. Note that any violation of the original time windows of the starting node $n+1$ or the endpoint node $n+2$ will not be reflected in the component objectives F_1, F_2, F_3 or F_4. The starting position $n+1$ can be the same as the depot, or a customer's node (in which case the corresponding customer should not be included in the customer set $\{1, 2, ..., n\}$), or any other position specified by its Cartesian coordinates (X_{n+1}, Y_{n+1}), with zero service time and time windows equal to $[0, +\infty)$. Since each instance of the Delayed TSPTW is based on a feasible solution for a TSPTW instance, this implies that the capacity constraint is automatically satisfied at all times.

Note that some obvious bounds for the first two component objectives are: $0 \leq F_1 \leq n$, $0 \leq F_2 \leq n$, and $0 \leq F_1 + F_2 \leq n$.

6.2.1 Objective

The objective in the first heuristic approach is to minimize function G, defined below:

$$minimize \quad G = \sum_{i=1}^{5} \beta_i F_i \tag{6.2.1}$$

6.2.2 Solution Notation

We define the following:

- $R = (R_1, R_2, ..., R_{n+2})$ is a vector of length $n+2$ which represents the vehicle's route. Vector R starts with node $n+1$ (i.e. $R_1 = n+1$), followed by a permutation of the set of customers {1,2,...,n} and ends with node $n+2$, which corresponds to the depot (i.e. $R_{n+2} = n+2$).

- $\widetilde{W} = (\widetilde{W}_1, \widetilde{W}_2, ..., \widetilde{W}_{n+2})$ is a vector of length $n+2$ which contains the *time of start of service* $\widetilde{W}_i$ of each node i $(i = 1, 2, ..., n+2)$, when the vehicle follows route R and *while idle waiting time at nodes is not allowed*. Of course, specifically for the starting and endpoint nodes, $\widetilde{W}_{n+1}$ represents the time of departure from node $n+1$ and $\widetilde{W}_{n+2}$ the time of arrival at node $n+2$ (since $s_i = 0$ for the nodes $i = n+1, n+2$ which do not represent customers).

- $\widetilde{G}$ is a real number representing the value of the objective (6.2.1) *without allowing idle waiting times at nodes*, when following route R and visiting the customers at times $\widetilde{W}$. Essentially, $\widetilde{G}$ represents the value of G over a subset Π_1 of the feasible region Π, in which no waiting times are allowed at any node. More formally, $\widetilde{G}$ is the restriction of the function G over the set Π_1; i.e. $\widetilde{G} = G|_{\Pi_1}$. So, $\widetilde{G}$ is also calculated by equation (6.2.1).

- $W = (W_1, W_2, ..., W_{n+2})$ is a vector of length $n + 2$ which contains the time of start of service W_i of each node i ($i = 1, ..., n + 2$), when the vehicle follows route R and *while idle waiting time at nodes is allowed.*

- G is a real number representing the value of the objective (6.2.1) *when allowing idle waiting time at nodes*, when visiting the customers in the order described by the collection of routes R and at times W. So, G is calculated over the whole feasible region Π.

A solution to the problem can be described by vector R representing the route, along with the respective vector W containing the time of start of service of each node while allowing idle waiting time at nodes, together with the respective values for the 5 component objectives $F_1, F_2, ..., F_5$. However, in the context of the heuristic and throughout this chapter, whenever we use the term *solution* S we will be referring to the quintuple $S = (R, \widetilde{W}, \widetilde{G}, W, G)$.

In a similar way, we define the quintuples S^1, S^2, S^{best} and S^* as: $S^1 = (R^1, \widetilde{W}^1, \widetilde{G}^1, W^1, G^1)$, $S^2 = (R^2, \widetilde{W}^2, \widetilde{G}^2, W^2, G^2)$, $S^{best} = (R^{best}, \widetilde{W}^{best}, \widetilde{G}^{best}, W^{best}, G^{best})$ and $S^* = (R^*, \widetilde{W}^*, \widetilde{G}^*, W^*, G^*)$. From now on, whenever we say that a solution S, S^1, S^2, S^{best} or S^* is created or updated, we will imply that the respective quintuple is created or updated accordingly, unless otherwise stated.

It is requested to find the Solution S^* which minimizes the objective function G defined by equation (6.2.1), over the whole feasible space Π, i.e. when allowing idle waiting time at nodes, while respecting the necessary constraints.

6.2.3 Description of the main algorithmic functions

Given a route R, we can calculate or update the complete solution S, composed of the quintuple $(R, \widetilde{W}, \widetilde{G}, W, G)$, using the following algorithmic functions:

(i) Given R, how to calculate vector $\widetilde{\mathbf{W}}$ using the algorithmic function *time*: $\widetilde{\mathbf{W}} = \textit{time}(\mathbf{R})$:

Given a route R, the algorithmic function *time* calculates the times of start of service at each node when following route R, if there is no idle waiting time at any node, and stores the result in the vector $\widetilde{W}$ of length $n+2$ (i.e. $\widetilde{W} = time(R)$).

For this, it sets $\widetilde{W}(R_1) = T_1$ for the starting node R_1 of the route, and then for all $i = 2, 3, \ldots, n+2$, it sets $\widetilde{W}(R_i) = \widetilde{W}(R_{i-1}) + t(R_{i-1}, R_i) + s(R_{i-1})$. Note that we use the notation $\widetilde{W}(R_i)$ and $\widetilde{W}_{R_i}$ interchangeably; i.e. $\widetilde{W}(R_i) = \widetilde{W}_{R_i}$. Similarly, $t(R_i, R_j) = t_{R_i,R_j}$ and $s(R_i) = s_{R_i}$.

(ii) Given R and $\widetilde{\mathbf{W}}$, how to calculate $\widetilde{\mathbf{G}}$ using the algorithmic function *cost*: $\widetilde{\mathbf{G}} = \mathbf{cost}(\mathbf{R}, \widetilde{\mathbf{W}})$:

Given R and $\widetilde{W}$, the algorithmic function *cost* uses equation (6.2.1) to calculate the value $\widetilde{G}$ of the objective function without allowing idle waiting times at nodes, when following route R and visiting the nodes at times $\widetilde{W}$.

This function compares the time of start of service $\widetilde{W}_i$ of each node i with the respective time window $[a_i, b_i]$ and thus calculates the value of each component objective F_j $(j = 1, \ldots, 5)$. Initially each F_j is set to zero. Then, for all nodes $i = 1, 2, \ldots n$ that represent customers, if $\widetilde{W}_i > b_i$, then F_1 is increased by 1 and F_3 is increased by $\widetilde{W}_i - b_i$. Similarly, for all $i = 1, \ldots n$, if $\widetilde{W}_i < a_i$, then F_2 is increased by 1 and F_4 is increased by $a_i - \widetilde{W}_i$. Finally, F_5 is defined as $\widetilde{W}_{n+2}$. Therefore, the values of all five component objectives F_j $(j = 1, \ldots, 5)$ are defined for the specific vector $\widetilde{W}$ and route R.

Finally, the function calculates the corresponding value of the objective function (6.2.1), i.e. $\sum_{i=1}^{5} \beta_i F_i$, and returns this as $\widetilde{G}$, i.e. $\widetilde{G} = cost(R, \widetilde{W})$.

Note that, more generally, given R and a vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes (for example, we may use $\widetilde{W}$ or W as $\widehat{W}$), we can still use the same algorithmic function *cost* to calculate the respective value of the objective function $\widehat{G}$, as $\widehat{G} = cost(R, \widehat{W})$. Specifically, given R and the times of start of service W while allowing idle waiting times at nodes, we can use the same algorithmic function *cost* to calculate G, as $G = cost(R, W)$.

Up to this point we have described how, given a route R, we can create or update the triplet $(R, \widetilde{G}, \widetilde{W})$. We will now describe how, given the triplet $(R, \widetilde{G}, \widetilde{W})$, we can use the algorithmic function *optimal waiting times* to calculate G and W and thus get the complete solution $S = (R, \widetilde{G}, \widetilde{W}, G, W)$.

Note that the triplet (R, G, W) is enough to represent a complete solution, since $\widetilde{G}$ and $\widetilde{W}$ serve as intermediate values. Therefore, once we have the optimal values for the triplet (R, G, W), there is no need to update $\widetilde{G}$ or

$\widetilde{W}$.

(iii) Given *R* and $\widetilde{\mathbf{W}}$, how to calculate *W* and *G* using the function *optimal waiting times*: *[W, G]=optimal_waiting_times*$(\mathbf{R}, \widetilde{\mathbf{W}})$:

Given R and $\widetilde{W}$, the function *optimal waiting times* calculates a heuristic approximation of the optimal times of start of service W and the respective value of the objective function G involved when following the route R and while idle waiting time at nodes is allowed. The function works as described below:

- First initialize W and G by letting $W := \widetilde{W}$ and $G := cost(R, W)$. Then, for all $i = 2, 3, ..., n + 1$, perform the procedure described in the following paragraph:
- Check if by visiting the customers at times W, then the customer stored at position R_i of the route R will be visited earlier than his or her ready time. If the answer is yes, then define m_1 as the amount of earliness of customer R_i; that is $m_1 := a_{R_i} - W_{R_i}$. Also, define $m_2 := min\{b_{R_k} - W_{R_k} | k = i+1, i+2, \ldots, n+1\}$; that is, m_2 is the minimum difference between the due date and the time of start of service, amongst the customer nodes following the node R_i in the route R. Note that m_2 is not necessarily positive. In fact, m_2 is negative if there is at least one late customer among the customers R_{i+1}, $R_{i+2}, \ldots, R_{n+1}$. Then, define $m := min\{m_1, m_2\}$. If $m > 0$, then set $W_{temp} := W$. Add a waiting time of m time units to customer R_i in the vector W_{temp}. This is done by increasing the times of visit of all nodes including and following node R_i (i.e. nodes $R_i, R_{i+1}, ..., R_{n+2}$) by m units, in vector W_{temp}. Then calculate $G_{temp} := cost(R, W_{temp})$. If $G_{temp} < G$, this means that adding a waiting time of m time units to customer R_i gives an improvement to the current times of start of service (vector W), with respect to the objective function. Therefore, if $G_{temp} < G$, then update $W := W_{temp}$, $G := G_{temp}$ and continue to the next iteration of the index i.
- Finally, return G and W.

Note that within the function *optimal waiting times*, vector $\widetilde{W}$ may be replaced by $\widehat{W}$, which may or may not involve some idle waiting times at nodes.

6.2.4 Description of the Algorithm

1. Parameters input & Initialization:
   - Input: n, T_1, TL (Tabu Length), d=number of different restarts of the algorithm, weights $(\beta_1, \beta_2, ..., \beta_5)$.
   - Input: (X_i, Y_i), $[a_i, b_i]$, s_i, $\forall\ i = 1, 2, \ldots, n + 2$.

- Input: J_1, J_2, J_3, $\hat{d}$. Here J_1 and J_2 denote the number of Tabu Search iterations without an improvement before Stages A and B terminate, respectively. Also, J_3 and $\hat{d}$ are parameters controlling whether or not in the current iteration the two-opt moves will be considered.
- For all $i, j \in \{1, 2, \ldots, n+2\}$, calculate the travel time $t_{i,j}$ as the Euclidean distance $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ between each pair of nodes (i, j), which is first truncated into an integer using the floor function (as done in Dumas et al. (1995)), and then modified by applying the *Triangle Inequality Correction* as follows:

```
for all $i, j, k \in \{1, 2, \ldots, n+2\}$
    if $t_{i,j} > t_{i,k} + t_{k,j}$
        $t_{i,j} := t_{i,k} + t_{k,j}$
    end if
end for
```

2. Calculate lower bounds for component objectives F_1 and F_3:
- Identify the set of *Definitely Late customers*, $I_{DL} = \left\{ i \in I_C \;\middle|\; T_1 + s_{n+1} + t_{n+1,i} > b_i \right\}$, as described in chapter 5. Its cardinality $|I_{DL}|$ is the number of definitely late customers, which serves as a lower bound for F_1; that is, $LB_{F_1} = |I_{DL}|$.
- Additionally, in some cases we can raise the lower bound LB_{F_1}, by identifying the set of *Additional Late customers*, I_{AL}. This uses a similar reasoning as before, to identify pairs of customers (i, j) not belonging to I_{DL}, among which at least one will be served with a delay. This is achieved by the following subroutine:

```
$I_{AL} := \emptyset$
for $i = 1, 2, \ldots, n-1$
  if $(T_1 + s_{n+1} + t_{n+1,i} \leq b_i)$
    for $j = i+1, i+2, \ldots, n$
      if $(T_1 + s_{n+1} + t_{n+1,j} \leq b_j)$
        if $[(T_1 + s_{n+1} + t_{n+1,i} + s_i + t_{i,j} > b_j)$ &
          & $(T_1 + s_{n+1} + t_{n+1,j} + s_j + t_{j,i} > b_i)]$
            $\Rightarrow I_{AL} := I_{AL} \cup \{(i, j)\}$
        end if
      end if
    end for
  end if
end for
```

```
        return $I_{AL}$
```

In cases where the set I_{AL} is not empty, we can increase the lower bound LB_{F_1} accordingly. For example, if I_{AL} contains only one pair of customers (i, j), then we set $LB_{F_1} = |I_{DL}|+1$. If $I_{AL} = \{(i_1, i_2), (i_3, i_4)\}$ with i_1, i_2, i_3, i_4 all different, then we set $LB_{F_1} = |I_{DL}| + 2$. However, if $I_{AL} = \{(i_1, i_2), (i_1, i_3)\}$ with i_1, i_2, i_3 different between each other, then we can only set $LB_{F_1} = |I_{DL}| + 1$.

From now on, we will assume that LB_{F_1} is enhanced, whenever possible, by using the above subroutine and thus taking into account any *additionally late customers*. Note that the above subroutine can be further generalized.

- Evaluate the *minimum amount of lateness*, defined as: $LB_{F_3} := \sum\limits_{i \in I_{DL}} (T_1 + s_{n+1} + t_{n+1,i} - b_i)$, which serves as a lower bound for F_3.

Objective:

The objective is to minimize function G defined by equation (6.2.1), over the whole feasible region Π, as was already explained.

For diversification purposes, we perform d different restarts of the Tabu Search, each time starting from a different initial solution.

Therefore, for all $i = 1, 2, ..., d$ repeat steps 3 and 4 below to find the *best solution found in run i*, $S^{best}(i)$:

3. Construct the Initial Route R^0:

   For run $i = 1$, the initial route is constructed as follows: We sort all nodes representing customers (after removing served customers) in increasing order of b_i's and store these in the row vector I_S. The initial route R^0 is defined as: $R^0 = [n + 1, I_S, n + 2]$.

   For runs $i = 2, 3$ in the case where $2 \leq d \leq 3$, or for runs $i = 2, 3, ..., d-1$ in the case where $4 \leq d \leq 6$, or for runs $i = 2, 3, ..., d - 3$ in the case where $d \geq 7$, in order to construct the initial route R^0 we start with the best solution found in run $i-1$. We then perform k_1 random moves (exchange, insertion or two-opt), where k_1 is chosen separately for each run i as a random integer in the interval $[k_2,\ k_3]$. Here k_2 and k_3 are input parameters defined by the user (throughout our experiments we used $k_2 = 11$ and $k_3 = 31$).

For run $i = d$ in the case where $4 \leq d \leq 6$, or for runs $i = d-2, d-1, d$ in the case where $d \geq 7$, the initial route is defined as $R^0 = [n+1, I_S^*, n+2]$, where I_S^* is a random permutation of the set of customers {1,2,...,n}.

For each run $i = 1, 2, ..., d$ separately, once the initial route $R^0 = (R_1^0, R_2^0, ..., R_{n+2}^0) = (n+1, R_2^0, R_3^0, ..., R_{n+1}^0, n+2)$ is defined, we calculate the respective full initial solution $S^0 = (R^0, \widetilde{W}^0, \widetilde{G}^0, W^0, G^0)$. We also let $S := S^0$ and $S^{best}(i) := S^0$.

4. - Initialize the *current solution* S^2, by letting $S^2 := S^0$.

   - Apply Tabu Search with best improvement to find the best solution of run i, $S^{best}(i)$, as described in Stages A and B which follow. Throughout the search, three types of neighborhood operators are used: exchange, insertion and two-opt. However, the computationally more expensive two-opt moves are only considered while there was an improvement in the last J_3 iterations, or if the number of iterations performed so far is a multiple of $\hat{d}$ (we used $\hat{d} = 19$ in our experiments).

   - Tabu Search - Stage A:
     - Start with solution S^2.
     - Find the best solution S^1 in a neighborhood $N(S^2)$ of S^2, with respect to the *objective function*. The neighborhood $N(S^2)$ of S^2 is constructed starting from S^2 and performing any single possible move, using one of the following three neighborhood operators: exchange, insertion and two-opt (although the two-opt move is only considered in some cases, as was previously explained).
     - To avoid going around in circles, moves involving nodes that were involved in any of the previous TL moves are declared tabu and thus forbidden. A tabu move however is allowed as an exception to this rule, if it leads to a solution that improves the best solution found so far in run i, $S^{best}(i)$.

     - We also include a diversification mechanism which prevents the frequent use of moves which involve the same nodes. Specifically, moves involving nodes that were involved in more than $\theta_1 + \lceil \theta_2 \cdot \mu \rceil$ iterations are also declared forbidden, where μ is the average number of times a node was involved in a move so far, and θ_1, θ_2 are parameters that need to be tuned experimentally (after some experimentation, we decided to use $\theta_1 = 3$ and $\theta_2 = 1.4$). Again, such a move is allowed as an exception to this rule, if it leads to a solution that improves the best solution found so far in run i, $S^{best}(i)$.

- If S^1 is better than $S^{best}(i)$, then set $S^{best}(i) := S_1$.
- Set $S^2 := S^1$.
- Repeat Stage A, until there is no improvement in solution $S^{best}(i)$ in the last J_1 iterations.

Note that, during this first stage of Tabu search, whenever a solution S is calculated or updated, only the triplet $(R, \widetilde{W}, \widetilde{G})$ is updated; i.e. the function *optimal waiting times* is not employed, nor are the corresponding W and G calculated or updated.

- <u>Tabu Search - Stage B:</u>
  Repeat the same process described by Stage A, but this time replace J_1 by J_2 (where generally J_2 is advised to be smaller than J_1) and replace 'Stage A' by 'Stage B'. In this second stage, whenever a solution S is calculated or updated, the full quintuple $(R, \widetilde{W}, \widetilde{G}, W, G)$ is updated, meaning that the function *optimal waiting times* is now called to calculate or update W and G.

  Stage B is repeated, until there is no improvement in solution $S^{best}(i)$ in the last J_2 iterations.

- Once the two stages are finished, we get the best solution found in run i, $S^{best}(i)$.
- The reason why we split the Tabu Search in two stages, is because this saves us computational time, something which was confirmed through experiments. Stage A is a fast way to get close to a very good solution. Once this is achieved, stage B repeats the same process as in stage A, but this time the more computationally expensive function *optimal waiting times* is employed, which involves the exploration of the different solutions when allowing different amounts of waiting times at different customers, and the search for the optimal allocation of waiting times at customers for the corresponding set of routes. In fact, one could ignore stage A and go directly to stage B of Tabu Search, which should give similar results (provided that J_2 would be approximately the same or greater than J_1); however, this would be computationally more

expensive and the whole algorithm would be substantially slower than the proposed method.

5. Compare all d solutions $S^{best}(i)$, for $i = 1, 2, ..., d$ and return the best one, S^*, with respect to the objective. Return also the values of the five component objectives $F_1, F_2, ..., F_5$ that correspond to this best solution, in the quintuple $(F_1^*, F_2^*, ..., F_5^*)$.

6.2.5 Parameter values used in Heuristic Approach 1

After some experimentation, we decided to use the following values for the parameters, throughout our experiments of Heuristic Approach 1 for problem 2:

d	θ_1	θ_2	β_1	β_2	β_3	β_4	β_5	J_1	J_2	J_3	$\hat{d}$	k_2	k_3
6	3	1.4	10^8	10^7	10^3	1	10^{-3}	148	49	12	19	11	31

The Tabu Length TL is defined as a function of n, as shown below:

$$TL = \begin{cases} \lfloor \frac{n}{4} \rfloor & \text{if } n \leq 30 \\ \max\{\lfloor \frac{n}{6} \rfloor, 7\} & \text{if } 30 < n \leq 100 \\ 15 & \text{if } n > 100 \end{cases}$$

Note that the following parameters are given as an input separately for each instance: $n, X_i, Y_i, a_i, b_i, s_i, T_1$.

6.2.6 Heuristic Approach 1b: A Tabu Search heuristic for the Multi-Objective Delayed TSPTW with 5 Objectives, using the Weighted-Sum method

The algorithm presented in subsection 6.2.4 solves the Delayed-TSPTW-5 heuristically, assuming a specific lexicographic preference of the 5 component objectives. We can generalize this algorithm by following the procedure described in the following paragraph, which we will call *Heuristic Approach 1b*. This approach solves heuristically the more general variant of problem 2, i.e. the *Multi-Objective Delayed TSPTW with 5 objectives* (without preference information), using the weighted-sum method and Tabu Search.

In Heuristic Approach 1b, we vary systematically the set of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ (not necessarily normalized), and for each different choice of weights we run the algorithm of Heuristic Approach 1, which was described in subsection 6.2.4. The solutions derived from all these runs are compared with each other and any dominated or duplicated solutions are discarded. The remaining set of solutions constitutes an approximate representative set of the Pareto front.

6.3 Heuristic Approach 2: A Tabu Search heuristic for the Lexicographic Delayed TSPTW with 3 Objectives

In this section we describe a heuristic approach to solve the *Lexicographic Delayed TSPTW with 3 objectives (Delayed-TSPTW-Lex3).* We will refer to this approach as *Heuristic Approach 2* for problem 2.

In this approach, we use the same notation and algorithm that were used in heuristic approach 1 and described in section 6.2, but with fewer component objectives. For this, we make the assumptions and employ the technique described in section 5.12, in order to reduce the 5 component objectives of the Delayed-TSPTW-5 down to 3 objectives. Specifically, recall that in the first heuristic approach we had used 5 component objectives F_1, F_2,..., F_5 which were transformed into a single aggregated objective function $G := \sum_{i=1}^{5} \beta_i F_i$ to be minimized. Instead, in heuristic approach 2 we use the 3 component objectives f_1, f_2 and f_3, defined as $f_1 := F_1 + F_2$, $f_2 := F_3 + F_4$ and $f_3 := F_5$, which are also combined into a single objective function. Therefore, the objective in the second heuristic approach is:

$$minimize \quad G_3 := \sum_{i=1}^{3} \beta_i' f_i \tag{6.3.1}$$

Note that the lower bound LB_{F_1} of F_1 is also a lower bound for f_1 (i.e. $LB_{f_1} := LB_{F_1}$). Similarly, the lower bound LB_{F_3} of F_3 is also a lower bound for f_2 (i.e. $LB_{f_2} := LB_{F_3}$).

By systematically varying the triplet of positive weights $(\beta_1', \beta_2', \beta_3')$ (not necessarily normalized), and for each different choice of weights running the algorithm of heuristic approach 1, but using G_3 as the objective function and f_1, f_2, f_3 as the component objectives, instead of G and F_1, F_2,..., F_5, we can get an approximate representative set of the Pareto front for the Multi-Objective Delayed-TSPTW-3 (after removing any duplicated or dominated solutions). This approach constitutes *Heuristic Approach 2b* for problem 2, which uses the weighted-sum method and Tabu Search to solve heuristically the Multi-Objective Delayed TSPTW-3.

As a special case of Heuristic Approach 2b, if we perform a single run of the weighted-sum method described in the previous paragraph, using just one triplet of weights $(\beta'_1, \beta'_2, \beta'_3)$ with $\beta'_1 >> \beta'_2 >> \beta'_3 > 0$, we can solve heuristically the Lexicographic Delayed TSPTW with 3 objectives (Delayed-TSPTW-Lex3), which constitutes *Heuristic Approach 2* for problem 2.

Note that heuristic approach 2b is equivalent to using heuristic approach 1b with different sets of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ such that $\beta_1 = \beta_2$ and $\beta_3 = \beta_4$. Also, heuristic approach 2 is equivalent to using heuristic approach 1 with a single set of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ such that $\beta_1 = \beta_2 >> \beta_3 = \beta_4 >> \beta_5 > 0$.

Heuristic approaches 2 and 2b are slight simplifications of heuristic approaches 1 and 1b, respectively. However, it is generally easier to handle fewer objectives, especially if we are interested in finding not just a single solution but the whole set of non-dominated solutions (or even a representative subset of this). Therefore, there are some obvious advantages for using the 3-objective variants over the 5-objective counterparts. There is also the possibility that for the specific problem under study, the 3-objective variants may sometimes produce similar solutions to the more general 5-objective counterparts - although further experimentation is needed to investigate this claim. Furthermore, heuristic approach 2 serves as a bridge for comparing solutions between the first and third heuristic approaches: Since it is not obvious how to directly compare the results of heuristic approaches 1 and 3, we can do this indirectly by comparing the results from each one of those two approaches separately with the results of heuristic approach 2. Finally, we needed a separate and precise description of heuristic approach 2, because the algorithm of the more complex heuristic approach 3 calls the algorithm of heuristic approach 2 several times, as a subroutine.

6.3.1 Parameter values for Heuristic Approach 2

Throughout experiments with Heuristic Approach 2, we can use the same algorithm used in Heuristic Approach 1 (described in section 6.2) and the same parameter values that were reported in subsection 6.2.5, with the only difference that in Heuristic Approach 2 we can choose the following set of weights:

β_1	β_2	β_3	β_4	β_5
10^8	10^8	10^3	10^3	10^{-3}

We note that this is equivalent to using the algorithm described in section 6.3 with the following set of weights:

β'_1	β'_2	β'_3
10^8	10^3	10^{-3}

6.4 Heuristic Approach 3: A Tabu Search heuristic for the Delayed TSPTW with 3 objectives, using the ECM framework

In this section we describe a heuristic based on *Tabu Search* and the *Epsilon Constraint Method (ECM)*, which is designed to solve the *Delayed TSPTW with 3 objectives (Delayed-TSPTW-3)*. We will refer to this approach as *Heuristic Approach 3* for Problem 2.

As in section 6.3, there are three component objectives to be minimized, namely f_1, f_2 and f_3. In order to address the 3-objective variant of the problem using the ECM, we treat f_2 as the actual objective function to be minimized (min f_2) and convert the other two objectives into constraints ($f_1 \leq \epsilon_1$ and $f_3 \leq \epsilon_3$). This way we arrive at the *ECM formulation for the Delayed-TSPTW-3*, or in short *(MILP 3E)*, which is composed of all constraints (5.6.6)-(5.6.24) of (MOMILP 2A) that were presented in section 5.6, together with objective 6.4.1 and constraints 6.4.2 and 6.4.3 which follow:

$$\text{minimize} \;\; f_2 := \sum_{i \in I_C} \rho_i + \sum_{i \in I_C} \sigma_i \tag{6.4.1}$$

$$f_1 := \sum_{i \in I_C} \mu_i + \sum_{i \in I_C} \lambda_i \leq \epsilon_1 \tag{6.4.2}$$

$$f_3 := w_0 \leq \epsilon_3 \tag{6.4.3}$$

(where w_0 is also denoted by w_{n+2} in this chapter). The above mathematical program is a single-objective MILP, which depends, among others, on the parameters ϵ_1 and ϵ_3. Therefore, in the remainder of this chapter we will be referring to this formulation as *MILP-2E-ECM*(ϵ_1, ϵ_3). Note that this program constitutes an exact formulation for the Delayed-TSPTW-3, using the ECM method. In theory, this can be implemented in an optimization solver and, by systematically varying the values of ϵ_1 and ϵ_3, one can get a representative subset of the set of non-dominated solutions for the problem. We will refer to this approach as *Exact Approach 3* for Problem 2.

Note that the experimental results from Exact Approach 1, which will be presented later in section 6.5, indicate that such exact approaches based on formulations (MILP 2B) or (MOMILP 2A) are not promising for solving moderate-sized or large instances in real time. Since Exact Approach 3 involves solving multiple such programs of the form (MILP 2E), it is even more computationally expensive than Exact Approach 1. Therefore, we decided not to perform any experiments with Exact Approach 3.

Instead, we will use the above framework of the ECM to construct a heuristic algorithm, based on formulation (MILP 2E). *Heuristic Approach 3* involves varying systematically the values of ϵ_1 and ϵ_3 within appropriate ranges, and for each pair (ϵ_1, ϵ_3) of these parameter values, solving heuristically MILP-2E-ECM(ϵ_1, ϵ_3). The solutions derived from solving all of the above programs are compared with each other and any duplicated or dominated ones are discarded. The resulting set of solutions constitutes an approximate subset of the set of non-dominated solutions for the Delayed-TSPTW-3.

6.4.1 ECM solutions table & ranges of the epsilons

Suppose that ϵ_1 will take Φ different values in the range $[E^1_\Phi, E^1_1]$ (perhaps equidistant, but not necessarily). Likewise, suppose that ϵ_3 will take Θ different values in the range $[E^3_\Theta, E^3_1]$ (perhaps equidistant, but not necessarily). Let $E^1 = (E^1_1, E^1_2, ..., E^1_\Phi)$ be a vector containing all Φ possible values that ϵ_1 may take, in descending order. Similarly, let $E^3 = (E^3_1, E^3_2, ..., E^3_\Theta)$ be a vector containing all Θ possible values that ϵ_3 may take, in descending order.

It is requested to solve heuristically $\Phi \times \Theta$ programs of the form MILP-2E-ECM(ϵ_1, ϵ_3), one for each possible combination of (ϵ_1, ϵ_3). Assume that $f^2_{i,j}$ is the optimal value for f_2 in the program MILP-2E-ECM(E^1_i, E^3_j), and that for the specific solution, the corresponding values for f_1 and f_3 are $f^1_{i,j}$ and $f^3_{i,j}$ respectively, for all i=1,2,...,Φ, j=1,2,...,Θ.

Essentially, by solving all $\Phi \times \Theta$ programs we will get a table of the same form as table 6.1, where in each cell (i, j) we record $f^2_{i,j}$ ($f^1_{i,j}$, $f^3_{i,j}$), i.e. the value of the main objective f_2 for the respective pair of values for ϵ_1 and ϵ_3, followed by the values of the other two objectives f_1 and f_3 in parentheses.

Therefore, for each pair of indexes (i, j) we are trying to find the optimal solution $f^2_{i,j}$ ($f^1_{i,j}$, $f^3_{i,j}$) to the program MILP-2E-ECM(E^1_i, E^3_j). This solution gives the minimum value $f^2_{i,j}$ of f_2, while satisfying the constraints $f^1_{i,j} \leq E^1_i$ and $f^3_{i,j} \leq E^3_j$.

6.4.2 Objectives

In the algorithm of heuristic approach 3, which is described step by step in subsection 6.4.5, we use different objectives at different stages. Note that, at any stage of the algorithm, the value of the objective is stored as $\widetilde{G}$ when

Table 6.1: ECM solutions table

ϵ_3 / ϵ_1	E^3_1	E^3_2	...	E^3_j	...	E^3_Θ
E^1_1	$f^2_{1,1}$ $(f^1_{1,1}, f^3_{1,1})$	$f^2_{1,2}$ $(f^1_{1,2}, f^3_{1,2})$	...	...	...	...
E^1_2	...	...	...	...	...	...
...	...	...	...	...	...	...
E^1_i	...	...	...	$f^2_{i,j}$ $(f^1_{i,j}, f^3_{i,j})$	...	...
...	...	...	...	...	...	...
E^1_Φ	...	...	...	...	...	...

idle waiting times at nodes are not allowed, or as G when idle waiting times are allowed.

- **Objective throughout Step 4:**

  Throughout Step 4, for the calculation of the objective function we use function $G_3 := \sum_{i=1}^{3} \beta'_i f_i$ that was previously defined by equation (6.3.1), which we seek to minimize.

  Therefore, throughout Step 4, the value of the objective function without allowing idle waiting times at nodes, and when following route R and visiting the customers at times $\widetilde{W}$, is stored as $\widetilde{G}$, and is calculated using the function G_3 defined by equation (6.3.1). Similarly, throughout Step 4, the value of the objective function when allowing idle waiting times at nodes, and when following route R and visiting the customers at times W, is stored as G, and is also calculated using the function G_3 defined by equation (6.3.1).

- **Objective throughout Step 6:**

  Throughout Step 6, for the calculation of the objective function we use function G_ϵ defined below, which we seek to minimize:

$$G_\epsilon := f_2 + \begin{cases} 0 & \text{if } f_1 \leq \epsilon_1 \text{ and } f_3 \leq \epsilon_3 \\ (f_1 - \epsilon_1) \cdot P_1 + P^*_1 & \text{if } f_1 > \epsilon_1 \text{ and } f_3 \leq \epsilon_3 \\ (f_3 - \epsilon_3) \cdot P_3 + P^*_3 & \text{if } f_1 \leq \epsilon_1 \text{ and } f_3 > \epsilon_3 \\ (f_1 - \epsilon_1) \cdot P_1 + P^*_1 + (f_3 - \epsilon_3) \cdot P_3 + P^*_3 & \text{if } f_1 > \epsilon_1 \text{ and } f_3 > \epsilon_3 \end{cases} \tag{6.4.4}$$

  Essentially, G_ϵ is defined as f_2 if the epsilon constraints are respected; otherwise it is penalized by a large penalty which is proportional to the amount of violation, plus a large constant. Specifically, G_ϵ is increased by $(f_1 - \epsilon_1) \cdot P_1 + P_1^*$ if $f_1 > \epsilon_1$, and/or it is increased by $(f_3 - \epsilon_3) \cdot P_3 + P_3^*$ if $f_3 > \epsilon_3$, where P_1, P_1^*, P_3 and P_3^* are appropriate penalty coefficients.[1]

Therefore, throughout Step 6, the value of the objective function without allowing idle waiting times at nodes, and when following route R and visiting the customers at times $\widetilde{W}$, is stored as $\widetilde{G}$, and is calculated using the function G_ϵ defined by equation (6.4.4). Similarly, throughout Step 6, the value of the objective function when allowing idle waiting times at nodes, and when following route R and visiting the customers at times W, is stored as G, and is also calculated using the function G_ϵ defined by equation (6.4.4).

6.4.3 Solution Notation

Apart from the difference in the calculation of the objectives $\widetilde{G}$ and G, which was explained in the previous subsection, throughout heuristic approach 3 we will use the same notation used in heuristic approach 1 and described in section 6.2 (e.g. for $R, \widetilde{W}, W, S$ etc.), unless otherwise stated or implied. Furthermore, we will use the same algorithmic functions, *time*, *cost* and *optimal waiting times* that were used in heuristic approach 1 and described in section 6.2, but adjusted accordingly, so that at different stages the appropriate function G_3 or G_ϵ (defined by equations (6.3.1) and (6.4.4), respectively) is involved in the calculation of the objective function. More details on the main algorithmic functions involved can be found in the following subsection.

6.4.4 Descriptions of the main algorithmic functions

As in heuristic approach 1, given a route R, we can calculate or update the complete solution $S := (R, \widetilde{W}, \widetilde{G}, W, G)$ using the following algorithmic functions:

(i) Given R, we can calculate the vector $\widetilde{\mathbf{W}}$ using the algorithmic function *time*: $\widetilde{\mathbf{W}} = \boldsymbol{time}(\mathbf{R})$ (as described in subsection 6.2.3).

(ii) Given R and $\widetilde{\mathbf{W}}$, we can calculate $\widetilde{\mathbf{G}}$ using the algorithmic function *cost_2*: $\widetilde{\mathbf{G}} = \mathbf{cost\_2}(\mathbf{R}, \widetilde{\mathbf{W}})$:

Given R and $\widetilde{W}$, the algorithmic function *cost_2* calculates the value of the objective function without allowing idle waiting times at nodes, when following the route R and visiting the nodes at times $\widetilde{W}$. This value is stored

[1]After some experimentation, we decided to use $P_1 = 400$, $P_1^* = 50000000$, $P_3 = 100$ and $P_3^* = 20000000$.

as $\widetilde{G}$ and for its calculation we use either function G_3 and equation (6.3.1) whenever *cost_2* is called in step 4 of the algorithm, or function G_e and equation (6.4.4) whenever *cost_2* is called in step 6 of the algorithm.

In more detail, given a route R and a vector $\widetilde{W}$, the algorithmic function *cost_2* compares the time of start of service $\widetilde{W_i}$ of each node with the respective time window $[a_i, b_i]$ and thus calculates the values of each function F_j $(j = 1, \ldots, 5)$. Initially each F_j is set to zero. Then, for all nodes $i = 1, \ldots n$ that represent customers, if $\widetilde{W_i} > b_i$, then F_1 is increased by 1 and F_3 is increased by $\widetilde{W_i} - b_i$. Similarly, for all $i = 1, \ldots n$, if $\widetilde{W_i} < a_i$, then F_2 is increased by 1 and F_4 is increased by $a_i - \widetilde{W_i}$. Finally, F_5 is defined as $\widetilde{W_{n+2}}$. Then the values of f_j $(j = 1, 2, 3)$ are defined as $f_1 := F_1 + F_2$, $f_2 := F_3 + F_4$ and $f_3 := F_5$. Hence $G_3 := \sum_{i=1}^{3} \beta_i' f_i$ is defined. Also, given the values of f_j $(j = 1, 2, 3)$, ϵ_1 and ϵ_3, the function G_ϵ is straightforward to calculate using equation (6.4.4). Note that at the different stages of the algorithm, only one of the two functions G_3 or G_ϵ is calculated, and the corresponding objective value is returned as the output $\widetilde{G}$ of the algorithmic function *cost_2*; i.e. $\widetilde{G} = cost\_2(R, \widetilde{W})$.

Note that, more generally, given R and a vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes (where for instance, we may use $\widetilde{W}$ or W as $\widehat{W}$), we can still use the same algorithmic function *cost_2* to calculate the respective value of the objective function $\widehat{G}$, as $\widehat{G} = cost\_2(R, \widehat{W})$. Specifically, given R and the times of start of service W while allowing idle waiting times at nodes, we can use the same algorithmic function *cost_2* to calculate G, as $G = cost\_2(R, W)$.

(iii) Given R and $\widetilde{W}$, we can calculate W and G using the algorithmic function *optimal waiting times 2*: *[W, G]=optimal_waiting_times_2(R,$\widetilde{W}$)*:

This algorithmic function is defined the same way that the algorithmic function *optimal waiting times* was defined in section 6.2, but with the following difference: Instead of calling internally the algorithmic function *cost*, the algorithmic function *optimal waiting times 2* calls the function *cost_2*.

Therefore, given R and $\widetilde{W}$, the algorithmic function *optimal waiting times 2* calculates and returns a heuristic approximation of the optimal times of start of service W and the corresponding value of the objective function G when allowing idle waiting times at nodes and when following the collection of routes R and visiting the nodes at times W, where G is calculated either using function G_3 and equation (6.3.1) if it is called in step 4 of the algorithm, or using function G_ϵ and equation (6.4.4) if it is called in step 6 of the algorithm.

Note that we can still use the function *optimal waiting times 2* if we replace vector $\widetilde{W}$ in the input with vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes.

6.4.5 Algorithm description

In this subsection we present the algorithm involved in Heuristic Approach 3, which uses the framework of the Epsilon Constraint Method and Tabu Search with multiple restarts, to construct an approximate representative subset of the set of non-dominated solutions for the Delayed-TSPTW-3. These are the main steps of the algorithm:

1. Parameters input & Initialization:
   - Input: n, d, T_1, TL (Tabu Length), Θ, Φ, weights $(\beta_1, \beta_2, ..., \beta_5)$.
   - Input: (X_i, Y_i), $[a_i, b_i]$, s_i $\forall\ i = 1, 2, \ldots, n+2$.
   - Input: J_1, J_2, J_3, $\hat{d}$. The parameters J_1 and J_2 denote the number of Tabu Search iterations without an improvement, before Stages A and B terminate, respectively. Also, J_3 and $\hat{d}$ are parameters controlling whether or not the two-opt moves will be considered in the current iteration.

   - For all $i, j \in \{1, 2, \ldots, n+2\}$, calculate the travel time $t_{i,j}$ as the Euclidean distance $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ between each pair of nodes (i, j), which is first truncated into an integer value using the floor function (as done in Dumas et al. (1995)), and then slightly adjusted by applying the *Triangle Inequality Correction* where necessary, as follows:

   ```
   for all $i, j, k \in \{1, 2, \ldots, n+2\}$
       if $t_{i,j} > t_{i,k} + t_{k,j}$
           $t_{i,j} := t_{i,k} + t_{k,j}$
       end if
   end for
   ```

2. Calculate lower bounds for component objectives f_1 and f_2:
   - Identify the set of *Definitely Late customers*, $I_{DL} = \left\{ i \in I_C \;\middle|\; T_1 + s_{n+1} + t_{n+1,i} > b_i \right\}$, as described in chapter 5. Its cardinality $|I_{DL}|$ is the number of definitely late customers, which serves as a lower bound for f_1; that is, $LB_{f_1} = |I_{DL}|$.

   - We will assume that LB_{f_1} is enhanced, whenever possible, by using the subroutine described at the second step of the algorithm that was described in subsection 6.2.4, which identifies any *additionally late customers* to increase LB_{f_1}.

- Evaluate the *minimum amount of lateness*, defined as: $LB_{f_2} := \sum_{i \in I_{DL}} (T_1 + s_{n+1} + t_{n+1,i} - b_i)$, which serves as a lower bound for f_2.

3. <u>Construct the Initial Route R^0:</u>

   The initial route R^0 is constructed as follows: First we sort all nodes of the customers' set I_C in increasing order of the b_i's and store these in the row vector I_S. Then, among all nodes of the vector I_S, we identify those that also belong to the set I_{DL} of definitely late customers, and move them at the end of I_S, while preserving their order. The initial route is defined as the vector $R^0 = [n+1, I_S, n+2]$, where node $n+1$ represents the starting node (i.e. the position of the vehicle at time T_1) and node $n+2$ represents the end-point node (i.e. the depot).

   Given the initial route $R^0 = (R^0_1, R^0_2, ..., R^0_{n+2}) = (n+1, R^0_2, R^0_3, ..., R^0_{n+1}, n+2)$, we then calculate the respective $\widetilde{W}^0$. We also let $R := R^0$ and $\widetilde{W} := \widetilde{W}^0$.

4. <u>Objective throughout Step 4:</u>
   Throughout Step 4, the objective is to minimize the function G_3 defined by equation (6.3.1), and the respective value found at any given time is stored as $\widetilde{G}$ when idle waiting time at nodes is not allowed, or as G when idle waiting time at nodes is allowed.

   <u>Step 4:</u>

   - Using the initial route R^0 found in the previous step, *perform three separate runs of the algorithm described by Heuristic Approach 2 (in section 6.3)*, with G_3 as the objective to be minimized, where each time we use one of the following three sets of weights $(\beta'_1, \beta'_2, \beta'_3)$ respectively: $(1,0,0)$, $(0,1,0)$ and $(0,0,1)$.

   Note that when calling the algorithm of heuristic approach 2, we assume the following additional modifications: (i) we only employ a single restart of the algorithm in each case (i.e. we use $d = 1$), and (ii) we skip step 3 of that algorithm, and instead use R^0 described above as the initial solution.

   Essentially, in each one of these three runs the algorithm solves heuristically the mathematical program *(MOMILP 2C)* that was defined in section 5.12, which is transformed into a single-objective program, with only one of f_1, f_2 and f_3 as the objective to be minimized, neglecting the other two component objectives (but storing their values each time).

From each one of the three different runs, we get a triplet of the form (f_1, f_2, f_3) containing the values of the three component objectives. Then, separately for each $i = 1, 2, 3$, the three values found for each f_i are compared with each other to find the smallest value f_i^m and the largest value f_i^M among the three values of each component objective. Also, let min_{f_i} and max_{f_i} denote the true minimum and maximum value of f_i, respectively, for each $i = 1, 2, 3$. Note that f_i^m is not necessarily the same as the true minimum value min_{f_i} of f_i, but is an overestimate of min_{f_i} that should be close to the true value. On the other hand, not only is f_i^M generally different to the true maximum value max_{f_i} of f_i, but it is an underestimate of max_{f_i} that may be substantially smaller than the true maximum value. However, since we are minimizing, we do not really need the maximum value for any of the three component objectives f_1, f_2 and f_3. In fact, we only treat the values f_i^m and f_i^M as reference values that specify a part of the range of each component objective f_i which is of interest. Some of those values are used below to construct the set of different values that the epsilons will take.

We use the number of definitely late customers $|I_{DL}|$ as the lower bound LB_{f_1} for f_1, and the number of all active customers n as the upper bound UB_{f_1} for f_1; i.e. we set $LB_{f_1} := |I_{DL}|$ and $UB_{f_1} := n$. Note that these are both true bounds.

Regarding f_1, the bounds LB_{f_1} and UB_{f_1} are used below to define the range of values that ϵ_1 may take. Also, regarding f_3, the reference values f_3^m and f_3^M are used to define the range of values that ϵ_3 may take. Note that we do not make use of and therefore we do not need any bounds or reference values for f_2.

Also, regarding the true maximum value of f_3, this could be substantially greater than the value f_3^M that was calculated above. However, the true maximum value of f_3 corresponds to a solution with a maximum sum of arrival times at the endpoint node, which is comparable to a solution involving routes of maximal total length, which is very undesirable and irrelevant in problems like the one under study. Hence, we believe that f_3^M is in fact more meaningful and more appropriate to use than the true maximum value max_{f_3}, for the purpose of defining the largest value that ϵ_3 is allowed to take.

5. Calculate the different values that ϵ_1 and ϵ_3 will take:
   We assume that ϵ_1 takes Φ different values, where Φ is a user-input parameter such that $\Phi \geq 2$. These values are calculated as follows: The interval $[LB_{f_1}, UB_{f_1}]$ is divided into $\Phi - 1$ equal intervals of length $(UB_{f_1} - LB_{f_1})/(\Phi - 1)$, and their lower end-points, together with UB_{f_1}, are collected to create the set of Φ different values that ϵ_1 will take. In other words, ϵ_1 takes the values LB_{f_1}, $LB_{f_1} + (UB_{f_1} - LB_{f_1})/(\Phi - 1)$, $LB_{f_1} + 2(UB_{f_1} - LB_{f_1})/(\Phi - 1)$, ..., UB_{f_1}.

   Since f_1 takes only integer values, in the case where the fraction $(UB_{f_1} - LB_{f_1})/(\Phi - 1)$ is less than 1 (or equivalently, if $\Phi > UB_{f_1} - LB_{f_1} + 1$), we can speed up the algorithm by setting $\Phi := UB_{f_1} - LB_{f_1} + 1$, i.e. reducing Φ to a smaller and more meaningful value, which results in ϵ_1 taking all the integer values from LB_{f_1} to UB_{f_1} inclusive.

   Similarly, we assume that ϵ_3 takes Θ different values, where Θ is a positive integer user-input parameter. These values are calculated as follows: The interval $[f_3^m, f_3^M]$ is divided into $\Theta - 1$ equal intervals of length $(f_3^M - f_3^m)/(\Theta - 1)$, and their lower end-points, together with f_3^M, are collected to create the set of Θ values that ϵ_3 will take.

   Let $E^1 = (E_1^1, E_2^1, ..., E_\Phi^1)$ and $E^3 = (E_1^3, E_2^3, ..., E_\Theta^3)$ be the vectors containing all possible values in descending order, that ϵ_1 and ϵ_3 take, respectively.

6. Objective throughout Step 6:

   Throughout Step 6, the objective is to minimize the function G_ϵ defined by equation (6.4.4), and the respective value found at any given time is stored as $\widetilde{G}$ when idle waiting time at nodes is not allowed, or as G when idle waiting time at nodes is allowed.

   Step 6:

   For all $i = 1, 2, ..., \Phi$ and for all $j = 1, 2, ..., \Theta$ do:

   - Let $\epsilon_1 := E_i^1$ and $\epsilon_3 := E_j^3$.
   - Try to solve heuristically MILP-2E-ECM(ϵ_1, ϵ_3) using Tabu Search, by following the procedure described below:

   - Initialize the *current route* R^2 by letting $R^2 := R^0$. Then create or update the complete *current solution* $S^2 = (R^2, \widetilde{W}^2, \widetilde{G}^2, W^2, G^2)$ accordingly.

- Initialize the *best solution found in iteration* (i,j), denoted by $S^{best\_ECM}(i,j)$, by letting $S^{best\_ECM}(i,j) := S^2$.
- Apply Tabu Search with the best improvement criterion, using the neighborhood moves exchange, insertion and two-opt, to find the best solution $S_{best\_ECM}(i,j)$ of iteration (i,j), with respect to the objective function G_ϵ (defined by equation (6.4.4)). More specifically, apply Tabu Search Stages A and B of Step 4 of the Heuristic Approach 2, as described in section 6.3 (which refer to the respective parts of Heuristic Approach 1, described in subsection 6.2.4, but while having the 3 component objectives f_1, f_2, f_3 involved, instead of the 5 component objectives $F_1, F_2, ..., F_5$), with the following changes: (i) Replace $S^{best}(i)$ by $S^{best\_ECM}(i,j)$ as the best solution found so far in the current iteration, (ii) replace the function *optimal waiting times* by the function *optimal waiting times 2*, and (iii) use G_ϵ as the objective function to be minimized. As in Heuristic Approach 1, the two-opt moves are only considered while there was an improvement to the best solution found in the last J_3 iterations, or if the number of iterations performed so far is a multiple of $\hat{d}$ (we used $\hat{d} = 19$ in our experiments).
- Once the two stages A and B of Step 4 of the Tabu Search (in Heuristic Approach 1) are finished, we have the best solution $S^{best\_ECM}(i,j)$ found in iteration (i,j), corresponding to the epsilons $(\epsilon_1, \epsilon_3) = (E_i^1, E_j^3)$, i.e. the best solution found for MILP-2E-ECM(E_i^1, E_j^3). For each iteration (i,j), we record the values of the functions f_1, f_2, f_3 that correspond to this best solution, in the triplet $(f_{i,j}^1, f_{i,j}^2, f_{i,j}^3)$.

7. Return all non-dominated solutions:

   - Compare all the best solutions found, $S^{best\_ECM}(i,j)$, between each other, for all values of (i,j), by comparing the corresponding triplets $(f_{i,j}^1, f_{i,j}^2, f_{i,j}^3)$. Remove all dominated solutions from the solution set.

   For instance, if $(f_{i_1,j_1}^1, f_{i_1,j_1}^2, f_{i_1,j_1}^3)$ and $(f_{i_2,j_2}^1, f_{i_2,j_2}^2, f_{i_2,j_2}^3)$ are two such solutions found so that $f_{i_1,j_1}^1 \le f_{i_2,j_2}^1$, $f_{i_1,j_1}^2 \le f_{i_2,j_2}^2$, $f_{i_1,j_1}^3 \le f_{i_2,j_2}^3$ and at least one of the previous three inequalities is strict, then the second solution is said to be dominated by the first solution and is therefore removed from the set of solutions.[2]

[2]Actually, even if none of those three inequalities is strict, we can still remove one of the two solutions, since they give exactly the same values to the three objectives f_1, f_2, f_3 and are therefore equivalent.

- Return the resulting set of all non-dominated solutions found, which is an approximate subset of the true set of non-dominated solutions for the Delayed-TSPTW-3.
- Then plot the respective Value Paths, showing the corresponding triplets $(f^1_{i,j}, f^2_{i,j}, f^3_{i,j})$ of all the non-dominated solutions found.

6.4.6 Parameter values used in Heuristic Approach 3

After some experimentation, we decided to use the following parameter values in our experiments with Heuristic Approach 3:

P_1	P_1^*	P_3	P_3^*	θ_1	θ_2	Θ	$\hat{d}$
400	50000000	100	20000000	3	1.4	4	19

Throughout our experiments, the parameter Φ is calculated as a function of n and LB_{f_1}, by letting $\Phi := n - LB_{f_1} + 1$. Therefore, ϵ_1 takes all values in the set $\{LB_{f_1}, LB_{f_1} + 1, ..., n\}$ (i.e. all consecutive integer values from LB_{f_1} up to and including n).

	β'_1	β'_2	β'_3
First set of weights used in Step 4:	1	0	0
Second set of weights used in Step 4:	0	1	0
Third set of weights used in Step 4:	0	0	1

	J_1	J_2	J_3
In Step 4:	98	4	10
In Step 6:	98	9	15

The Tabu Length (TL) is defined as a function of n, as follows:

$$TL = \begin{cases} \lfloor \frac{n}{4} \rfloor & \text{if } n \leq 30 \\ \max\{\lfloor \frac{n}{6} \rfloor, 7\} & \text{if } 30 < n \leq 100 \\ 15 & \text{if } n > 100 \end{cases}$$

Note that the parameter values used in Step 4, where the algorithm calls the algorithm of Heuristic Approach 2, are the same as the ones described in subsection 6.3.1. Note also that the following parameters are given as an input separately for each instance: $n, X_i, Y_i, a_i, b_i, s_i, T_1$.

6.5 Experimental Results

In this section we present the experimental results for different methods proposed to solve the *Delayed TSPTW*, including exact and heuristic approaches.

As an overview, the experimental results of Exact Approach 1 are summarized in table 6.2 and discussed in subsection 6.5.1. Then in subsection 6.5.2 we discuss the results of Heuristic Approach 1, which are presented in table 6.3. Finally, the experimental results of Heuristic Approach 3 are presented in Appendix A and discussed in subsection 6.5.3.

In order to test the solution methods proposed for the different variations of the Delayed TSPTW, we created a dataset of 27 new instances for this problem. For this, 9 benchmark instances were selected from the TSPTW dataset proposed by Dumas et al. (1995). Specifically, we selected the following instances: n20w20.001, n40w40.001, n40w60.001, n20w60.001, n80w60.001, n80w20.001, n100w20.001, n100w60.001 and n60w20.001. Based on each one of these standard TSPTW instances, which will be referred to as *classes* 1 through 9, three different instances for the *Delayed TSPTW* were created (e.g. *instances* 1-A, 1-B and 1-C, which correspond to class 1). For any specific class, each one of the three instances assumes that there is a different amount of delay while following the optimal solution plan for the original TSPTW problem, with a different subset of customers having already been served and with the vehicle starting from a different position and at a different time in the revised plan.

We will now explain in more detail how the three instances A, B and C of each class were created. We select a benchmark TSPTW instance from Dumas et al. (1995) dataset, which is assumed to be the original TSPTW problem (e.g. instance n20w20.001 for problems 1-A, 1-B, 1-C of class 1). Then, while following the optimal solution reported in the literature for this underlying problem, we assume that a delay occurs at some point. For each instance A, B and C, we define T_1 as the 25%, 50% and 75% of the optimal route duration (makespan) in the original TSPTW plan, respectively, truncated to an integer using the floor function. Then, we assume that all customers in the original plan up to and including time $\lfloor \frac{T_1}{2} \rfloor$ have already been served, and that at time T_1 the vehicle is ready to depart from the new starting node SN, which is the last customer in the original plan who was supposed to be visited on or before time $\lfloor \frac{T_1}{2} \rfloor$. Therefore, the vehicle has fallen behind schedule, since by time T_1 the vehicle has only served the customers that it should have served during the first half (approximately) of the time after departure from the starting node, wasting the other half of the time. Obviously, in order to create a new instance for the Delayed TSPTW, we must remove the original starting node and the customers that have already been served from the set of nodes, and relabel the remaining customers as nodes $1, 2, ..., n$ (where n is the number of customers that are still unserved at time T_1), and the starting node SN as node $n + 1$.

Given a TSPTW instance, by following the above scheme and using the values of T_1 and SN indicated in table 6.2 (where in SN we use the same numbering of nodes as in the original TSPTW instance), as well as the optimal solution reported in the literature for the original TSPTW instance, one can reconstruct the three respective instances A, B and C for the Delayed TSPTW. Using the above reasoning and starting from the 9 different TSPTW instances mentioned before, we created 3 new instances of the Delayed TSPTW for each one TSPTW instance; thus creating a total of 27 new instances for the Delayed TSPTW. The number of active customers in these 27 instances ranges from 5 to 84, with a mean of 36.5 customers and a standard deviation of 4.2 customers.

6.5.1 Experimental Results of Exact Approach 1

Table 6.2 presents the experimental results of Exact Approach 1, which was described in section 5.9 of the previous chapter, for solving the Lexicographic Delayed TSPTW with 5 objectives. Specifically, formulation (MILP 2B) of section 5.7 was implemented in AIMMS and was used to solve each one of the 27 instances of the dataset. The columns of this table provide the following information for each instance: the instance name, the number of customers n_0 in the underlying TSPTW instance, the number of customers n in the new/disrupted instance (i.e. number of unserved or active customers), the time T_1 when the vehicle is ready to depart after rescheduling, its new starting node SN (where SN is indicated using the original numbering of nodes in the underlying TSPTW problem), the values of the five component objectives F_1, F_2, ..., F_5, the runtime in seconds, the value of the objective $(\sum_{i=1}^{5} \beta_i F_i)$, which is denoted by G^{EA_1} in this approach, the lower bound $LB_G^{EA_1}$ of G^{EA_1} found by AIMMS, the optimality gap provided by AIMMS, which is calculated as: $\frac{G^{EA_1} - LB_G^{EA_1}}{G^{EA_1}} \cdot 100\%$, and finally whether the solution is Optimal, Feasible or Infeasible (O/F/I). Note that the value of G^{EA_1} depends on the specific choice of weights $\beta_1, \beta_2, ..., \beta_5$, which in our case was: $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (10^8, 10^7, 10^3, 1, 10^{-3})$. Note also that in our experiments we used $M = 5000$ and $\epsilon = 0.00001$.

For the experiments of Exact Approach 1, a maximum runtime of 10 minutes (600 seconds) was allowed. From table 6.2 we can see that the exact approach was able to find the optimal solution within 10 minutes in only 6 instances out of 27. Specifically, this occurred in the three instances of Class 1 and in the three instances of Class 4, where the number of active customers was very small (from 5 up to 15 customers). Additionally, a solution with optimality gap less than 0.22% was found in 4 other instances (specifically, in instances 2-C, 3-C, 6-C and 9-C, with 20, 19, 34 and 30 active customers, respectively). In 3 other instances the optimality gap was between 3% and 8%. In the remaining 14 instances the optimality gap was between 10% and 80%. The average optimality gap for all 27 instances was 24.24%, with a standard deviation of 5.48%. In all 27 instances this exact approach gave a feasible solution within the available time of 10 minutes; however, in many cases the solution reported was of poor quality, not only compared to the lower bound found by this approach, but also compared to the best solution reported by Heuristic Approach 1. The largest instance for which AIMMS provided an optimality gap of less than 10% was instance 8-C which involved 40 active customers, where AIMMS gave a solution with an optimality gap of 3.07%. It seems that within 10 minutes, AIMMS was unable to produce a solution with an optimality gap of less than 10% in any of the instances with more than 40 active customers. Moreover, there were instances with fewer than 40 active customers where AIMMS didn't provide a solution with an optimality gap of less than 10% within 10 minutes (instances 2-B, 9-B, 3-A and 2-A, with 25, 39, 30 and 31 customers, respectively). These results justify the need of a heuristic method that can reach a very good solution within reasonable time in all cases (not just in instances with very few customers).

Finally, note that in this table we have included the results of two additional runs of instances 9-A and 9-C, where we allowed longer runtime (8 hours 47 minutes in the first case, and 50.4 minutes in the second case). Of course, in the analysis described above, we used only the 1st run for each of the two instances mentioned here, so that all results refer to a maximum runtime of 10 minutes. Note however that if we had used the 2nd run we would get the exact same analysis described in the previous paragraph, with the only difference that the average optimality gap for all 27 instances would be 22.19%, with a standard deviation of 5.16%.

6.5.2 Experimental Results of Heuristic Approach 1

In section 6.2 we described Heuristic Approach 1, which solves the Lexicographic variant of the Delayed TSPTW with 5 objectives (Delayed-TSPTW-Lex5). We implemented the respective algorithm in MATLAB and used it to solve the 27 instances that were created for problem 2. The results are summarized in table 6.3.

The first 8 columns of this table provide the following information for each instance: the instance name, the values of the five component objectives F_1, F_2, ..., F_5, the runtime in seconds, and the value of the objective function ($G := \sum_{i=1}^{5} \beta_i F_i$), which is denoted by G^{HA_1} in heuristic approach 1. The following two columns give the lower bounds LB_{F_1} and LB_{F_3} for the component objectives F_1 and F_3 respectively. The 11th column indicates whether

Table 6.2: Experimental Results of *Exact Approach 1* for the *Lexicographic Delayed TSPTW with 5 objectives* (in AIMMS)

Instance Name	$\mathbf{n_0}$	**n**	$\mathbf{T_1}$	**SN**	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**runtime (sec)**	$\mathbf{G^{EA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_G^{EA_1}}$	**% of $\mathbf{G^{EA_1}}$ from $\mathbf{LB_G^{EA_1}}$**	**O/ F/ I**
Class 1 (n20w20.001)														
Instance 1-A	20	14	96	17	5	0	966	0	404	6.14	500966000.404	500966000.404	0.00%	O
Instance 1-B	20	9	193	12	5	0	747	0	453	1.95	500747000.453	500747000.453	0.00%	O
Instance 1-C	20	5	290	2	4	0	344	0	458	0.05	400344000.458	400344000.458	0.00%	O
Class 2 (n40w40.001)														
Instance 2-A	40	31	127	21	8	0	1932	0	642	600.47	801932000.642	500207590.1	37.62%	F
Instance 2-B	40	25	255	28	14	0	2807	0	621	600.31	1402807000.621	1200761662	14.40%	F
Instance 2-C	40	20	382	16	15	0	3128	0	599	600.28	1503128000.599	1501795242	0.09%	F
Class 3 (n40w60.001)														
Instance 3-A	40	30	133	19	9	0	857	0	553	600.30	900857000.553	700131389.5	22.28%	F
Instance 3-B	40	22	267	25	13	0	2909	0	593	600.39	1302909000.593	1200637750	7.85%	F
Instance 3-C	40	19	401	31	15	0	4087	0	659	600.16	1504087000.659	1501955210	0.14%	F
Class 4 (n20w60.001)														
Instance 4-A	20	15	100	1	4	0	715	0	400	11.34	400715000.4	400715000.4	0.00%	O
Instance 4-B	20	11	200	14	7	0	479	0	419	147.80	700479000.419	700479000.419	0.00%	O
Instance 4-C	20	8	300	16	6	0	938	0	445	1.08	600938000.445	600938000.445	0.00%	O
Class 5 (n80w60.001)														
Instance 5-A	80	64	168	29	45	1	25713	450	1450	600.39	4535713451.450	1200239000	73.54%	F
Instance 5-B	80	49	337	5	36	0	16364	0	1170	600.41	3616364001.170	2101438000	41.89%	F
Instance 5-C	80	35	505	69	27	0	8045	0	857	600.02	2708045000.857	2603267838	3.87%	F

(Continued on the next page)

Table 6.2 – *Continued from the previous page*

Instance Name	$\mathbf{n_0}$	**n**	$\mathbf{T_1}$	**SN**	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**runtime (sec)**	$\mathbf{G^{EA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_G^{EA_1}}$	**% of $\mathbf{G^{EA_1}}$ from $\mathbf{LB_G^{EA_1}}$**	**O/ F/ I**
Class 6 (n80w20.001)														
Instance 6-A	80	63	182	33	51	7	34017	1176	1509	600.22	5204018177.509	1710665214	67.13%	F
Instance 6-B	80	46	364	44	37	3	14535	299	1024	600.09	3744535300.024	2602160001	30.51%	F
Instance 6-C	80	34	546	32	29	0	11509	0	961	600.02	2911509000.961	2904966249	0.22%	F
Class 7 (n100w20.001)														
Instance 7-A	100	84	206	50	72	4	60961	381	2050	600.16	7300961383.050	1700932221	76.70%	F
Instance 7-B	100	65	413	34	58	0	27509	0	1449	600.02	5827509001.449	3603438000	38.17%	F
Instance 7-C	100	47	620	27	43	0	21650	0	1423	600.02	4321650001.423	3806545221	11.92%	F
Class 8 (n100w60.001)														
Instance 8-A	100	78	206	80	69	6	54460	1572	1809	600.41	7014461573.809	1410598501	79.89%	F
Instance 8-B	100	57	412	96	46	1	20136	111	1282	600.03	4630136112.282	2302411001	50.27%	F
Instance 8-C	100	40	618	62	35	0	12675	0	1062	600.03	3512675001.062	3404891298	3.07%	F
Class 9 (n60w20.001)														
Instance 9-A	60	46	146	22	36	3	15057	395	1115	600.45	3345057396.115	910341097.1	72.79%	F
Instance 9-A (2nd run)	60	46	146	22	11	0	3414	0	727	31646.06	1103414000.727	910351708.8	17.50%	F
Instance 9-B	60	39	293	52	24	3	8323	36	782	600.13	2438323036.782	1901385288	22.02%	F
Instance 9-C	60	30	439	16	25	0	7565	0	781	600.33	2507565000.781	2503096221	0.18%	F
Instance 9-C (2nd run)	60	30	439	16	25	0	5861	0	693	3021.11	2505861000.693	2503100311	0.11%	F

the solution found is Optimal, Feasible or Infeasible (O/F/I). The next column gives the lower bound $LB_G^{HA_1}$ of G^{HA_1} that was found by MATLAB in heuristic approach 1, which is calculated as: $LB_G^{HA_1} = \beta_1 \cdot LB_{F_1} + \beta_3 \cdot LB_{F_3}$, for the specific choice of weights $\beta_1 = 10^8$ and $\beta_3 = 10^3$. The 13th column shows the percentage gap between the objective value G^{HA_1} found in Heuristic Approach 1, and the lower bound $LB_G^{HA_1}$ also found in Heuristic Approach 1, calculated as: $\frac{G^{HA_1} - LB_G^{HA_1}}{LB_G^{HA_1}} \cdot 100\%$. The 14th column shows the percentage gap between the objective value G^{HA_1} found in Heuristic Approach 1, and the corresponding objective value G^{EA_1} found in Exact Approach 1 (which was reported in table 6.2), calculated as: $\frac{G^{HA_1} - G^{EA_1}}{G^{EA_1}} \cdot 100\%$. Finally, the last column shows the percentage gap between the objective value G^{HA_1} found in Heuristic Approach 1, and the lower bound $LB_G^{EA_1}$ calculated by AIMMS in Exact Approach 1 (which was also reported in table 6.2), calculated as: $\frac{G^{HA_1} - LB_G^{EA_1}}{LB_G^{EA_1}} \cdot 100\%$. Note that the values of G^{HA_1}, G^{EA_1} and corresponding lower bounds depend on the specific choice of weights $\beta_1, \beta_2, ..., \beta_5$, which in our case was: $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (10^8, 10^7, 10^3, 1, 10^{-3})$.

Apart from the 27 instances of the Delayed TSPTW, table 6.3 includes the results of 9 additional instances, labeled as instances 1-T, 2-T,..., 9-T. These are regular TSPTW instances, where each one of them corresponds to the original TSPTW instance before disruption that is the basis of the disrupted problem of the respective class (i.e. an instance of the Delayed-TSPTW with $T_1 = 0$). We used the algorithm of Heuristic Approach 1 to run these TSPTW instances, in order to test whether our algorithm performs well.

In each one of these 9 TSPTW instances, the algorithm was able to find a solution where all customers are served within their time windows (i.e. with $F_1 = F_2 = F_3 = F_4 = 0$), and thus a feasible solution to the TSPTW problem. In fact, in each one of these 9 TSPTW instances, the solution found using Heuristic Approach 1 matches precisely the optimal solution reported in the literature for the TSPTW-M; i,e, the TSPTW with the objective to minimize the total tour duration or 'makespan'. To clarify this, from table 6.3 we can see that in each one of these 9 instances, the solution found by the heuristic involves $F_1 = F_2 = F_3 = F_4 = 0$, whereas F_5 matches the optimal total tour duration reported in the literature for the corresponding TSPTW-M. This is an indication that the proposed algorithm performs well - at least under certain conditions. Also, this algorithm can be used to solve standard TSPTW instances. Note that our algorithm does not use the solution of the original plan of the underlying TSPTW as the starting solution.

We should mention that in the case where a solution to the Delayed

TSPTW exists where $F_1 = F_2 = F_3 = F_4 = 0$, then this problem is equivalent to the problem of the TSPTW where the objective is to minimize the time of arrival at the endpoint node. Hence the Delayed TSPTW can be considered to be a generalization of the TSPTW variant where the objective is to minimize the time of arrival at the endpoint node.

Note that there is a small difference between minimizing the total tour duration and minimizing the time of arrival at the endpoint node. This difference is connected to whether or not the vehicle departs right-away from the starting point, or if it waits before it starts its journey towards the first customer. Therefore, it is possible that the optimal solution to the TSPTW-M differs from the optimal solution to the TSPTW where the objective is to minimize the time of arrival at the endpoint node (given that the vehicle is free to depart at any time ≥ 0). However, in cases where the optimal solution to the TSPTW-M involves a solution where the vehicle departs at time 0 from the starting node, then this solution should match the optimal solution of the TSPTW with the objective to minimize the time of arrival at the endpoint node.

By looking at the percentage gaps in the 13th column of table 6.3 and considering the 27 instances of the Delayed TSPTW (i.e. excluding the 9 instances of the TSPTW), we can make the following observations: The percentage gap $\frac{G^{HA_1} - LB_G^{HA_1}}{LB_G^{HA_1}} \cdot 100\%$ between the objective value G^{HA_1} found in Heuristic Approach 1 (HA1) and the lower bound $LB_G^{HA_1}$ also found in HA1 for these 27 instances, had a mean of 3.79%, a median of 0.42%, a standard deviation of 8.19%, and was ranging from 0.02% up to 40.29%. Moreover, in 17 instances this percentage gap was below 0.82%, in 7 other instances it ranged between 2% and 7%, in 2 more instances it was between 11% and 15%, whereas in one instance this figure was 40.29%.

Furthermore, by examining the 14th column of the table, we can see that the percentage gap $\frac{G^{HA_1} - G^{EA_1}}{G^{EA_1}} \cdot 100\%$ between the objective value G^{HA_1} found in HA1 and the corresponding objective value G^{EA_1} found in Exact Approach 1 (EA1) for the 27 instances of the Delayed TSPTW, had a mean of -15.79%, a median of -2.64%, a standard deviation of 23.31%, and was ranging from -68.59% up to 1.94%. Moreover, in 20 out of 27 instances this percentage gap was negative, in 4 other cases it was zero, whereas in the remaining 3 instances it was positive. This means that in 20 out of 27 instances (i.e. in 74.07% of the cases), HA1 produced a better solution than EA1. For those 20 instances where HA1 produced better results, this percentage gap had a mean of -21.45% (as well as a median of -11.82% and a standard deviation of 24.76%). Also, the four percentage values that equal to zero indicate that the solution found by HA1 matches precisely the one from EA1 in 4 more instances (i.e. in 14.81% of the cases); these are instances where the optimal solution was found by both methods. For the remaining 3 instances (i.e. in 11.11% of the cases considered) where the percentages were positive, the solution found by EA1 was optimal and outperformed the one found by HA1; however, in each one of these 3 cases the solution found by HA1 lies within 2% of the proven optimal. In fact, the exact figures of the percentage gaps for these three instances were 0.01%, 0.67% and 1.94%, with a mean of 0.87%.

Finally, by examining the last column of the table, we can see that the percentage gap $\frac{G^{HA_1}-LB_G^{EA_1}}{LB_G^{EA_1}} \cdot 100\%$ between the objective value G^{HA_1} found in HA1 and the lower bound $LB_G^{EA_1}$ found in EA1 for these 27 instances, had a mean of 13.38%, a median of 8.45%, a standard deviation of 17.95%, and was ranging from 0.00% up to 69.81%.

Comparing the lower bounds of the objective G that were found by the two approaches, we can see that in 19 out of 27 instances of the Delayed TSPTW (i.e. in 70.37% of the cases), the lower bound $LB_G^{HA_1}$ found by HA1 was greater and thus better than the lower bound $LB_G^{EA_1}$ found by EA1.

We should mention that the runtime of HA1 had a mean of 410.1 seconds (6 minutes 50.1 seconds), ranging from 8 seconds to 1835.1 seconds (30.6 minutes). This was below 10 minutes in 20 out of 27 instances (i.e. in 74.07% of the cases), was equal to 10 minutes plus 4 seconds in 1 other instance, and between 13 and 31 minutes in the remaining 6 instances.

Regarding table 6.3, note that whenever the value of LB_{F_1} is marked with an asterisk (*), i.e. in instances 3-B, 4-B, 5-C, 6-A and 7-A, the subroutine *additional late customers* did have some effect, since it was able to increase the lower bound $|I_{DL}|$ by at least 1 unit. Note also that the percentages in the last two entries of the corresponding rows of instances 9-A and 9-C, which are marked with two asterisks (**), indicate that these percentages were calculated using the best values of G^{EA_1} and $LB_G^{EA_1}$ from table 6.2, corresponding to the '2nd runs' of Exact Approach 1, where AIMMS was allowed to work for longer than 10 minutes.

To summarize, the results justify that HA1 performs fairly well and generally better than EA1. In other words, HA1 generally outperforms EA1, except perhaps when solving some very small instances where EA1 is able to find the proven optimal in real time. This does not mean that EA1 is not helpful though; since it provides lower bounds and it does find the proven optimal fast in some cases. In practice, a possible strategy is to have both approaches running in parallel and take the best solution out of the two methods in the available time.

Table 6.3: Experimental Results of *Heuristic Approach 1* for the *Lexicographic Delayed TSPTW with 5 objectives* (in MATLAB)

Instance name	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**run-time (sec)**	$\mathbf{G^{HA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_{F_1}}$	$\mathbf{LB_{F_3}}$	**O/ F/ I**	$\mathbf{LB_G^{HA_1} := \beta_1 \cdot LB_{F_1} + +\beta_3 \cdot LB_{F_3}}$	**% of $\mathbf{G^{HA_1}}$ from $\mathbf{LB_G^{HA_1}}$**	**% of $\mathbf{G^{HA_1}}$ from $\mathbf{G^{EA_1}}$**	**% of $\mathbf{G^{HA_1}}$ from $\mathbf{LB_G^{EA_1}}$**
Class 1														
1-T (TSPTW)	0	0	0	0	387	33.1	0.387	–	–	O	–	–	–	–
1-A	5	1	695	89	387	39.9	510695089.387	5	167	F	500167000	2.10%	1.94%	1.94%
1-B	5	0	800	0	451	22	500800000.451	5	240	F	500240000	0.11%	0.0106%	0.0106%
1-C	4	0	344	0	458	9.3	400344000.458	4	180	O	400180000	0.04%	0.000%	0.000%
Class 2														
2-T (TSPTW)	0	0	0	0	510	525.3	0.510	–	–	O	–	–	–	–
2-A	7	0	1724	0	583	287.2	701724000.583	5	193	F	500193000	40.29%	-12.50%	40.29%
2-B	14	1	2134	5	564	141.1	1412134005.564	14	934	F	1400934000	0.80%	0.665%	17.60%
2-C	15	0	3100	0	595	64.5	1503100000.595	15	1904	F	1501904000	0.08%	-0.00186%	0.0869%
Class 3														
3-T (TSPTW)	0	0	0	0	535	220	0.535	–	–	O	–	–	–	–
3-A	8	0	634	0	570	100.3	800634000.570	7	162	F	700162000	14.35%	-11.13%	14.35%
3-B	13	0	2102	0	596	43.5	1302102000.596	13*	617	F	1300617000	0.11%	-0.0619%	8.45%
3-C	15	0	3956	0	682	30	1503956000.682	15	2165	F	1502165000	0.12%	-0.00871%	0.133%
Class 4														
4-T (TSPTW)	0	0	0	0	400	4	0.400	–	–	O	–	–	–	–
4-A	4	0	715	0	400	13.6	400715000.400	4	101	O	400101000	0.15%	0.000%	0.000%
4-B	7	0	479	0	419	7.8	700479000.419	7*	327	O	700327000	0.02%	0.000%	0.000%
4-C	6	0	938	0	445	15.7	600938000.445	6	514	O	600514000	0.07%	0.000%	0.000%
Class 5														
5-T (TSPTW)	0	0	0	0	674	1529.3	0.674	–	–	O	–	–	–	–
5-A	16	0	2957	0	674	1043.6	1602957000.674	15	580	F	1500580000	6.82%	-64.66%	33.55%
5-B	24	0	6237	0	716	551	2406237000.716	24	1878	F	2401878000	0.18%	-33.46%	14.50%

(Continued on the next page)

Table 6.3 – *Continued from the previous page*

Instance name	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**run-time (sec)**	$\mathbf{G^{HA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_{F_1}}$	$\mathbf{LB_{F_3}}$	**O/ F/ I**	$\mathbf{LB_G^{HA_1} := \beta_1 \cdot LB_{F_1} + \beta_3 \cdot LB_{F_3}}$	**% of** $\mathbf{G^{HA_1}}$ **from** $\mathbf{LB_G^{HA_1}}$	**% of** $\mathbf{G^{HA_1}}$ **from** $\mathbf{G^{EA_1}}$	**% of** $\mathbf{G^{HA_1}}$ **from** $\mathbf{LB_G^{EA_1}}$
5-C	27	0	6609	0	796	147.4	2706609000.796	27*	3678	F	2703678000	0.11%	-0.0530%	3.97%
Class 6														
6-T (TSPTW)	0	0	0	0	729	1924.4	0.729	–	–	O	–	–	–	–
6-A	19	6	5749	649	730	801	1965749649.730	19*	894	F	1900894000	3.41%	-62.23%	14.91%
6-B	29	2	6331	66	809	331.2	2926331066.809	29	2606	F	2902606000	0.82%	-21.85%	12.46%
6-C	29	0	11050	0	893	87.1	2911050000.893	29	5376	F	2905376000	0.20%	-0.0158%	0.209%
Class 7														
7-T (TSPTW)	0	0	0	0	827	3985.1	0.827	–	–	O	–	–	–	–
7-A	28	8	8292	981	782	1835.1	2888292981.782	27*	1340	F	2701340000	6.92%	-60.44%	69.81%
7-B	40	2	11273	140	928	1209.7	4031273140.928	39	4170	F	3904170000	3.26%	-30.82%	11.87%
7-C	41	1	12635	20	1025	347.6	4122635021.025	41	7160	F	4107160000	0.38%	-4.61%	8.30%
Class 8														
8-T (TSPTW)	0	0	0	0	824	3041.4	0.824	–	–	O	–	–	–	–
8-A	22	0	3176	0	824	1498.7	2203176000.824	21	864	F	2100864000	4.87%	-68.59%	56.19%
8-B	29	0	8168	0	874	1154.5	2908168000.874	26	2884	F	2602884000	11.73%	-37.19%	26.31%
8-C	34	1	10031	79	962	256.7	3420031079.962	34	5774	F	3405774000	0.42%	-2.64%	0.445%
Class 9														
9-T (TSPTW)	0	0	0	0	586	1462.1	0.586	–	–	O	–	–	–	–
9-A	10	4	2206	304	627	603.9	1042206304.627	10	339	F	1000339000	4.19%	-5.55% **	14.48% **
9-B	21	1	6397	170	704	320.7	2116397170.704	21	1643	F	2101643000	0.70%	-13.20%	11.31%
9-C	25	0	5443	0	700	109.6	2505443000.700	25	3319	F	2503319000	0.08%	-0.0167% **	0.0936% **

6.5.3 Experimental Results of Heuristic Approach 3

In this subsection we discuss the experimental results of Heuristic Approach 3 (HA3) for solving the Multi-Objective Delayed TSPTW with 3 objectives. This approach, which was described in section 6.4, employs Tabu Search within the framework of the Epsilon Constraint Method, to find an approximate representative subset of the set of non-dominated solutions for the 3-objective variant of problem 2. The algorithm was implemented in MATLAB and the respective results can be found in tables A.1 - A.27 of Appendix A.

We present a separate table for each instance. Each table contains at its top part, which is composed of the first two rows, the following information: the runtime in seconds, the lower bounds LB_{f_1} and LB_{f_2} for the component objectives f_1 and f_2 respectively, and a reference value for f_3. In the main part of the table, which spans from the third row until the last row of the table, all the non-dominated solutions found for the instance are listed. Each solution corresponds to one row, for which we record the solution's serial number and the corresponding values of the three objectives f_1, f_2 and f_3. Each entry of f_1 is followed by a percentage, which is calculated as $\frac{f_1 - f_1^{min}}{f_1^{max} - f_1^{min}} \cdot 100\%$, where f_1^{max} and f_1^{min} stand, respectively, for the maximum (worst) and minimum (best) values of f_1, amongst all non-dominated solutions found. The percentages which follow the entries of f_2 and f_3 are calculated in a similar way. In each table, all non-dominated solutions found are sorted in ascending order of f_1 and, in the case where we have solutions with the same value for f_1, these are sorted in ascending order of f_2, and so on.

Regarding these tables, note that whenever the value of LB_{f_1} is marked with an asterisk (*), i.e. in instances 3-B, 4-B, 5-C, 6-A and 7-A, the subroutine *additional late customers* did have some effect, since it was able to increase the lower bound $|I_{DL}|$ by at least 1 unit. Also, the *reference value for* f_3 is the minimum value that the heuristic was able to find at the stage when it was searching for the minimum of f_3, while neglecting the other two objectives.

By examining the tables of results we can make some observations regarding the trade-offs between the three objectives f_1, f_2 and f_3. Specifically, in instances 1-A, 2-B, 2-C, 3-C, 5-B, 5-C, 6-B, 6-C, 7-B, 8-B, 8-C and 9-B, a solution with the maximum value of f_1 gives the minimum value of f_2. Also, in instances 2-B, 3-C, 5-B, 5-C, 6-A, 6-B, 6-C, 7-A, 7-C, 8-C, 9-A, 9-B and 9-C, a solution with the minimum value of f_1 gives the maximum value of f_2. Furthermore, in instances 3-A, 4-A, 5-C, 6-C, 7-A and 9-A, a solution with the maximum value of f_1 gives the minimum value of f_3 (although the exact opposite occurs in instance 8-B, where the solution with the maximum value of f_1 gives the maximum value of f_3). Similarly, in instances 2-A, 2-B, 3-B, 3-C, 5-C, 6-A, 7-A, 8-C, 9-A, 9-B and 9-C, a solution with the minimum value of f_1 gives the maximum value of f_3. Therefore, solutions where f_1 takes its minimum value (or simply a low value), do not necessarily give the minimum value (or even a low value) for f_2 or f_3; and vice versa. In fact, it seems that there might be an indication of a negative correlation between f_1 and f_2 (which may be surprising to some extent). However, we will make no attempt of drawing any general conclusions about correlations between pairs of component objectives. This is something that can be further investigated, by performing more experiments and appropriate statistical hypothesis tests, and is left open for future research.

Note that in the observations reported in the previous paragraph, instances 1-C, 4-B and 4-C were not included, because only 3 or fewer non-dominated solutions were reported for these instances and can therefore be considered degenerate cases. In other words, we cannot safely draw any conclusions regarding the trade-offs between the objectives by studying these particular instances, so we decided that it would be better not to include them in the relevant discussion of the previous paragraph.

Now, comparing HA3 with HA1 may not be very meaningful, because these two approaches solve different variants of problem 2. However, we will briefly compare their runtimes. For the 27 instances of the Delayed TSPTW, the ratio $\frac{runtime\ of\ HA_3}{runtime\ of\ HA_1}$ ranges from 0.59 up to 26.2, with an average of 5.2 and a standard deviation of 5.3. Therefore, the average runtime of HA3 is 5.2 times as large as the runtime of HA1. In one instance (instance 1-C), where the ratio equals to 0.59 and is therefore below 1, HA3 is actually faster than HA1. In 10 other instances the above ratio ranges between 1.1 and 2.82. In 12 other instances, this figure lies between 3 and 7, whereas in the remaining 4 instances this ranges between 10 and 26.2.

These two heuristics are different and involve different sets of parameters that control, e.g., the numbers of cycles of outer iterations (or restarts). The values chosen for these parameters may have an actual impact on the runtime of these two heuristics. First of all, in HA1 we use d different restarts of the algorithm, whereas in HA3 we use $\Phi \times \Theta$ cycles of outer iterations. Second of all, the values of the parameters J_1 and J_2 that control the number of Tabu Search iterations without an improvement before the two main steps of the search terminate, also greatly affect the runtime. This is especially true for the parameter J_2 which controls the slowest part of the Tabu Search. For instance, in our experiments with HA1 we used $J_2 = 49$, whereas in HA3 we used $J_2 = 4$ and $J_2 = 9$ at steps 4 and 6 of the algorithm, respectively. Therefore, in our tests with HA1 we allowed a higher number of inner iterations without an improvement before Tabu Search terminates, compared to

HA3. If instead we had used the same value of $J_2 = 49$ in HA3, it would take much longer to terminate.

Moreover, running HA1 with just 1 restart, i.e. with $d = 1$ (instead of $d = 6$ that we used in our experiments), is likely to give comparable results to those reported for HA1 (maybe slightly worse in some cases), but significantly faster, in approximately $\frac{1}{6}$th of the reported runtime. Further experimentation with HA1 and HA3 with different choices of the parameter values is left open for further research. Performing experiments with HA2 and comparing with HA3 is also left open for further research.

Lastly, HA3 aims to find a representative subset of the set of non-dominated solutions for the Delayed TSPTW with 3 objectives, rather than a single solution. Therefore, it takes longer to run than HA1, which only aims to find a single non-dominated solution for a particular set of weights. In some circumstances, the runtime for HA3 may be too long in practice, though it could be stopped before a full set of non-dominated solutions has been found, if one of those found is acceptable.

6.6 Conclusions

The *Delayed Traveling Salesman Problem with Time Windows* (in short, the *Delayed TSPTW*), also referred to as *Problem 2*, was introduced in chapter 5, along with notation, formulations and exact approaches. In this chapter we described three heuristic approaches (HA1, HA2 and HA3) for different variants of this problem. The first heuristic approach solves the *Delayed-TSPTW-Lex5*; i.e. the Delayed TSPTW with 5 objectives of lexicographic preference. The second one solves the *Delayed-TSPTW-Lex3*; i.e. the variant of problem 2 with 3 objectives of lexicographic preference. Both HA1 and HA2 involve aggregating the component objectives into a single objective function defined as their weighted sum, and use Tabu Search to find a single optimal or near-optimal solution, for a single choice of weights. Finally, HA3 solves the Delayed TSPTW with 3 objectives, without assuming or making use of any preference information concerning the component objectives. This approach uses Tabu Search with multiple restarts within the framework of the ϵ-Constraint Method, in order to create an approximate representation of the set of non-dominated solutions. The algorithm of HA3 calls the algorithm of HA2 several times, as a subroutine.

In section 6.5 we presented the experimental results for heuristic approaches 1 and 3, as well as the results derived from exact approach 1 (EA1), which was described in the previous chapter. The experimental results of EA1 imply that exact approaches based on the formulations presented in the previous chapter, cannot be directly used in a commercial solver to solve instances with more than 40 customers. Of course, the exact formulations presented in the previous chapter can be used in more sophisticated exact approaches or frameworks, e.g. in conjunction with branch-and-cut, to solve larger instances. This is left open for further research.

HA1 and EA1 both solve the same variant, namely the *Delayed-TSPTW-Lex5*. Comparison of the two methods showed that HA1 generally outperformed EA1. Also, in 70.37% of the instances considered, the lower bound that was found by HA1 was better than the one provided by AIMMS while using EA1. Furthermore, HA1 was tested on 9 standard TSPTW instances with success, since it was capable of finding the optimal solution in every single case. To sum up, HA1 generally performed well.

Finally, HA3 is a method that attempts to find, not just one solution, but instead a representative subset of the set of non-dominated solutions for the Delayed TSPTW with 3 objectives. It generally takes longer to run than HA1 and in some cases it may be impractical to use. However, by proper parameter calibration and perhaps some guidance from the decision maker that can help reduce the ranges of acceptable values for the component objectives, it is possible that HA3 can find a small number of different Pareto-optimal solutions in a short time, thus aiding the decision maker by providing several alternatives.

Chapter 7

Problem 3 - The Single-Commodity Delayed VRPTW: Formulations & Exact Approaches

7.1 Introduction

Suppose that we have created a feasible and optimal or near-optimal solution plan for a given instance of the single-commodity VRPTW with heterogeneous fleet. Assume that during the execution of this solution plan there is a major delay in one or more routes, so that if we keep following the original plan, at least one customer in each one of the affected routes will be served with a delay. This causes a disruption to the original plan, which is no longer feasible with respect to the time windows.

Obviously, one option is to keep following the original plan and visit the remaining customers as it was originally scheduled. However, this could cause a delay in serving some - or even all - of the remaining customers of the delayed vehicles. Can we revise the plan and have fewer customers served with a delay, or perhaps have shorter delays?

In general, servicing customers outside their promised time windows may cause substantial direct or indirect costs for the enterprise. For instance, the enterprise could suffer direct costs by having to reimburse potential dissatisfied customers because of the delay in delivery. In some extreme cases, a dissatisfied customer may be able to take legal actions against the company. Indirect costs include the cost of losing some customers, along with the long-term cost resulting from the potential criticisms from those dissatisfied customers to other potential or current customers of the enterprise. Indirect costs are hard to estimate and may be huge in some cases. Clearly the plan has to be revised, if possible, so that the negative effect of the disruption on the enterprise is minimized.

Essentially, once the disruption takes place we are facing a new problem, which is substantially different from the original VRPTW with heterogeneous fleet, with different constraints and objectives. In this chapter we will define this new problem and propose mathematical formulations and exact approaches which theoretically can be used to solve the problem to optimality.

7.2 The easy plan

In general, there is an obvious and very simple way to construct a solution to the problem, following the so-called *easy plan* which is described below.

On the one hand, once the disruption occurs, all vehicles that have not experienced a delay will continue following their routes as scheduled in the original plan, i.e. serve all of their remaining customers and finish at the depot. On the other hand, any vehicle k that has fallen behind schedule will reschedule as follows: It will skip the first few customers and be diverted to visit next that specific customer which, if visited first after rescheduling, then all of the following customers in the order of the original plan will be served within their requested time windows. Then the vehicle will visit the first few customers that were skipped in their original order, and finish at the endpoint node. This way, only the customers that were at the first part of the original route and were skipped, may be served with delay.

In more detail, suppose that vehicle k has fallen behind schedule. Assume that $R_k = (R_{k,1}, R_{k,2}, ..., R_{k,l_{R_k}})$ is the remaining part of the route of vehicle k at the time of rescheduling, with l_{R_k} denoting the length of route R_k. Here, $R_{k,1}$ denotes the starting node (or current position) of vehicle k at the time of rescheduling, and $R_{k,l_{R_k}}$ denotes the endpoint node, whereas nodes $R_{k,2}, R_{k,3}, ..., R_{k,l_{R_k}-1}$ represent customer nodes. With trivial calculations, we can find the smallest index j such that, if vehicle k skips the first j customers and goes directly from its starting node $R_{k,1}$ to its $(j+1)$-th customer $R_{k,j+2}$, then all of the following customers will be served within their promised time windows. In that case, vehicle k can follow the revised route $R'_k = (R_{k,1}, R_{k,j+2}, R_{k,j+3}, ..., R_{k,l_{R_k}-1}, R_{k,2}, R_{k,3}, ..., R_{k,j+1}, R_{k,l_{R_k}})$. In other words, the vehicle will be redirected from its starting position $R_{k,1}$ to serve the $(j+1)$-th customer $R_{k,j+2}$ and all of its following customers $R_{k,j+3}, ..., R_{k,l_{R_k}-1}$ within their time windows, in the same order as scheduled in the original plan. Then it will go back to serve the first customer $R_{k,2}$ and all of its following customers $R_{k,3}, ..., R_{k,j+1}$, possibly with some delay, in the same order as scheduled in the original plan, before finishing its route at the endpoint node $R_{k,l_{R_k}}$. Finally, this process is repeated separately for each vehicle that experiences a delay.

In general, the easy plan described here should be preferable to staying with the original plan. However, rescheduling in a more sophisticated way, may lead to a new plan where fewer customers are served outside their time windows and where the amount of violation of the time windows may be lower, compared to following either the original plan or the easy plan. Therefore, in the remaining part of this chapter, as well as in the following chapter, we describe more appropriate methods to solve the problem.

7.3 Problem description and notation

We will now describe in detail *the Single-Commodity Delayed Vehicle Routing Problem with Time Windows (SCD-VRPTW)*, or in short *problem 3*, which is the problem that we study in this chapter.

Suppose that we have created a feasible and optimal or near-optimal solution plan for a given instance of the VRPTW with heterogeneous fleet, where vehicles carry a single commodity. Assume that, during the execution of this solution plan, there is a major delay in one or more of the routes, causing a disruption to the plan. Specifically, assume that at some time instant T, the operations team realizes that there has been a severe delay in at least one of the routes, and the respective vehicles have fallen behind schedule. In the context of this problem, by the term *severe delay* we refer to a delay which causes a violation of the time-window constraint for at least one of the remaining customers in each one of the affected routes, if the original routes are followed as planned.[1] This means that the remaining part of the original plan is now infeasible, with respect to the time windows. Therefore, the plan has to be revised in order to minimize the negative impact of the disruption. The new problem that arises is the *SCD-VRPTW*, i.e. *the Single-Commodity Delayed VRPTW* with heterogeneous fleet (or *the disrupted VRPTW with non-customer-specific orders and vehicle delay*). The operations team can use the methods proposed in this chapter and in the following one, to revise the plan in an appropriate way.

[1]This distinguishes the case of a severe delay, from the case of a *minor delay* in a route. In the latter case the vehicle has fallen slightly behind schedule, but by continuing to follow the original plan, the vehicle can still serve all of the remaining customers within their requested time windows. For example, in case of a flat tire in a medium-sized motor vehicle (e.g. van), the driver may be able to replace it with a spare one within 15-30 minutes. If this small delay does not cause a violation of the time windows of any of the remaining customers, then it does not account for a severe delay, and the driver can continue on its route as originally planned.

The original problem is assumed to be a VRPTW with a heterogeneous fleet, in which vehicles depart from a single depot (or from different positions, more generally), and are required to serve a set of customers within their requested time windows and return to the depot. We assume that the vehicles deliver a single commodity, such as oil or gas, so that any vehicle can serve any customer. We also assume that there are no extra vehicles available and that all customers have to be served. Furthermore, we assume that all vehicles had departed fully loaded from the depot (or from their starting positions, more generally). Having the vehicles departing from the depot fully loaded is a strategy which provides flexibility in case of such a disruption. If, on the other hand, the vehicles depart with only the total load they are supposed to deliver, then generally it is less likely that the operations team will be able to provide a good revised plan in case of a disruption (especially if the demands of the customers differ from one another).

Assume that during the execution stage of the original plan, at some time instant T the operations team realizes that a severe delay has occurred in at least one of the routes and starts constructing the new plan. Suppose that T_0 is the estimated time needed for the operations team to construct a new plan and communicate this to the drivers.[2] Then the position of each vehicle at time $T + T_0$ can be estimated. Suppose that there are d active vehicles and let $K = \{1, 2, \dots, d\}$ be the set of all these vehicles. We denote by N_k the node representing the expected or estimated position of vehicle k at time $T + T_0$, for each $k \in K$. We wish to define T_k as the time of departure of vehicle k from node N_k. For this, we need to consider the two possible cases.

In the first case, at time $T + T_0$ vehicle k is projected to be at a customer node. In this case, it may happen that by time $T+T_0$, vehicle k is expected to have just arrived at the customer, to have just finished servicing the customer, or to have already started but not finished servicing the customer. Therefore, in the case where at time $T + T_0$ vehicle k is projected to be at a customer node, we define T_k to be the expected time by which servicing the customer will have finished (where $T_k \geq T + T_0$). We also define N_k to be the node corresponding to the geographic location of that customer.

In the second case, at time $T + T_0$ vehicle k is projected to be on the road, traveling from one node to another. In this case, we can define node N_k to be a new node that corresponds to the expected or estimated geographic location of the vehicle at time $T + T_0$ (if following the route of the original plan), and also define $T_k := T + T_0$.

Thus, in any case, each vehicle k is expected to be ready to depart from

[2]Obviously, in the case where the expected time necessary to construct the new plan and communicate it to the drivers is negligible, one can simply set $T_0 = 0$.

node N_k at time T_k, where $T_k \geq T + T_0$, for all $k = 1, 2, ..., d$. Note that node N_k and time T_k can be uniquely defined, for each $k = 1, 2, ..., d$. In the case where node N_k represents a customer location, since by time T_k the particular customer will have already been served, he or she is removed from the set containing the customers that are still unserved by time $T + T_0$ when the revised plan starts.[3] Note that in all cases, node N_k will have zero service time and zero demand. Also, we assume that $T \geq 0$ and $T_0 \geq 0$; therefore $T_k \geq 0$ for all $k \in K$ (since $T_k \geq T + T_0$).

Now, having chosen the starting node N_k and the corresponding time of departure T_k of vehicle k from its starting node N_k, for all $k = 1, 2, ..., d$, we can define the set of customers for the revised plan. Specifically, the set of customers I_C for the revised plan is defined as the union of the sets containing the customers of vehicle k that are still unserved by time T_k, excluding any customers whose nodes were marked as starting nodes N_k (if any), for all $k = 1, 2, ..., d$. Suppose that there are n such customers. We relabel the corresponding customer nodes as nodes $1, 2, ..., n$. From now on, we will refer to the set $I_C = \{1, 2, ..., n\}$ as 'the set of customers that are unserved at time $T + T_0$', or simply as 'the set of customers' (in the revised plan). Furthermore, from now each starting node N_k will also be denoted as node $n + k$.

Let $I_D = \{n + 1, n + 2, \ldots, n + d\}$ be the set containing the starting positions of the vehicles. Let node 0 denote the depot, which marks the endpoint of each route. Let $I = I_C \cup I_D \cup \{0\}$ be the set of all nodes, which can be written explicitly as $I = \{0, 1, \ldots, n, n + 1, \ldots, n + d\}$.

Let $A = A_1 \cup A_2$ be the set of arcs, where $A_1 = \{(i, j) : i \in I_D,\ j \in I_C \cup \{0\}\}$ and $A_2 = \{(i, j) : i \in I_C,\ j \in I_C \cup \{0\}, i \neq j\}$. Clearly $A \subseteq I \times I$. Let $\Delta^+(i) = \{j \in I : (i, j) \in A\}$ be the set of nodes that are directly reachable from node i, for all $i \in I$. Let $\Delta^-(i) = \{j \in I : (j, i) \in A\}$ be the set of nodes from which node i is directly reachable, for all $i \in I$. The *Single-Commodity Delayed VRPTW* is formulated on the directed graph $\Gamma = (I, A)$.

Each active vehicle $k \in K$ will depart from node $n + k$ at time T_k, will then visit some of the customers in the set I_C and will finish at the endpoint node 0 (depot). Each node in I_C must have exactly 1 vehicle entering and the same vehicle leaving the node. Each node in I_D must have no vehicles entering and exactly 1 vehicle leaving the node. The depot (node 0) should have exactly d vehicles entering and no vehicles leaving.

[3]To be more precise, by saying 'the set containing the customers that are still unserved by time $T + T_0$ when the revised plan starts', we actually refer to the union of the sets containing the customers of vehicle k that are still unserved by time T_k, for all $k = 1, 2, ..., d$.

Let x_{ijk} be a binary variable representing the number of times that vehicle k traverses arc (i, j), for all $(i, j) \in A,\ k \in K$. Let w_{ik} be a nonnegative variable representing the time of start of service of node i by vehicle k, in the case where vehicle k actually visits node i, for all $i \in I,\ k \in K$. If node i is not visited by vehicle k, then we set $w_{ik} = 0$. Note that $w_{ik} = 0$ means that either vehicle k does not visit node i, or that it visits node i at time 0 (although the latter case can never happen if $T_k > 0$). Obviously, if node i is visited by vehicle k, then we should have $w_{ik} \geq T_k$. Also, let W_i be a nonnegative variable representing the time of start of service of node i, for $i \in I \setminus \{0\}$. Note that W_0 is not defined. Obviously, we should have $W_i \geq T_{min}\ \forall\ i \in I_C$, where $T_{min} := \min\{T_k | k \in K\}$.

Let t_{ij} be the travel time from node i to node j, for all $(i, j) \in A$. Assume that $t_{ij} \geq 0\ \ \forall\ (i, j) \in A$. Let $[a_i, b_i]$, s_i and D_i be the time window, service time and demand associated with node i, respectively, for all $i \in I$. We set $s_i = 0$ and $D_i = 0$ for all nodes $i \in I_D \cup \{0\}$; i.e. the starting nodes and endpoint node all have zero service time and zero demand. Let C_k be the quantity of the commodity carried by vehicle k at time instant T_k, for $k \in K$. Assume that $C_k \geq 0\ \forall\ \ k \in K$.

Let M be a large positive constant (e.g. $M = 10^{10}$) and let ϵ be a small positive constant (e.g. $\epsilon = 10^{-8}$). The quantities d, n, M, ϵ, C_k's, T, T_0, T_k's, t_{ij}'s, D_i's, s_i's, a_i's and b_i's are assumed to be known parameters.

For all $i \in I_C$, we introduce the variables μ_i, λ_i, ρ_i and σ_i as follows:
(i) The variable μ_i is a binary variable indicating whether or not customer i is served with a delay. For this, μ_i is equal to 1 if and only if $W_i > b_i$ (i.e. if customer i is served with a delay, that is, later than b_i); otherwise, $\mu_i = 0$ (i.e. $\mu_i = 0$ if and only if $W_i \leq b_i$).
(ii) The variable λ_i is a binary variable indicating whether or not customer i is served earlier than promised. For this, λ_i is equal to 1 if and only if $W_i < a_i$ (i.e. if customer i is served earlier than a_i, that is, earlier than promised); otherwise, $\lambda_i = 0$ (i.e. $\lambda_i = 0$ if and only if $W_i \geq a_i$).
(iii) The variable ρ_i represents the amount of lateness of customer i, if this customer is served with a delay (later than b_i); otherwise, $\rho_i = 0$. For this, ρ_i is equal to $W_i - b_i$ if $W_i > b_i$; otherwise, ρ_i is equal to 0 if $W_i \leq b_i$. Mathematically, this can be expressed as:

$$\rho_i = (W_i - b_i)\mu_i = \begin{cases} 0 & \text{if } W_i \leq b_i \qquad (\Leftrightarrow \mu_i = 0) \\ W_i - b_i & \text{if } W_i > b_i \qquad (\Leftrightarrow \mu_i = 1) \end{cases}$$

(iv) The variable σ_i represents the amount of earliness of customer i, if this customer is served early (earlier than a_i); otherwise, $\sigma_i = 0$. For this, σ_i is equal to $a_i - W_i$ if $W_i < a_i$; otherwise, σ_i is equal to 0 if $W_i \geq a_i$.

Mathematically, this can be expressed as:

$$\sigma_i = (a_i - W_i)\lambda_i = \begin{cases} 0 & \text{if } W_i \geq a_i \quad (\Leftrightarrow \lambda_i = 0) \\ a_i - W_i & \text{if } W_i < a_i \quad (\Leftrightarrow \lambda_i = 1) \end{cases}$$

7.4 Objectives

The *Single-Commodity Delayed VRPTW with 5 objectives (SCD-VRPTW-5)* is the problem of determining the appropriate allocation of customers to vehicles and the appropriate sequence in which each vehicle $k \in K$ has to visit its corresponding customers, once leaving the starting position $n + k$ and before arriving at the endpoint node 0 (the depot), as well as the appropriate times of start of service at each node, so that the following 5 goals are achieved: 1) have as few late customers as possible, 2) have as few early customers as possible, 3) have as low amount of total lateness as possible, 4) have as low amount of total earliness as possible, and 5) finish the routes as early as possible.

Therefore, we define the SCD-VRPTW-5 as a Multi-Objective Optimization (MOO) problem with the following 5 objectives:

1. Minimize $F_1 := \sum\limits_{i \in I_C} \mu_i$, i.e. minimize the number of customers served with a delay (later than b_i).
2. Minimize $F_2 := \sum\limits_{i \in I_C} \lambda_i$, i.e. minimize the number of customers served earlier than promised (earlier than a_i).
3. Minimize $F_3 := \sum\limits_{i \in I_C} \rho_i$, i.e. minimize the total amount of lateness (combined for all customers being served with a delay).
4. Minimize $F_4 := \sum\limits_{i \in I_C} \sigma_i$, i.e. minimize the total amount of earliness (combined for all customers being served earlier than promised).
5. Minimize $F_5 := \sum_{k \in K} w_{0,k}$, i.e. minimize the sum of arrival times of vehicles at the endpoint node. [4]

[4]The fifth objective is equivalent to minimizing the total travel time plus waiting time (i.e. total journey time), combined for all vehicles. As an alternative to objective 5, one can use the following objective:
Minimize $\sum\limits_{(i,j) \in A, k \in K} t_{ij} x_{ijk}$, i.e. minimize the total travel time combined for all vehicles, without including waiting times.

7.5 Preprocessing stage

Before solving the mathematical program which is presented in the following section, a preprocessing stage takes place. During this stage, we identify some customers that will be definitely served with a delay and some others that will definitely not be served earlier than promised. For this, we define *the set of definitely late customers*, $I_{DL} = \left\{ i \in I_C \;\middle|\; \min_{k \in K}(T_k + s_{n+k} + t_{n+k,i}) > b_i \right\}$ and *the set of definitely not early customers*, $I_{DNE} = \left\{ i \in I_C \;\middle|\; \min_{k \in K}(T_k + s_{n+k} + t_{n+k,i}) \geq a_i \right\}$. Since $a_i \leq b_i$ for all $i \in I$, it is trivial to show that $I_{DL} \subseteq I_{DNE}$.

7.6 General MOMILP Formulation for the SCD-VRPTW with 5 objectives

The more general variant of problem 3, namely *the Single-Commodity Delayed VRPTW with 5 objectives (SCD-VRPTW-5)* can be formulated as the Multi-Objective Mixed Integer Linear Program *(MOMILP 3A)* composed of equations (7.6.1) - (7.6.32), as follows:

$$\min \;\; F_1 := \sum_{i \in I_C} \mu_i \tag{7.6.1}$$

$$\min \;\; F_2 := \sum_{i \in I_C} \lambda_i \tag{7.6.2}$$

$$\min \;\; F_3 := \sum_{i \in I_C} \rho_i \tag{7.6.3}$$

$$\min \;\; F_4 := \sum_{i \in I_C} \sigma_i \tag{7.6.4}$$

$$\min \;\; F_5 := \sum_{k \in K} w_{0,k} \tag{7.6.5}$$

subject to:

$$\sum_{j \in \Delta^+(i)} x_{ijk} = \sum_{j \in \Delta^-(i)} x_{jik} \qquad \forall\, i \in I_C,\; k \in K \tag{7.6.6}$$

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1 \qquad \forall\, i \in I \setminus \{0\} \tag{7.6.7}$$

$$\sum_{j \in \Delta^-(0)} x_{j0k} = 1 \qquad \forall\ k \in K \tag{7.6.8}$$

$$\sum_{j \in \Delta^+(n+k)} x_{n+k,j,k} = 1 \qquad \forall\ k \in K \tag{7.6.9}$$

$$w_{ik} \leq M \cdot \sum_{j \in \Delta^+(i)} x_{ijk} \quad \forall\ i \in I \setminus \{0\},\ k \in K \tag{7.6.10}$$

$$W_i = \sum_{k \in K} w_{ik} \qquad \forall\ i \in I \setminus \{0\} \tag{7.6.11}$$

$$w_{n+k,k} = T_k \qquad \forall\ k \in K \tag{7.6.12}$$

$$w_{ik} + s_i + t_{ij} - w_{jk} \leq (1 - x_{ijk})M \qquad \forall\ \ (i,j) \in A,\ k \in K \tag{7.6.13}$$

$$\sum_{i \in I_C \cup \{n+k\}} D_i \sum_{j \in \Delta^+(i)} x_{ijk} \leq C_k \qquad \forall\ \ k \in K \tag{7.6.14}$$

$$W_i - b_i \leq M\mu_i \qquad \forall\ i \in I_C \setminus I_{DL} \tag{7.6.15}$$

$$b_i - W_i + \epsilon \leq M(1 - \mu_i) \qquad \forall\ i \in I_C \setminus I_{DL} \tag{7.6.16}$$

$$\rho_i \leq M\mu_i \qquad \forall\ i \in I_C \tag{7.6.17}$$

$$\rho_i \leq (W_i - b_i) - (1 - \mu_i)(-M) \qquad \forall\ i \in I_C \tag{7.6.18}$$

$$\rho_i \geq (W_i - b_i) - (1 - \mu_i)M \qquad \forall\ i \in I_C \tag{7.6.19}$$

$$a_i - W_i \leq M\lambda_i \qquad \forall\ i \in I_C \setminus I_{DNE} \tag{7.6.20}$$

$$W_i - a_i + \epsilon \leq M(1 - \lambda_i) \qquad \forall\ i \in I_C \setminus I_{DNE} \tag{7.6.21}$$

$$\sigma_i \leq M\lambda_i \qquad \forall\ i \in I_C \tag{7.6.22}$$

$$\sigma_i \leq (a_i - W_i) - (1 - \lambda_i)(-M) \qquad \forall\ i \in I_C \tag{7.6.23}$$

$$\sigma_i \geq (a_i - W_i) - (1 - \lambda_i)M \qquad \forall\ i \in I_C \tag{7.6.24}$$

$$\mu_i + \lambda_i \leq 1 \qquad \forall\ i \in I_C \setminus I_{DNE} \tag{7.6.25}$$

$$\mu_i = 1 \qquad \forall\ i \in I_{DL} \tag{7.6.26}$$

$$\lambda_i = 0 \qquad \forall\ i \in I_{DNE} \tag{7.6.27}$$

$$x_{ijk} \in \{0, 1\} \qquad \forall\ \ (i, j) \in A,\ k \in K \tag{7.6.28}$$

$$w_{ik} \geq 0 \quad \forall\ i \in I,\ k \in K \tag{7.6.29}$$

$$\lambda_i, \mu_i \in \{0, 1\} \quad \forall\ i \in I_C \tag{7.6.30}$$

$$\rho_i, \sigma_i \geq 0 \qquad \forall\ i \in I_C \tag{7.6.31}$$

$$W_i \geq 0 \quad \forall\ i \in I \setminus \{0\} \tag{7.6.32}$$

A part of the above formulation is based on the multi-commodity network flow model for the VRPTW (e.g. see Toth & Vigo (2014)). Equations (7.6.1)-(7.6.5) define the 5 objectives, as described previously. Equation (7.6.6) ensures that the same vehicle enters and exits node i, $\forall\ i \in I_C$. Equation (7.6.7) ensures that exactly 1 vehicle leaves node i, $\forall\ i \in I \setminus \{0\}$. Therefore, so far we have established that exactly 1 vehicle enters and the same 1 vehicle exits node i, $\forall\ i \in I_C$. Also, we have established that exactly 1 vehicle exits node i, $\forall\ i \in I_D$. Equation (7.6.8) states that each vehicle must arrive at the endpoint node exactly once. This establishes that exactly d vehicles enter node 0. Equation (7.6.9) states that each vehicle k departs from node $n + k$, for all $k \in K$. Constraint (7.6.10) states that if vehicle k does not visit node i, then $w_{ik} = 0$, for all $i \in I \setminus \{0\}, k \in K$. Constraint (7.6.11) gives the definition of the variables W_i, for $i \in I \setminus \{0\}$ (W_0 is not defined). Constraint (7.6.12) states that vehicle k will start servicing node $n + k$ at time T_k. Constraint (7.6.13) guarantees schedule feasibility with respect to time considerations. Constraint (7.6.14) is the capacity constraint. Constraints (7.6.15) - (7.6.19) relate any potential violation of the upper limits of the time windows (due dates) with the variables μ_i and ρ_i. Constraints (7.6.20) - (7.6.24) relate any potential violation of the lower limits of the time windows (ready times) with the variables λ_i and σ_i. Constraint (7.6.25) states that no customer can violate both the lower limit and the upper limit of the time windows; i.e. no customer can be both an early and a late customer. Note that constraint (7.6.25) is not necessary for the model; however, it is a valid cut that reduces the feasible region, and is therefore included. Constraint (7.6.26) fixes the variables $\mu_i = 1$ for those customers who are identified as 'definitely late customers' during the preprocessing stage. Similarly, constraint (7.6.27) fixes the variables $\lambda_i = 0$ for those customers who are identified as 'definitely not early customers' during the preprocessing stage. Finally, constraints (7.6.28) - (7.6.32) give the ranges of the variables.

Note that an additional constraint that should be satisfied but is not really necessary for the model (i.e. another valid cut) is the following:

$$W_i \geq T_{min} \qquad \forall\, i \in I_C \tag{7.6.33}$$

We recognize that, since one of the objectives is to minimize $\sum_{i \in I_C} \mu_i$, constraint (7.6.16) is not needed. Similarly, since one of the objectives is to minimize $\sum_{i \in I_C} \lambda_i$, constraint (7.6.21) is not needed. The same argument applies when minimizing a weighted sum of the five component objectives, with positive weights. However, even if these constraints may be regarded as redundant, they are valid cuts and are thus included as part of the main formulation. Also, they were included when performing our experiments. The task of experimenting both with and without these cuts and assessing the performance, in order to decide which version of the formulation produces results faster, is left open for future researchers.

7.7 A MILP formulation with a single aggregated objective for the SCD-VRPTW-5

One way to address the Single-Commodity Delayed VRPTW with 5 objectives, is to combine these 5 component objectives into a single aggregated objective function, defined as their weighted sum.

Specifically, if in the formulation (MOMILP 3A) of section 7.6 we replace the 5 component objectives (7.6.1)-(7.6.5) with the single objective function defined by equation (7.7.1) shown below, and keep all other constraints (7.6.6)-(7.6.32), then we get the Mixed Integer Linear Programming formulation *(MILP 3B)* for the SCD-VRPTW-5, with the following aggregated objective function:

$$\text{minimize} \quad \beta_1 \sum_{i \in I_C} \mu_i + \beta_2 \sum_{i \in I_C} \lambda_i + \beta_3 \sum_{i \in I_C} \rho_i + \beta_4 \sum_{i \in I_C} \sigma_i + \beta_5 \sum_{k \in K} w_{0,k} \tag{7.7.1}$$

Note that equation (7.7.1) can equivalently be written, either as equation (7.7.2), or in a more concise form as equation (7.7.3):

$$\text{minimize} \quad \sum_{i \in I_C} (\beta_1 \mu_i + \beta_2 \lambda_i + \beta_3 \rho_i + \beta_4 \sigma_i) + \beta_5 \sum_{k \in K} w_{0,k} \tag{7.7.2}$$

$$\text{minimize} \quad \sum_{i=1}^{5} \beta_i F_i \tag{7.7.3}$$

The weights β_i are assumed to be known parameters, tuned according to the decision maker's guidelines, where $\beta_i \in [0, +\infty)$ $\forall$ $i = 1, 2, ..., 5$. In general, the β_i's do not necessarily need to be normalized.

By varying the values of the weights β_i's, with $\beta_i > 0$ for all $i = 1, \ldots, 5$ and $\sum_{i=1}^{5} \beta_i = 1$, and solving (MILP 3B), we can get different solutions to the *Multi-Objective Single-Commodity Delayed VRPTW with 5 objectives* (SCD-VRPTW-MOO-5, or simply SCD-VRPTW-5), all of which will belong to the Pareto front. This will be further discussed in section 7.10 when describing *Exact Approach 2*.

Note that by solving (MILP 3B) with $\beta_l = 1$ and $\beta_r = 0$ $\forall$ $r = 1, ..., 5$, $r \neq l$, and repeating this for each $l = 1, ..., 5$, we can get lower bounds for each one of the 5 component objectives $F_1, ..., F_5$.

7.8 Notes

1. Problem 3 can be considered to be a generalization of the VRPTW. If we let $T = 0$, $T_0 = 0$, $T_k = 0$, nodes $n + k$ to represent the same geographic location as node 0 (i.e. copies of the depot) for all $k \in K$, and if we define C_k as the capacity of vehicle k, then the problem reduces to a VRPTW with heterogeneous fleet, with the objective to minimize the sum of arrival times at the depot. If additionally we let $C_k = C$ for all $k \in K$, then the problem reduces further to a regular VRPTW. Although in problem 3 the time windows are treated as soft constraints, in the presence of a solution that respects all time windows, the last reduction should make the problem equivalent to a VRPTW with hard time windows. Therefore, problem 3 is NP-hard.

2. If we let $T_k = 0$ and define C_k as the capacity of vehicle k, for all $k \in K$, then the problem reduces to an Open-VRPTW with heterogeneous fleet, with the objective to minimize the sum of arrival times at the depot.[5]

[5]Note that here, by the term *Open-VRPTW* we refer to the variant of the VRPTW where routes start from a fixed location and finish at the depot.

3. We are assuming that, in the original plan, the time of start of service of customer i was supposed to lie within the time window $[a_i, b_i]$. This time window was a hard constraint in the original problem, but in the new problem after disruption it is treated as a soft constraint.

4. If in the original problem customer i was promised to be served at a specific time instant τ_i^*, instead of a time window, we can define the respective time window as $[a_i, b_i] = [\tau_i^*, \tau_i^*]$. In fact, if in the original problem customer i was supposed to be served at a specific time instant τ_i representing his or her *estimated time of start of service*, and if ξ_1 and ξ_2 are non-negative numbers representing the amount of time allowed to deviate below and above the estimated time of start of service without being considered an early arrival or a delay, respectively, then this can be converted into a time window in the new problem, by setting $[a_i, b_i] = [\tau_i - \xi_1, \tau_i + \xi_2]$.

5. Suppose that the operations team decides that customer i would not mind having a delivery time outside their promised time window, either earlier or later by some specific amount of time. For instance, this decision may be taken after communicating and reaching an agreement with the customer. Then the operations team can expand or update their time window $[a_i, b_i]$ accordingly, and use the updated time window when constructing the revised plan.

6. In our experiments, for the sake of simplicity, we assume that the Cartesian coordinates (X_i, Y_i) of each node $i \in I$ are also given as an input, which are used to calculate the travel time $t_{i,j}$ between each pair of nodes i and j as the Euclidean distance between the nodes, i.e. as $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$, rounded to 2 decimal places.

7.9 Exact Approach 1: Lexicographic Exact Approach for the 5-objective variant

A relatively simple approach to solve the multi-objective optimization problem described in this chapter, is to use the Lexicographic Preference method. For this, if we further assume that there is a lexicographic preference of the five component objectives, so that objectives $F_1, F_2, ..., F_5$ are in strict descending order of importance, then the more complex multi-objective problem simplifies to a multi-criteria problem. The objective vectors $(F_1, F_2, ..., F_5)$ are first compared on the most important component, namely F_1, and only in case of equality the next most important component (F_2) is considered, and so on.

Thus, we can solve the *Single-Commodity Delayed VRPTW with 5 objectives of Lexicographic Preference (SCD-VRPTW-Lex5)* by using formulation (MILP 3B) of section 7.7 with a single aggregated objective (equation (7.7.1)) and with a single choice of weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ so that $\beta_1 >> \beta_2 >> \beta_3 >> \beta_4 >> \beta_5 > 0$. For instance, in our experiments we used the following set of weights: $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (10^{10}, 10^8, 10^4, 1, 10^{-4})$. This can be implemented in an optimization solver (e.g. Cplex or AIMMS) to directly solve different problem instances. We will be referring to this method as *Exact Approach 1* for solving Problem 3.

We implemented formulation (MILP 3B) in AIMMS and used Exact Approach 1 to solve a number of instances which were created for the Single-Commodity Delayed VRPTW. These instances are divided into 5 classes A, B, C, D and E, with 11, 100, 81, 54 and 77 customers respectively. In classes A, C and D we have 3 vehicles, whereas in classes B and E we have 4 and 10 vehicles, respectively. More details about these instances can be found in section 8.5.

The experimental results of Exact Approach 1 are presented in table 8.5 of the following chapter, along with results from the heuristic approaches, comparisons and detailed discussion.

In short, the exact approach failed to provide an optimal, or even an acceptable solution within 5 minutes in all classes apart from the instances of class A (with 11 customers). The results imply that this method cannot be used in practice for solving instances of more than 54 customers within a few minutes. Instead, a heuristic method may be more appropriate for this problem.

7.10 Exact Approach 2: Weighted Sum Exact Approach for the 5-objective variant

A second and more general exact approach to solve the problem under study, is to use the Weighted Sum Method, without making any assumption about any preference regarding the 5 component objectives. Thus, we can solve the *Multi-Objective Single-Commodity Delayed VRPTW with 5 objectives* (*SCD-VRPTW-MOO-5*, or simply *SCD-VRPTW-5*) by using the weighted-sum method and formulation (MILP 3B) of section 7.7. As before, a single aggregated objective (equation (7.7.1)) is involved. However, instead of simply using a single choice of weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ as done in Exact Approach

1, in this approach we consider many different sets of weights. More specifically, we vary systematically the values of the weights $\beta_1, \beta_2, ..., \beta_5$ with $\beta_i > 0$ for all $i = 1, \ldots, 5$, and for each set of weights we solve (MILP 3B); thus getting several different solutions to the MOO problem, all of which will belong to the Pareto front. This can be implemented in an optimization solver to directly solve different problem instances. We will be referring to this method as *Exact Approach 2* for solving Problem 3. Note that the weights may be normalized (i.e. to have $\sum_{i=1}^{5} \beta_i = 1$), or not.

The method implies that to solve a single instance using Exact Approach 2, we must solve an adequate number of programs of the form (MILP 3B), each time with a different combination of weights. This way, we can get a subset of the set of non-dominated solutions. However, we would probably have to solve too many MILPs, choosing some of the very many possible combinations, and even then there is no guarantee that we will get a representative subset of the Pareto front. More importantly, since Exact Approach 1 which involves solving just one MILP failed to provide an acceptable solution in a short time for instances with more than 54 nodes, it is only reasonable to expect even worse results with exact approach 2. Therefore, we decided not to perform any experiments with exact approach 2, nor with any other exact approach. We believe that heuristic approaches are more appropriate for this problem, and this is the focus of the following chapter.

7.11 Problem extension with customer priorities

The Single-Commodity Delayed VRPTW can be extended to include different priorities for different customers. This may be crucial in real applications, where e.g. the company has some key customers that are more important than others, and thus may prefer having these key customers affected the least in case of a disruption - at the expense of having some customers of less importance being affected more.

For this, let p_i be the priority of customer i, where p_i is a known parameter with $p_i > 0$ for all $i \in I_C$. The more important a customer is, the bigger the value of p_i should be. Therefore, the more important a customer is, the more a potential violation of his or her time window is penalized in the objective(s).

In order to include different priorities to different customers, in the more general formulation *(MOMILP 3A)* presented in section 7.6, we can replace the 5 component objectives (7.6.1) - (7.6.5) with the following 5 equations:

$$\min \ F_1^p := \sum_{i \in I_C} p_i \mu_i \tag{7.11.1}$$

$$\min \ F_2^p := \sum_{i \in I_C} p_i \lambda_i \tag{7.11.2}$$

$$\min \ F_3^p := \sum_{i \in I_C} p_i \rho_i \tag{7.11.3}$$

$$\min \ F_4^p := \sum_{i \in I_C} p_i \sigma_i \tag{7.11.4}$$

$$\min \ F_5^p := \sum_{k \in K} w_{0,k} \tag{7.11.5}$$

Likewise, if we want to involve customer priorities in formulation *(MILP 3B)* presented in section 7.7, we can replace the single aggregated objective function (7.7.1), with the following more general objective:[6]

$$\min \quad \beta_1 \sum_{i \in I_C} p_i \mu_i + \beta_2 \sum_{i \in I_C} p_i \lambda_i + \beta_3 \sum_{i \in I_C} p_i \rho_i + \beta_4 \sum_{i \in I_C} p_i \sigma_i + \beta_5 \sum_{k \in K} w_{0,k} \tag{7.11.7}$$

Note that, starting from the more general models presented above that involve customer priorities, if we simply set $p_i = 1$ for all $i \in I_C$, then these reduce down to models *(MOMILP 3A)* and *(MILP 3B)* without priorities.

No experiments were performed using the generalized formulations which involve customer priorities. This is left open for future research.

7.12 Reduction to 3 objectives

In general, for a multi-objective optimization (MOO) problem, the greater the number of objectives, the more complex and time-consuming it is to construct the set of non-dominated solutions (or a representative subset of this set). For example, finding the set of non-dominated solutions of a MOO problem with 5 objectives, is considered significantly more complex than for a MOO problem with 3 objectives. Even trying to visualize the Pareto

[6]Obviously, equation (7.11.7) can be written in a more concise form as:

$$\min \quad \sum_{i \in I_C} p_i(\beta_1 \mu_i + \beta_2 \lambda_i + \beta_3 \rho_i + \beta_4 \sigma_i) + \beta_5 \sum_{k \in K} w_{0,k} \tag{7.11.6}$$

front and decide among the (possibly many) alternative solutions may be not straightforward when handling 5 objectives.

For this, the 5 objectives of this MOO problem can be reduced down to 3 objectives, if one is willing to make the following two assumptions: (i) that having one early customer is equally as bad as having one late customer, and (ii) that having a delay in service of χ time units is equally as bad as having an early start of service by χ time units.

Under the above additional assumptions, the Single-Commodity Delayed VRPTW with 5 objectives (SCD-VRPTW-5) reduces to *the Single-Commodity Delayed VRPTW with 3 objectives (SCD-VRPTW-3).* Specifically, the latter is a MOO problem which is similar to the former one, but instead of having 5 objectives, it only involves the following 3 objectives:

1. Minimize $f_1 := F_1 + F_2 = \sum_{i \in I_C} (\mu_i + \lambda_i)$, i.e. minimize the number of customers served outside their promised time windows.

2. Minimize $f_2 := F_3 + F_4 = \sum_{i \in I_C} (\rho_i + \sigma_i)$, i.e. minimize the total amount of time by which the time windows are violated.

3. Minimize $f_3 := F_5 = \sum_{k \in K} w_{0,k}$, i.e. minimize the sum of arrival times of the vehicles at the depot.

The SCD-VRPTW-3 can therefore be formulated as a Multiple Objective Mixed Integer Linear Program *(MOMILP 3C)*, composed of all constraints (7.6.6)-(7.6.32) of (MOMILP 3A) presented in section 7.6, but with the following three objectives:

$$\min \ f_1 := \sum_{i \in I_C} (\mu_i + \lambda_i) \tag{7.12.1}$$

$$\min \ f_2 := \sum_{i \in I_C} (\rho_i + \sigma_i) \tag{7.12.2}$$

$$\min \ f_3 := \sum_{k \in K} w_{0,k} \tag{7.12.3}$$

Furthermore, the SCD-VRPTW-3 can also be solved using the weighted-sum method, the same way that SCD-VRPTW-5 can be solved, if in equation (7.7.1) we let $\beta_1 = \beta_2$, $\beta_3 = \beta_4$, and define $(\beta_1', \beta_2', \beta_3') = (\beta_1, \beta_3, \beta_5)$, which will transform it into the following equation:

$$\text{minimize} \ \ \beta_1' \sum_{i \in I_C} (\mu_i + \lambda_i) + \beta_2' \sum_{i \in I_C} (\rho_i + \sigma_i) + \beta_3' \sum_{k \in K} w_{0,k} \tag{7.12.4}$$

or equivalently

$$\text{minimize} \quad \sum_{i=1}^{3} \beta'_i f_i \tag{7.12.5}$$

For this, starting from formulation (MOMILP 3C) presented above, if we replace the three objectives (7.12.1) - (7.12.3) with equation (7.12.5) (or its equivalent (7.12.4)) and keep all other equations (7.6.6)-(7.6.32), we get the single-objective formulation *(MILP 3D)* for the SCD-VRPTW-3.

Then, formulation (MILP 3D) can be used to solve the 3-objective variant of the problem, i.e. SCD-VRPTW-3, either assuming a Lexicographic Preference (i.e. as in Exact Approach 1 of section 7.9), or as a MOO problem using the Weighted Sum approach and varying systematically the set of weights (i.e. as in Exact Approach 2 of section 7.10).

7.13 Applications: other types of disruption

The models described in this chapter can be used to handle delays in a single-commodity VRPTW. Additionally, the same models may be used to handle other forms of disruption for a single-commodity VRPTW. Examples include:

- one or multiple roads/arcs blocked, or more generally parts of the network blocked, affecting the routes of some delivery vehicles. For instance, this may occur due to an accident which does not involve a delivery vehicle, due to a landslide etc.
- traffic congestion in some parts of the network, which was not taken into account or was not known when constructing the original plan.

In each one of the above cases, the same formulation and models can be used, with the only difference that we assume that the operations team updates the travel times accordingly before rescheduling, as follows: If arc (i, j) is blocked, then set $t_{i,j} = \infty$. If there is traffic congestion and the travel time $t_{i,j}$ has increased, then $t_{i,j}$ is updated using a new estimated travel time $t'_{i,j}$, or an overestimate if it is impossible to predict the travel jam effect on arc (i, j). Historical data of travel times under traffic jams may be used, if available and if this is the case.

7.14 Other problem variants and solution methods, scope for further research and conclusions

It is possible to extend the formulations described in this chapter to include other constraints (such as certain customers being served by particular vehicles), if that is relevant for a particular application. Additionally, other variants of the problem can arise if we consider a different set of objectives.

For instance, there are no restrictions on the amounts of delay or earliness in the models described in this chapter. However, if desired, such restrictions can be easily incorporated to the current models as follows:
(a) If we want to have a maximum delay of ρ_{max} per customer, we can add the constraint $\rho_i \leq \rho_{max} \;\; \forall\, i \in I_C$ (or for a specific subset of I_C).
(b) Similarly, if we want to have a maximum earliness of σ_{max} per customer, we can include the constraint $\sigma_i \leq \sigma_{max} \;\; \forall\, i \in I_C$ (or for a subset of I_C).
(c) If we want to have a maximum delay of ρ'_{max} combined for all customers, we can add the constraint $\sum_{i \in I_C} \rho_i \leq \rho'_{max}$.
(d) Likewise, if we want to have a maximum earliness of σ'_{max} combined for all customers, we can include the constraint $\sum_{i \in I_C} \sigma_i \leq \sigma'_{max}$.

Obviously there are many different variants that can occur from the problem under study, by considering different combinations of constraints and objectives. Additionally, each one of these variants may be solved using different methods. In this chapter we only considered the lexicographic approach and the weighted-sum approach as exact methods to address the 5-objective variant of problem. Apart from these methods, of course, one can apply other multi-objective optimization methods to address each variant of the problem, such as the Epsilon Constraint Method.

Since the results from Exact Approach 1 indicate that exact approaches based on the formulations presented in this chapter are not promising for quickly solving medium-sized or large instances, we decided not to perform any experiments using other exact approaches. Therefore, we leave open for further research the task of experimenting with other multi-objective optimization methods to find the set of non-dominated solutions in an exact approach. Instead, in the following chapter we focus on heuristic approaches that may be more appropriate for solving such problems in practice. Note also that in the following chapter we do consider another MOO method, namely the Epsilon Constraint Method, to solve the 3-objective variant of problem 3; however, we use this method in a heuristic framework.

Chapter 8

Problem 3 - The Single-Commodity Delayed VRPTW: Heuristic Approaches & Experimental Results

8.1 Introduction

In this chapter we describe three heuristic approaches for the Single-Commodity Delayed VRPTW (SCD-VRPTW), to which we will also be referring as *problem 3.* The first heuristic approach solves the Lexicographic variant of the problem with 5 objectives. The second one solves the Lexicographic variant with 3 objectives, which is a simplification of the first variant and is mainly included so that valid comparisons between the first and third approaches can be made. Finally, the third heuristic approach uses the framework of the ϵ-Constraint Method to solve heuristically the 3-objective version of the problem, and finds a representative subset of the set of non-dominated solutions, as a more general multi-objective optimization (MOO) approach. All of the above approaches employ the Tabu Search metaheuristic.

In section 8.5 we present the experimental results of the three heuristic approaches, as well as the results of exact approach 1 which was described in the previous chapter. Comparisons of the different methods and conclusions follow.

In general, throughout this chapter we use the notation described in the previous chapter, unless it is explicitly stated otherwise.

8.2 Heuristic Approach 1: A Tabu Search heuristic for the Lexicographic SCD-VRPTW with 5 Objectives

In this section we describe a heuristic based on Tabu Search, to solve the *Lexicographic Single-Commodity Delayed VRPTW with 5 objectives (SCD-VRPTW-Lex5).* We refer to this approach as *Heuristic Approach 1* for problem 3.

In this variant of the problem, the 5 objective functions F_1, F_2,..., F_5 which were defined in section 7.4, are aggregated into a single objective, namely $\sum_{i=1}^{5} \beta_i F_i$, which is to be minimized. In other words, we employ the weighted-sum method to transform the MOO problem into one with a single objective. Furthermore, we choose weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ with $\beta_1 >> \beta_2 >> \beta_3 >> \beta_4 >> \beta_5$, so that function F_1 is minimized first, then among the solutions which involve the minimum value of F_1, the one(s) which also minimize F_2 are chosen, and so on. This way, we solve the so-called Lexicographic variant of the problem with 5 objectives (i.e. assuming Lexicographic preference of the 5 component objectives). Note that this variant was also described in section 7.9, where Exact Approach 1 was presented for solving the same variant.

In the Single-Commodity Delayed VRPTW, we have n customers and d vehicles. Each vehicle k $(k = 1, ..., d)$ departs from node $n + k$ (initial position) at time T_k, then visits a subset of the set of customers (nodes $1, 2, \ldots, n$) in the appropriate order, and finishes at the depot. In the heuristic, for technical reasons we use d copies of the depot, denoted by nodes $n+d+1$, $n+d+2$,..., $n+2d$, so that vehicle k $(k = 1, ..., d)$ finishes at node $n+d+k$.[1] All customers must be served. All nodes must be visited exactly once. At any time, the capacity constraint must be satisfied. Each node i $(i = 1, 2, ..., n + 2d)$ is associated with its Cartesian coordinates (X_i, Y_i), a service time s_i, a demand D_i and a time window $[a_i, b_i]$. All time windows are treated as soft constraints, so any intermediate or final solutions may violate some or even all of the time windows; this is not considered to be an infeasibility. The starting positions $n + k$ $(k = 1, ..., d)$ can be copies of the depot, or customer nodes (in which case the corresponding customers should not be included in the customer set $\{1, 2, ..., n\}$), or any other position specified by its Cartesian coordinates (X_{n+k}, Y_{n+k}), with zero service time, zero demand and time windows $[0, +\infty)$.

[1]This is a difference between the notation of this chapter and the corresponding notation used in the previous chapter, where the depot was represented by a single node, namely node 0.

Notes:

1. The *lexicographic approach* employed here can be viewed as a special case of the *weighted-sum method*, with a choice of weights so that $\beta_1 >> \beta_2 >> \beta_3 >> \beta_4 >> \beta_5$.

2. Any violation of the time windows of the starting nodes $n + k$ or the endpoint nodes $n + d + k$, will not affect the values of the component objectives F_1, F_2, F_3 or F_4.

3. Some obvious bounds for the first two component objectives are: $0 \le F_1 \le n,\ \ 0 \le F_2 \le n,\ \ \text{and}\ \ 0 \le F_1 + F_2 \le n.$

8.2.1 Objective

The objective in the first heuristic approach is to minimize function G defined below:

$$minimize \quad G := \sum_{i=1}^{5} \beta_i F_i + \sum_{k=1}^{d} (P_0 \xi_k + P_0^* \xi_k^*) \tag{8.2.1}$$

where

$$\xi_k := \begin{cases} 1, & \text{if } \sum_{i \in R_k} D_i > C_k \\ 0, & \text{otherwise} \end{cases} \tag{8.2.2}$$

$$\xi_k^* := \begin{cases} \sum_{i \in R_k} D_i - C_k, & \text{if } \sum_{i \in R_k} D_i > C_k \\ 0, & \text{otherwise} \end{cases} \tag{8.2.3}$$

The objective function G is equal to $\sum_{i=1}^{5} \beta_i F_i$ if the capacity constraint is respected; otherwise a penalty function is added, which increases as the amount of capacity violation increases. The second summation term in G is the penalty term for capacity violation, where ξ_k is equal to 1 if the capacity constraint is violated in route R_k, or 0 otherwise. Also, ξ_k^* is the amount of capacity violation in route R_k (ξ_k^* is zero if there is no capacity violation in route R_k). Finally, P_0 and P_0^* are fixed penalty coefficients.

The idea is to allow intermediate solutions which may violate the capacity constraint, but to have a final solution with zero penalty and thus feasible with respect to the capacity constraint.

8.2.2 Solution Notation

We define the following:

- $R = (R_1, R_2, ..., R_d)'$ is an ordered collection of d vectors R_k of different and variable sizes l_k, for $k = 1, ..., d$. Each vector R_k starts with node $n + k$, is then followed by a subset of the set of customers {1,2,...,n}, and ends with node $n + d + k$. Obviously, R_k represents the route of the k-th vehicle.

- $\widetilde{W} = (\widetilde{W_1}, \widetilde{W_2}, ..., \widetilde{W_{n+2d}})$ is a vector of length $n+2d$ which contains the time of start of service $\widetilde{W_i}$ of each node i ($i = 1, ..., n+2d$), when vehicles follow the routes specified by the collection of routes R and *while idle waiting times at nodes are not allowed.* Of course, $s_i = 0$ for nodes $i = n+1, n+2, ..., n+2d$ which do not represent customers; therefore, $\widetilde{W_i}$ coincides with the time of departure from node i for starting nodes $i = n + 1, n + 2, ..., n + d$, and with the time of arrival at node i for endpoint nodes $i = n + d + 1, n + d + 2, ..., n + 2d$. From now on, whenever we mention the *time of start of service at node i, without idle waiting times*, we will refer to $\widetilde{W_i}$ as described above ($i = 1, ..., n+2d$). Also, whenever we mention the *times of start of service without idle waiting times*, we will refer to the vector $\widetilde{W}$.

- $\widetilde{G}$ is a real number representing the value of the objective (8.2.1) *without allowing idle waiting times at nodes*, when following the collection of routes R and visiting the customers at times $\widetilde{W}$. Essentially, $\widetilde{G}$ represents the value of G over a subset Π_1 of the feasible region Π, where by Π_1 we denote the subset of the feasible space in which no waiting times are allowed at any node. More formally, $\widetilde{G}$ is the restriction of the function G over the set Π_1; i.e. $\widetilde{G} = G|_{\Pi_1}$. So, $\widetilde{G}$ is also calculated by equation (8.2.1).

- $W = (W_1, W_2, ..., W_{n+2d})$ is a vector of length $n + 2d$ which contains the time of start of service W_i of each node i ($i = 1, ..., n + 2d$), when vehicles follow the routes specified by the collection of routes R and *while idle waiting time at nodes is allowed.* From now on, whenever we mention the *time of start of service at node i, with idle waiting times*, we will refer to W_i as described above ($i = 1, ..., n+2d$). Also, whenever we mention the *times of start of service with idle waiting times*, we will refer to the vector W.

- G is a real number representing the value of the objective (8.2.1) *when allowing idle waiting time at nodes*, when visiting the customers in the order described by the collection of routes R and at times W. So, G is calculated over the whole feasible region Π.

A solution to the problem can be described by a collection of routes R, along with the respective vector W containing the time of start of service of each node while allowing idle waiting time at nodes, together with the respective values for the 5 objectives $F_1, F_2, ..., F_5$. However, in the context of the heuristic and throughout this chapter, when we talk about a *solution* S of the heuristic we will refer to a quintuple of the form $S = (R, \widetilde{W}, \widetilde{G}, W, G)$.

In a similar way, we define the quintuples S^1, S^2, S^{best} and S^* as: $S^1 = (R^1, \widetilde{W}^1, \widetilde{G}^1, W^1, G^1)$, $S^2 = (R^2, \widetilde{W}^2, \widetilde{G}^2, W^2, G^2)$, $S^{best} = (R^{best}, \widetilde{W}^{best}, \widetilde{G}^{best}, W^{best}, G^{best})$ and $S^* = (R^*, \widetilde{W}^*, \widetilde{G}^*, W^*, G^*)$. From now on, whenever we say that a solution S, S^1, S^2, S^{best} or S^* is created or updated, we will imply that the respective quintuple is created or updated accordingly, unless otherwise stated.

It is requested to find the Solution S^* which minimizes the objective function G over the feasible space Π, i.e. when allowing idle waiting time at nodes, while respecting the necessary constraints.

8.2.3 Description of the basic algorithmic functions

Given a route R, we will now show how to calculate or update the complete solution S, composed of the quintuple $(R, \widetilde{W}, \widetilde{G}, W, G)$.

(i) Given R, how to calculate vector $\widetilde{\mathbf{W}}$ using the algorithmic function *time*: $\widetilde{\mathbf{W}} = \boldsymbol{time}(\mathbf{R})$:

Given a collection of routes R (or, in short, a route R), the algorithmic function *time* calculates the times of start of service at each node when following the route R, if there are no idle waiting times at nodes, and stores the result in the vector $\widetilde{W}$ of length $n + 2d$ (i.e. $\widetilde{W} = time(R)$).

Specifically, for all $j = 1, ...d$ the function *time* sets $\widetilde{W}(R_{j,1}) = T_j$ for the starting node $R_{j,1}$ of route R_j, and then for each one of the following nodes $R_{j,i}$ of route R_j, it sets $\widetilde{W}(R_{j,i}) = \widetilde{W}(R_{j,i-1}) + t(R_{j,i-1}, R_{j,i}) + s(R_{j,i-1})$, for all $i = 2, 3, \dots, length(R_j)$.

Note that we use the notation $\widetilde{W}(R_{j,i})$ and $\widetilde{W}_{R_{j,i}}$ interchangeably; i.e. $\widetilde{W}(R_{j,i}) = \widetilde{W}_{R_{j,i}}$. Similarly, $t(R_{j_1,i_1}, R_{j_2,i_2}) = t_{R_{j_1,i_1}, R_{j_2,i_2}}$ and $s(R_{j,i}) = s_{R_{j,i}}$.

(ii) Given R and $\widetilde{\mathbf{W}}$, how to calculate $\widetilde{\mathbf{G}}$ using the algorithmic function *cost*: $\widetilde{\mathbf{G}} = \mathbf{cost}(\mathbf{R}, \widetilde{\mathbf{W}})$:

Given R and $\widetilde{W}$, the algorithmic function *cost* uses equation (8.2.1) to calculate the value $\widetilde{G}$ of the objective function without allowing idle waiting times at nodes, when following the collection of routes R and visiting the nodes at times $\widetilde{W}$.

In more detail, this function compares the time of start of service $\widetilde{W}_i$ of each node with the respective time window $[a_i, b_i]$ and thus calculates the values of each component objective F_j $(j = 1, \ldots, 5)$. Initially each F_j is set to zero. Then, for all nodes $i = 1, \ldots n$ that represent customers, if $\widetilde{W}_i > b_i$, then F_1 is increased by 1 and F_3 is increased by $\widetilde{W}_i - b_i$. Similarly, for all $i = 1, \ldots n$, if $\widetilde{W}_i < a_i$, then F_2 is increased by 1 and F_4 is increased by $a_i - \widetilde{W}_i$. Finally, F_5 is defined as $\sum_{j=1}^{d} \widetilde{W}_{n+d+j}$. Therefore, the values of all 5 objectives F_j $(j = 1, \ldots, 5)$ and hence the first summation term $\sum_{j=1}^{5} \beta_i F_i$ of equation (8.2.1) are defined for the specific vector $\widetilde{W}$ and collection of routes R.

The second summation term of equation (8.2.1), which is the penalty term for capacity constraint violation, is straightforward to calculate, as explained in subsection 8.2.1. Therefore, $\widetilde{G}$ is calculated and returned as the output of the function *cost*; i.e. $\widetilde{G} = cost(R, \widetilde{W})$.

Note that, more generally, given R and a vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes (for example, we may use $\widetilde{W}$ or W as $\widehat{W}$), we can still use the same algorithmic function *cost* to calculate the respective value of the objective function $\widehat{G}$, as $\widehat{G} = cost(R, \widehat{W})$. Specifically, given R and the times of start of service W while allowing idle waiting times at nodes, we can use the same algorithmic function *cost* to calculate G, as $G = cost(R, W)$.

Up to this point, we have described how, given R we can create or update the triplet $(R, \widetilde{G}, \widetilde{W})$. We will now describe how, given the triplet $(R, \widetilde{G}, \widetilde{W})$, we can use the algorithmic function *optimal waiting times* to calculate G and W and thus get the full quintuple $(R, \widetilde{G}, \widetilde{W}, G, W)$, i.e. the full solution S.

Note that the triplet (R, G, W) is enough to represent a complete solution, since $\widetilde{G}$ and $\widetilde{W}$ serve as intermediate values. Therefore, once we have the optimal values for the triplet (R, G, W), there is no need to update $\widetilde{G}$ and $\widetilde{W}$.

(iii) Given *R* and $\widetilde{\mathbf{W}}$ (or $\widehat{\mathbf{W}}$), how to calculate *W* and *G* using the algorithmic function *optimal waiting times*: *[W,G]=optimal_waiting_ times*$(\mathbf{R}, \widetilde{\mathbf{W}})$:

Given R and $\widetilde{W}$, the function *optimal waiting times* calculates a heuristic approximation of the optimal times of start of service, W and the respective value of the objective function G involved when following the routes defined by R and while idle waiting time at nodes is allowed. The function works as described below.

First initialize W and G by letting $W := \widetilde{W}$ and $G := cost(R, W)$. Then, for all $j = 1, 2, ...d$, for all $i = 2, 3, ..., length(R_j) - 1$, perform the procedure described in the following paragraph:

Check if by visiting customers at times W, then the i^{th} customer of the j^{th} route, denoted by $R_{j,i}$ will be visited earlier than his or her ready time. If the answer is yes, then define m_1 as the amount of earliness of customer $R_{j,i}$; that is $m_1 := a_{R_{j,i}} - W_{R_{j,i}}$. Also, define $m_2 := min\{b_{R_{j,k}} - W_{R_{j,k}} | k = i+1, \dots, length(R_j) - 1\}$; that is, m_2 is the minimum difference between the due date and the time of start of service, amongst the nodes following the node $R_{j,i}$ in the route R_j. Note that m_2 is not necessarily positive. In fact, m_2 is negative if there is at least one late customer among the customers $R_{j,i+1}$, $R_{j,i+2}, \dots, R_{j,length(R_j)-1}$. Then, define $m := min\{m_1, m_2\}$. If $m > 0$, then set $W_{temp} := W$. Add a waiting time of m time units to customer $R_{j,i}$ in the vector W_{temp}. This is done by increasing the times of visit of all nodes including and following node $R_{j,i}$ by m, in vector W_{temp}. Then calculate $G_{temp} := cost(R, W_{temp})$. If $G_{temp} < G$, this means that adding a waiting time of m time units to customer $R_{j,i}$ gives an improvement to the current times of start of service (vector W), with respect to the *objective function*. Therefore, if $G_{temp} < G$, then update $W := W_{temp}$, $G := G_{temp}$ and continue to the next iteration of the variables i and j.

Finally, the function returns G and W.

Note that within the function *optimal waiting times*, the vector $\widetilde{W}$ may be replaced by $\widehat{W}$, which may or may not involve some idle waiting times at nodes.

8.2.4 Description of the Algorithm

1. <u>Parameters input & Initialization:</u>
   - Input: n, d, C_k and T_k (for $k = 1, ..., d$), TL (Tabu Length).
   - Define $T_{min} := min\{T_k | k = 1, ..., d\}$.
   - Input: (X_i, Y_i), $[a_i, b_i]$, s_i, D_i $\forall\ i = 1, 2, \dots, n + 2d$.
   - Input: J_1, J_2 (parameters controlling the number of Tabu Search iterations without an improvement before the algorithm terminates, in Tabu Stages A and B, respectively).
   - For all $i, j \in \{1, 2, \dots, n + 2d\}$, calculate the travel time $t_{i,j}$ as the Euclidean distance $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ between each pair of nodes (i, j), rounded to 2 decimal points.

2. <u>Calculate lower bounds for the component objectives F_1 and F_3:</u>
   - Identify the set of *Definitely Late customers*, $I_{DL} = \left\{ i \in I_C \;\middle|\; \min_{k \in K}(T_k + s_{n+k} + t_{n+k,i}) > b_i \right\}$, as described in chapter 7. Its cardinality $|I_{DL}|$ is

the number of definitely late customers, which serves as a lower bound for F_1; that is, $LB_{F_1} = |I_{DL}|$.

- Evaluate the *minimum amount of lateness*, defined as: $LB_{F_3} := \sum\limits_{i \in I_{DL}} [\min\limits_{k \in K}(T_k + s_{n+k} + t_{n+k,i}) - b_i]$, which serves as a lower bound for F_3.

Objective:

The objective is to minimize function G defined by equation (8.2.1) over the whole feasible region Π, as was explained before.

3. Construct the Initial Route R^0:

   In order to construct the *initial route* R^0, we use the set of routes involved in the original plan (which is supposed to be the best possible plan found for the original problem, optimal or near-optimal), after removing the first part of each route containing the customers that have already been served.

   Then in route R_k^0 of vehicle k (for all $k = 1, ..., d$) we use the position of vehicle k at time T_k as the starting position (represented by node $n + k$) and include this as the first node of the route. The depot is the end-point of each route R_k^0, so we also include a copy of the depot (represented by node $n + d + k$) as the last node of route R_k^0.

   The collection of routes $R^0 = (R_1^0, R_2^0, ..., R_d^0)'$ is used as the initial solution's collection of routes and will be simply referred to as *the initial route.*

4. - Given the initial route R^0, calculate the full initial solution $S^0 = (R^0, \widetilde{G}^0, G^0, \widetilde{W}^0, W^0)$.
   - Initialize the *current solution* S^2, by letting $S^2 := S^0$.
   - Initialize the *best solution found*, S^{best}, by letting $S^{best} := S^2$.

   - Apply Tabu Search with best improvement and the exchange and insertion neighborhood moves, to find the best solution S^{best}, as described in Stages A and B which follow:

   • Tabu Search - Stage A:
   - Start with solution S^2.
   - Find the best solution S^1 in a neighborhood $N(S^2)$ of S^2, with respect to the *objective function.* The neighborhood $N(S^2)$ of S^2 is constructed starting from S^2 and performing any single possible move, using one of the two following neighborhood operators: exchange and insertion.

- To avoid going around in circles, moves involving nodes that were involved in any of the previous TL moves are declared tabu and thus forbidden. A tabu move however is allowed as an exception to this rule, if it leads to a solution that improves the best solution found so far, S^{best}.
Once a local optimum is reached, we want the algorithm to be able to escape that local optimum and move to another neighborhood. However, sometimes when a local optimum is reached in this problem, there are several possible moves that lead to exactly the same objective (ties). To avoid having the algorithm wandering around a plateau containing a local optimum and prevent it from getting trapped there, we disallow moves that lead to a solution which gives the same objective value as the current solution. In fact, when designing the algorithm, we tried allowing moves that lead to the same objective value, but after some experimentation we concluded that forbidding those moves gave a better performance. Therefore, we do not allow moves that do not lead to a change in the objective value (we obviously allow improving moves, as well as worsening moves to escape a local optimum).

- We also include a diversification mechanism which prevents the frequent use of moves involving the same nodes. Specifically, moves involving nodes that were involved in more than $\theta_1 + \theta_2 \cdot \mu$ iterations so far are also declared forbidden, where μ is the average number of times a node was involved in a move so far, and θ_1, θ_2 are parameters that need to be tuned experimentally.[2] Again, such a move is allowed as an exception to this rule, if it leads to an improvement to the best solution found so far, S^{best}.

- If S^1 is better than S^{best}, then set $S^{best} := S_1$.
- Set $S^2 := S^1$.
- Repeat Stage A, until there is no improvement in solution S^{best} in the last J_1 iterations.

Note that, during this first stage of Tabu search, whenever a solution S is calculated or updated, only the triplet $(R, \widetilde{W}, \widetilde{G})$ is updated; i.e. the function *optimal waiting times* is not employed, nor are the corresponding W and G calculated or updated.

[2]After some experimentation, we decided to use $\theta_1 = 3$ and $\theta_2 = 1.7$

- Tabu Search - Stage B:
  Repeat the same process described in Stage A, but this time replace J_1 by J_2 (where generally J_2 is advised to be smaller than J_1) and replace 'Stage A' by 'Stage B'. In this second stage, whenever a solution S is calculated or updated, the full quintuple $(R, \widetilde{W}, \widetilde{G}, W, G)$ is updated, meaning that the function *optimal waiting times* is now called to calculate or update W and G.

  Stage B is repeated, until there is no improvement in solution S^{best} in the last J_2 iterations.

- Once the two stages are finished, we return the best solution found, S^{best}. We also return the values of the five component objective functions $F_1, F_2, ..., F_5$ that correspond to this best solution, in the quintuple $(F_1^*, F_2^*, ..., F_5^*)$.
- The reason why we split the Tabu Search in two stages, is because this saves us computational time, something which was confirmed through experiments. Stage A is a fast way to get close to a very good solution. Once this is achieved, stage B repeats the same process as in stage A, but this time the more computationally expensive function *optimal waiting times* is employed, which involves the exploration of the different solutions when allowing different amounts of waiting times at different customers, and the search for the optimal allocation of waiting times at customers for the corresponding set of routes. In fact, one could ignore stage A and go directly to stage B of Tabu Search, which should give similar results (provided that J_2 would be approximately the same or greater than J_1); however, this would be computationally more expensive and the whole algorithm would be substantially slower than the proposed method.

8.2.5 Heuristic Approach 1b: A Tabu Search heuristic for the Multi-Objective SCD-VRPTW with 5 Objectives, using the Weighted-Sum method

The algorithm presented in the previous subsection solves the SCD-VRPTW-5 heuristically, assuming Lexicographic Preference of the 5 component objectives. This algorithm can be generalized as follows:

We vary systematically the set of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ (perhaps normalized but not necessarily), and for each different choice of weights we run the algorithm of heuristic approach 1, which was presented in subsection 8.2.4. The solutions derived from each run are then compared with each other and any dominated solutions, or any duplicated solutions, are removed from the solution set. The remaining set of solutions constitutes an approximate representative set of the Pareto front. We will refer to this approach as *Heuristic Approach 1b*, which essentially solves heuristically the Multi-Objective SCD-VRPTW-5 using the weighted-sum method and Tabu Search.

8.3 Heuristic Approach 2: A Tabu Search heuristic for the Lexicographic SCD-VRPTW with 3 Objectives

In this section we discuss a heuristic approach to solve the *Lexicographic Single-Commodity Delayed VRPTW with 3 objectives (SCD-VRPTW-Lex3).* We will be referring to this approach as *Heuristic Approach 2* for problem 3. In this approach we use the same notation and algorithm that were used in the first heuristic approach, as described in section 8.2, but with a different set of component objectives. Essentially, we make the assumptions and use the technique described in section 7.12, to reduce the 5 objectives of the SCD-VRPTW down to 3 objectives.

Specifically, in heuristic approach 1 we used the 5 component objectives F_1, F_2,..., F_5 that were transformed into a single aggregated objective $\sum_{i=1}^{5} \beta_i F_i$, which composed the first summation term of the objective function G that was defined by equation (8.2.1). Instead, in heuristic approach 2 we use the 3 component objectives f_1, f_2 and f_3, defined as $f_1 := F_1 + F_2$, $f_2 := F_3 + F_4$ and $f_3 := F_5$, which are then aggregated into a single objective $\sum_{i=1}^{3} \beta'_i f_i$ that constitutes the first summation term of the objective function G_3, defined below. Therefore, the objective function of the second heuristic approach is:

$$minimize \quad G_3 := \sum_{i=1}^{3} \beta'_i f_i + \sum_{k=1}^{d} (P_0 \xi_k + P_0^* \xi_k^*) \tag{8.3.1}$$

where ξ_k and ξ_k^* were defined by equations (8.2.2) and (8.2.3) in subsection 8.2.1.

Obviously, the lower bound LB_{F_1} of the component objective F_1, discussed in the previous section, is also a lower bound for the component objective f_1 (since $f_1 := F_1 + F_2$ and $F_2 \geq 0$). Similarly, the lower bound

LB_{F_3} of the component objective F_3 is also a lower bound for the component objective f_2 (since $f_2 := F_3 + F_4$ and $F_4 \geq 0$). Hence we can define $LB_{f_1} := LB_{F_1}$ and $LB_{f_2} := LB_{F_3}$.

By varying systematically the set of positive weights $(\beta'_1, \beta'_2, \beta'_3)$ (perhaps normalized but not necessarily), and for each different choice of weights running the algorithm of heuristic approach 1, but with G_3 as the objective function, instead of G, and with f_1, f_2, f_3 as the 3 component objectives to replace $F_1, F_2, ..., F_5$, we can get an approximate representative set of the Pareto front for the SCD-VRPTW-3 (after removing any duplicates or dominated solutions from the resulting set of solutions derived from all the runs). This approach constitutes *Heuristic Approach 2b*, which solves heuristically the Multi-Objective SCD-VRPTW-3 using the weighted-sum method and Tabu Search.

As a special case of Heuristic Approach 2b, if we perform a single run of the weighted-sum approach described in the previous paragraph, using a single set of weights $(\beta'_1, \beta'_2, \beta'_3)$ such that $\beta'_1 >> \beta'_2 >> \beta'_3 > 0$, we can solve heuristically the 3-objective variant of the problem with Lexicographic Preference. This constitutes *Heuristic Approach 2*, which solves heuristically the Lexicographic SCD-VRPTW-3 using Tabu Search.

Essentially, heuristic approach 2b is equivalent to employing heuristic approach 1b with different sets of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ such that $\beta_1 = \beta_2$ and $\beta_3 = \beta_4$. Also, heuristic approach 2 is equivalent to employing heuristic approach 1 with a single choice of positive weights $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5)$ such that $\beta_1 = \beta_2 >> \beta_3 = \beta_4 >> \beta_5 > 0$.

Heuristic approach 2, which is a slight simplification of heuristic approach 1, is presented mainly for two reasons. Firstly, we want to show that narrowing the 5 objectives down to 3 objectives in the way that we chose to do so, and in the context of the problem under study, may sometimes not affect the solution very much. Secondly, heuristic approach 2 serves as a bridge between the first and third heuristic approaches. Since it is not obvious how to directly compare the results of heuristic approaches 1 and 3, we can do this indirectly by comparing the results of heuristic approaches 1 and 2 between each other, as well as those of heuristic approaches 2 and 3 between each other.

8.4 Heuristic Approach 3: A Tabu Search heuristic for the SCD-VRPTW with 3 objectives, using the ECM framework

In this section we describe a heuristic based on the *Epsilon Constraint Method (ECM)* and Tabu Search, which is designed to solve the *Single-Commodity Delayed VRPTW with 3 objectives (SCD-VRPTW-3).* This heuristic constitutes the *3rd heuristic approach* for problem 3.

As in section 8.3, there are three component objectives to be minimized, namely f_1, f_2 and f_3. In order to address the 3-objective variant of the problem using the ECM, we treat f_2 as the actual objective function to be minimized (min f_2) and we convert the remaining two objectives into constraints ($f_1 \leq \epsilon_1$ and $f_3 \leq \epsilon_3$). Therefore, we arrive at the *first ECM formulation for the SCD-VRPTW-3*, which is composed of constraints (7.6.6)-(7.6.32) of (MOMILP 3A) that were presented in section 7.6, together with objective (8.4.1) and constraints (8.4.2) and (8.4.3) which follow:

$$\text{minimize} \quad f_2 := \sum_{i \in I_C} \rho_i + \sum_{i \in I_C} \sigma_i \tag{8.4.1}$$

$$f_1 := \sum_{i \in I_C} \mu_i + \sum_{i \in I_C} \lambda_i \leq \epsilon_1 \tag{8.4.2}$$

$$f_3 := \sum_{k \in K} w_{0,k} \leq \epsilon_3 \tag{8.4.3}$$

The above mathematical program, to which we will be referring as *(MILP 3E)*, constitutes an exact formulation for the SCD-VRPTW-3 using the ECM method. In theory, this can be implemented in an optimization solver and, by varying systematically the values of ϵ_1 and ϵ_3, one can get a representative subset of the set of non-dominated solutions for the problem. We will refer to this approach as *Exact Approach 3* for problem 3.

However, the experimental results from Exact Approach 1, which are presented later in section 8.5, show that exact approaches based on formulations (MILP 3B) or (MOMILP 3A) are not promising for solving medium-sized or large instances within a few minutes. Exact Approach 3 presented above involves solving multiple such programs, so it is even more computationally expensive than Exact Approach 1 and is expected to take even longer time to use. Therefore, we decided not to perform any experiments with exact approach 3.

Instead, we will use the above framework of the epsilon constraint method to construct a heuristic algorithm. In fact, we created such a heuristic that was based on the above formulation (MILP 3E). However, after some experimentation we concluded that better results can be achieved by the heuristic if we replace objective (8.4.1) with objective (8.4.4) which is shown below, and with a choice of positive weights $(\beta'_1, \beta'_2, \beta'_3)$ such that $\beta'_2 = 1$, $\beta'_1 << 1$ and $\beta'_3 << 1$:

$$\text{minimize} \quad \sum_{i=1}^{3} \beta'_i f_i \tag{8.4.4}$$

Objective (8.4.4), together with constraints (8.4.2), (8.4.3) and constraints (7.6.6)-(7.6.32) of (MOMILP 3A), will constitute the *second ECM formulation for the SCD-VRPTW-3*, which we will denote by (MILP 3F). Essentially, (MILP 3F) is the same as (MILP 3E), after replacing its objective (8.4.1) with objective (8.4.4). This program is a single-objective MILP which depends - among others - on the parameters ϵ_1 and ϵ_3. Therefore, in the remainder of this chapter we will refer to this second ECM formulation for the SCD-VRPTW-3 as *MILP-3F-ECM*(ϵ_1, ϵ_3).

Heuristic Approach 3 involves varying the values of ϵ_1 and ϵ_3 within appropriate ranges in a systematic way, and for each pair of parameter values (ϵ_1, ϵ_3), solving heuristically MILP-3F-ECM(ϵ_1, ϵ_3). The solutions resulting from solving all the above programs are then compared with each other and any duplicates or dominated ones are removed from the solution set. The resulting set of solutions is an approximate subset of the set of non-dominated solutions for the SCD-VRPTW-3.

8.4.1 Objectives

In the algorithm of heuristic approach 3, which is described step by step in subsection 8.4.4, we use different objectives at different stages. Note that, at any stage of the algorithm, the value of the objective is stored as $\widetilde{G}$ when idle waiting times at nodes are not allowed, or as G when idle waiting times are allowed.

- **Objective throughout Step 4:**

  Throughout Step 4, for the calculation of the objective function we use the following function, which we seek to minimize:

$$G_3 = \sum_{i=1}^{3} \beta'_i f_i + \sum_{k=1}^{d} (P_0 \xi_k + P_0^* \xi_k^*) \tag{8.4.5}$$

  where

$$\xi_k := \begin{cases} 1, & \text{if } \sum_{i \in R_k} D_i > C_k \\ 0, & \text{otherwise} \end{cases} \tag{8.4.6}$$

$$\xi_k^* := \begin{cases} \sum_{i \in R_k} D_i - C_k, & \text{if } \sum_{i \in R_k} D_i > C_k \\ 0, & \text{otherwise} \end{cases} \qquad (8.4.7)$$

Essentially, function G_3 is equal to $\sum_{i=1}^{3} \beta_i' f_i$ if the capacity constraints are respected; otherwise a penalty function is added, which increases as the amount of capacity violation increases. The second summation term in G_3 is the penalty term for capacity violation, where ξ_k is equal to 1 if the capacity constraint is violated in route R_k, or 0 otherwise. Also, ξ_k^* is the amount of capacity violation in route R_k, if any (ξ_k^* is zero if there is no capacity violation in route R_k). Finally, P_0 and P_0^* are fixed penalty coefficients. This allows intermediate solutions which may violate the capacity constraints, but the aim is to have a final solution with zero penalty and thus feasible with respect to the capacity constraints.

Therefore, throughout Step 4, the value of the objective function without allowing idle waiting times at nodes, and when following the collection of routes R and visiting the customers at times $\widetilde{W}$, is stored as $\widetilde{G}$, and is calculated using the function G_3 defined by equation (8.4.5). Similarly, throughout Step 4, the value of the objective function when allowing idle waiting times at nodes, and when following the collection of routes R and visiting the customers at times W, is stored as G, and is also calculated using the function G_3 defined by equation (8.4.5).

- **Objective throughout Step 6:**

  Throughout Step 6, for the calculation of the objective function we use function G_ϵ defined below, which we seek to minimize:

$$G_\epsilon := \sum_{i=1}^{3} \beta_i' f_i + \sum_{k=1}^{d} (P_0 \xi_k + P_0^* \xi_k^*) + \\ + \begin{cases} 0 & \text{if } f_1 \le \epsilon_1 \text{ and } f_3 \le \epsilon_3 \\ (f_1 - \epsilon_1) \cdot P_1 + P_1^* & \text{if } f_1 > \epsilon_1 \text{ and } f_3 \le \epsilon_3 \\ (f_3 - \epsilon_3) \cdot P_3 + P_3^* & \text{if } f_1 \le \epsilon_1 \text{ and } f_3 > \epsilon_3 \\ (f_1 - \epsilon_1) \cdot P_1 + P_1^* + (f_3 - \epsilon_3) \cdot P_3 + P_3^* & \text{if } f_1 > \epsilon_1 \text{ and } f_3 > \epsilon_3 \end{cases} \qquad (8.4.8)$$

  with a choice of weights $(\beta_1', \beta_2', \beta_3')$ such that $\beta_2' = 1$, $\beta_1' << 1$ and $\beta_3' << 1$, where ξ_k and ξ_k^* were defined by equations (8.4.6) and (8.4.7).

  Essentially, G_ϵ is defined as $\sum_{i=1}^{3} \beta_i' f_i$ if the epsilon constraints and the capacity constraints are respected; otherwise G_ϵ is penalized by a large penalty which is proportional to the amount of violation, plus a large constant. Specifically, if $f_1 > \epsilon_1$, then G_ϵ is increased by $(f_1 - \epsilon_1) \cdot P_1 + P_1^*$. If $f_3 > \epsilon_3$, then G_ϵ is increased by $(f_3 - \epsilon_3) \cdot P_3 + P_3^*$, where P_1, P_1^*, P_3 and P_3^* are appropriate penalty coefficients.[3] Also, the second summation term $\sum_{k=1}^{d}(P_0\xi_k + P_0^*\xi_k^*)$ of G_ϵ is the penalty term for capacity constraint violation and was explained before, when discussing equation (8.4.5).

Therefore, throughout Step 6, the value of the objective function without allowing idle waiting times at nodes, and when following the collection of routes R and visiting the customers at times $\widetilde{W}$, is stored as $\widetilde{G}$, and is calculated using the function G_ϵ defined by equation (8.4.8). Similarly, throughout Step 6, the value of the objective function when allowing idle waiting times at nodes, and when following the collection of routes R and visiting the customers at times W, is stored as G, and is also calculated using the function G_ϵ defined by equation (8.4.8).

8.4.2 Solution Notation

Apart from the difference in the calculation of the objectives $\widetilde{G}$ and G, which was explained in the previous subsection, throughout heuristic approach 3 we will use the same notation used in heuristic approach 1, as described in section 8.2 (e.g. for $R, \widetilde{W}, W, S, R_k$ etc.), unless otherwise stated or implied.

Furthermore, we will use the algorithmic functions *time*, *cost* and *optimal waiting times* that were used in heuristic approach 1 and described in section 8.2, but modified accordingly, so that at different stages the appropriate function G_3 or G_e (i.e. equation (8.4.5) or (8.4.8)) is involved in the calculation of the objective value.

8.4.3 Description of the main algorithmic functions

As in heuristic approach 1, given a route R, we can calculate or update the solution $S := (R, \widetilde{W}, \widetilde{G}, W, G)$ as follows:

(i) Given R, we can calculate the vector $\widetilde{\mathbf{W}}$ using the algorithmic function *time*: $\widetilde{\mathbf{W}} = \boldsymbol{time}(\mathbf{R})$ (as defined in subsection 8.2.3).

(ii) Given R and $\widetilde{\mathbf{W}}$, we can calculate $\widetilde{\mathbf{G}}$ using the algorithmic function *cost_2*: $\widetilde{\mathbf{G}} = \mathbf{cost\_2}(\mathbf{R}, \widetilde{\mathbf{W}})$:

[3]After some experimentation, we decided to use $P_1 = 400$, $P_1^* = 50000000$, $P_3 = 100$ and $P_3^* = 20000000$.

Given R and $\widetilde{W}$, the algorithmic function *cost_2* calculates the value of the objective function without allowing idle waiting times at nodes, when following the collection of routes R and visiting the nodes at times $\widetilde{W}$. This value is stored as $\widetilde{G}$ and for its calculation we use either function G_3 and equation (8.4.5) whenever *cost_2* is called in step 4 of the algorithm, or function G_ϵ and equation (8.4.8) whenever *cost_2* is called in step 6 of the algorithm.

In more detail, given a collection of routes R and a vector $\widetilde{W}$, the algorithmic function *cost_2* compares the time of start of service $\widetilde{W}_i$ of each node with the respective time window $[a_i, b_i]$ and thus calculates the values of each function F_j ($j = 1, \dots, 5$). Initially each F_j is set to zero. Then, for all nodes $i = 1, \dots n$ that represent customers, if $\widetilde{W}_i > b_i$, then F_1 is increased by 1 and F_3 is increased by $\widetilde{W}_i - b_i$. Similarly, for all $i = 1, \dots n$, if $\widetilde{W}_i < a_i$, then F_2 is increased by 1 and F_4 is increased by $a_i - \widetilde{W}_i$. Finally, F_5 is defined as $\sum_{j=1}^{d} \widetilde{W}_{n+d+j}$. Then the values of f_j ($j = 1, 2, 3$) are defined as $f_1 := F_1 + F_2$, $f_2 := F_3 + F_4$ and $f_3 := F_5$. Hence the first summation term $\sum_{j=1}^{3} \beta_i' f_i$ of equations (8.4.5) and (8.4.8) is defined. The second summation term of equations (8.4.5) and (8.4.8), which is the penalty term for capacity constraint violation, is straightforward to calculate, as explained in subsection 8.2.1. The third part of equation (8.4.8) that involves penalizing G_e in case of a violation of an epsilon constraint, is also straightforward to calculate, as explained in subsection 8.4.1. Note that at different stages of the algorithm, only one of the two functions G_3 and G_e (i.e. equations (8.4.5) and (8.4.8)) is involved in the calculation of the objective value, as was previously explained. Therefore, the value of the objective function is calculated and returned as the output $\widetilde{G}$ of the algorithmic function *cost_2*; i.e. $\widetilde{G} = cost\_2(R, \widetilde{W})$.

Note that, more generally, given R and a vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes (for instance, we may use $\widetilde{W}$ or W as $\widehat{W}$), we can still use the same algorithmic function *cost_2* to calculate the respective value of the objective function $\widehat{G}$, as $\widehat{G} = cost\_2(R, \widehat{W})$. Specifically, given R and the times of start of service W while allowing idle waiting times at nodes, we can use the same algorithmic function *cost 2* to calculate G, as $G = cost\_2(R, W)$.

(iii) Given $\boldsymbol{R}$ and $\widetilde{W}$, we can calculate $\boldsymbol{W}$ and $\boldsymbol{G}$ using the algorithmic function *optimal waiting times 2*: *[W, G]=optimal_waiting_times_2(R,$\widetilde{W}$)*:

This algorithmic function is defined the same way that the algorithmic function *optimal waiting times* was defined in section 8.2, but with the following difference: Instead of calling internally the algorithmic function *cost*, the algorithmic function *optimal waiting times 2* calls the function *cost_2*.

Therefore, given R and $\widetilde{W}$, the algorithmic function *optimal waiting times 2* calculates and returns a heuristic approximation of the optimal times of start of service W and the corresponding value of the objective function G when allowing idle waiting times at nodes and when following the collection of routes R and visiting the nodes at times W, where G is calculated either using function G_3 and equation (8.4.5) if it is called in step 4 of the algorithm, or using function G_ϵ and equation (8.4.8) if it is called in step 6 of the algorithm.

Note that we can still use the function *optimal waiting times 2* if we replace vector $\widetilde{W}$ in the input with vector $\widehat{W}$, which may or may not involve some idle waiting times at nodes.

8.4.4 Algorithm description

In this subsection we present the algorithm involved in Heuristic Approach 3. This algorithm uses the framework of the Epsilon Constraint Method and Tabu Search with multiple restarts, to construct an approximate representative subset of the set of non-dominated solutions for the SCD-VRPTW-3. These are the main steps of the algorithm:

1. Parameters input & Initialization:
   - Input: n, d, C_k and T_k (for $k = 1, ..., d$), Θ, Φ, TL (Tabu Length).
   - Define $T_{min} := min\{T_k | k = 1, ..., d\}$.
   - Input: (X_i, Y_i), $[a_i, b_i]$, s_i, D_i $\forall\ i = 1, 2, \ldots, n + 2d$.
   - Input: J_1, J_2 (parameters controlling the number of Tabu Search iterations without an improvement before the algorithm terminates, in Tabu Stages A and B, respectively).
   - For all $i, j \in \{1, 2, \ldots, n + 2d\}$, calculate the travel time $t_{i,j}$ as the Euclidean distance $t_{i,j} := \sqrt{(X_i - X_j)^2 + (Y_i - Y_j)^2}$ between each pair of nodes (i, j), rounded to 2 decimal points.

2. Calculate lower bounds for the component objectives f_1 and f_2:
   - Identify the set of *Definitely Late customers*, $I_{DL} = \left\{ i \in I_C \;\middle|\; \min_{k \in K}(T_k + s_{n+k} + t_{n+k,i}) > b_i \right\}$, as described in chapter 7. Its cardinality $|I_{DL}|$ is the number of definitely late customers, which serves as a lower bound for f_1 (and also for F_1), and is thus denoted by $LB_{f_1} := |I_{DL}|$.
   - Evaluate the *minimum amount of lateness*:= $\sum_{i \in I_{DL}} [\min_{k \in K}(T_k + s_{n+k} +$

$t_{n+k,i}) - b_i]$, which serves as a lower bound for f_2 (and also for F_3), and is therefore denoted by LB_{f_2}.

3. Construct the Initial Route R^0:

   In order to construct the *initial route* R^0, we use the set of routes involved in the original plan (which is supposed to be the best possible plan found for the original problem, optimal or near-optimal), after removing the first part of each route containing the customers that have already been served.

   Then in route R_k^0 of vehicle k (for all $k = 1, ..., d$) we use the position of vehicle k at time T_k as the starting position (represented by node $n + k$) and include this as the first node of the route. The depot is the end-point of each route R_k^0, so we also include a copy of the depot (represented by node $n + d + k$) as the last node of route R_k^0. [4]

   The collection of routes $R^0 = (R_1^0, R_2^0, ..., R_d^0)'$ is used as the initial solution's collection of routes and will be simply referred to as *the initial route*. Given the initial route R^0, we then calculate the respective $\widetilde{W}^0$. We also let $R := R^0$ and $\widetilde{W} := \widetilde{W}^0$.

4. Objective throughout Step 4:

   Throughout Step 4, the objective is to minimize the function G_3 defined by equation (8.4.5), and the respective value found at any time is stored as $\widetilde{G}$ when idle waiting times at nodes are not allowed, or as G when idle waiting times at nodes are allowed.

   Step 4a:

   - Given the initial route R^0 and the corresponding vector $\widetilde{W}^0$ found in the previous step, calculate the full initial solution $S^0 = (R^0, \widetilde{G}^0, G^0, \widetilde{W}^0, W^0)$.

   - Initialize S by letting $S := S^0$.

[4]Note that any violation of the time windows of the starting nodes $n+k$ or the endpoint nodes $n+d+k$ will not be reflected in the objectives f_1 or f_2 (for $k = 1, ..., d$). For instance, if a vehicle arrives late at the endpoint node $n+d+k$, this will not be considered a violation of the respective time window, so it will not be counted in f_1 (number of customers served outside their time windows) and its amount of violation of the time window will not be added to f_2.

- Using S^0 as the initial solution, *run the algorithm described by Heuristic Approach 2 (in section 8.3)*, with G_3 as the objective to be minimized and with a choice of weights $(\beta'_1, \beta'_2, \beta'_3)$ such that $\beta'_1 >> \beta'_2 >> \beta'_3 > 0$, to find an optimal or near-optimal solution for the Lexicographic SCD-VRPTW with 3 Objectives.[5] Store this solution as S^{Lex}, where $S^{Lex} = (R^{Lex}, \widetilde{W}^{Lex}, \widetilde{G}^{Lex}, W^{Lex}, G^{Lex})$. Throughout this section, we will call this solution the *lexicographic solution.*

Step 4b:

- Using S^{Lex} as the initial solution, *perform three different restarts of the algorithm described by Heuristic Approach 2 (in section 8.3)*, with G_3 as the objective to be minimized, where each time we use one of the following three sets of weights $(\beta'_1, \beta'_2, \beta'_3)$ respectively: $(1, 0, 0)$, $(0, 1, 0)$ and $(0, 0, 1)$.

Essentially, in each one of these three runs the algorithm solves heuristically the mathematical program *(MOMILP 3C)* that was defined in section 7.12, which is transformed into a single-objective program, with only one of f_1, f_2 and f_3 as the objective to be minimized, neglecting the other two component objectives (but storing their values each time).

From each one of the three different runs, we get a triplet of the form (f_1, f_2, f_3) containing the values of the three component objectives. Then, separately for each $i = 1, 2, 3$, the three values found for each f_i are compared with each other to find the smallest value f_i^m and the largest value f_i^M among the three values of each component objective. Also, let min_{f_i} and max_{f_i} denote the true minimum and maximum value of f_i, respectively, for each $i = 1, 2, 3$. Note that f_i^m is not necessarily the same as the true minimum value min_{f_i} of f_i, but is an overestimate of min_{f_i} that should be close to the true value. On the other hand, not only is f_i^M generally different to the true maximum value max_{f_i} of f_i, but it is an underestimate of max_{f_i} that may be substantially smaller than the true maximum value. However, since we are minimizing, we do not really need the maximum value for any of the three component objectives f_1, f_2 and f_3. In fact, we only treat the values f_i^m and f_i^M as reference values that specify a part of the range of each component objective f_i which is of interest. Some of those values are used below to construct the set of different values that the epsilons will take.

[5]In our experiments, we have used the values $(10^8, 10^3, 10^{-3})$ for the parameters $(\beta'_1, \beta'_2, \beta'_3)$ of Step 4a.

We use the number of definitely late customers $|I_{DL}|$ as the lower bound LB_{f_1} for f_1, and the number of all active customers n as the upper bound UB_{f_1} for f_1; i.e. we set $LB_{f_1} := |I_{DL}|$ and $UB_{f_1} := n$. Note that these are both true bounds.

Regarding f_1, the bounds LB_{f_1} and UB_{f_1} are used below to define the range of values that ϵ_1 may take. Also, regarding f_3, the reference values f_3^m and f_3^M are used to define the range of values that ϵ_3 may take. Note that we do not make use of and therefore we do not need any bounds or reference values for f_2.

Also, regarding the true maximum value of f_3, this could be substantially greater than the value f_3^M that was calculated above. However, the true maximum value of f_3 corresponds to a solution with a maximum sum of arrival times at the endpoint node, which is comparable to a solution involving routes of maximal total length, which is very undesirable and irrelevant in problems like the one under study. Hence, we believe that f_3^M is in fact more meaningful and more appropriate to use than the true maximum value max_{f_3}, for the purpose of defining the largest value that ϵ_3 is allowed to take.

5. Calculate the different values that ϵ_1 and ϵ_3 will take:
We assume that ϵ_1 takes Φ different values, where Φ is a user-input parameter such that $\Phi \geq 2$. These values are calculated as follows: The interval $[LB_{f_1}, UB_{f_1}]$ is divided into $\Phi - 1$ equal intervals of length $(UB_{f_1} - LB_{f_1})/(\Phi - 1)$, and their lower end-points, together with UB_{f_1}, are collected to create the set of Φ different values that ϵ_1 will take. In other words, ϵ_1 takes the values LB_{f_1}, $LB_{f_1} + (UB_{f_1} - LB_{f_1})/(\Phi - 1)$, $LB_{f_1} + 2(UB_{f_1} - LB_{f_1})/(\Phi - 1)$, ..., UB_{f_1}.

   Since f_1 takes only integer values, in the case where the fraction $(UB_{f_1} - LB_{f_1})/(\Phi - 1)$ is less than 1 (or equivalently, if $\Phi > UB_{f_1} - LB_{f_1} + 1$), we can speed up the algorithm by setting $\Phi := UB_{f_1} - LB_{f_1} + 1$, i.e. reducing Φ into a smaller and more meaningful value, which results in ϵ_1 taking all the integer values from LB_{f_1} to UB_{f_1} inclusive.

   Similarly, we assume that ϵ_3 takes Θ different values, where Θ is a positive integer user-input parameter. These values are calculated as follows: The interval $[f_3^m, f_3^M]$ is divided into $\Theta - 1$ equal intervals of length $(f_3^M - f_3^m)/(\Theta - 1)$, and their lower end-points, together with f_3^M, are collected to create the set of Θ values that ϵ_3 will take.

   Let $E^1 = (E_1^1, E_2^1, ..., E_\Phi^1)$ and $E^3 = (E_1^3, E_2^3, ..., E_\Theta^3)$ be the vectors containing all possible values in descending order, that ϵ_1 and ϵ_3 take, respectively.

6. Objective throughout Step 6:

   Throughout Step 6, the objective is to minimize the function G_ϵ defined by equation (8.4.8), and the respective value found at any time is stored as $\widetilde{G}$ when idle waiting times at nodes are not allowed, or as G when idle waiting times at nodes are allowed.

   Step 6:

   For all $i = 1, 2, ..., \Phi$ and for all $j = 1, 2, ..., \Theta$ do:
   - Let $\epsilon_1 := E_i^1$ and $\epsilon_3 := E_j^3$.
   - Check if iteration (i, j), which corresponds to solving *MILP-3F-ECM* $(\epsilon_1,\ \epsilon_3)$, can be skipped, according to the criteria and following the procedures described in subsection 8.4.5. If not, then try to solve heuristically MILP-3F-ECM(ϵ_1, ϵ_3) using Tabu Search, by following the procedure described below:

   - Initialize the *current solution* S^2 as follows: If (ϵ_1, ϵ_3) is the first element of table 8.1, then the *current route* R^2 is set to be the route R^{Lex} corresponding to the *lexicographic solution* S^{Lex} found in Step 4. If (ϵ_1, ϵ_3) is an element of the first row of the table, but not the one in the first row and first column, i.e. in cell $(1, j)$ with $j > 1$, then the starting route is set to be the route corresponding to the best solution found in the previous iteration $(1, j - 1)$, i.e. the best route found corresponding to the cell of the same row and the previous column of the table. Finally, if (ϵ_1, ϵ_3) is not an element of the first row, i.e. in cell (i, j) with $i > 1$, then the starting route is set to be the route corresponding to the best solution found in iteration $(i - 1, j)$, i.e. the best route found in the previous row and same column.
   - Once the *current route* R^2 is defined, create or update the complete *current solution* S^2 accordingly.
   - Initialize the *best solution found in iteration* (i, j), $S^{best\_ECM}(i, j)$, by letting $S^{best\_ECM}(i, j) := S^2$.
   - Apply Tabu Search with best improvement criterion, using the exchange and insertion neighborhood moves, to find the best solution of iteration (i, j), that is $S_{best\_ECM}(i, j)$. More specifically, apply Tabu Search Stages A and B of Step 4 of the Heuristic

Approach 2, as described in section 8.3 (which refers to the respective part of Heuristic Approach 1, described in subsection 8.2.4, but while having the 3 component objectives f_1, f_2, f_3 involved, instead of the 5 component objectives $F_1, F_2, ..., F_5$), with the following changes: (i) Replace S^{best} by $S^{best\_ECM}(i,j)$, where $S^{best\_ECM}(i,j)$ is the best solution found so far in iteration (i,j) and (ii) replace the function *optimal waiting times* by the function *optimal waiting times 2*.

- Once the Stages A and B of Step 4 of the Tabu Search (in Heuristic Approach 1) are finished, we have the best solution $S^{best\_ECM}(i,j)$ found in iteration (i,j), corresponding to the epsilons $(\epsilon_1, \epsilon_3) = (E_i^1, E_j^3)$, i.e. the best solution found for MILP-3F-ECM(E_i^1, E_j^3). For each iteration (i,j), we record the values of the three component objective functions f_1, f_2, f_3 that correspond to this best solution, in the triplet $(f_{i,j}^1, f_{i,j}^2, f_{i,j}^3)$.

7. - Compare all the best solutions found, $S^{best\_ECM}(i,j)$, between each other, for all values of (i,j), by comparing the corresponding triplets $(f_{i,j}^1, f_{i,j}^2, f_{i,j}^3)$. Remove all dominated solutions from the solution set.

   For instance, if $(f_{i_1,j_1}^1, f_{i_1,j_1}^2, f_{i_1,j_1}^3)$ and $(f_{i_2,j_2}^1, f_{i_2,j_2}^2, f_{i_2,j_2}^3)$ are two such solutions found so that $f_{i_1,j_1}^1 \leq f_{i_2,j_2}^1$, $f_{i_1,j_1}^2 \leq f_{i_2,j_2}^2$, $f_{i_1,j_1}^3 \leq f_{i_2,j_2}^3$ and at least one of the previous three inequalities is strict, then the second solution is said to be dominated by the first solution and is therefore removed from the set of solutions.[6]

   - Return the resulting set of all non-dominated solutions found, which is an approximate subset of the true set of non-dominated solutions for the SCD-VRPTW-3.
   - Then plot the respective Value Paths, showing the corresponding triplets $(f_{i,j}^1, f_{i,j}^2, f_{i,j}^3)$ for all the non-dominated solutions found.

8.4.5 Two procedures to speed up the ECM algorithm

Let $E^1 = (E_1^1, E_2^1, ..., E_\Phi^1)$ be the vector containing all Φ possible values that ϵ_1 may take, in descending order. Similarly, let $E^3 = (E_1^3, E_2^3, ..., E_\Theta^3)$ be the vector containing all Θ possible values that ϵ_3 may take, in descending order.

Suppose that we want to solve heuristically $\Phi \times \Theta$ programs of the form *(MILP 3E)*, denoted by MILP-3E-ECM(ϵ_1, ϵ_3), one for each possible com-

[6]Actually, even if none of those three inequalities is strict, we can still remove one of the two solutions, since they give exactly the same values to the three objectives f_1, f_2, f_3 and are therefore equivalent.

bination of (ϵ_1, ϵ_3), where the objective (8.4.1) is to minimize f_2. Assume that $f^2_{i,j}$ is the optimal solution of the program MILP-3E-ECM(E^1_i, E^3_j), and that for this solution the corresponding values of f_1 and f_3 are $f^1_{i,j}$ and $f^3_{i,j}$ respectively, for all i=1,2,...,Φ, j=1,2,...,Θ.

By solving all $\Phi \times \Theta$ programs we will get a table of the same form as table 8.1, where in each cell (i, j) we record $f^2_{i,j}$ ($f^1_{i,j}$, $f^3_{i,j}$), i.e. the value of the main objective f_2 for the respective pair of values for ϵ_1 and ϵ_3, followed by the values of the other two objectives f_1 and f_3 in parentheses.

Table 8.1: ECM solutions table

ϵ_3 / ϵ_1	E^3_1	E^3_2	...	E^3_j	...	E^3_Θ
E^1_1	$f^2_{1,1}$ ($f^1_{1,1}$, $f^3_{1,1}$)	$f^2_{1,2}$ ($f^1_{1,2}$, $f^3_{1,2}$)	...	...	...	...
E^1_2	...	...	...	...	...	...
...	...	...	...	...	...	...
E^1_i	...	...	...	$f^2_{i,j}$ ($f^1_{i,j}$, $f^3_{i,j}$)	...	...
...	...	...	...	...	...	...
E^1_Φ	...	...	...	...	...	...

We will now describe the first procedure employed, in order to speed up the ECM algorithm. Assume that for some indexes (i, j), we have found the optimal solution $f^2_{i,j}$ ($f^1_{i,j}$, $f^3_{i,j}$) to the program MILP-3E-ECM(E^1_i, E^3_j). Therefore, this solution gives the minimum value $f^2_{i,j}$ of f_2, while satisfying the constraints $f^1_{i,j} \leq E^1_i$ and $f^3_{i,j} \leq E^3_j$. Then we can perform the following check:

We take the next row-index $i + 1$ and we check if $f^1_{i,j} \leq E^1_{i+1}$. If yes, then this obviously means that the solution found above is also the optimal solution to the program MILP-3E-ECM(E^1_{i+1}, E^3_j). We continue increasing the row-index by one and performing a similar check, as long as there is no violation of the ϵ-constraint for f_1. Assume that $f^1_{i,j} \leq E^1_{i+2}$, $f^1_{i,j} \leq E^1_{i+3}$, ..., $f^1_{i,j} \leq E^1_{i+\chi}$ are all satisfied, while $f^1_{i,j} \leq E^1_{i+\chi+1}$ is not satisfied.

We then take the next column-index $j + 1$ and we check if $f^3_{i,j} \leq E^3_{j+1}$. If yes, then this means that the solution found above is also the optimal solution to the program MILP-3E-ECM(E^1_i, E^3_{j+1}). We continue increasing the column-index by one and performing a similar check, as long as there is no violation to the ϵ-constraint for f_3. Assume that $f^3_{i,j} \leq E^3_{j+2}$, $f^3_{i,j} \leq E^3_{j+3}$, ..., $f^3_{i,j} \leq E^3_{j+\psi}$ are all satisfied, while $f^3_{i,j} \leq E^3_{j+\psi+1}$ is not satisfied.

Therefore, the optimal solution to the program MILP-3E-ECM(E^1_i, E^3_{j+1}) found before, is also the optimal solution to the program MILP-3E-ECM(E^1_ζ, E^3_η), for all pairs of indexes (ζ, η) with $\zeta = i, i+1, ..., i+\chi$, $\eta = j, j+1, ..., j+\psi$ and $(\zeta, \eta) \neq (i, j)$.

The conclusion is that there is no need to solve any of the above $(\chi+1)\times(\psi+1)-1$ programs, which can considerably speed up the ECM algorithm. This idea can be visualized in table 8.2, where the programs whose solution can be skipped are marked with an 'X'.

Table 8.2: First procedure to speed up the ECM Algorithm

ϵ_3 / ϵ_1	E^3_1	...	E^3_j	E^3_{j+1}	E^3_{j+2}	...	$E^3_{j+\psi}$	$E^3_{j+\psi+1}$	...	E^3_Θ
E^1_1	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...
E^1_i	...	...	$f^2_{i,j}$ $(f^1_{i,j}, f^3_{i,j})$ with $f^1_{i,j} \leq E^1_{i+\chi}$, $f^3_{i,j} \leq E^3_{j+\psi}$	X	X	...	X	...	...	...
E^1_{i+1}	...	...	X	X	X	...	X	...	...	...
E^1_{i+2}	...	...	X	X	X	...	X	...	...	...
...	...	...	...	...	...	...	...	...	...	...
$E^1_{i+\chi}$	...	...	X	X	X	...	X	...	...	...
$E^1_{i+\chi+1}$	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...
E^1_Φ	...	...	...	...	...	...	...	...	...	...

Moving on, we will now describe the second procedure employed, in order to speed up the ECM algorithm. Assume that for some indexes (i, j), the program MILP-3E-ECM(E^1_i, E^3_j) is infeasible, violating either the constraint $f^1_{i,j} \leq E^1_i$, or $f^3_{i,j} \leq E^3_j$ or both.

If the constraint $f_1 \leq E^1_i$ is violated, then so will the constraints $f_1 \leq E^1_\zeta$, for all $\zeta = i+1, i+2, ..., \Phi$, since the values $E^1_1, E^1_2, ..., E^1_\Phi$ were sorted in decreasing order. Similarly, if the constraint $f_3 \leq E^3_j$ is violated, then so will the constraints $f_3 \leq E^3_\eta$, for all $\eta = j+1, j+2, ..., \Theta$, since the values $E^3_1, E^3_2, ..., E^3_\Phi$ were also sorted in decreasing order.

Therefore, if for some indexes (i, j) the program MILP-3E-ECM(E^1_i, E^3_j) is infeasible, violating either the constraint $f^1_{i,j} \leq E^1_i$, or $f^3_{i,j} \leq E^3_j$ or both, then the programs MILP-3E-ECM(E^1_ζ, E^3_η) will also be infeasible, for all pairs of indexes (ζ, η) with $\zeta = i, i+1, ..., \Phi$, $\eta = j, j+1, ..., \Theta$ and $(\zeta, \eta) \neq (i, j)$.

The conclusion is that there is no need to solve any of the above $(\Phi - i + 1) \times (\Theta - j + 1) - 1$ programs, which can considerably speed up the ECM algorithm. This idea can be visualized in table 8.3, where the programs whose solution can be skipped are marked with an 'X'.

Note that the discussion in this subsection, up to this point, refers to programs MILP-3E-ECM(E^1_i, E^3_j) which are based on formulation *(MILP 3E)*, where the objective (8.4.1) is to minimize f_2. Throughout Step 6 of the algorithm of Heuristic Approach 3, described in subsection 8.4.4, we apply the same reasoning when solving heuristically programs MILP-3F-ECM(E^1_i, E^3_j) which are based on formulation *(MILP 3F)*, where the objective (8.4.8) is to minimize G_ϵ. If the heuristic can reach a feasible solution, then G_ϵ reduces

Table 8.3: Second procedure to speed up the ECM Algorithm

ϵ_3 / ϵ_1	E^3_1	...	E^3_j	E^3_{j+1}	E^3_{j+2}	...	E^3_Θ
E^1_1	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...
E^1_i	...	...	if MILP-3E-ECM(E^1_i, E^3_j) is infeasible, wrt $f_1 \leq E^1_i$ and/or $f_3 \leq E^3_j$	X	X	...	X
E^1_{i+1}	...	...	X	X	X	...	X
E^1_{i+2}	...	...	X	X	X	...	X
...	...	...	...	...	...	...	
E^1_Φ	...	...	X	X	X	...	X

to $\sum_{i=1}^{3} \beta'_i f_i$. Since we use positive weights $(\beta'_1, \beta'_2, \beta'_3)$ such that $\beta'_2 = 1$, $\beta'_1 << 1$ and $\beta'_3 << 1$, i.e. β'_1 and β'_3 are marginal, this is almost equivalent to minimizing f_2. Therefore, we can apply the same ideas of this subsection when solving heuristically programs of the form MILP-3F-ECM(E^1_i, E^3_j).

8.5 Experimental Results

This section presents the experimental results from the different solution methods proposed for the *Single-Commodity Delayed VRPTW* (Problem 3), including both exact and heuristic approaches.

Starting from 4 different VRPTW instances that were selected from the classic Solomon (1987) dataset (namely, instances rc201, rc202, c203 and c101), we created 36 instances for the *SCD-VRPTW*. Note that in our tables we include the results from one additional instance, namely instance 9 (rc201-B1), which is identical to the original VRPTW instance rc201 without disruption. This instance was included in the tables of results to show that, since SCD-VRPTW is a generalization of the VRPTW, we can use the SCD-VRPTW to solve a standard VRPTW (but with the objective of minimizing the sum of arrival times at the endpoint node, instead of the total travel distance). Therefore our dataset contains 37 instances in total.

In short, starting from each one of the four benchmark VRPTW instances mentioned above, and taking into account the respective best solution reported in the literature for each instance, table 8.4 provides the additional information needed to construct the corresponding instances with disruption for the SCD-VRPTW. Table 8.5 presents the experimental results of using Exact Approach 1, which was described in section 7.9 of the previous chapter and implemented in AIMMS, to solve the 37 instances of Problem 3. Table 8.6 presents the respective experimental results of Heuristic Approach 1, which was described in section 8.2 and implemented in MATLAB. Table 8.7 presents the summary results of Heuristic Approach 2, described in section 8.3, also implemented in MATLAB. Finally, the experimental results of

Heuristic Approach 3, which was described in section 8.4 and implemented in MATLAB, are discussed in subsection 8.5.8, whereas the corresponding tables of results can be found in Appendix B.

8.5.1 Discussion about the SCD-VRPTW instances

The 37 instances for the SCD-VRPTW are divided into 5 classes: A, B, C, D and E. Each class contains different instances for problem 3 that are based on the same baseline VRPTW instance from Solomon (1987) dataset. Specifically, problems in classes A and B were created based on the VRPTW instance rc201, whereas problems in classes C, D and E were created based on the VRPTW instances rc202, c203 and c101 respectively. Table 8.4 presents some additional parameters that are necessary to construct the instances for the SCD-VRPTW, given the parameters of the underlying undisrupted VRPTW instance and the respective best solution reported in the literature. Specifically, the columns of this table present the following information for each instance of the SCD-VRPTW: the serial number and instance name, the number of customers n, the number of vehicles d, the times of departure T_1, T_2,..., T_d of the vehicles from the starting nodes, the quantities C_1, C_2,..., C_d of the commodity carried by the vehicles at their respective times of departure, and finally the starting nodes of each vehicle (indicated by their original numbering, as in the underlying VRPTW instance), which will be relabeled as nodes $n+1$, $n+2$,..., $n+d$ in the disrupted problem. Note that, given T_1, T_2,..., T_d as input parameters, the parameters T and T_0 are not actually required to define a SCD-VRPTW instance.

Each instance of the same class in the SCD-VRPTW among classes B, C, D and E, corresponds to a variation of the underlying VRPTW, with the same number of vehicles as in the original VRPTW, assuming different amounts of delay in different routes, which is implied by having the same starting nodes but different set of times of departure $T_1, T_2, ..., T_d$.

For example, starting from instance rc202 of the standard VRPTW, we can construct instance 17 (rc202-C1) by using the input parameters of the original instance rc202, the optimal or best solution reported in the literature for this instance, as well as the information provided in the respective row of table 8.4, as follows: Given the starting nodes 28, 37, 69 of the three vehicles in the disrupted problem, we can first go through the routes of the best solution for the original VRPTW instance up to the point where we meet the above nodes. Then, to create the new instance for the SCD-VRPTW, we must remove the customers that have already been served up to that point and the new starting nodes from the set of customer nodes, define the new starting nodes and label them as $n+1, ..., n+d$, and relabel the remaining

Serial no. & Instance name	**n**	**d**	$\mathbf{T_1, ..., T_d}$	$\mathbf{C_1, ..., C_d}$	**starting nodes (original numbering)**
Class A					
1 - rc201-A1	11	3	713, 703, 710.5	1000, 1000, 1000	89, 17, 35
2 - rc201-A2	11	3	800, 703, 710.5	210, 210, 210	89, 17, 35
3 - rc201-A3	11	3	830, 830, 830	100, 70, 70	89, 17, 35
4 - rc201-A4	11	3	870, 870, 870	100, 70, 70	89, 17, 35
5 - rc201-A5	11	3	900, 870, 840	150, 150, 150	89, 17, 35
6 - rc201-A6	11	3	900, 870, 840	100, 70, 70	89, 17, 35
7 - rc201-A7	11	3	900, 870, 840	82, 82, 82	89, 17, 35
8 - rc201-A8	11	3	900, 870, 840	210, 210, 210	89, 17, 35
Class B					
9 - rc201-B1	100	4	0, 0, 0, 0	1000,1000,1000,1000	0,0,0,0
10 - rc201-B2	100	4	100, 100, 100, 100	1000,1000,1000,1000	0,0,0,0
11 - rc201-B3	100	4	100, 100, 0, 0	1000,1000,1000,1000	0,0,0,0
12 - rc201-B4	100	4	200, 100, 0, 0	1000,1000,1000,1000	0,0,0,0
13 - rc201-B5	100	4	200, 100, 200, 100	1000,1000,1000,1000	0,0,0,0
14 - rc201-B6	100	4	200, 200, 200, 200	1000,1000,1000,1000	0,0,0,0
15 - rc201-B7	100	4	300, 200, 100, 0	1000,1000,1000,1000	0,0,0,0
16 - rc201-B8	100	4	300, 0, 0, 0	1000,1000,1000,1000	0,0,0,0
Class C					
17 - rc202-C1	81	3	200, 204, 230.94	962, 920, 866	28, 37, 69
18 - rc202-C2	81	3	250, 204, 230.94	962, 920, 866	28, 37, 69
19 - rc202-C3	81	3	250, 250, 230.94	962, 920, 866	28, 37, 69
20 - rc202-C4	81	3	300, 250, 230.94	962, 920, 866	28, 37, 69
21 - rc202-C5	81	3	300, 300, 300	962, 920, 866	28, 37, 69
22 - rc202-C6	81	3	400, 300, 300	962, 920, 866	28, 37, 69
23 - rc202-C7	81	3	400, 400, 230.94	962, 920, 866	28, 37, 69
Class D					
24 - c203-D1	54	3	1500, 1527.76, 1531.15	500, 410, 400	49, 28, 91
25 - c203-D2	54	3	1500, 1650, 1531.15	500, 410, 400	49, 28, 91
26 - c203-D3	54	3	1500, 1650, 1650	500, 410, 400	49, 28, 91
27 - c203-D4	54	3	1650, 1650, 1650	500, 410, 400	49, 28, 91
28 - c203-D5	54	3	1500, 1800, 1531.15	500, 410, 400	49, 28, 91
29 - c203-D6	54	3	1900, 1900, 1900	500, 410, 400	49, 28, 91
30 - c203-D7	54	3	1500, 2000, 2000	500, 410, 400	49, 28, 91
31 - c203-D8	54	3	2000, 1527.76, 2300	500, 410, 400	49, 28, 91
Class E					
32 - c101-E1	77	10	195, 195, 195, 195, 195, 206.62, 219.01, 218.19, 198.13, 232.43	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76
33 - c101-E2	77	10	195, 195, 195, 195, 195, 300, 300, 218.19, 198.13, 232.43	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76
34 - c101-E3	77	10	195, 195, 300, 300, 195, 206.62, 219.01, 218.19, 198.13, 232.43	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76
35 - c101-E4	77	10	300, 300, 300, 300, 300, 300, 300, 300, 300, 300	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76
36 - c101-E5	77	10	195, 250, 300, 350, 195, 206.62, 219.01, 218.19, 198.13, 400	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76
37 - c101-E6	77	10	250, 350, 450, 400, 300, 500, 350, 400, 198.13, 232.43	190, 190, 190, 170, 200, 170, 130, 170, 180, 150	24, 65, 42, 17, 0, 86, 31, 95, 7, 76

Table 8.4: Additional information needed to construct the SCD-VRPTW instances

customers as nodes 1,2,...,n (where n is the number of customers that have not already been served at the time of disruption, excluding customers that become starting nodes, if any). Instead of having the vehicles departing from nodes 28, 37, 69 at the times specified in or implied by the best solution of the VRPTW, we assume that vehicles will depart from these nodes at different times $T_1 = 200$, $T_2 = 204$, $T_3 = 230.94$ (which must be greater than or equal to the respective times in the original VRPTW). Also, note that in the original problem, the capacity of each vehicle was equal to 1000 units. We assume that each vehicle had departed fully loaded from the depot, thus carrying 1000 units of the commodity. For vehicle 1, the total demand of the customers already served up to time T_1 is equal to 38. Therefore, C_1 is defined as $1000 - 38 = 962$. This is the quantity of the commodity that vehicle 1 carries at time T_1 (which can also be thought as the new 'capacity' of vehicle 1 in the disrupted problem). We can calculate $C_2 = 920$ and $C_3 = 866$ using the same reasoning. Therefore, the new instance for the SCD-VRPTW is now fully characterized.

Instances of class A were created in a slightly different way. Specifically, starting from instance rc201 of the standard VRPTW, which involved 4 vehicles, we first remove the vehicle which was originally labeled as vehicle 2 and its corresponding customers. Then we use nodes 89, 17 and 35 as the new starting nodes for the remaining 3 vehicles, and varied the corresponding times of departure T_1, T_2,..., T_d to create different variations. Additionally, in order to test the formulations and algorithms that we proposed for Problem 3, we experimented by modifying the quantities carried by the three vehicles that are used in the disrupted problem - even if these quantities may not necessarily be in accordance to the quantities carried or to the vehicles' capacities, as reported in the underlying problem. Therefore, once constructed, instances of class A should be seen as stand-alone problems for the SCD-VRPTW, which are not necessarily based on the respective underlying problem. These were the first instances created for this problem, involving just 11 customers, and were mainly used to test several features of the different formulations and approaches of the problem under study.

Finally, we should mention that all instances 9-16 of Class B involve the same set of 100 customers as in the underlying VRPTW instance rc201, and have the depot (node 0) as the starting point of each vehicle. As mentioned before, instance 9 represents a regular VRPTW. All of the remaining instances 10-16 of Class B correspond to a disruption in the form of one or more vehicles departing from the depot with some amount of delay. This case of a delay in the departure of some of the vehicles from the depot has been studied before as a separate problem in the context of disruption management in vehicle routing and scheduling (but not necessarily under the same assumptions and with the same constraints and objectives). One variation of this problem is known as the *Disrupted Capacitated VRP with Order Release Delay* and was studied by Mu & Eglese (2013). Note that the formulations and proposed methods for SCD-VRPTW can also solve the above problem, by treating it as a special case of the SCD-VRPTW (under the same assumptions and with the same objectives and constraints as the SCD-VRPTW).

8.5.2 Discussion about the Experimental Results of Exact Approach 1

Table 8.5 presents the experimental results of using Exact Approach 1, which was described in section 7.9 of the previous chapter, to solve the 37 instances of Problem 3. In this approach, formulation (MILP 3B) of section 7.7 was implemented in AIMMS, in an attempt to solve the Lexicographic SCD-VRPTW with 5 objectives. The columns of table 8.5 provide the following information for each instance: The serial number and code-name of the instance, the values of the 5 component objectives F_1, F_2, ..., F_5, the runtime in seconds, the value of the objective $(\sum_{i=1}^{5} \beta_i F_i)$, which is denoted by G^{EA_1} (to denote the objective G in Exact Approach 1), the lower bound $LB_G^{EA_1}$ of G^{EA_1} found by AIMMS, the optimality gap provided by AIMMS, which is calculated as $\frac{G^{EA_1} - LB_G^{EA_1}}{G^{EA_1}} \cdot 100\%$, and finally whether the solution is Optimal, Feasible or Infeasible (O/F/I). Note that in the experiments with Exact Approach 1, we used the following parameter values: $M = 5000$, $\epsilon = 0.00001$ and $(\beta_1, \beta_2, \beta_3, \beta_4, \beta_5) = (10^8, 10^7, 10^3, 1, 10^{-3})$.

We tried several different cutoff times when experimenting with Exact Approach 1. Specifically, in instances 1-3 we applied a 15-minute cutoff time, in instances 4-9 a 10-minute cutoff time, and in instances 10-31 a 5-minute cutoff time. In instances 32-37 we did not apply any cutoff time, but instead we interrupted these instances after allowing different amounts of runtimes, varying from 14.5 minutes up to an hour. From table 8.5 we can see that, apart from the instances of Class A, where only 11 customers are involved, in all the remaining problems, Exact Approach 1 fails to find a satisfactory solution within reasonable time. First of all, in all the problems of Classes B, C, D and E, the optimality gap is between 80 and 100%. Second of all, comparing either the full aggregated objective function G, or simply the most important objective F_1 (under lexicographic preference), we can see that, even though AIMMS was able to find a feasible solution in every occasion, these solutions are much worse than those provided by Heuristic Approach 1, which are presented in table 8.6.

Serial no. & Instance name	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**runtime (sec)**	$\mathbf{G^{EA_1} = \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_G^{EA_1}}$	**% of $\mathbf{G^{EA_1}}$ from $\mathbf{LB_G^{EA_1}}$**	**O/ F/ I**
Class A										
1 - rc201-A1	0	0	0	0	2560.82	871.3	2.56082	2.56082	0.00%	O
2 - rc201-A2	0	0	0	0	2674.47	413.3	2.67447	2.67447	0.00%	O
3 - rc201-A3	5	0	409.23	0	2939.55	900.1	500409232.93955	500052542.4	0.07%	F
3b - rc201-A3	5	0	409.23	0	2939.55	4026.61	500409232.93955	500409232.93955	0.00%	O
4 - rc201-A4	8	0	605.52	0	2998.08	600.1	800605522.998080	600000169.5	25.06%	F
5 - rc201-A5	8	0	676.35	0	3071.79	600.2	800676353.071790	700028664	12.57%	F
6 - rc201-A6	8	0	651.27	0	3037.61	600.3	800651273.037610	700005476.8	12.57%	F
7 - rc201-A7	8	0	651.27	0	3037.61	600.0	800651273.037610	700034660.1	12.57%	F
8 - rc201-A8	8	0	651.27	0	3037.61	600.1	800651273.037610	700025493.6	12.57%	F
Class B										
9 - rc201-B1	62	27	23672.27	8894.92	4691.87	602.3	6493681169.61187	0	100%	F
10 - rc201-B2	67	21	37182.55	5309.10	5367.32	301.1	6947187864.46732	0	100%	F
11 - rc201-B3	64	22	35156.10	6121.90	5265.73	301.2	6655162227.165464	0	100%	F
12 - rc201-B4	64	22	35322.06	6121.90	5279.30	301.3	6655328187.1793	0	100%	F
13 - rc201-B5	69	20	37485.97	4885.80	5326.15	301.3	7137490861.125959	0	100%	F
14 - rc201-B6	72	15	41454.93	3786.42	5591.51	301.1	7391458722.01151	1400000000	81.06%	F
15 - rc201-B7	69	18	39728.59	4573.12	5484.68	301.1	7119733168.60468	0	100%	F
16 - rc201-B8	64	22	36955.51	5178.24	5347.64	301.1	6656960693.58764	0	100%	F
Class C										
17 - rc202-C1	58	7	29878.96	1841.05	4645.32	304.88	5899880805.695319	100000000.1	98.31%	F
18 - rc202-C2	62	8	45392.86	1744.96	5719.77	300.72	6325394610.67977	100000000.1	98.42%	F
19 - rc202-C3	59	8	30828.73	2186.26	4584.80	300.45	6010830920.844799	100000000.1	98.34%	F
20 - rc202-C4	59	8	31928.73	2036.26	4634.80	300.42	6011930770.8948	300000000.1	95.01%	F
21 - rc202-C5	61	7	35650.05	1700.22	4855.54	300.41	6205651755.07554	800000000.1	87.11%	F
22 - rc202-C6	63	7	36115.58	1400.22	4850.01	300.39	6406116985.069949	900000000.2	85.95%	F
23 - rc202-C7	64	7	36988.42	1676.46	4867.45	300.44	6506990101.327372	300000000.1	95.39%	F
Class D										
24 - c203-D1	19	5	14394.48	3941.66	11108.33	300.22	1964398432.76833	400000000.3	79.64%	F
25 - c203-D2	23	5	15166.99	3662.03	11096.35	300.92	2365170663.12635	400000000.3	83.09%	F
26 - c203-D3	23	7	14353.64	3876.51	11158.92	300.22	2384357527.668881	400000000.3	83.22%	F
27 - c203-D4	22	5	17466.02	3210.14	11328.77	300.23	2267469241.46876	500000000.4	77.95%	F
28 - c203-D5	21	3	17810.55	2105.26	11657.82	300.22	2147812666.917777	400000000.3	81.38%	F
29 - c203-D6	29	2	26696.77	809.78	12206.33	300.20	2946697591.98633	800000000.4	72.85%	F
30 - c203-D7	23	2	24903.30	1050.12	12410.03	300.20	2344904362.52993	400000000.3	82.94%	F
31 - c203-D8	31	3	23966.55	1912.81	12113.91	300.22	3153968474.923841	400000000.3	87.32%	F
Class E										
32 - c101-E1	44	5	23732.11	1314.85	14327.43	10801.89	4473733439.17743	200000000	95.53%	F
33 - c101-E2	48	7	30337.53	1884.62	14754.07	3601.72	4900339429.374057	200000000	95.92%	F
34 - c101-E3	50	8	28304.17	2385.64	14447.07	1801.92	5108306570.086989	200000000	96.08%	F
35 - c101-E4	55	5	31345.73	1025.51	14883.86	873.50	5581346770.39386	1000000000	82.08%	F
36 - c101-E5	50	8	29865.67	1921.34	14532.14	1202.22	5109867605.87214	200000000	96.09%	F
37 - c101-E6	61	4	33448.08	827.77	15210.65	1202.08	6173448922.98065	800000000.1	87.04%	F

Table 8.5: Experimental Results of Exact Approach 1 in AIMMS (for the Lexicographic SCD-VRPTW with 5 objectives)

For example, in instance 25, the solution provided by exact approach 1 within 5 minutes involves 23 late customers and 5 early customers, out of 54 customers in total, whereas the solution found by heuristic approach 1 (see table 8.6) involves 4 late customers and no early customer. Moreover, the lower bounds found by either method verify that the latter values of 4 and 0 match the lower bounds and are thus the optimal values for objectives F_1 and F_2, respectively. Comparable results are found e.g. in instance 25. Even worse, in instance 9 the solution provided by AIMMS within 10 minutes involves 62 late and 27 early customers, out of a total of 100 customers (i.e. only 11 out of the 100 customers are served within their time windows). The heuristic, on the other hand, finds a good enough solution within 3 minutes, with zero late and zero early customers; i.e. a solution in which all 100 customers are served within their time windows. Note that this instance is equivalent to regular VRPTW without delay, and was included both for testing purposes and to show that Problem 3 generalizes the VRPTW.

Furthermore, both methods provide lower bounds for the objective $G := \sum_{i=1}^{5} \beta_i F_i$, which is denoted by G^{EA_1} in Exact Approach 1, and by G^{HA_1} in Heuristic Approach 1. The lower bound $LB_G^{EA_1}$ found by Exact Approach 1 is the one provided by AIMMS, whereas the lower bound $LB_G^{HA_1}$ found by Heuristic Approach 1 is calculated as $LB^{HA_1} := \beta_1 \cdot LB_{F_1} + \beta_3 \cdot LB_{F_3}$ (where the calculation of LB_{F_1} and LB_{F_3} was explained in the second step of the algorithm in subsection 8.2.4).

Comparing tables 8.5 and 8.6, we can see that the lower bound found by heuristic approach 1 is strictly better than the one found by exact approach 1 in 28 out of 37 cases (without counting instance 3b, which is discussed below). The two lower bounds are exactly the same in 7 other cases where they are equal to zero. Finally, the lower bound found by the heuristic is worse than the one found by the exact approach in just 2 cases, which are the first 2 very small instances (as well as in instance 3b), where AIMMS is able to prove the optimality and thus the final lower bound equals the objective value. Therefore, when assessing the quality of the heuristic in table 8.6, we decided to calculate the optimality gap based on the lower bound LB^{HA_1} found in the heuristic approach, instead of using LB^{EA_1} found by AIMMS.

Note that in table 8.5, apart from the results for the 37 instances of our dataset, we include an additional row marked as 'instance 3b'. This is because, after solving instance 3 with a cutoff time of 15 minutes and getting a solution within 0.07% of the optimal, we tried solving the same instance without a cutoff time and recorded the results in this table as 'instance 3b'. It took AIMMS around 67 minutes to find the optimal solution, even for this small problem with $n = 11$ customers. The solution found in instance 3b was the same as the one found in instance 3 within 15 minutes, where the remaining runtime was necessary for AIMMS to prove that this solution is in fact optimal.

Exact Approach 1 may be interesting from a theoretical point of view, since in theory it can find the optimal solution, if it is given enough time. Also, it may be possible to use the formulations of chapter 7 in more sophisticated exact frameworks, such as Branch-and-Bound, Branch-and-Cut, Branch-and-Cut-and-Price etc. However, at this point we can conclude that Exact Approach 1 cannot be used directly to solve instances of the SCD-VRPTW-Lex5 in a short amount of time and without using a more sophisticated framework (such as Branch-and-Cut-and-Price), unless the instances are very small. Therefore, heuristic methods may be more appropriate to use in practice.

8.5.3 Discussion about the Experimental Results of Heuristic Approach 1

Table 8.6 presents the experimental results of Heuristic Approach 1 concerning the SCD-VRPTW-Lex5, which was implemented in MATLAB. The columns of this table provide the following information for each instance: The serial number and code-name of the instance, the values of the 5 component objectives F_1, F_2, ..., F_5, the runtime in seconds, the value of the objective function in Heuristic Approach 1, which is denoted by G^{HA_1} and is equal to $\sum_{i=1}^{5} \beta_i F_i$, provided that the corresponding solution is feasible, the lower bounds LB_{F_1} and LB_{F_3} for the component objectives F_1 and F_3 respectively, the number of definitely late customers (indicated by $|I_{DNE}|$), the total travel time, whether the solution is Optimal, Feasible or Infeasible (O/F/I), the lower bound $LB_G^{HA_1}$ of G^{HA_1}, defined as $LB_G^{HA_1} := \beta_1 \cdot LB_{F_1} + \beta_3 \cdot LB_{F_3}$, the percentage gap between the heuristic's objective value G^{HA_1} and the heuristic's lower bound $LB_G^{HA_1}$, calculated as $\frac{G^{HA_1} - LB_G^{HA_1}}{G^{HA_1}} \cdot 100\%$, and finally the percentage gap between the objective value G^{HA_1} reported in Heuristic Approach 1 and the corresponding objective value G^{EA_1} reported in Exact Approach 1, calculated as $\frac{G^{HA_1} - G^{EA_1}}{G^{EA_1}} \cdot 100\%$.

We can recall that the objective function in Heuristic Approach 1 was denoted by G and was defined by equation (8.2.1) of subsection 8.2.1, which is more complex than $\sum_{i=1}^{5} \beta_i F_i$. However, in the case of a feasible solution - and specifically in all 37 instances of our dataset - the objective value G of Heuristic Approach 1, which is denoted by G^{HA_1} in this table, is equal to $\sum_{i=1}^{5} \beta_i F_i$. Note also that $LB_G^{HA_1}$ is the lower bound found in Heuristic Approach 1, not to be confused with the lower bound $LB_G^{EA_1}$ found by AIMMS in Exact Approach 1.

Table 8.6: Experimental Results of Heuristic Approach 1 in MATLAB (for the Lexicographic SCD-VRPTW with 5 objectives)

Serial no. & Instance name	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**run-time (sec)**	$\mathbf{G^{HA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_{F_1}}$	$\mathbf{LB_{F_3}}$	$\mathbf{\lvert I_{DNE} \rvert}$	**total travel time**	**O/ F/ I**	$\mathbf{LB_G^{HA_1} :=}$ $\beta_1 \cdot \mathbf{LB_{F_1}} +$ $+\beta_3 \cdot \mathbf{LB_{F_3}}$	**% of** $\mathbf{G^{HA_1}}$ **from** $\mathbf{LB_G^{HA_1}}$	**% of** $\mathbf{G^{HA_1}}$ **from** $\mathbf{G^{EA_1}}$
Class A															
1 - rc201-A1	0	0	0	0	2565.82	2.0	2.56582	0	0	4	299.32	F	0	100.00%	0.20%
2 - rc201-A2	0	0	0	0	2675.6	1.9	2.6756	0	0	4	322.1	F	0	100.00%	0.04%
3 - rc201-A3	6	0	307.2	0	2896.97	1.8	600307202.89697	5	172.09	11	266.97	F	500172090	16.68%	19.96%
4 - rc201-A4	8	0	601.07	0	3032.77	1.8	800601073.03277	7	396.17	11	282.77	F	700396170	12.52%	0.00%
5 - rc201-A5	8	0	651.27	0	3037.61	2.0	800651273.03761	7	440.54	11	287.61	F	700440540	12.52%	0.00%
6 - rc201-A6	8	0	651.27	0	3037.61	1.9	800651273.03761	7	440.54	11	287.61	F	700440540	12.52%	0.00%
7 - rc201-A7	8	0	651.27	0	3037.61	1.7	800651273.03761	7	440.54	11	287.61	F	700440540	12.52%	0.00%
8 - rc201-A8	8	0	651.27	0	3037.61	1.8	800651273.03761	7	440.54	11	287.61	F	700440540	12.52%	0.00%
Class B															
9 - rc201-B1	0	0	0	0	3276.4	184.2	3.2764	0	0	8	1983.92	F	0	100.00%	-100.00%
10 - rc201-B2	5	0	392.7	0	3277.87	148.6	500392703.27787	0	0	17	1863.08	F	0	100.00%	-92.80%
11 - rc201-B3	1	0	22.61	0	3263.11	233.1	100022613.26311	0	0	8	1956.62	F	0	100.00%	-98.50%
12 - rc201-B4	4	0	385.07	0	3231.49	251.2	400385073.23149	0	0	8	1896.95	F	0	100.00%	-93.98%
13 - rc201-B5	15	5	2665.25	370.08	3365.44	208.2	1552665623.44544	0	0	17	1740.35	F	0	100.00%	-78.25%
14 - rc201-B6	21	3	5108.66	268.65	3438.1	307.9	2135108932.0881	14	855.33	34	1627.86	F	1400855330	34.39%	-71.11%
15 - rc201-B7	13	1	2627.1	48.44	3291.35	211.3	1312627151.73135	0	0	8	1662.02	F	0	100.00%	-81.56%
16 - rc201-B8	0	2	0	32.37	3261.25	422.9	20000035.63125	0	0	8	1915.12	F	0	100.00%	-99.70%
Class C															
17 - rc202-C1	2	2	751.32	306	2691.01	92.9	220751628.69101	1	54	29	1240.09	F	100054000	54.68%	-96.26%
18 - rc202-C2	3	2	823.45	306	2719.04	63.0	320823758.71904	1	54	31	1215.4	F	100054000	68.81%	-94.93%
19 - rc202-C3	4	5	1176.06	548.62	2795.11	153.1	451176611.41511	1	100	33	1253.53	F	100100000	77.81%	-92.49%
20 - rc202-C4	10	3	1668.71	320.39	2833.08	58.8	1031669033.22308	3	177.95	33	1242.14	F	300177950	70.90%	-82.84%
21 - rc202-C5	20	5	3309.44	541.68	2871.99	62.2	2053309984.55199	8	387.28	39	1161.99	F	800387280	61.02%	-66.91%

(Continued on the next page)

Table 8.6 – *Continued from the previous page*

Serial no. & Instance name	$\mathbf{F_1}$	$\mathbf{F_2}$	$\mathbf{F_3}$	$\mathbf{F_4}$	$\mathbf{F_5}$	**run-time (sec)**	$\mathbf{G^{HA_1} := \sum_{i=1}^{5} \beta_i F_i}$	$\mathbf{LB_{F_1}}$	$\mathbf{LB_{F_3}}$	$\mathbf{\lvert I_{DNE} \rvert}$	**total travel time**	**O/ F/ I**	$\mathbf{LB_G^{HA_1} := \beta_1 \cdot LB_{F_1} + {} + \beta_3 \cdot LB_{F_3}}$	**% of $\mathbf{G^{HA_1}}$ from $\mathbf{LB_G^{HA_1}}$**	**% of $\mathbf{G^{HA_1}}$ from $\mathbf{G^{EA_1}}$**
22 - rc202-C6	22	6	4208.96	419.38	2918.8	78.8	2264209382.2988	9	555.06	39	1108.8	F	900555060	60.23%	-64.66%
23 - rc202-C7	25	3	5387.26	361.22	2955.53	72.9	2535387624.17553	3	203.27	34	1114.59	F	300203270	88.16%	-61.04%
Class D															
24 - c203-D1	4	0	2626.89	0	10186.48	33.5	402626900.18648	4	665.36	32	497.57	F	400665360	0.49%	-79.50%
25 - c203-D2	4	0	4282.52	0	10398.13	31.2	404282530.39813	4	665.36	33	586.98	F	400665360	0.89%	-82.91%
26 - c203-D3	5	0	7813.69	0	10561.84	24.8	507813700.56184	4	665.36	33	631.84	F	400665360	21.10%	-78.70%
27 - c203-D4	6	0	9529.03	0	10658.95	24.6	609529040.65895	5	1304.36	34	578.95	F	501304360	17.76%	-73.12%
28 - c203-D5	5	6	5930.53	419.15	10581.46	35.3	565930959.73146	4	665.36	33	620.31	F	400665360	29.20%	-73.65%
29 - c203-D6	14	0	15153.94	0	11401.81	32.4	1415153951.40181	8	2978.6	36	571.81	F	802978600	43.26%	-51.97%
30 - c203-D7	13	0	15373.34	0	11291.12	29.8	1315373351.29112	4	665.36	33	661.12	F	400665360	69.54%	-43.91%
31 - c203-D8	15	0	18786.39	0	11554.8	28.0	1518786401.5548	4	959.31	33	597.04	F	400959310	73.60%	-51.85%
Class E															
32 - c101-E1	6	0	4061.46	0	10487.53	72.1	604061470.48753	2	215.06	7	698.15	F	200215060	66.86%	-86.50%
33 - c101-E2	8	0	5209.53	0	10647.11	69.4	805209540.64711	2	215.06	7	683.36	F	200215060	75.14%	-83.57%
34 - c101-E3	8	0	5377.42	0	10660.57	79.1	805377430.66057	2	215.06	7	661.19	F	200215060	75.14%	-84.23%
35 - c101-E4	17	0	10868.73	0	11536.14	109.2	1710868741.53614	10	867.18	17	796.14	F	1000867180	41.50%	-69.35%
36 - c101-E5	14	3	7735.89	55.53	11262.03	100.7	1437735956.79203	2	215.06	7	990.08	F	200215060	86.07%	-71.86%
37 - c101-E6	23	0	14551.55	0	12166.47	121.6	2314551562.16647	8	834.31	17	995.91	F	800834310	65.40%	-62.51%

The stopping criterion in Heuristic Approach 1 was to reach a predefined number of Tabu Search iterations without an improvement. Therefore, we did not use a predefined cutoff time. The runtime of the 37 instances varied from 1.7 to 422.9 seconds, with an average of 90.7 seconds. In 25 cases the runtime was below 100 seconds, in 11 other instances between 100 and 308 seconds, and in one instance 423 seconds. In all 37 instances the final solution reported was feasible.

The percentage gap between the heuristic's objective value G^{HA_1} and the heuristic's lower bound $LB_G^{HA_1}$ varies between 0 and 100%, with an average of 58.4%. The percentage gap between the objective value G^{HA_1} reported in Heuristic Approach 1 and the corresponding objective value G^{EA_1} reported in Exact Approach 1, varies between -100% and $+20\%$, with an average of -60.8%. These percentages show that the solution reported by heuristic approach 1 is better than the one found by exact approach 1 in 29 out of 37 cases, is the same and equal to zero in 5 other cases, and is worse only in the remaining 3 cases.

8.5.4 Parameter values used in Heuristic Approach 1

After some experimentation, we decided to use the following values for the parameters, throughout our experiments of Heuristic Approach 1:

P_0	P_0^*	θ_1	θ_2	β_1	β_2	β_3	β_4	β_5	J_1	J_2
10^{12}	10^{12}	3	1.7	10^8	10^7	10^3	1	10^{-3}	48	19

The Tabu Length (TL) that we used was defined as a function of n, as shown below:

$$TL = \begin{cases} \lfloor \frac{n}{3} \rfloor & \text{if } n \leq 11 \\ \lfloor \frac{n}{4} \rfloor & \text{if } 12 \leq n \leq 19 \\ \lfloor \frac{n}{5} \rfloor & \text{if } 20 \leq n \leq 39 \\ 7 & \text{if } 40 \leq n \leq 69 \\ \lfloor \frac{n}{10} \rfloor & \text{if } 70 \leq n \leq 119 \\ \lfloor \frac{n}{12} \rfloor & \text{if } n \geq 120 \end{cases}$$

Note that the following parameters are given as an input separately for each instance: $n, d, X_i, Y_i, a_i, b_i, s_i, D_i, C_k$ and T_k.

8.5.5 Discussion about the Experimental Results of Heuristic Approach 2

Table 8.7 presents the experimental results of Heuristic Approach 2 concerning the SCD-VRPTW-Lex3, which was implemented in MATLAB. The

Serial no. & Instance name	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$	**run-time (sec)**	$\mathbf{G^{HA_2} := \sum_{i=1}^{3} \beta'_i f_i}$	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**O/ F/ I**	$\mathbf{LB_G^{HA_2} := \beta'_1 \cdot LB_{f_1} + + \beta'_2 \cdot LB_{f_2}}$	**% of $\mathbf{G^{HA_2}}$ from $\mathbf{LB_G^{HA_2}}$**
Class A										
1 - rc201-A1	0	0	2565.82	1.9	2.56582	0	0	F	0	100%
2 - rc201-A2	0	0	2675.6	1.9	2.6756	0	0	F	0	100%
3 - rc201-A3	6	307.2	2896.97	1.8	600307202.89697	5	172.09	F	500172090	16.68%
4 - rc201-A4	8	601.07	3032.77	1.7	800601073.03277	7	396.17	F	700396170	12.52%
5 - rc201-A5	8	651.27	3037.61	1.7	800651273.03761	7	440.54	F	700440540	12.52%
6 - rc201-A6	8	651.27	3037.61	1.7	800651273.03761	7	440.54	F	700440540	12.52%
7 - rc201-A7	8	651.27	3037.61	1.9	800651273.03761	7	440.54	F	700440540	12.52%
8 - rc201-A8	8	651.27	3037.61	1.8	800651273.03761	7	440.54	F	700440540	12.52%
Class B										
9 - rc201-B1	0	0	3340.44	193.8	3.34044	0	0	F	0	100%
10 - rc201-B2	6	652.96	3290.46	175.9	600652963.29046	0	0	F	0	100%
11 - rc201-B3	4	156.72	3274.52	261	400156723.27452	0	0	F	0	100%
12 - rc201-B4	4	1025	3248.52	311.7	401025003.24852	0	0	F	0	100%
13 - rc201-B5	19	2567.05	3352.21	178.9	1902567053.35221	0	0	F	0	100%
13b - rc201-B5	17	2821.49	3380.23	328.9	1702821493.38023	0	0	F	0	100%
14 - rc201-B6	25	4180.33	3386.93	144.7	2504180333.38693	14	855.33	F	1400855330	44.06%
15 - rc201-B7	13	2595.75	3314.19	252.1	1302595753.31419	0	0	F	0	100%
16 - rc201-B8	6	754.9	3243.27	256.8	600754903.24327	0	0	F	0	100%
Class C										
17 - rc202-C1	4	382.44	2685.58	56.5	400382442.68558	1	54	F	100054000	75.01%
18 - rc202-C2	6	771.29	2722.2	62.1	600771292.7222	1	54	F	100054000	83.35%
19 - rc202-C3	7	1017.41	2751.51	67.9	701017412.75151	1	100	F	100100000	85.72%
20 - rc202-C4	13	1393.46	2806.78	82.3	1301393462.80678	3	177.95	F	300177950	76.93%
21 - rc202-C5	25	3403.01	2947.89	68.2	2503403012.94789	8	387.28	F	800387280	68.03%
22 - rc202-C6	26	3787.1	2912.69	72.7	2603787102.91269	9	555.06	F	900555060	65.41%
23 - rc202-C7	27	5038.14	2922.35	101.2	2705038142.92235	3	203.27	F	300203270	88.90%
Class D										
24 - c203-D1	4	2626.89	10186.48	31.9	402626900.18648	4	665.36	F	400665360	0.49%
25 - c203-D2	4	4282.22	10397.65	25.3	404282230.39765	4	665.36	F	400665360	0.89%
26 - c203-D3	5	7813.69	10561.84	23.4	507813700.56184	4	665.36	F	400665360	21.10%
27 - c203-D4	6	9529.03	10658.95	23.5	609529040.65895	5	1304.36	F	501304360	17.76%
28 - c203-D5	6	4619.62	10543.8	26.2	604619630.5438	4	665.36	F	400665360	33.73%
29 - c203-D6	14	15156.3	11398.3	30.2	1415156311.3983	8	2978.6	F	802978600	43.26%
30 - c203-D7	13	14749.79	11265.64	29	1314749801.26564	4	665.36	F	400665360	69.53%
31 - c203-D8	15	16514.86	11543.81	28.7	1516514871.54381	4	959.31	F	400959310	73.56%
Class E										
32 - c101-E1	6	4061.46	10487.53	70.9	604061470.48753	2	215.06	F	200215060	66.86%
33 - c101-E2	8	5209.53	10647.11	66.8	805209540.64711	2	215.06	F	200215060	75.14%
34 - c101-E3	8	5377.42	10660.57	80.2	805377430.66057	2	215.06	F	200215060	75.14%
35 - c101-E4	17	10863.57	11485.55	133.5	1710863581.48555	10	867.18	F	1000867180	41.50%
36 - c101-E5	17	7517.24	11100.82	97.8	1707517251.10082	2	215.06	F	200215060	88.27%
37 - c101-E6	28	14036.46	12049.55	94.4	2814036472.04955	8	834.31	F	800834310	71.54%

Table 8.7: Experimental Results of Heuristic Approach 2 in MATLAB (for the Lexicographic SCD-VRPTW with 3 objectives)

columns of this table provide the following information for each instance: The serial number and code-name of the instance, the values of the 3 component objectives f_1, f_2 and f_3, the runtime in seconds, the value of the objective function in Heuristic Approach 2, which is denoted by G^{HA_2} and is equal to $\sum_{i=1}^{3} \beta'_i f_i$, provided that the corresponding solution is feasible, the lower bounds LB_{f_1} and LB_{f_2} for the component objectives f_1 and f_2 respectively, whether the solution is Optimal, Feasible or Infeasible (O/F/I), the lower bound $LB_G^{HA_2}$ of G^{HA_2}, defined as $LB_G^{HA_2} := \beta'_1 \cdot LB_{f_1} + \beta'_2 \cdot LB_{f_2}$, and finally the percentage gap between this heuristic's objective value G^{HA_2} and the heuristic's lower bound $LB_G^{HA_2}$, calculated as $\frac{G^{HA_2} - LB_G^{HA_2}}{G^{HA_2}} \cdot 100\%$.

We can recall that the objective function in Heuristic Approach 2 was denoted by G_3 and was defined by equation (8.3.1) of subsection 8.3, which is more complex than $\sum_{i=1}^{3} \beta'_i f_i$. However, in the case of a feasible solution - and specifically in all 37 instances of our dataset - the objective value G_3 of Heuristic Approach 2 simplifies to $\sum_{i=1}^{3} \beta'_i f_i$, which is denoted by G^{HA_2} in this table.

The stopping criterion in Heuristic Approach 2 was to terminate the algorithm once it reaches a predefined number of Tabu Search iterations without an improvement. Therefore, we did not use a predefined fixed cutoff time. The runtime of the 37 instances varied from 1.7 to 311.7 seconds, with an average of 82.8 seconds. In 27 cases the runtime was below 100 seconds, in 6 other instances between 100 and 200 seconds, and in the remaining 4 instances between 200 and 312 seconds. In all 37 instances the final solution reported was feasible. The percentage gap between the heuristic's objective value G^{HA_2} and the heuristic's lower bound $LB_G^{HA_2}$ varies between 0 and 100%, with an average of 60.7%. Note that in table 8.7, apart from the 37 instances, we include an additional entry which is marked as 'instance 13b', which corresponds to a second run of instance 13 with a different set of parameters, where the solution found was better than the respective solution of instance 13 (at the expense of longer runtime).

Comparing the results from Heuristic Approaches 1 and 2, we can see that in 14 out of 37 instances, (namely, in instances 1-8, 24, 26, 27 and 32-34), the two methods provide exactly the same solution. Note that in all of the above instances, the best solution reported by Heuristic Approach 1 involved a zero value for F_2 (which also implies that $F_4 = 0$). Furthermore, in 5 other instances (namely, instances 15, 25, 30, 31 and 35), the solution provided by Heuristic Approach 2 outperforms the solution found by Heuristic Approach 1.

These results may be considered to give a partial justification for the simplification that occurs through the reduction of the 5 objectives of the problem down to 3 objectives, which was discussed in section 7.12. This reduction in the number of objectives is crucial so that Heuristic Approach 3 can be used, which involves using the Epsilon Constraint Method to find an approximation of a representative subset of the set of non-dominated solutions for the SCD-VRPTW with 3 objectives. It would be substantially more complex and more time-consuming to apply a method like Heuristic Approach 3 for the 5-objective variant of the problem.

8.5.6 Parameter values used in Heuristic Approach 2

Throughout our experiments with Heuristic Approach 2, we used the same algorithm used in Heuristic Approach 1, described in section 8.2, and the same parameter values reported in subsection 8.5.4, with the only difference that in Heuristic Approach 2 we used the following set of weights:

β_1	β_2	β_3	β_4	β_5
10^8	10^8	10^3	10^3	10^{-3}

Note that this is equivalent to using the algorithm described in section 8.3 with the following set of weights:

β'_1	β'_2	β'_3
10^8	10^3	10^{-3}

8.5.7 Parameter values used in Heuristic Approach 3

After some experimentation, we decided to use the following parameter values in our experiments with Heuristic Approach 3:

P_0	P_0^*	P_1	P_1^*	P_3	P_3^*	θ_1	θ_2	Φ	Θ
10^{12}	10^{12}	10^{20}	10^{20}	10^{20}	10^{20}	3	1.7	15	5

	β'_1	β'_2	β'_3
Set of weights used in Step 4a:	10^8	10^3	10^{-3}
First set of weights used in Step 4b:	1	0	0
Second set of weights used in Step 4b:	0	1	0
Third set of weights used in Step 4b:	0	0	1
Set of weights used in Step 6:	10^{-3}	1	10^{-8}

	J_1	J_2
In Step 4a:	48	19
In Step 4b:	28	9
In Step 6:	8	2

The tabu length (TL) was defined as a function of n, as follows:

$$TL = \begin{cases} \lfloor \frac{n}{3} \rfloor & \text{if } n \leq 11 \\ \lfloor \frac{n}{4} \rfloor & \text{if } 12 \leq n \leq 19 \\ \lfloor \frac{n}{5} \rfloor & \text{if } 20 \leq n \leq 39 \\ 7 & \text{if } 40 \leq n \leq 69 \\ \lfloor \frac{n}{10} \rfloor & \text{if } 70 \leq n \leq 119 \\ \lfloor \frac{n}{12} \rfloor & \text{if } n \geq 120 \end{cases}$$

Note that the parameter values used in Step 4, where the algorithm calls the algorithm of Heuristic Approach 2, are the same as the ones described in subsection 8.5.6. Note also that the following parameters are given as an input separately for each instance: $n, d, X_i, Y_i, a_i, b_i, s_i, D_i, C_k$ and T_k.

8.5.8 Experimental Results of Heuristic Approach 3

In this subsection we discuss the experimental results of Heuristic Approach 3 (HA3) for solving the SCD-VRPTW-3 (or SCD-VRPTW-MOO-3); i.e. the variant of Problem 3 with 3 objectives, without assuming any preference information about the objectives. This approach involves the Tabu Search metaheuristic in an ECM framework, and was described in detail in section 8.4. For each one of the 37 instances that were constructed for Problem 3, we present a separate table that contains all non-dominated solutions found by implementing Heuristic Approach 3 in MATLAB. These tables of results can be found in Appendix B.

Each table is split into two parts. The first part is composed of the first two rows and contains the runtime in seconds, the lower bounds LB_{f_1} and LB_{f_2} for the component objectives f_1 and f_2 respectively, and a reference value for f_3. This reference value shows the sum of arrival times at the endpoint node in the original undisrupted VRPTW, and is the same for all instances of the same class. The second part of each table spans from the third row until the last row of the table, and contains all non-dominated solutions found for the instance. Each non-dominated solution corresponds to one row and records the solution's serial number and the values of the three component objectives f_1, f_2 and f_3. Note that each entry of the component objectives is followed by a percentage. Each percentage that follows a value of f_1, is calculated as $\frac{f_1 - f_1^{best}}{f_1^{worst} - f_1^{best}} \cdot 100\%$, or equivalently as $\frac{f_1 - f_1^{min}}{f_1^{max} - f_1^{min}} \cdot 100\%$, where f_1^{best} corresponds to the best result (i.e. the minimum value f_1^{min}) for f_1, and f_1^{worst} to the worst result (i.e. the maximum value f_1^{max}) for f_1, among all non-dominated solutions found. The percentages that follow the values of f_2 and f_3 are calculated in a similar way.

Regarding the trade-offs between the three objectives f_1, f_2 and f_3, we can make the following observations from the tables:

- On the one hand, in instances 1, 2, 4-6, 8, 9, 11 and 12, a solution with the minimum value of f_1 gives the minimum value of f_2. On the other hand, in instances 3, 6, 25-28 and 37, a solution with the minimum value of f_1 gives the maximum value of f_2.

- In instances 1, 5, 8, 13-17, 21 and 24, a solution with the maximum value of f_1 gives the maximum value of f_2. However, in instance 3, a solution with the maximum value of f_1 gives the minimum value of f_2.

- In instances 1-6, 8-12, 18, 22, 23, 25-28 and 33-36, a solution with the minimum value of f_1 gives the maximum value of f_3. However, in instances 3 and 6, a solution with the minimum value of f_1 gives the minimum value of f_3.

- In instances 1, 4, 5, 8, 11, 13-17, 21, 24 and 33, a solution with the maximum value of f_1 gives the minimum value of f_3. No instance was found where a solution with the maximum value of f_1 gives the maximum value of f_3.

- In instances 1, 2, 4-6, 8, 9, 11, 12, 16, 17, 20, 31 and 32, a solution with the minimum value of f_2 gives the maximum value of f_3. No instance was found where a solution with the minimum value of f_2 gives the minimum value of f_3.

- In instances 1-3, 5, 6, 8 and 12-24, a solution with the maximum value of f_2 gives the minimum value of f_3. However, in instances 25-28, a solution with the maximum value of f_2 gives the maximum value of f_3.

There are no obvious conclusions that can be drawn from these observations, regarding the trade-offs between the objectives, since some of the results are contradictory. Further investigation is needed, which is left open for future research.

Caution may be needed when interpreting the above results for instances 3-6 and 8, where f_1 only takes 2 different values in the set of non-dominated solutions found, and therefore can be considered as degenerate cases. Moreover, instance 7 was not included in the above discussion about the trade-offs, because only 2 non-dominated solutions were reported for this instance, both with the same value for f_1; therefore this can also be considered a degenerate case.

HA2 and HA3 both solve 3-objective variants of the problem; although HA2 further assumes a lexicographic preference of the component objectives. On the contrary, HA3 doesn't make use of any preference information regarding the objectives, but attempts to generate an approximate representation of the Pareto front. These are our findings from the comparison of the runtimes of HA2 and HA3: For the 37 instances of the SCD-VRPTW, the ratio $\frac{runtime\ of\ HA_3}{runtime\ of\ HA_2}$ ranges from 5.5 up to 31.4, with an average of 14.3 and a standard deviation of 6.3. Therefore, the runtime of HA3 is, on average, 14 times as large as the runtime of HA2. More specifically, in 12 instances the above ratio ranges between 5.5 and 10, in 21 other instances this figure lies between 10 and 20, whereas in the remaining 4 instances this ranges between 25 and 32.

In general, the runtime of each one of the three heuristics described in this chapter, is greatly affected by the choice of certain parameters that control the number of iterations performed; for example, parameters J_1 and J_2 for HA1 and HA2. In HA3, in addition to J_1 and J_2, we have multiple cycles of up to $\Phi \times \Theta$ outer iterations, although under certain conditions the algorithm may skip some of these cycles, if found unnecessary. For example, by reducing both values of Φ and Θ by 50%, we can expect a reduction of up to 75% in the runtime (since we would then have to perform around 75% fewer cycles of outer iterations[7]). Nevertheless, choosing different combinations of values for the parameters J_1, J_2, Φ and Θ, can have a huge impact on the runtime. Further experimentation may be needed to examine the effect of changing each one of these parameters on the runtime, which is left for future researchers.

In most instances (and specifically in 31 out of 37 instances), the solutions found by HA3 include the solution found by HA2 for the same instance. However, there are 6 instances out of the 37 tested where HA3 finds a solution which dominates the solution found by HA2 (these are instances 9, 13b, 21, 24, 29 and 30). This is due to a different heuristic and stopping criterion being used.

The results also show how the solutions relate to lower bounds or reference values for the three different objectives.

[7]The figure of 75% refers to the case where the algorithm does not skip any cycles of outer iterations.

8.6 Conclusions

To sum up, in this chapter we described three different heuristic methods for different variants of the *Single-Commodity Delayed VRPTW* (problem 3). A set of instances was created and the three heuristic approaches (HA1, HA2 and HA3), as well as exact approach 1 (EA1) that was described in the previous chapter, were tested on these instances.

EA1 and HA1 both solve the same variant, the *SCD-VRPTW-Lex5*, which is the 5-objective variant of the problem assuming a lexicographic preference of the component objectives. It was shown that EA1 requires too much computation time for instances involving higher numbers of customers. HA1 generally outperforms EA1, with the exception of a few small problems where the reverse is true. However, even then, HA1 gives good results compared to the exact method.

HA2 and HA3 both solve 3-objective variants of the problem; however, HA2 assumes a lexicographic preference of the component objectives, whereas HA3 makes no such an assumption. Instead, HA3 aims to find a representative subset of the set of non-dominated solutions for the SCD-VRPTW with 3 objectives, rather than a single solution and so takes longer to run than Heuristic Approach 2 which only aims to find a single non-dominated solution for a particular set of weights. Comparison of runtimes for the same instances shows that HA3 takes about 14 times as long as HA2 (for the specific choice of parameters used). In some circumstances, the runtime for HA3 may be too long in practice, though it could be stopped before a full set of non-dominated solutions has been found, if one of those found is acceptable.

Chapter 9

CONCLUSIONS

This chapter summarizes the main findings and contributions of this thesis, and discusses possible directions for future research.

9.1 Research Summary & Scope for Further Research

In this thesis we studied applications of disruption management in the area of vehicle routing and scheduling for road freight transport. We considered two types of disruption, namely vehicle breakdown and vehicle delay. We identified three interesting problems resulting from such disruptions, and we devoted two chapters to the study of each problem: one chapter for mathematical programming formulations and exact approaches, and one chapter for heuristic methods and experimental results. Exact approaches were implemented in AIMMS, whereas heuristic approaches were implemented in MATLAB.

A dataset of instances was constructed for each problem, in order to test the proposed solution methods. The mathematical programming formulations and exact approaches proposed were successful in solving small instances to optimality. However, because of the complexity of the three problems under study, which are variants of the TSP and the VRP, there is a limit in the size of problems that the exact approaches can solve within a few minutes. Beyond a certain size of problems, either more sophisticated exact approaches may be used, or heuristic methods, or combinations, e.g. math-heuristics. Therefore, for each problem we also proposed at least one heuristic algorithm, which can solve larger instances in real-time. Of course, even in the case where a heuristic approach provides a superior solution within a certain time limit, compared to an exact approach, the latter may still be important by providing a useful lower bound, and thus a way to calculate a guaranteed bound on the optimality gap and assess the quality of the heuristic.

9.1.1 Problem 1 - Summary

In chapter 3 we described *the disrupted Vehicle Routing Problem with customer-specific orders and Vehicle Breakdown* (problem 1) and presented an exact approach (EA) and a MILP formulation. For small instances, or for instances where only a few customers were originally assigned to the disabled vehicle, the exact approach succeeded in finding the proven optimal solution within a few minutes. Specifically, in 40 out of 101 instances considered (i.e. in 39.6% of the cases), the exact approach succeeded in finding the proven optimal solution within 5 minutes, whereas in 35 other instances (i.e. in 34.7% of the cases) this method reported a solution with an optimality gap below 10%. In 99 out of 101 instances (i.e. in 98% of the cases) a feasible solution was found within 5 minutes, and in all cases a lower bound was reported. However, there is a size limit, beyond which the exact approach fails to provide an acceptable solution within a few minutes. In such cases a heuristic is more appropriate. The experimental results suggest that the runtime of the EA highly depends on the number of customers assigned to the broken-down vehicle. Also, we verified that the optimal solution may sometimes involve multiple visits to the disabled vehicle, either by a single active vehicle or by multiple ones.

In chapter 4 we described a heuristic algorithm based on Tabu Search, as a second approach for solving problem 1. This method was also tested on the 101 instances that were created for this problem. Although we did not apply a cutoff time when performing experiments with this heuristic approach (HA), in 97 out of 101 instances the algorithm terminated within 5 minutes, and in the remaining 4 instances within 8 minutes; thus we can think of the two methods (EA and HA) as being tested using similar or comparable cutoff times. The solution found by the HA was feasible in all 101 cases, and proven optimal in 35 instances (i.e. in 34.7% of the cases). Considering the remaining 66 instances, the percentage gap between the solution found by the HA and the lower bound found by AIMMS, was below 5% in 37 cases, and below 22% in all 66 cases.

Comparing the results found by the two approaches, out of 101 instances, the heuristic solution was better in 35 instances, exactly the same in 53 instances, and worse than the exact approach in the remaining 13 instances. For the latter category of those 13 cases, the percentage gap of the heuristic solution was within 8.61% of the exact solution, with an average percentage gap of 2.02%. The results verify that the heuristic is capable of finding optimal or near-optimal solutions within a few minutes, and that this approach may be more promising for larger problems. In general, both methods are important and useful. In practice, we recommend having both approaches running simultaneously, and take the best solution found by either one of the two methods within the available time. For larger instances, the heuristic will be the only real choice.

In section 3.6, we described an extension of problem 1, *the disrupted VRPTW with Vehicle Breakdown (d-VRPTW-VB)*, under the assumptions of a heterogeneous fleet and customer-specific orders. This problem refers to the direct generalization of problem 1, where additionally each customer was associated with a time window in the original plan. We described three possible scenarios that may occur, and showed how the first two cases can be treated using the methods and formulations that were proposed for solving problem 1. Further investigation, precise definition and proper modeling of the third and more general scenario, as well as performing experiments with all three scenarios, are tasks which are left to be carried out by future researchers. New models and formulations may be necessary for the third case, for which a possible direction may be to combine the models and algorithms that were described for problems 1 and 3.

9.1.2 Problem 2 - Summary

In chapter 5 we defined problem 2, *the Delayed Traveling Salesman Problem with Time Windows (Delayed TSPTW)*, and presented formulations and exact approaches. In its more general version, the *Delayed-TSPTW-5* (or *Delayed-TSPTW-MOO-5*) was defined as a multi-objective optimization problem with 5 component objectives, for which a general MOMILP formulation was presented.

We then described exact approach 1 (EA1) for solving the *Delayed-TSPTW-Lex5*, which refers to the variant of problem 2 which additionally assumes a predefined lexicographic preference of the 5 component objectives. This method aggregates the 5 component objectives into a single objective, transforming the more general MOMILP into a single-objective MILP. This MILP was implemented in AIMMS and tested on a set of 27 instances that were created for the purpose (the respective experimental results were presented in chapter 6). Allowing a maximum runtime of 10 minutes, AIMMS found a feasible solution in all instances, with an average optimality gap of 24.24% for the whole dataset. In 6 out of 27 instances, the proven optimal solution was reported. Results showed that within 10 minutes, EA1 was unable to produce a solution with an optimality gap of less than 10% in any of the instances which involved more than 40 active customers. These results justify the need of a heuristic algorithm capable of reaching high-quality solutions within reasonable time for larger instances.

Additionally, in chapter 5 we described exact approach 2 (EA2) for addressing the Delayed-TSPTW-MOO-5 using the weighted-sum method, which can be viewed as a generalization of EA1, where the weights of the 5 component objectives are systematically changed, and a MILP is solved for each different choice of weights, for the purpose of getting a representative subset of the set of non-dominated solutions. However, because of the poor performance of EA1 in solving instances with more than 40 customers, we did not perform any experiments with EA2, which should be computationally even harder than EA1. Instead, performing experiments with EA2, as well as modeling and experimenting with other multi-objective optimization methods that can be based on the formulations described in this chapter to find the set of Pareto optimal solutions in an exact approach, are tasks left open for further research.

In section 5.13 we introduced *the disrupted VRPTW with customer-specific orders and vehicle delay*, which is another generalization of problem 2 where, during the execution of the operational plan of a multi-commodity VRPTW with a heterogeneous fleet, we have multiple vehicles experiencing delays at various times. Because of the assumption of customer-specific orders, solving one instance of the above problem is shown to be equivalent to solving multiple independent instances of the *Delayed TSPTW* - one for each affected route. Therefore, the study of this more general problem can also be considered as completed.

In chapter 5 we also presented an extension of problem 2 that involves customer priorities, as well as the 3-objective variant of the problem (*the Delayed-TSPTW-3*). Moreover, we discussed other applications of problem 2 to handle other types of disruption, other variants, additional constraints that can be included etc. Further investigation of other solution methods, modeling and performing experiments with other problem variants that were described or discussed in this chapter for which we did not provide complete models or performed experiments, are left open for further research.

In chapter 6, we described three Tabu Search heuristics for solving variants of the Delayed TSPTW. Heuristic approaches 1 and 2 solve the problem variants with 5 and 3 objectives, respectively, assuming a predefined lexicographic ordering of the component objectives. We created a dataset of 27 instances for the Delayed TSPTW, based on benchmark instances taken from Dumas et al. (1995) dataset, and used these to test our approaches. The experimental results of exact approach 1 (EA1), heuristic approach 1 (HA1) and comparisons are included in chapter 6.

These are the main findings derived from the comparison of HA1 and EA1 for solving the lexicographic variant of the Delayed TSPTW with 5 objectives *(Delayed-TSPTW-Lex5)*: Out of the 27 instances, HA1 outperformed EA1 in 20 cases (i.e. in 74.07% of the cases), with an average percentage gap of 21.45% for these 20 cases. In 4 other instances (i.e. in 14.81% of the cases) the solutions provided by HA1 and EA1 were both optimal and hence equivalent. For the remaining 3 instances (i.e. in 11.11% of the instances), HA1 was outperformed by EA1, but in each one of these 3 cases the solution found by HA1 was within 2% of the proven optimal, with an average optimality gap of 0.87% for the 3 cases. In short, HA1 performs relatively well, generally outperforming EA1 (although there were some exceptions when solving some very small instances, where EA1 was able to find the proven optimal in real time, whereas HA1 found a slightly worse solution). Also, in 70.37% of the cases the lower bound found by HA1 was better than the one provided by AIMMS in EA1. To sum up, both EA1 and HA1 are important methods and therefore in practice we recommend running both approaches in parallel, and use the best solution and the best lower bounds provided by either one of the two methods in the available time. Of course, for larger instances EA1 is expected to be impractical, therefore HA1 will be the only real choice among these two options.

We also used HA1 to solve 9 standard TSPTW instances, both to test the performance of HA1 and to verify that the Delayed TSPTW generalizes the standard TSPTW.[1]

Moreover, in chapter 6 we described Heuristic Approach 3 (HA3), which uses the framework of the Epsilon Constraint Method to solve heuristically the MOO variant of the Delayed TSPTW with 3 objectives, and finds a representative subset of the set of non-dominated solutions. The respective experimental results can be found in Appendix A. HA3 gives an indication of the Pareto front and potential trade-offs between the 3 component objectives f_1, f_2 and f_3, although it may be too slow to apply in practice.

9.1.3 Problem 3 - Summary

In chapter 7 we described *problem 3*, i.e. *the Single-Commodity Delayed Vehicle Routing Problem with Time Windows (SCD-VRPTW)*[2], and presented formulations and exact approaches. In its general version, the *SCD-VRPTW-MOO-5* (or simply the *SCD-VRPTW-5*) was defined as a multi-

[1]To be more precise, both the Delayed-TSPTW-5 and the Delayed-TSPTW-Lex5 generalize the TSPTW variant where the objective is to minimize the time of arrival at the endpoint node.

[2]or *the disrupted VRPTW with non-customer-specific orders and vehicle delay*

objective optimization problem with 5 component objectives, for which a general MOMILP formulation was presented. It can be regarded as a generalization of the *Delayed-TSPTW-5* with multiple vehicles of different capacities, delivering a single commodity. Exact approaches 1 and 2 were presented, equivalent to those described for problem 2, as well as an extension of problem 3 involving customer priorities. We also presented some additional assumptions to reduce the number of objectives from 5 down to 3 (as done in problem 2), differentiating between the following two variants of problem 3: *the SCD-VRPTW with 5 objectives* (SCD-VRPTW-5) and *the SCD-VRPTW with 3 objectives* (SCD-VRPTW-3). Other applications of problem 3 to handle other types of disruption, as well as other variants or extensions involving different sets of constraints and objectives were also discussed. Further modeling and experimentation with other solution methods or with solving other variants suggested in this chapter, are left open for further research.

In more detail, exact approach 1 (EA1) addresses the variant of problem 3 with 5 objectives of lexicographic preference *(SCD-VRPTW-Lex5).* This approach aggregates the 5 component objectives into a single objective, defined as the weighted sum of the 5 components, and uses appropriate weights to impose a lexicographic preference of these objectives; thus transforming the more general MOMILP formulation into a single-objective MILP formulation (as in problem 2). We implemented the latter formulation in AIMMS and used it to solve 37 instances that were created specifically for this problem. The results can be found in chapter 8. In short, EA1 failed to provide an optimal, or even an acceptable solution within 5 minutes in all instances that involved more than 54 customers (in fact, in some of those instances we tried using larger cutoff times and still got unacceptable results). EA1 gave optimal or acceptable solutions in instances of Class A that only involved 11 customers, although in some of those instances with 11 customers, AIMMS was not able to prove that the final solutions were optimal within 10 minutes. We did not perform any experiments with instances of sizes above 11 and below 54 customers, and thus we did not precisely define the exact size of problems that can be solved with EA1. This can be investigated by further research, if necessary. However, our results were enough to show that EA1 cannot be used in practice for solving problems of more than 54 customers, since for instances of 54 or more customers the solutions were of very poor quality.

Chapter 7 also describes exact approach 2 for addressing the more general *SCD-VRPTW-MOO-5.* This method is similar to EA1, but instead of involving a single run with a single set of weights, it involves systematically varying the weights of the 5 component objectives and solving one MILP for each set of weights. Therefore, this approach involves solving a large number of MILPs, in order to get a large number of non-dominated solutions for this MOO problem with 5 objectives, which will comprise a representation of the Pareto front. Of course, as Branke et al. (2008) mentions, this method may not always give a good representation of the Pareto front. Instead, for a good representation of the Pareto front, other multi-objective optimization methods may be more appropriate, such as the Epsilon-Constraint Method. In any case, because of the poor performance of EA1, we decided not to perform any experiments with the more general and computationally more expensive EA2, nor with any other exact approaches or other multi-objective optimization methods that are used in an exact framework. This is also left open for further research.

In chapter 8 we described three heuristic approaches (HA1, HA2 and HA3), all based on Tabu Search, for solving variants of problem 3. Specifically, HA1 and HA2 solve the SCD-VRPTW with 5 and 3 objectives, respectively, assuming a lexicographic preference of the component objectives. HA3 uses the framework of the Epsilon Constraint Method to solve heuristically the SCD-VRPTW with 3 objectives, and finds a representative subset of the set of non-dominated solutions. The different methods are tested on a dataset of 37 instances that were created for the SCD-VRPTW. Chapter 8 includes the experimental results of EA1, HA1 and HA2, whereas the experimental results of HA3 can be found in Appendix B.

The stopping criterion of HA1 was to terminate after reaching a predefined number of iterations without an improvement; therefore no cutoff time was applied. However, the maximum runtime of HA1 for solving the 37 instances was around 7 minutes (422.9 seconds), with an average of 90.7 seconds for the whole dataset. HA1 gave a feasible solution in all of the cases. The average percentage gap between HA1 and the corresponding lower bound derived by HA1, was 58.4%.

Comparing the solutions reported by HA1 and EA1 for solving the SCD-VRPTW-Lex5, it was found that HA1 gave a better solution in 29 out of 37 cases, the same solution in 5 cases, and a worse solution in the remaining 3 cases, with a percentage deviation of HA1 from EA1 averaging at -60.8%. Specifically, for the 3 instances where EA1 gave a better solution than HA1, the three percentage gaps were 0.2%, 0.04% and 19.96%, with an average percentage deviation of 6.73%. For these three instances (which are the instances with serial number 1, 2 and 3 of Class A), the runtime of EA1 was between 400 and 900 seconds, whereas the runtime of HA1 was no more than 2 seconds. Comparing the lower bounds found by HA1 and EA1, we observed that the lower bound found by HA1 was better than the one found by EA1 in 28 out of 37 cases, the same in 7 other cases and worse in just

2 cases (which correspond to very small instances where AIMMS provided the proven optimal solution, and thus the final lower bound was matching the optimal solution). These results suggest that HA1 generally outperforms EA1, and in many cases it provides a better lower bound than the one found by AIMMS in EA1. However, as in problem 2, both EA1 and HA1 are important, and therefore in practice we recommend running both approaches in parallel and use the best lower bounds and the best solution reported in the available time, by either one of the two methods. Of course, for large instances, HA1 will almost certainly be the only practical choice among these two approaches.

HA2 solves *SCD-VRPTW-Lex3*, i.e. the Lexicographic variant with 3 objectives, and can be considered to be a special case of HA1. Its results are presented mainly to be used as reference values for HA3.

The last method, HA3, solves the *SCD-VRPTW-3*, i.e. the variant with 3 objectives, but without assuming a lexicographic preference of the 3 objectives. Therefore, HA3 is a more general multi-objective optimization approach that employs several nested loops, performing multiple Tabu Search iterations with multiple restarts, within the framework of the Epsilon Constraint Method. Its purpose is to provide a representative subset of the set of non-dominated solutions for the 3-objective variant. The results of HA3, which can be found in Appendix B, give an indication of the Pareto front and potential trade-offs between the 3 component objectives f_1, f_2 and f_3. Of course, HA3 is significantly slower than HA2 (and HA1).

9.1.4 Additional scope for further research

Below we list some of the potential problems or areas that can be further investigated by future researchers, in addition to those discussed in the previous subsections of this chapter:

- For the problems described in this thesis, there were several solution approaches, problem variations and extensions that were discussed or described but not formulated, or that were formulated and thoroughly described but not tested experimentally. Building appropriate models and formulations for the cases that were discussed but not modeled, as well as performing the relevant experiments, are tasks left open for further research. Specifically, these include the following variations and solution approaches for problems 2 and 3: (i) implementing exact approach 2 for the solution of the 5-objective variants using the weighted-sum approach and performing experiments, (ii) implementing other MOO methods, such as the ECM, in an exact approach, for the 3-objective and/or for the 5-objective variants, (iii) performing experiments with the extensions of problems 2 and 3 that involve customer priorities, (iv) performing experiments with the 3-objective variants of problems 2 and 3 using exact approaches, (v) performing experiments with HA2 for the 3-objective variant of problem 2 assuming a lexicographic ordering of the component objectives, (vi) applications of problems 2 and 3 to handle other types of disruption, other problem variants, additional constraints etc.

- Regarding heuristic approaches 3 which employ the Epsilon-Constraint Method to solve the MOO variants of problems 2 and 3 that involve 3 component objectives, further examination of possible correlations between the three component objectives f_1, f_2 and f_3 may be performed, by using appropriate statistical methods (e.g. hypothesis tests).

- When dealing with disruptions, it is crucial that the response time is very short; ideally this should happen in real-time. We recognize that some of the computational times reported in this thesis are prohibitively high. This is especially true for the heuristics that were developed for problems 2 and 3 and were based on the Epsilon Constraint Method.

  Further experimentation with the heuristics proposed after making minor changes may lead to improvements in the performance of the algorithms, in terms of both runtime and solution quality. Parameter tuning and parameter optimization, may lead to such an improvement, especially in terms of the runtime. In particular, fine-tuning the parameters that control the number of different restarts of each algorithm, as well as the number of iterations without an improvement before the algorithm terminates, can have a significant impact on the runtime. Of course, we should keep in mind that there is a trade-off between solution quality and computational time.

  Parallelization is another option which can be trivially achieved in some cases (e.g. in cases of multiple and independent restarts of the algorithm) and can also have an impact on the runtime. Other stopping criteria may be investigated. Another direction is a simultaneous use of exact and heuristic approaches, where the two approaches interact with each other and the search is guided by continuously comparing the best solutions with the best lower bounds already found, in a branch-and-bound framework.

- The formulations and exact approaches described for each one of the three problems may be used in combination with more sophisticated exact frameworks, such as in conjunction with branch-and-bound, branch-and-cut, branch-and-cut-and-price, etc.

- Other algorithms can be created, which may be based on other metaheuristics (such as Simulated Annealing, Variable Neighborhood Search etc.), or math-heuristics that combine metaheuristics and mathematical programming techniques, possibly combining some of the features proposed in this thesis, in order to create better or faster algorithms for the three problems and improve on the results found for the instances of our datasets.

- For the solution of the different problem variations, other multi-objective optimization methods can also be used, apart from the lexicographic approach, the weighting method and the epsilon-constraint method.

- The use of multi-objective optimization performance metrics, such as the hyper-volume indicator, may be used to measure the quality of approximation of the Pareto front, for heuristic approach 3 of problems 2 and 3.

- Regarding the travel times, in our models we are assuming that these are constant over time. This assumption was adopted for the sake of simplicity and convenience. We recognize that in reality, travel times are usually dynamic, especially in urban areas. Extending our models to include dynamic travel times is something that may be further investigated.

- There is a large number of other problem variants and extensions that can arise from each one of the three problems studied in this thesis, by considering alternative combinations of assumptions, constraints and objectives.

  For instance, regarding the assumptions, the following options may be particularly important: (i) whether or not there are extra vehicles and drivers available to use or hire in case of a disruption - and how many, (ii) single-commodity or multi-commodity, whether or not orders are customer-specific etc., (iii) whether it is possible to leave some customers unserved, (iv) whether it is possible and practical for vehicles to meet and exchange goods: special equipment may be needed to load certain items, (v) when vehicles can be re-routed to respond to a disruption.

  When modeling a disruption in vehicle routing and scheduling, the objective will generally have two parts, which can be further subdivided into more components. The first part will normally try and optimize the objective of the original problem, while the second part will try and minimize the negative impact of the disruption. The assumptions, objectives and constraints of the original problem will greatly affect and determine to some extent the respective counterparts of the problem after disruption. The original objective may have been to minimize total travel time, total travel cost, carbon emissions, total monetary cost, total distance traveled, sum of arrival times at the endpoint node etc. This will usually compose the first part of the objective in the resulting problem after disruption. The second part of the objective that seeks to minimize the negative impact of the disruption may be more complex, including functions that are associated with minimizing the impact of the disruption on customers, minimizing the deviation from the original plan for drivers, and minimizing the associated costs of deviation from the original plan for the enterprise. There are several ways to capture these deviations, and there is flexibility to leave out some of these, if they are not particularly useful.[3] Therefore, there is a large number of possibilities regarding to the number and choice of component objectives. For example, when modeling problems 2 and 3 in their more general versions, we used 5 component objectives, where 4 out of 5 components were seeking to minimize the negative impact of the disruption, whereas the last one was related to the objective of the original plan. We will normally have multiple conflicting objectives and therefore the problem should be tackled using multi-objective optimization techniques. Even for a single choice of objectives, additional variations may occur by considering the type of preference information available (such as the classification or ordering of component objectives). Of course, it may be possible to aggregate them all in a single objective, such as the total monetary cost to be minimized, and use classical single-objective optimization techniques.

In addition to the large number of combinations of objectives, there is also a large number of possible combinations of constraints that may be considered. For example, as was discussed in sections 5.15 and 7.14, interesting variations of problems 2 and 3 can be derived by including additional constraints in the current models, such as a maximum

[3]For example, in the case of the disrupted VRP with vehicle breakdown (problem 1), we decided not to include a component for drivers in the objective, because this was not particularly relevant. Alternative approaches may include an additional objective that seeks to find solutions where the routes are as close to the original routes as possible, since this may be slightly more convenient for drivers who may be more familiar with certain routes.

allowed deviation outside the promised time windows, or the use of customer priorities. Non-linear constraints or objectives may also be relevant to some applications, such as using a quadratic function to penalize the objective when violating the time windows, instead of a linear function (for problems 2 and 3).

  All these possibilities can be further investigated by future researchers. However, it seems that there is an endless number of variations that may be created. Therefore, perhaps an application-specific approach may be more appropriate; i.e. creating custom-based models for real case-studies.

- Other types of disruption may also be studied, including combinations of disruptions. All of the above can be guided by the pursuit of a more ambitious goal of eventually creating a flexible, real-time decision support system that can aid practitioners in addressing all usual types of disruptions and combinations.

9.2 Contributions

These are the main contributions of this thesis:

1. The introduction of two new problems, namely problems 2 and 3, that have never been studied before under the specific assumptions and from a similar point of view. We believe that these problems model actual situations that occur frequently in road freight transport, but we are not aware of similar studies that proposed efficient methods to deal with vehicle delays in real time.

2. We proposed mathematical programming formulations (MILPs and/or MOMILPs) for each one of the three problems.

3. We constructed at least one heuristic for each problem.

4. We created a dataset of test instances for each problem.

5. We have performed experiments on these problem instances using at least one exact and one heuristic method for each problem.

6. We have drawn some important conclusions from these experiments. Specifically, we give some idea about the size of instances that may be solved to optimality using the exact approaches. For smaller instances, this provides a way to measure the quality of the heuristics.

For larger instances, we used lower bounds provided either by the exact approach (from the LP relaxation of the MILP formulations), or problem-specific lower bounds that were found and included in the heuristic approach. These lower bounds allow us to get proven optimality gaps and to assess the quality of the heuristics. Furthermore, by examining the experimental results we can draw some conclusions about how the computational times depend on the instance size.

7. We provided two simple - yet powerful - techniques to speed-up the Epsilon-Constraint Method.

8. Some helpful lower bounds were suggested for problems 2 and 3.

9. We suggested several variants and extensions of the problems, involving different sets of objectives, constraints and assumptions. Some of the models and solution methods proposed can also be used to cope with other types of disruptions.

Bibliography

Altinkemer, K., & Gavish, B. (1991). Parallel savings based heuristics for the delivery problem. *Operations Research*, *39*(3), 456–469.

Alves, M. J., & Climaco, J. (2007). A review of interactive methods for multiobjective integer and mixed-integer programming. *European Journal of Operational Research*, *180*(1), 99-115.

Ascheuer, N., Fischetti, M., & Grötschel, M. (2001). Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Mathematical Programming*, *90*(3), 475–506.

Audet, C., Bigeon, J., Cartier, D., Le Digabel, S., & Salomon, L. (2018). Performance indicators in multiobjective optimization. *Technical Report, Les Cahiers du GERAD G-2018-90, 38 pages*.

Augerat, P., Belenguer, J. M., Benavent, E., Corberan, A., Naddef, D., & Rinaldi, G. (1995, 01). Computational results with a branch and cut code for the capacitated vehicle routing problem.

Baker, E. (1983). An exact algorithm for the time-constrained traveling salesman problem. *Operations Research*, *31*(5), 938–945.

Balas, E., & Simonetti, N. (2001). Linear time dynamic-programming algorithms for new classes of restricted TSPs: A computational study. *INFORMS journal on Computing*, *13*(1), 56–75.

Baldacci, R., Hadjiconstantinou, E., & Mingozzi, A. (2004). An exact algorithm for the capacitated vehicle routing problem based on a two-commodity network flow formulation. *Operations Research*, *52*(5), 723-738.

Baldacci, R., Mingozzi, A., & Roberti, R. (2012). New state-space relaxations for solving the traveling salesman problem with time windows. *INFORMS Journal on Computing*, *24*(3), 356–371.

Basu, S. (2012). Tabu search implementation on traveling salesman problem and its variations: a literature survey. *American Journal of Operations Research*, *2*(02), 163.

Baxter, J. (1984). Depot location: A technique for the avoidance of local optima. *European journal of operational research*, *18*(2), 208–214.

Beasley, J. E. (1983). Route first-cluster second methods for vehicle routing. *Omega*, *11*(4), 403–408.

Bektaş, T., Repoussis, P. P., & Tarantilis, C. D. (2014). Chapter 11: Dynamic vehicle routing problems. In P. Toth & D. Vigo (Eds.), *Vehicle routing: Problems, methods, and applications, second edition* (pp. 299–347). SIAM, Philadelphia.

Boland, N., Charkhgard, H., & Savelsbergh, M. (2017). The quadrant shrinking method: A simple and efficient algorithm for solving tri-objective integer programs. *European Journal of Operational Research*, *260*(3), 873–885.

Bramel, J., & Simchi-Levi, D. (1995). A location based heuristic for general routing problems. *Operations research*, *43*(4), 649–660.

Branke, J., Deb, K., Miettinen, K., & Slowinski, R. (Eds.). (2008). *Multiobjective optimization: Interactive and evolutionary approaches* (Vol. 5252). Springer-Verlag, Berlin, Heidelberg.

Burke, E. K., Kendall, G., et al. (2005). *Search methodologies: Introductory tutorials in optimization and decision support techniques*. Springer.

Cacchiani, V., Huisman, D., Kidd, M., Kroon, L., Toth, P., Veelenturf, L., & Wagenaar, J. (2014). An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological*, *63*, 15–37.

Calvo, R. W. (2000). A new heuristic for the traveling salesman problem with time windows. *Transportation Science*, *34*(1), 113–124.

Carlton, W. B., & Barnes, J. W. (1996). Solving the traveling-salesman problem with time windows using tabu search. *IIE transactions*, *28*(8), 617–629.

Carosi, S., Gualandi, S., Malucelli, F., & Tresoldi, E. (2015). Delay management in public transportation: service regularity issues and crew rescheduling. *Transportation Research Procedia*, *10*, 483–492.

Chankong, V., & Haimes, Y. Y. (1983). Multiobjective decision making: Theory and methodology. In *North holland series in system science and engineering.* North-Holland.

Chen, P., Huang, H.-K., & Dong, X.-Y. (2010). Iterated variable neighborhood descent algorithm for the capacitated vehicle routing problem. *Expert Systems with Applications*, *37*(2), 1620–1627.

Cheng, C.-B., & Mao, C.-P. (2007). A modified ant colony system for solving the travelling salesman problem with time windows. *Mathematical and Computer Modelling*, *46*(9), 1225–1235.

Christofides, N., Mingozzi, A., & Toth, P. (1979). The vehicle routing problem. In N. Christofides, A. Mingozzi, P. Toth, & C. Sandi (Eds.), *Combinatorial optimization* (pp. 315–338). Wiley, Chichester, UK.

Christofides, N., Mingozzi, A., & Toth, P. (1981). State-space relaxation procedures for the computation of bounds to routing problems. *Networks*, *11*(2), 145–164.

Clarke, G., & Wright, J. W. (1964). Scheduling of vehicles from a central depot to a number of delivery points. *Operations research*, *12*(4), 568–581.

Clausen, J., Larsen, A., Larsen, J., & Rezanova, N. J. (2010). Disruption management in the airline industry - concepts, models and methods. *Computers & Operations Research*, *37*(5), 809-821.

Cordeau, J.-F., & Laporte, G. (2002). Tabu search heuristics for the vehicle routing problem. *Les Cahiers du GERAD G-2002-15*, 1-14.

Cordeau, J.-F., Laporte, G., & Mercier, A. (2001). A unified tabu search heuristic for vehicle routing problems with time windows. *Journal of the Operational research society*, *52*(8), 928–936.

Dantzig, G. B., & Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, *6*, 80-91.

Das, I., & Dennis, J. E. (1997). A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Structural optimization*, *14*(1), 63–69.

Da Silva, R. F., & Urrutia, S. (2010). A general VNS heuristic for the traveling salesman problem with time windows. *Discrete Optimization*, *7*(4), 203–211.

Desrochers, M., & Verhoog, T. (1989). A matching based savings algorithm for the vehicle routing problem. *Technical Report Cahiers du GERAD G-89-04*.

Dhahri, A., Zidi, K., & Ghedira, K. (2013). Vehicle routing problem with time windows under availability constraints. In *2013 international conference on advanced logistics and transport* (p. 308-314).

Doerner, K. F., & Hartl, R. F. (2018). Comments on: Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP*, *26*(1), 21–24.

Dueck, G. (1993). New optimization heuristics: The great deluge algorithm and the record-to-record travel. *Journal of Computational physics*, *104*(1), 86–92.

Dueck, G., & Scheuer, T. (1990). Threshold accepting: a general purpose optimization algorithm appearing superior to simulated annealing. *Journal of computational physics*, *90*(1), 161–175.

Dumas, Y., Desrosiers, J., Gelinas, E., & Solomon, M. M. (1995). An optimal algorithm for the traveling salesman problem with time windows. *Operations research*, *43*(2), 367–371.

Eglese, R., & Zambirinis, S. (2018a). Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP*, *26*(1), 1–17.

Eglese, R., & Zambirinis, S. (2018b). Rejoinder on: Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP*, *26*(1), 27–29.

Ernst, A. T., Horn, M., Krisnamoorthy, M., Kilby, P., Degenhardt, P., & Moran, M. (2007). Static and dynamic order scheduling for recreational rental vehicles at tourism holdings limited. *Interfaces*, *37*(4), 334-341.

Fisher, M. L., & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. *Networks*, *11*(2), 109–124.

Focacci, F., Lodi, A., & Milano, M. (2002). A hybrid exact algorithm for the TSPTW. *INFORMS Journal on Computing*, *14*(4), 403–417.

Foster, B. A., & Ryan, D. M. (1976). An integer programming approach to the vehicle scheduling problem. *Journal of the Operational Research Society*, *27*(2), 367–384.

Freling, R., Wagelmans, A., & Paixao, J. (2001). Models and algorithms for single-depot vehicle scheduling. *Transportation Science*, *35*, 165-180.

Gambardella, L. M., Taillard, É., & Agazzi, G. (1999). MACS-VRPTW: A multiple ant colony system for vehicle routing problems with time windows. In *New ideas in optimization* (pp. 63–76).

Gaskell, T. (1967). Bases for vehicle fleet scheduling. *Journal of the Operational Research Society*, *18*(3), 281–295.

Gendreau, M. (2018). Comments on: Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP*, *26*(1), 18–20.

Gendreau, M., Hertz, A., & Laporte, G. (1994). A tabu search heuristic for the vehicle routing problem. *Management science*, *40*(10), 1276–1290.

Gendreau, M., Hertz, A., Laporte, G., & Stan, M. (1998). A generalized insertion heuristic for the traveling salesman problem with time windows. *Operations Research*, *46*(3), 330–335.

Gendreau, M., Laporte, G., & Potvin, J.-Y. (2002). Metaheuristics for the capacitated VRP. In *The vehicle routing problem, P. Toth and D. Vigo* (pp. 129–154). SIAM.

Gendreau, M., & Potvin, J.-Y. (2010). *Handbook of metaheuristics* (Vol. 2). Springer.

Gillett, B. E., & Miller, L. R. (1974). A heuristic algorithm for the vehicle-dispatch problem. *Operations research*, *22*(2), 340–349.

Glover, F. (1977). Heuristics for integer programming using surrogate constraints. *Decision sciences*, *8*(1), 156–166.

Glover, F. (1986). Future paths for integer programming and links to artificial intelligence. *Computers & operations research*, *13*(5), 533–549.

Glover, F. (1989). Tabu search-part i. *ORSA Journal on computing*, *1*(3), 190–206.

Glover, F. (1990). Tabu search-part ii. *ORSA Journal on computing*, *2*(1), 4–32.

Golden, B. L., Magnanti, T. L., & Nguyen, H. Q. (1977). Implementing vehicle routing algorithms. *Networks*, *7*(2), 113–148.

Haimes, Y. Y., Lasdon, L. S., & Widsmer, D. A. (1971). On a bicriterion formation of the problems of integrated system identification and system optimization. *IEEE Transactions on Systems, Man, and Cybernetics*, *1*(3), 296–297.

Hansen, P., Mladenović, N., Brimberg, J., & Pérez, J. A. M. (2010). Variable neighborhood search. In *Handbook of metaheuristics* (pp. 61–86). Springer.

Hu, X., & Sun, L. (2012). Knowledge-based modeling for disruption management in urban distribution. *Expert Systems with Applications*, *39*, 906-916.

Hu, X., Sun, L., & Liu, L. (2013). A PAM approach to handling disruptions in real-time vehicle routing problems. *Decision Support Systems*, *54*(3), 1380-1393.

Jiang, L., Wang, H., & Ding, B. (2013). Disruption management recovery model of distribution delay with service priority. *Asian Social Science*, *9*(2), 170-179.

Jozefowiez, N., Semet, F., & Talbi, E. (2008). Multi-objective vehicle routing problems. *European Journal of Operational Research*, *189*(2), 293-309.

Kara, I., & Derya, T. (2015). Formulations for minimizing tour duration of the traveling salesman problem with time windows. *Procedia Economics and Finance*, *26*, 1026–1034.

Kindervater, G. A. P., & Savelsbergh, M. W. P. (1997). Vehicle routing: handling edge exchanges. In *Local search in combinatorial optimization* (pp. 337–360). Wiley, Chichester, UK.

Kona, H., Burde, A., & Zanwar, D. (2015). A review of traveling salesman problem with time window constraint. *IJIRST - International Journal for Innovative Research in Science & Technology*, *2*, 253–256.

Kytöjoki, J., Nuortio, T., Bräysy, O., & Gendreau, M. (2007). An efficient variable neighborhood search heuristic for very large scale vehicle routing problems. *Computers & operations research*, *34*(9), 2743–2757.

Langevin, A., Desrochers, M., Desrosiers, J., Gélinas, S., & Soumis, F. (1993). A two-commodity flow formulation for the traveling salesman and the makespan problems with time windows. *Networks*, *23*(7), 631–640.

Laporte, G., Ropke, S., & Vidal, T. (2014). Chapter 4: Heuristics for the Vehicle Routing Problem. In *Vehicle Routing: Problems, Methods, and Applications, second edition* (pp. 87–116). SIAM.

Laporte, G., & Semet, F. (2002). Classical heuristics for the capacitated VRP. In *The vehicle routing problem, P. Toth and D. Vigo* (p. 109-128). SIAM.

Letchford, A. N., Lysgaard, J., & Eglese, R. W. (2007). A branch-and-cut algorithm for the capacitated open vehicle routing problem. *Journal of the Operational Research Society*, *58*(12), 1642-1651.

Li, F., Golden, B., & Wasil, E. (2005). Very large-scale vehicle routing: new test problems, algorithms, and results. *Computers & Operations Research*, *32*(5), 1165–1179.

Li, J.-Q., Borenstein, D., & Mirchandani, P. B. (2007a). A decision support system for single-depot vehicle rescheduling problem. *Computers & Operations Research*, *34*(4), 1008-1032.

Li, J.-Q., Borenstein, D., & Mirchandani, P. B. (2008a). Truck schedule recovery for solid waste collection in Porto Alegre, Brazil. *International Transactions in Operational Research*, *15*, 565-582.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2007b). Vehicle rescheduling problem: Model and algorithms. *Networks*, *50*(3), 211-229.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2008b). Parallel auction algorithm for bus rescheduling. In *Computer-aided systems in public transport* (pp. 281–299). Springer.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009a). Real-time vehicle rerouting problems with time windows. *European Journal of Operational Research*, *194*, 711-727.

Li, J.-Q., Mirchandani, P. B., & Borenstein, D. (2009b). A vehicle rescheduling problem with real-time vehicle reassignments and trip cancellations. *Transportation Research Part E: Logistics and Transportation Review*, *45*, 419-433.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *Bell System Technical Journal*, *44*(10), 2245–2269.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, *21*(2), 498–516.

López-Ibáñez, M., Blum, C., Ohlmann, J. W., & Thomas, B. W. (2013). The travelling salesman problem with time windows: Adapting algorithms from travel-time to makespan optimization. *Applied Soft Computing*, *13*(9), 3806–3815.

Lust, T., & Teghem, J. (2010). The multiobjective traveling salesman problem: a survey and a new approach. In C. Coello, C. Dhaenens, & L. Jourdan (Eds.), *Advances in multi-objective nature inspired computing, studies in computational intelligence* (Vol. 272, p. 119-141). Springer, Berlin.

Mamasis, K., Minis, I., & Dikas, G. (2013). Managing vehicle breakdown incidents during urban distribution of a common product. *Journal of the Operational Research Society*, *64*, 925-937.

Marler, R., & Arora, J. (2004). Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, *26*(6), 369-395.

McKinnon, A. (2015). Performance measurement in freight transport: Its contribution to the design, implementation and monitoring of public policy. *Report prepared for the Roundtable on Logistics Development Strategies and their Performance Measurements: 9-10 March 2015, Queretaro*.

McKinnon, A., Palmer, A., Edwards, J., & Piecyk, M. (2008). Reliability of road transport from the perspective of logistics managers and freight operators. *Final report by Logistics Research Centre Heriot-Watt University Edinburgh, UK*.

Miettinen, K. (1999). *Nonlinear multiobjective optimization* (Vol. 12). Springer Science & Business Media.

Minis, I., Mamasis, K., & Zeimpekis, V. (2012). Real-time management of vehicle breakdowns in urban freight distribution. *Journal of Heuristics*, *18*(3), 375-400.

Mirchandani, P. B., Li, J.-Q., & Hickman, M. (2010). A macroscopic model for integrating bus signal priority with vehicle rescheduling. *Public Transport*, *2*(3), 159–172.

Mladenović, N., Todosijević, R., & Urošević, D. (2012). An efficient GVNS for solving traveling salesman problem with time windows. *Electronic Notes in Discrete Mathematics*, *39*, 83–90.

Mole, R., & Jameson, S. (1976). A sequential route-building algorithm employing a generalised savings criterion. *Journal of the Operational Research Society*, *27*(2), 503–511.

Mu, Q., & Eglese, R. (2013). Disrupted capacitated vehicle routing problem with order release delay. *Annals of Operations Research*, *207*, 201-216.

Mu, Q., Fu, Z., Lysgaard, J., & Eglese, R. (2011). Disruption management of the vehicle routing problem with vehicle breakdown. *Journal of the Operational Research Society*, *62*, 742-749.

Nagata, Y., & Bräysy, O. (2009). Edge assembly-based memetic algorithm for the capacitated vehicle routing problem. *Networks: An International Journal*, *54*(4), 205–215.

Nelson, M. D., Nygard, K. E., Griffin, J. H., & Shreve, W. E. (1985). Implementation techniques for the vehicle routing problem. *Computers & Operations Research*, *12*(3), 273–283.

Ng, K. K. H., Lee, C. K. M., Zhang, S. Z., Wu, K., & Ho, W. (2017). A multiple colonies artificial bee colony algorithm for a capacitated vehicle routing problem and re-routing strategies under time-dependent traffic congestion. *Computers & Industrial Engineering*, *109*, 151–168.

Ngai, E. W. T., Leung, T. K. P., Wong, Y. H., Lee, M. C. M., Chai, P. Y. F., & Choi, Y. S. (2012). Design and development of a context-aware decision support system for real-time accident handling in logistics. *Decision Support Systems*, *52*, 816-827.

Ohlmann, J. W., & Thomas, B. W. (2007). A compressed-annealing heuristic for the traveling salesman problem with time windows. *INFORMS Journal on Computing*, *19*(1), 80–90.

Ombuki, B., Ross, B. J., & Hanshar, F. (2006). Multi-objective genetic algorithms for vehicle routing problem with time windows. *Applied Intelligence*, *24*(1), 17-30.

Or, I. (1976). Traveling salesman-type combinatorial optimization problems and their relation to the logistics of regional blood banking. *PhD dissertation, Department of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.*

Osman, I. H. (1993). Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. *Annals of operations research*, *41*(4), 421–451.

Pesant, G., Gendreau, M., Potvin, J.-Y., & Rousseau, J.-M. (1998). An exact constraint logic programming algorithm for the traveling salesman problem with time windows. *Transportation Science*, *32*(1), 12–29.

Pillac, V., Gendreau, M., Guéret, C., & Medaglia, A.-L. (2013). A review of dynamic vehicle routing problems. *European Journal of Operational Research*, *225*(1), 1-11.

Poggi, M., & Uchoa, E. (2014). Chapter 3: New exact algorithms for the Capacitated Vehicle Routing Problem. In *Vehicle routing: Problems, methods, and applications, second edition* (pp. 59–86). SIAM.

Potvin, J.-Y., & Bengio, S. (1996). The vehicle routing problem with time windows Part II: Genetic search. *INFORMS journal on Computing*, *8*(2), 165–172.

Prins, C. (2004). A simple and effective evolutionary algorithm for the vehicle routing problem. *Computers & Operations Research*, *31*(12), 1985–2002.

Psaraftis, H. N., Wen, M., & Kontovas, C. A. (2016). Dynamic vehicle routing problems: Three decades and counting. *Networks*, *67*(1), 3-31.

Rancourt, M.-E., & Paquette, J. (2014). Multicriteria optimization of a long-haul routing and scheduling problem. *Journal of Multi-Criteria Decision Analysis*, *21*(5-6), 239-255.

Reimann, M., Doerner, K., & Hartl, R. F. (2004). D-Ants: Savings based ants divide and conquer the vehicle routing problem. *Computers & Operations Research*, *31*(4), 563–591.

Renaud, J., Boctor, F. F., & Laporte, G. (1996). An improved petal heuristic for the vehicle routeing problem. *Journal of the operational Research Society*, *47*(2), 329–336.

Resende, M. G., & Ribeiro, C. C. (2010). Greedy randomized adaptive search procedures: Advances, hybridizations, and applications. In *Handbook of metaheuristics* (pp. 283–319). Springer.

Rochat, Y., & Taillard, É. D. (1995). Probabilistic diversification and intensification in local search for vehicle routing. *Journal of heuristics*, *1*(1), 147–167.

Ryan, D. M., Hjorring, C., & Glover, F. (1993). Extensions of the petal method for vehicle routing. *Journal of the Operational Research Society*, *44*(3), 289–296.

Sankaran, J. K., Gore, A., & Coldwell, B. (2005). The impact of road traffic congestion on supply chains: insights from Auckland, New Zealand. *International Journal of Logistics: Research and Applications*, *8*(2), 159–180.

Savelsbergh, M. W. P. (1985). Local search in routing problems with time windows. *Annals of Operations Research*, *4*(1), 285–305.

Semet, F., Toth, P., & Vigo, D. (2014). Chapter 2: Classical Exact Algorithms for the Capacitated Vehicle Routing Problem. In *Vehicle routing: Problems, methods, and applications, second edition* (pp. 37–57). SIAM.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problems with time window constraints. *Operations Research*, *35*(2), 254-265.

Spliet, R., Gabor, A. F., & Dekker, R. (2014). The vehicle rescheduling problem. *Computers & Operations Research*, *43*, 129-136.

Subramanian, A., Uchoa, E., & Ochi, L. S. (2013). A hybrid algorithm for a class of vehicle routing problems. *Computers & Operations Research*, *40*(10), 2519–2531.

Sun, L.-J., Li, F., & Hu, X.-P. (2016). An ontology-based model for typical-context awareness in the oil products distribution system. *Procedia Computer Science*, *96*, 1156–1165.

Thompson, P. M., & Psaraftis, H. N. (1993). Cyclic transfer algorithm for multivehicle routing and scheduling problems. *Operations research*, *41*(5), 935–946.

Tilk, C., & Irnich, S. (2016). Dynamic programming for the minimum tour duration problem. *Transportation Science*, *51*(2), 549–565.

Toth, P., & Vigo, D. (Eds.). (2014). *Vehicle routing: Problems, methods, and applications, second edition.* SIAM, Philadelphia, USA.

Uçar, E., Birbil, Ş. İ., & Muter, İ. (2017). Managing disruptions in the multi-depot vehicle scheduling problem. *Transportation Research Part B: Methodological*, *105*, 249–269.

Van Breedam, A. (1994). *An analysis of the behavior of heuristics for the vehicle routing problem for a selection of problems with vehicle-related, customer-related, and time-related constraints.* Ph.D. dissertation, University of Antwerp.

Vidal, T., Crainic, T. G., Gendreau, M., Lahrichi, N., & Rei, W. (2012). A hybrid genetic algorithm for multidepot and periodic vehicle routing problems. *Operations Research*, *60*(3), 611–624.

Vigo, D. (2018). Comments on: Disruption management in vehicle routing and scheduling for road freight transport: a review. *TOP*, *26*(1), 25–26.

Visentini, M.-S., Borenstein, D., Li, J.-Q., & Mirchandani, P.-B. (2013). Review of real-time vehicle schedule recovery methods in transportation services. *Journal of Scheduling*, *17*(6), 541-567.

Wang, X., & Cao, H. (2008). A dynamic vehicle routing problem with backhaul and time window. In *Proceedings of the 2008 IEEE international conference on service operations and logistics, and informatics* (p. 1256-1261).

Wang, X., Ruan, J., & Shi, Y. (2012). A recovery model for combinational disruptions in logistics delivery: Considering the real-world participators. *International Journal of Production Economics*, *140*(1), 508-520.

Wang, X., Wu, X., & Hu, X. (2010). A study of urgency vehicle routing disruption management problem. *Journal of Networks*, *5*(12), 1426-1433.

Wang, X., Wu, X., Wang, Z., & Hu, X. (2009a). A model and an improved genetic algorithm for the vehicle routing problem with break-down vehicles. In *Fourth international conference on innovative computing, information and control (ICICIC)* (p. 696-699).

Wang, X., Xu, C., & Yang, D. (2009b). Disruption management for vehicle routing problem with the request changes of customers. *International Journal of Innovative Computing, Information and Control*, *5*(8), 2427-2438.

Wang, Z., & Shi, J. (2009). A model for urban distribution system under disruptions of vehicle travel time delay. In *Second international conference on intelligent computation technology and automation* (Vol. 3, p. 433-436).

Wark, P., & Holt, J. (1994). A repeated matching heuristic for the vehicle routeing problem. *Journal of the Operational Research Society*, *45*(10), 1156–1167.

Wren, A., & Carr, J. (1971). *Computers in transport planning and operation.* Ian Allan, London.

Wren, A., & Holliday, A. (1972). Computer scheduling of vehicles from one or more depots to a number of delivery points. *Journal of the Operational Research Society*, *23*(3), 333–344.

Yellow, P. (1970). A computational modification to the savings method of vehicle scheduling. *Journal of the Operational Research Society*, *21*(2), 281–283.

Yu, G., & Qi, X. (2004). *Disruption management: Framework, models and applications.* World Scientific Publishing Co. Pte Ltd, Singapore.

Zhang, X., & Tang, L. (2007). Disruption management for the vehicle routing problem with time windows. In D. Huang, L. Heutte, & M. Loog (Eds.), *Advanced intelligent computing theories and applications with aspects of contemporary intelligent computing techniques* (Vol. 2, p. 225-234).

Zitzler, E., Knowles, J., & Thiele, L. (2008). Quality assessment of Pareto set approximations. In *Multiobjective Optimization. Lecture Notes in Computer Science, Vol. 5252* (pp. 373–404). Springer.

Appendix A

Experimental Results of Heuristic Approach 3 for Problem 2 (Delayed-TSPTW-3)

Table A.1: Experimental results for instance 1-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB$_{f_1}$**	**LB$_{f_2}$**	**Reference value for f_3**
88.7	5	167	281
Solution S/N	**f_1**	**f_2**	**f_3**
1	6 (0%)	743 (56.91%)	398 (90.53%)
2	6 (0%)	784 (63.73%)	387 (78.95%)
3	7 (14.29%)	645 (40.6%)	387 (78.95%)
4	7 (14.29%)	661 (43.26%)	381 (72.63%)
5	8 (28.57%)	596 (32.45%)	381 (72.63%)
6	9 (42.86%)	1002 (100%)	325 (13.68%)
7	10 (57.14%)	424 (3.83%)	399 (91.58%)
8	10 (57.14%)	466 (10.82%)	388 (80%)
9	10 (57.14%)	545 (23.96%)	381 (72.63%)
10	10 (57.14%)	877 (79.2%)	329 (17.89%)
11	11 (71.43%)	435 (5.66%)	387 (78.95%)
12	11 (71.43%)	867 (77.54%)	312 (0%)
13	12 (85.71%)	415 (2.33%)	407 (100%)
14	12 (85.71%)	516 (19.13%)	382 (73.68%)
15	12 (85.71%)	662 (43.43%)	332 (21.05%)
16	13 (100%)	401 (0%)	405 (97.89%)
17	13 (100%)	425 (3.99%)	385 (76.84%)

Table A.2: Experimental results for instance 1-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
24.9	5	240	366
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	5 (0%)	800 (93.49%)	451 (100%)
2	6 (25%)	536 (15.38%)	439 (78.57%)
3	6 (25%)	822 (100%)	437 (75%)
4	7 (50%)	492 (2.37%)	413 (32.14%)
5	8 (75%)	484 (0%)	416 (37.5%)
6	8 (75%)	621 (40.53%)	395 (0%)
7	9 (100%)	609 (36.98%)	402 (12.5%)

Table A.3: Experimental results for instance 1-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
5.5	4	180	409
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	4 (0%)	344 (72.13%)	458 (100%)
2	5 (100%)	300 (0%)	439 (44.12%)
3	5 (100%)	361 (100%)	424 (0%)

Table A.4: Experimental results for instance 2-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
809.9	5	193	353
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	7 (0%)	1255 (60.63%)	581 (100%)
2	8 (5.56%)	881 (32.53%)	540 (75.15%)
3	9 (11.11%)	856 (30.65%)	505 (53.94%)
4	9 (11.11%)	1130 (51.24%)	481 (39.39%)
5	10 (16.67%)	830 (28.7%)	533 (70.91%)
6	10 (16.67%)	834 (29%)	520 (63.03%)
7	10 (16.67%)	953 (37.94%)	480 (38.79%)
8	11 (22.22%)	834 (29%)	519 (62.42%)

Continued on the next page

Table A.4 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
9	11 (22.22%)	937 (36.74%)	480 (38.79%)
10	11 (22.22%)	1089 (48.16%)	476 (36.36%)
11	11 (22.22%)	1466 (76.48%)	471 (33.33%)
12	12 (27.78%)	807 (26.97%)	523 (64.85%)
13	13 (33.33%)	697 (18.71%)	510 (56.97%)
14	14 (38.89%)	798 (26.3%)	466 (30.3%)
15	14 (38.89%)	1779 (100%)	417 (0.61%)
16	16 (50%)	785 (25.32%)	475 (35.76%)
17	17 (55.56%)	679 (17.36%)	512 (58.18%)
18	18 (61.11%)	493 (3.38%)	522 (64.24%)
19	18 (61.11%)	1398 (71.37%)	417 (0.61%)
20	19 (66.67%)	448 (0%)	528 (67.88%)
21	19 (66.67%)	701 (19.01%)	481 (39.39%)
22	20 (72.22%)	1265 (61.38%)	416 (0%)
23	21 (77.78%)	564 (8.72%)	455 (23.64%)
24	22 (83.33%)	532 (6.31%)	480 (38.79%)
25	22 (83.33%)	1005 (41.85%)	417 (0.61%)
26	25 (100%)	857 (30.73%)	417 (0.61%)

Table A.5: Experimental results for instance 2-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
225.4	14	934	465
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	15 (0%)	2139 (20.91%)	564 (100%)
2	15 (0%)	2975 (100%)	553 (75.56%)
3	16 (20%)	1972 (5.11%)	540 (46.67%)
4	16 (20%)	2017 (9.37%)	519 (0%)
5	17 (40%)	1960 (3.97%)	526 (15.56%)
6	18 (60%)	1972 (5.11%)	523 (8.89%)
7	20 (100%)	1918 (0%)	543 (53.33%)
8	20 (100%)	1939 (1.99%)	520 (2.22%)

Table A.6: Experimental results for instance 2-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for $\mathbf{f_3}$**
75.3	15	1904	554
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	15 (0%)	3100 (81.05%)	595 (33.33%)
2	15 (0%)	3136 (86.47%)	592 (25%)
3	16 (33.33%)	3050 (73.53%)	619 (100%)
4	16 (33.33%)	3091 (79.7%)	604 (58.33%)
5	16 (33.33%)	3226 (100%)	589 (16.67%)
6	17 (66.67%)	2587 (3.91%)	583 (0%)
7	18 (100%)	2561 (0%)	593 (27.78%)

Table A.7: Experimental results for instance 3-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for $\mathbf{f_3}$**
685.7	7	162	330
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	8 (0%)	473 (27.44%)	539 (95.61%)
2	9 (11.11%)	290 (0%)	544 (100%)
3	9 (11.11%)	447 (23.54%)	539 (95.61%)
4	9 (11.11%)	720 (64.47%)	533 (90.35%)
5	10 (22.22%)	532 (36.28%)	533 (90.35%)
6	11 (33.33%)	323 (4.95%)	519 (78.07%)
7	12 (44.44%)	872 (87.26%)	432 (1.75%)
8	13 (55.56%)	676 (57.87%)	432 (1.75%)
9	13 (55.56%)	723 (64.92%)	431 (0.88%)
10	15 (77.78%)	573 (42.43%)	432 (1.75%)
11	15 (77.78%)	957 (100%)	430 (0%)
12	17 (100%)	791 (75.11%)	430 (0%)

Table A.8: Experimental results for instance 3-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
165.7	13*	617	460
Solution S/N	**f_1**	**f_2**	**f_3**
1	13 (0%)	2102 (90.6%)	596 (100%)
2	14 (20%)	1653 (43.19%)	558 (55.29%)
3	15 (40%)	1518 (28.93%)	546 (41.18%)
4	15 (40%)	2191 (100%)	532 (24.71%)
5	16 (60%)	1309 (6.86%)	569 (68.24%)
6	16 (60%)	1328 (8.87%)	561 (58.82%)
7	16 (60%)	1625 (40.23%)	511 (0%)
8	17 (80%)	1244 (0%)	555 (51.76%)
9	17 (80%)	1386 (14.99%)	532 (24.71%)
10	18 (100%)	1329 (8.98%)	529 (21.18%)

Table A.9: Experimental results for instance 3-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
64.3	15	2165	569
Solution S/N	**f_1**	**f_2**	**f_3**
1	15 (0%)	4075 (100%)	658 (100%)
2	16 (33.33%)	3097 (7.82%)	654 (93.65%)
3	16 (33.33%)	3605 (55.7%)	595 (0%)
4	17 (66.67%)	3060 (4.34%)	601 (9.52%)
5	18 (100%)	3014 (0%)	647 (82.54%)
6	18 (100%)	3029 (1.41%)	631 (57.14%)
7	18 (100%)	3044 (2.83%)	611 (25.4%)

Table A.10: Experimental results for instance 4-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
89.3	4	101	271
Solution S/N	**f_1**	**f_2**	**f_3**
1	4 (0%)	715 (85.99%)	400 (69.37%)

Continued on the next page

Table A.10 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
2	5 (20%)	693 (82.18%)	434 (100%)
3	5 (20%)	699 (83.22%)	401 (70.27%)
4	5 (20%)	796 (100%)	359 (32.43%)
5	6 (40%)	351 (23.01%)	424 (90.99%)
6	6 (40%)	777 (96.71%)	362 (35.14%)
7	7 (60%)	218 (0%)	411 (79.28%)
8	7 (60%)	474 (44.29%)	366 (38.74%)
9	8 (80%)	239 (3.63%)	377 (48.65%)
10	9 (100%)	451 (40.31%)	323 (0%)

Table A.11: Experimental results for instance 4-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
32.1	7*	327	359
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	7 (0%)	479 (0%)	419 (100%)
2	7 (0%)	736 (100%)	402 (0%)
3	8 (100%)	490 (4.28%)	402 (0%)

Table A.12: Experimental results for instance 4-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
95.9	6	514	444
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	6 (0%)	938 (100%)	445 (0%)
2	7 (100%)	653 (0%)	454 (100%)

Table A.13: Experimental results for instance 5-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
10581.4	15	580	471
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	16 (0%)	2957 (55.9%)	705 (91.07%)

Continued on the next page

Table A.13 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
2	17 (4.17%)	2826 (51.57%)	715 (100%)
3	17 (4.17%)	2920 (54.68%)	702 (88.39%)
4	18 (8.33%)	2817 (51.27%)	704 (90.18%)
5	18 (8.33%)	2821 (51.4%)	703 (89.29%)
6	19 (12.5%)	2707 (47.64%)	704 (90.18%)
7	19 (12.5%)	2718 (48%)	702 (88.39%)
8	20 (16.67%)	2657 (45.98%)	688 (75.89%)
9	23 (29.17%)	2037 (25.49%)	692 (79.46%)
10	26 (41.67%)	1950 (22.61%)	696 (83.04%)
11	30 (58.33%)	1297 (1.02%)	711 (96.43%)
12	30 (58.33%)	1385 (3.93%)	702 (88.39%)
13	31 (62.5%)	1282 (0.53%)	690 (77.68%)
14	31 (62.5%)	4291 (100%)	612 (8.04%)
15	32 (66.67%)	2950 (55.67%)	609 (5.36%)
16	34 (75%)	1266 (0%)	689 (76.79%)
17	34 (75%)	3366 (69.42%)	603 (0%)
18	37 (87.5%)	1767 (16.56%)	612 (8.04%)
19	39 (95.83%)	1755 (16.17%)	612 (8.04%)
20	40 (100%)	1731 (15.37%)	612 (8.04%)

Table A.14: Experimental results for instance 5-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
2544.8	24	1878	591
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	24 (0%)	6237 (100%)	716 (67.86%)
2	25 (5.56%)	6044 (92.69%)	710 (57.14%)
3	25 (5.56%)	6109 (95.15%)	693 (26.79%)
4	26 (11.11%)	6017 (91.67%)	725 (83.93%)
5	27 (16.67%)	5785 (82.89%)	694 (28.57%)
6	27 (16.67%)	6048 (92.84%)	692 (25%)
7	28 (22.22%)	5397 (68.19%)	692 (25%)
8	28 (22.22%)	5715 (80.23%)	690 (21.43%)
9	29 (27.78%)	5832 (84.66%)	678 (0%)
10	30 (33.33%)	4821 (46.38%)	729 (91.07%)
11	30 (33.33%)	5169 (59.56%)	697 (33.93%)
12	31 (38.89%)	4781 (44.87%)	734 (100%)

Continued on the next page

Table A.14 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
13	32 (44.44%)	4090 (18.71%)	698 (35.71%)
14	32 (44.44%)	4809 (45.93%)	697 (33.93%)
15	32 (44.44%)	4813 (46.08%)	693 (26.79%)
16	34 (55.56%)	5138 (58.39%)	692 (25%)
17	38 (77.78%)	3837 (9.13%)	717 (69.64%)
18	39 (83.33%)	3857 (9.88%)	696 (32.14%)
19	39 (83.33%)	3962 (13.86%)	694 (28.57%)
20	40 (88.89%)	3730 (5.07%)	714 (64.29%)
21	41 (94.44%)	3747 (5.72%)	693 (26.79%)
22	42 (100%)	3596 (0%)	725 (83.93%)

Table A.15: Experimental results for instance 5-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
446.9	27*	3678	725
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	27 (0%)	6609 (100%)	796 (100%)
2	28 (33.33%)	6237 (53.38%)	783 (78.33%)
3	28 (33.33%)	6598 (98.62%)	749 (21.67%)
4	29 (66.67%)	5946 (16.92%)	786 (83.33%)
5	29 (66.67%)	5965 (19.3%)	771 (58.33%)
6	29 (66.67%)	6073 (32.83%)	761 (41.67%)
7	30 (100%)	5811 (0%)	736 (0%)

Table A.16: Experimental results for instance 6-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
10146.2	19*	894	495
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	21 (0%)	8741 (100%)	809 (100%)
2	22 (3.23%)	7720 (83.84%)	766 (79.81%)
3	23 (6.45%)	6071 (57.75%)	736 (65.73%)
4	24 (9.68%)	5416 (47.38%)	733 (64.32%)
5	25 (12.9%)	5355 (46.42%)	740 (67.61%)

Continued on the next page

Table A.16 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
6	25 (12.9%)	5744 (52.57%)	719 (57.75%)
7	25 (12.9%)	6625 (66.51%)	695 (46.48%)
8	26 (16.13%)	5073 (41.95%)	740 (67.61%)
9	26 (16.13%)	5570 (49.82%)	716 (56.34%)
10	27 (19.35%)	4763 (37.05%)	731 (63.38%)
11	27 (19.35%)	5591 (50.15%)	705 (51.17%)
12	28 (22.58%)	4801 (37.65%)	729 (62.44%)
13	28 (22.58%)	5752 (52.7%)	704 (50.7%)
14	29 (25.81%)	4574 (34.06%)	729 (62.44%)
15	29 (25.81%)	5300 (45.55%)	704 (50.7%)
16	30 (29.03%)	4184 (27.88%)	729 (62.44%)
17	30 (29.03%)	5126 (42.79%)	701 (49.3%)
18	31 (32.26%)	4176 (27.76%)	729 (62.44%)
19	31 (32.26%)	5085 (42.14%)	702 (49.77%)
20	32 (35.48%)	4098 (26.52%)	725 (60.56%)
21	32 (35.48%)	5109 (42.52%)	690 (44.13%)
22	33 (38.71%)	3758 (21.14%)	725 (60.56%)
23	33 (38.71%)	4801 (37.65%)	702 (49.77%)
24	34 (41.94%)	3078 (10.38%)	729 (62.44%)
25	34 (41.94%)	4357 (30.62%)	696 (46.95%)
26	34 (41.94%)	5876 (54.66%)	598 (0.94%)
27	35 (45.16%)	2438 (0.25%)	729 (62.44%)
28	35 (45.16%)	4334 (30.26%)	691 (44.6%)
29	35 (45.16%)	5020 (41.11%)	596 (0%)
30	36 (48.39%)	3673 (19.8%)	691 (44.6%)
31	37 (51.61%)	2422 (0%)	729 (62.44%)
32	37 (51.61%)	3567 (18.12%)	726 (61.03%)
33	37 (51.61%)	4765 (37.08%)	599 (1.41%)
34	38 (54.84%)	3672 (19.78%)	725 (60.56%)
35	39 (58.06%)	3592 (18.52%)	691 (44.6%)
36	39 (58.06%)	4698 (36.02%)	600 (1.88%)
37	40 (61.29%)	3572 (18.2%)	690 (44.13%)
38	41 (64.52%)	3322 (14.24%)	691 (44.6%)
39	41 (64.52%)	4451 (32.11%)	600 (1.88%)
40	42 (67.74%)	3179 (11.98%)	704 (50.7%)
41	42 (67.74%)	3209 (12.45%)	703 (50.23%)
42	43 (70.97%)	3183 (12.04%)	703 (50.23%)
43	43 (70.97%)	3232 (12.82%)	700 (48.83%)

Continued on the next page

Table A.16 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
44	43 (70.97%)	4181 (27.84%)	600 (1.88%)
45	44 (74.19%)	3256 (13.2%)	696 (46.95%)
46	44 (74.19%)	4306 (29.81%)	599 (1.41%)
47	45 (77.42%)	3545 (17.77%)	600 (1.88%)
48	46 (80.65%)	3375 (15.08%)	600 (1.88%)
49	49 (90.32%)	3224 (12.69%)	600 (1.88%)
50	49 (90.32%)	3359 (14.83%)	599 (1.41%)
51	52 (100%)	2427 (0.08%)	702 (49.77%)

Table A.17: Experimental results for instance 6-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
1764.1	29	2606	629
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	30 (0%)	7762 (100%)	792 (88.6%)
2	31 (6.25%)	7327 (84.18%)	805 (100%)
3	31 (6.25%)	7404 (86.98%)	788 (85.09%)
4	31 (6.25%)	7489 (90.07%)	776 (74.56%)
5	32 (12.5%)	6307 (47.07%)	793 (89.47%)
6	33 (18.75%)	6431 (51.58%)	786 (83.33%)
7	33 (18.75%)	6992 (71.99%)	748 (50%)
8	34 (25%)	6258 (45.29%)	782 (79.82%)
9	34 (25%)	6478 (53.29%)	734 (37.72%)
10	35 (31.25%)	5852 (30.52%)	750 (51.75%)
11	35 (31.25%)	6055 (37.9%)	733 (36.84%)
12	36 (37.5%)	5463 (16.37%)	730 (34.21%)
13	36 (37.5%)	5881 (31.58%)	717 (22.81%)
14	36 (37.5%)	6119 (40.23%)	707 (14.04%)
15	37 (43.75%)	5451 (15.93%)	734 (37.72%)
16	38 (50%)	5417 (14.7%)	729 (33.33%)
17	38 (50%)	5784 (28.05%)	709 (15.79%)
18	39 (56.25%)	5372 (13.06%)	725 (29.82%)
19	39 (56.25%)	5511 (18.12%)	708 (14.91%)
20	40 (62.5%)	5533 (18.92%)	691 (0%)
21	41 (68.75%)	5207 (7.06%)	740 (42.98%)
22	41 (68.75%)	5425 (14.99%)	718 (23.68%)
23	41 (68.75%)	5472 (16.7%)	711 (17.54%)

Continued on the next page

Table A.17 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
24	42 (75%)	5196 (6.66%)	751 (52.63%)
25	42 (75%)	5205 (6.98%)	725 (29.82%)
26	42 (75%)	5408 (14.37%)	714 (20.18%)
27	43 (81.25%)	5128 (4.18%)	725 (29.82%)
28	44 (87.5%)	5032 (0.69%)	749 (50.88%)
29	44 (87.5%)	5272 (9.42%)	712 (18.42%)
30	44 (87.5%)	5289 (10.04%)	711 (17.54%)
31	46 (100%)	5013 (0%)	755 (56.14%)
32	46 (100%)	5089 (2.76%)	715 (21.05%)

Table A.18: Experimental results for instance 6-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
243.8	29	5376	779
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	29 (0%)	11050 (100%)	893 (86.21%)
2	30 (20%)	10503 (85.72%)	909 (100%)
3	30 (20%)	10557 (87.13%)	842 (42.24%)
4	31 (40%)	8278 (27.62%)	832 (33.62%)
5	32 (60%)	7595 (9.79%)	870 (66.38%)
6	32 (60%)	7798 (15.09%)	828 (30.17%)
7	33 (80%)	7471 (6.55%)	812 (16.38%)
8	34 (100%)	7220 (0%)	793 (0%)

Table A.19: Experimental results for instance 7-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
24296.1	27*	1340	565
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	28 (0%)	12210 (100%)	886 (100%)
2	29 (2.38%)	9411 (69.05%)	868 (91.39%)
3	30 (4.76%)	9326 (68.11%)	868 (91.39%)
4	30 (4.76%)	11862 (96.15%)	860 (87.56%)
5	31 (7.14%)	8807 (62.37%)	839 (77.51%)

Continued on the next page

Table A.19 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
6	32 (9.52%)	8302 (56.78%)	868 (91.39%)
7	32 (9.52%)	8647 (60.6%)	850 (82.78%)
8	33 (11.9%)	7803 (51.27%)	832 (74.16%)
9	34 (14.29%)	6938 (41.7%)	860 (87.56%)
10	34 (14.29%)	9437 (69.34%)	784 (51.2%)
11	35 (16.67%)	7053 (42.97%)	847 (81.34%)
12	35 (16.67%)	7591 (48.92%)	832 (74.16%)
13	35 (16.67%)	8308 (56.85%)	787 (52.63%)
14	36 (19.05%)	6250 (34.09%)	831 (73.68%)
15	37 (21.43%)	6067 (32.07%)	832 (74.16%)
16	37 (21.43%)	6843 (40.65%)	785 (51.67%)
17	38 (23.81%)	5431 (25.04%)	832 (74.16%)
18	38 (23.81%)	5597 (26.87%)	831 (73.68%)
19	38 (23.81%)	6054 (31.93%)	827 (71.77%)
20	39 (26.19%)	5794 (29.05%)	830 (73.21%)
21	39 (26.19%)	6560 (37.52%)	789 (53.59%)
22	40 (28.57%)	4965 (19.88%)	831 (73.68%)
23	40 (28.57%)	5527 (26.1%)	784 (51.2%)
24	41 (30.95%)	4863 (18.75%)	780 (49.28%)
25	42 (33.33%)	4221 (11.66%)	831 (73.68%)
26	42 (33.33%)	5880 (30%)	754 (36.84%)
27	44 (38.1%)	4150 (10.87%)	828 (72.25%)
28	44 (38.1%)	4565 (15.46%)	787 (52.63%)
29	45 (40.48%)	3642 (5.25%)	827 (71.77%)
30	46 (42.86%)	4741 (17.41%)	786 (52.15%)
31	47 (45.24%)	4556 (15.36%)	789 (53.59%)
32	48 (47.62%)	3582 (4.59%)	827 (71.77%)
33	48 (47.62%)	3927 (8.4%)	784 (51.2%)
34	49 (50%)	3556 (4.3%)	827 (71.77%)
35	49 (50%)	3759 (6.55%)	789 (53.59%)
36	50 (52.38%)	3550 (4.24%)	827 (71.77%)
37	51 (54.76%)	3715 (6.06%)	789 (53.59%)
38	52 (57.14%)	3479 (3.45%)	829 (72.73%)
39	52 (57.14%)	3481 (3.47%)	827 (71.77%)
40	52 (57.14%)	3666 (5.52%)	789 (53.59%)
41	54 (61.9%)	5786 (28.96%)	677 (0%)
42	55 (64.29%)	3446 (3.09%)	827 (71.77%)
43	55 (64.29%)	3569 (4.45%)	789 (53.59%)

Continued on the next page

Table A.19 – *Continued from the previous page*

Solution S/N	f_1	f_2	f_3
44	55 (64.29%)	3689 (5.77%)	783 (50.72%)
45	55 (64.29%)	3936 (8.5%)	779 (48.8%)
46	55 (64.29%)	5310 (23.7%)	677 (0%)
47	56 (66.67%)	3412 (2.71%)	827 (71.77%)
48	57 (69.05%)	3167 (0%)	827 (71.77%)
49	57 (69.05%)	3593 (4.71%)	788 (53.11%)
50	57 (69.05%)	5148 (21.91%)	677 (0%)
51	58 (71.43%)	4894 (19.1%)	677 (0%)
52	60 (76.19%)	3255 (0.97%)	789 (53.59%)
53	62 (80.95%)	4829 (18.38%)	677 (0%)
54	67 (92.86%)	4634 (16.22%)	677 (0%)
55	70 (100%)	4477 (14.49%)	677 (0%)

Table A.20: Experimental results for instance 7-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	LB_{f_1}	LB_{f_2}	Reference value for f_3
5655.8	39	4170	744
Solution S/N	f_1	f_2	f_3
1	40 (0%)	13085 (97.89%)	905 (91.92%)
2	41 (4.76%)	13186 (100%)	900 (86.87%)
3	42 (9.52%)	12296 (81.37%)	887 (73.74%)
4	42 (9.52%)	12658 (88.95%)	882 (68.69%)
5	42 (9.52%)	12998 (96.06%)	856 (42.42%)
6	43 (14.29%)	12515 (85.95%)	835 (21.21%)
7	44 (19.05%)	11772 (70.4%)	814 (0%)
8	45 (23.81%)	11341 (61.38%)	913 (100%)
9	45 (23.81%)	11663 (68.12%)	834 (20.2%)
10	46 (28.57%)	10976 (53.74%)	913 (100%)
11	46 (28.57%)	11009 (54.43%)	863 (49.49%)
12	47 (33.33%)	11503 (64.77%)	841 (27.27%)
13	48 (38.1%)	11660 (68.06%)	832 (18.18%)
14	49 (42.86%)	10188 (37.24%)	834 (20.2%)
15	50 (47.62%)	11286 (60.23%)	829 (15.15%)
16	51 (52.38%)	9444 (21.67%)	841 (27.27%)
17	52 (57.14%)	10584 (45.53%)	828 (14.14%)
18	53 (61.9%)	9160 (15.72%)	855 (41.41%)
19	55 (71.43%)	9350 (19.7%)	844 (30.3%)

Continued on the next page

Table A.20 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
20	55 (71.43%)	9528 (23.42%)	840 (26.26%)
21	56 (76.19%)	8603 (4.06%)	869 (55.56%)
22	56 (76.19%)	8886 (9.99%)	864 (50.51%)
23	57 (80.95%)	8476 (1.4%)	847 (33.33%)
24	57 (80.95%)	8534 (2.62%)	841 (27.27%)
25	59 (90.48%)	8455 (0.96%)	843 (29.29%)
26	60 (95.24%)	8810 (8.39%)	828 (14.14%)
27	61 (100%)	8409 (0%)	860 (46.46%)
28	61 (100%)	8727 (6.66%)	838 (24.24%)

Table A.21: Experimental results for instance 7-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
728.1	41	7160	906
Solution S/N	**f_1**	**f_2**	**f_3**
1	41 (0%)	12571 (100%)	992 (65.79%)
2	42 (16.67%)	11864 (68.49%)	968 (2.63%)
3	44 (50%)	10835 (22.64%)	999 (84.21%)
4	44 (50%)	11105 (34.67%)	986 (50%)
5	45 (66.67%)	10811 (21.57%)	1005 (100%)
6	45 (66.67%)	11665 (59.63%)	967 (0%)
7	46 (83.33%)	10327 (0%)	989 (57.89%)
8	46 (83.33%)	10404 (3.43%)	976 (23.68%)
9	46 (83.33%)	10407 (3.57%)	974 (18.42%)
10	47 (100%)	10343 (0.71%)	978 (28.95%)

Table A.22: Experimental results for instance 8-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
39275.9	21	864	535
Solution S/N	**f_1**	**f_2**	**f_3**
1	22 (0%)	3190 (60.87%)	824 (97.09%)
2	23 (3.85%)	2910 (46.45%)	824 (97.09%)
3	24 (7.69%)	2684 (34.81%)	826 (99.03%)

Continued on the next page

Table A.22 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
4	26 (15.38%)	2437 (22.09%)	826 (99.03%)
5	26 (15.38%)	2496 (25.13%)	824 (97.09%)
6	27 (19.23%)	2432 (21.83%)	826 (99.03%)
7	28 (23.08%)	2298 (14.93%)	824 (97.09%)
8	28 (23.08%)	3652 (84.65%)	726 (1.94%)
9	29 (26.92%)	2274 (13.7%)	824 (97.09%)
10	29 (26.92%)	3950 (100%)	724 (0%)
11	30 (30.77%)	2268 (13.39%)	826 (99.03%)
12	30 (30.77%)	3031 (52.68%)	726 (1.94%)
13	31 (34.62%)	2735 (37.44%)	726 (1.94%)
14	32 (38.46%)	2719 (36.61%)	726 (1.94%)
15	33 (42.31%)	2119 (5.72%)	827 (100%)
16	34 (46.15%)	2113 (5.41%)	826 (99.03%)
17	34 (46.15%)	2604 (30.69%)	726 (1.94%)
18	36 (53.85%)	2575 (29.2%)	726 (1.94%)
19	38 (61.54%)	2059 (2.63%)	826 (99.03%)
20	38 (61.54%)	2066 (2.99%)	824 (97.09%)
21	41 (73.08%)	2008 (0%)	824 (97.09%)
22	42 (76.92%)	2464 (23.48%)	726 (1.94%)
23	44 (84.62%)	2358 (18.02%)	726 (1.94%)
24	48 (100%)	2337 (16.94%)	726 (1.94%)

Table A.23: Experimental results for instance 8-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
5159.9	26	2884	725
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	30 (0%)	7188 (58.04%)	860 (94%)
2	31 (6.67%)	6793 (44.61%)	851 (76%)
3	32 (13.33%)	6499 (34.61%)	849 (72%)
4	32 (13.33%)	8422 (100%)	848 (70%)
5	33 (20%)	6307 (28.09%)	850 (74%)
6	33 (20%)	7359 (63.86%)	829 (32%)
7	34 (26.67%)	6184 (23.9%)	848 (70%)
8	34 (26.67%)	6206 (24.65%)	846 (66%)
9	34 (26.67%)	6300 (27.85%)	841 (56%)
10	34 (26.67%)	6931 (49.3%)	838 (50%)

Continued on the next page

Table A.23 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
11	35 (33.33%)	6401 (31.28%)	836 (46%)
12	35 (33.33%)	6963 (50.39%)	814 (2%)
13	36 (40%)	6377 (30.47%)	827 (28%)
14	37 (46.67%)	6127 (21.97%)	824 (22%)
15	37 (46.67%)	6630 (39.07%)	816 (6%)
16	38 (53.33%)	5785 (10.34%)	837 (48%)
17	38 (53.33%)	5983 (17.07%)	813 (0%)
18	40 (66.67%)	5592 (3.77%)	856 (86%)
19	40 (66.67%)	5737 (8.7%)	831 (36%)
20	41 (73.33%)	5524 (1.46%)	845 (64%)
21	42 (80%)	5517 (1.22%)	814 (2%)
22	44 (93.33%)	5892 (13.97%)	813 (0%)
23	45 (100%)	5481 (0%)	863 (100%)

Table A.24: Experimental results for instance 8-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
450.54	34	5774	882
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	34 (0%)	10883 (75.82%)	991 (100%)
2	34 (0%)	10914 (76.69%)	989 (97.59%)
3	34 (0%)	11744 (100%)	968 (72.29%)
4	35 (25%)	9019 (23.48%)	911 (3.61%)
5	35 (25%)	9026 (23.67%)	909 (1.2%)
6	36 (50%)	8455 (7.64%)	921 (15.66%)
7	36 (50%)	8478 (8.28%)	918 (12.05%)
8	36 (50%)	8484 (8.45%)	912 (4.82%)
9	37 (75%)	8336 (4.3%)	912 (4.82%)
10	37 (75%)	8469 (8.03%)	908 (0%)
11	38 (100%)	8183 (0%)	941 (39.76%)

Table A.25: Experimental results for instance 9-A (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	**LB_{f_1}**	**LB_{f_2}**	**Reference value for f_3**
3054.4	10	339	418
Solution S/N	**f_1**	**f_2**	**f_3**
1	11 (0%)	2282 (100%)	646 (100%)
2	12 (3.85%)	2016 (83.82%)	644 (98.62%)
3	12 (3.85%)	2090 (88.32%)	638 (94.48%)
4	13 (7.69%)	1822 (72.02%)	636 (93.1%)
5	14 (11.54%)	1655 (61.86%)	598 (66.9%)
6	15 (15.38%)	2096 (88.69%)	584 (57.24%)
7	16 (19.23%)	1552 (55.6%)	584 (57.24%)
8	17 (23.08%)	1243 (36.8%)	600 (68.28%)
9	17 (23.08%)	1274 (38.69%)	577 (52.41%)
10	19 (30.77%)	1059 (25.61%)	589 (60.69%)
11	19 (30.77%)	1217 (35.22%)	584 (57.24%)
12	20 (34.62%)	942 (18.49%)	602 (69.66%)
13	20 (34.62%)	1037 (24.27%)	586 (58.62%)
14	20 (34.62%)	1254 (37.47%)	580 (54.48%)
15	21 (38.46%)	904 (16.18%)	586 (58.62%)
16	21 (38.46%)	1818 (71.78%)	503 (1.38%)
17	22 (42.31%)	841 (12.35%)	586 (58.62%)
18	23 (46.15%)	839 (12.23%)	586 (58.62%)
19	23 (46.15%)	1209 (34.73%)	503 (1.38%)
20	24 (50%)	752 (6.93%)	586 (58.62%)
21	24 (50%)	1333 (42.27%)	502 (0.69%)
22	25 (53.85%)	973 (20.38%)	577 (52.41%)
23	25 (53.85%)	1279 (38.99%)	502 (0.69%)
24	26 (57.69%)	721 (5.05%)	589 (60.69%)
25	26 (57.69%)	746 (6.57%)	587 (59.31%)
26	27 (61.54%)	703 (3.95%)	589 (60.69%)
27	27 (61.54%)	1198 (34.06%)	503 (1.38%)
28	28 (65.38%)	1145 (30.84%)	502 (0.69%)
29	29 (69.23%)	638 (0%)	587 (59.31%)
30	32 (80.77%)	1022 (23.36%)	503 (1.38%)
31	35 (92.31%)	903 (16.12%)	503 (1.38%)
32	35 (92.31%)	914 (16.79%)	502 (0.69%)
33	37 (100%)	1574 (56.93%)	501 (0%)

Table A.26: Experimental results for instance 9-B (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
1226.9	21	1643	530
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	21 (0%)	6824 (100%)	727 (100%)
2	22 (6.25%)	6080 (77.58%)	694 (78%)
3	23 (12.5%)	5904 (72.27%)	681 (69.33%)
4	23 (12.5%)	6095 (78.03%)	676 (66%)
5	24 (18.75%)	5767 (68.14%)	683 (70.67%)
6	24 (18.75%)	5801 (69.17%)	678 (67.33%)
7	25 (25%)	5663 (65.01%)	649 (48%)
8	25 (25%)	5794 (68.96%)	624 (31.33%)
9	26 (31.25%)	4945 (43.37%)	642 (43.33%)
10	26 (31.25%)	5643 (64.41%)	599 (14.67%)
11	27 (37.5%)	4275 (23.18%)	646 (46%)
12	27 (37.5%)	4326 (24.71%)	645 (45.33%)
13	27 (37.5%)	5641 (64.35%)	598 (14%)
14	28 (43.75%)	4268 (22.97%)	635 (38.67%)
15	28 (43.75%)	4473 (29.14%)	585 (5.33%)
16	29 (50%)	4110 (18.2%)	635 (38.67%)
17	30 (56.25%)	4587 (32.58%)	580 (2%)
18	31 (62.5%)	4339 (25.11%)	595 (12%)
19	32 (68.75%)	3834 (9.89%)	645 (45.33%)
20	32 (68.75%)	3856 (10.55%)	635 (38.67%)
21	32 (68.75%)	4282 (23.39%)	599 (14.67%)
22	33 (75%)	3741 (7.08%)	618 (27.33%)
23	33 (75%)	3827 (9.67%)	591 (9.33%)
24	34 (81.25%)	3780 (8.26%)	615 (25.33%)
25	35 (87.5%)	3693 (5.64%)	577 (0%)
26	36 (93.75%)	3511 (0.15%)	618 (27.33%)
27	36 (93.75%)	3578 (2.17%)	589 (8%)
28	37 (100%)	3506 (0%)	632 (36.67%)

Table A.27: Experimental results for instance 9-C (Heuristic Approach 3 - Delayed-TSPTW-3)

Runtime (sec)	$\mathbf{LB_{f_1}}$	$\mathbf{LB_{f_2}}$	**Reference value for** $\mathbf{f_3}$
193.65	25	3319	629
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	25 (0%)	5459 (67.52%)	699 (100%)
2	25 (0%)	5543 (76.5%)	693 (86.36%)
3	25 (0%)	5763 (100%)	683 (63.64%)
4	26 (25%)	5257 (45.94%)	692 (84.09%)
5	26 (25%)	5398 (61%)	674 (43.18%)
6	27 (50%)	4859 (3.42%)	683 (63.64%)
7	27 (50%)	5013 (19.87%)	660 (11.36%)
8	27 (50%)	5017 (20.3%)	655 (0%)
9	28 (75%)	4827 (0%)	676 (47.73%)
10	29 (100%)	4831 (0.43%)	675 (45.45%)

Appendix B

Experimental Results of Heuristic Approach 3 for Problem 3 (SCD-VRPTW-3)

Table B.1: Experimental results for instance 1 - rc201-A1 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
26.6	0	0	2609.92
Solution S/N	**f_1**	**f_2**	**f_3**
1	0 (0%)	0 (0%)	2565.82 (100%)
2	1 (20%)	53.96 (25.56%)	2560.49 (85.3%)
3	2 (40%)	77.89 (36.89%)	2546.16 (45.78%)
4	3 (60%)	151.75 (71.88%)	2533.65 (11.28%)
5	4 (80%)	198.6 (94.07%)	2532.8 (8.94%)
6	5 (100%)	211.12 (100%)	2529.56 (0%)

Table B.2: Experimental results for instance 2 - rc201-A2 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
26.2	0	0	2609.92
Solution S/N	**f_1**	**f_2**	**f_3**
1	0 (0%)	0 (0%)	2675.6 (100%)
2	1 (14.29%)	53.96 (25.29%)	2665.17 (83.09%)

Continued on the next page

Table B.2 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
3	1 (14.29%)	127.34 (59.69%)	2650.04 (58.55%)
4	2 (28.57%)	156.92 (73.55%)	2646.97 (53.58%)
5	3 (42.86%)	86.16 (40.38%)	2652.82 (63.06%)
6	3 (42.86%)	164.63 (77.16%)	2645.21 (50.72%)
7	3 (42.86%)	184.71 (86.58%)	2633.82 (32.25%)
8	4 (57.14%)	85 (39.84%)	2655.06 (66.69%)
9	4 (57.14%)	213.35 (100%)	2613.93 (0%)
10	5 (71.43%)	123.45 (57.86%)	2647.99 (55.23%)
11	6 (85.71%)	122.29 (57.32%)	2650.23 (58.86%)
12	6 (85.71%)	142.37 (66.73%)	2638.84 (40.39%)
13	6 (85.71%)	177.41 (83.15%)	2633.33 (31.46%)
14	6 (85.71%)	197.49 (92.57%)	2622.36 (13.67%)
15	7 (100%)	176.25 (82.61%)	2635.57 (35.09%)
16	7 (100%)	196.33 (92.02%)	2624.6 (17.3%)

Table B.3: Experimental results for instance 3 - rc201-A3 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ for $\mathbf{f_1}$	$\mathbf{LB'_2}$ for $\mathbf{f_2}$	Reference value for $\mathbf{f_3}$
13.4	5	172.09	2609.92
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	6 (0%)	307.2 (60.9%)	2896.97 (100%)
2	6 (0%)	319.74 (97.27%)	2874.41 (21.15%)
3	6 (0%)	320.68 (100%)	2868.36 (0%)
4	7 (100%)	286.2 (0%)	2893.7 (88.57%)
5	7 (100%)	298.74 (36.37%)	2871.14 (9.72%)

Table B.4: Experimental results for instance 4 - rc201-A4 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ for $\mathbf{f_1}$	$\mathbf{LB'_2}$ for $\mathbf{f_2}$	Reference value for $\mathbf{f_3}$
12.5	7	396.17	2609.92
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	8 (0%)	601.07 (0%)	3032.77 (100%)
2	8 (0%)	605.52 (81.65%)	2998.08 (21.89%)
3	8 (0%)	606.52 (100%)	2994.81 (14.52%)
4	9 (100%)	603.43 (43.3%)	2994.41 (13.62%)

Continued on the next page

Table B.4 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
5	9 (100%)	604.37 (60.55%)	2988.36 (0%)

Table B.5: Experimental results for instance 5 - rc201-A5 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
13.2	7	440.54	2609.92
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	8 (0%)	651.27 (0%)	3037.61 (100%)
2	8 (0%)	661.07 (2.72%)	3032.77 (90.26%)
3	8 (0%)	710.88 (16.57%)	2994.41 (13.04%)
4	8 (0%)	711.88 (16.85%)	2991.14 (6.46%)
5	9 (100%)	703.45 (14.5%)	3016.97 (58.45%)
6	9 (100%)	704.45 (14.78%)	3013.7 (51.87%)
7	9 (100%)	1011.03 (100%)	2987.93 (0%)

Table B.6: Experimental results for instance 6 - rc201-A6 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
13.4	7	440.54	2609.92
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	8 (0%)	651.27 (0%)	3037.61 (100%)
2	8 (0%)	661.07 (16.18%)	3032.77 (90.17%)
3	8 (0%)	710.88 (98.45%)	2994.41 (12.28%)
4	8 (0%)	711.82 (100%)	2988.36 (0%)
5	9 (100%)	703.45 (86.18%)	3016.97 (58.09%)
6	9 (100%)	704.45 (87.83%)	3013.7 (51.45%)

Table B.7: Experimental results for instance 7 - rc201-A7 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
10.5	7	440.54	2609.92
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	8 (NaN%)	651.27 (0%)	3037.61 (100%)

Continued on the next page

Table B.7 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
2	8 (NaN%)	661.07 (100%)	3032.77 (0%)

Table B.8: Experimental results for instance 8 - rc201-A8 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
13	7	440.54	2609.92
Solution S/N	**f_1**	**f_2**	**f_3**
1	8 (0%)	651.27 (0%)	3037.61 (100%)
2	8 (0%)	661.07 (2.72%)	3032.77 (90.26%)
3	8 (0%)	710.88 (16.57%)	2994.41 (13.04%)
4	8 (0%)	711.88 (16.85%)	2991.14 (6.46%)
5	9 (100%)	703.45 (14.5%)	3016.97 (58.45%)
6	9 (100%)	704.45 (14.78%)	3013.7 (51.87%)
7	9 (100%)	1011.03 (100%)	2987.93 (0%)

Table B.9: Experimental results for instance 9 - rc201-B1 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
3747.3	0	0	3358.42
Solution S/N	**f_1**	**f_2**	**f_3**
1	0 (0%)	0 (0%)	3251.37 (100%)
2	2 (2.56%)	42.86 (0.22%)	3198.74 (95.71%)
3	3 (3.85%)	42.18 (0.22%)	3199.42 (95.76%)
4	4 (5.13%)	42.81 (0.22%)	3198.92 (95.72%)
5	17 (21.79%)	1307.03 (6.82%)	3005.07 (79.92%)
6	18 (23.08%)	1266.05 (6.61%)	3005.07 (79.92%)
7	18 (23.08%)	1328.1 (6.93%)	3004.2 (79.85%)
8	18 (23.08%)	1330.32 (6.94%)	3003.09 (79.75%)
9	18 (23.08%)	1857.29 (9.69%)	2999.58 (79.47%)
10	20 (25.64%)	1909.02 (9.96%)	2999.4 (79.45%)
11	21 (26.92%)	1035.44 (5.4%)	3114.66 (88.85%)
12	21 (26.92%)	1440.3 (7.52%)	2984.62 (78.25%)
13	22 (28.21%)	1131.97 (5.91%)	3002.68 (79.72%)
14	22 (28.21%)	1891.67 (9.87%)	2983.54 (78.16%)
15	23 (29.49%)	1479.44 (7.72%)	2984.44 (78.23%)

Continued on the next page

Table B.9 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
16	23 (29.49%)	1504.8 (7.85%)	2971.92 (77.21%)
17	23 (29.49%)	2041.28 (10.65%)	2971.78 (77.2%)
18	24 (30.77%)	4414.86 (23.04%)	2910.35 (72.19%)
19	25 (32.05%)	2820.71 (14.72%)	2913.13 (72.42%)
20	25 (32.05%)	4044.18 (21.1%)	2910.35 (72.19%)
21	25 (32.05%)	4418.31 (23.05%)	2910.24 (72.18%)
22	26 (33.33%)	1090.1 (5.69%)	2988.23 (78.54%)
23	26 (33.33%)	2887.17 (15.06%)	2908.37 (72.03%)
24	26 (33.33%)	2895.5 (15.11%)	2899.9 (71.34%)
25	26 (33.33%)	3935.9 (20.54%)	2898.37 (71.22%)
26	28 (35.9%)	3921.36 (20.46%)	2898.37 (71.22%)
27	29 (37.18%)	1380.54 (7.2%)	2865.26 (68.52%)
28	30 (38.46%)	1318.46 (6.88%)	2883.31 (69.99%)
29	30 (38.46%)	1366.2 (7.13%)	2867.87 (68.73%)
30	31 (39.74%)	1310.08 (6.84%)	2883.31 (69.99%)
31	31 (39.74%)	1377.16 (7.19%)	2859.46 (68.04%)
32	33 (42.31%)	1969.44 (10.28%)	2852.63 (67.49%)
33	35 (44.87%)	2608.63 (13.61%)	2847.71 (67.08%)
34	37 (47.44%)	2568.81 (13.4%)	2842.2 (66.64%)
35	39 (50%)	2336.98 (12.19%)	2842.2 (66.64%)
36	43 (55.13%)	4848.73 (25.3%)	2763.47 (60.22%)
37	47 (60.26%)	5859.01 (30.57%)	2744.83 (58.7%)
38	48 (61.54%)	6596.19 (34.42%)	2535.25 (41.61%)
39	49 (62.82%)	6537.3 (34.11%)	2534.42 (41.54%)
40	51 (65.38%)	6383.3 (33.31%)	2744.83 (58.7%)
41	51 (65.38%)	6524.1 (34.04%)	2536.34 (41.69%)
42	52 (66.67%)	6885.32 (35.93%)	2533.49 (41.46%)
43	53 (67.95%)	4896.58 (25.55%)	2728.67 (57.38%)
44	54 (69.23%)	5372.81 (28.03%)	2696.71 (54.77%)
45	54 (69.23%)	5752.42 (30.01%)	2692.08 (54.39%)
46	54 (69.23%)	7658 (39.96%)	2533.01 (41.42%)
47	55 (70.51%)	5012.43 (26.15%)	2719.82 (56.66%)
48	56 (71.79%)	5038.85 (26.29%)	2715.66 (56.32%)
49	56 (71.79%)	10078.38 (52.59%)	2459.45 (35.42%)
50	57 (73.08%)	10125.62 (52.83%)	2452.72 (34.88%)
51	58 (74.36%)	10228.62 (53.37%)	2452.2 (34.83%)
52	63 (80.77%)	16487.17 (86.03%)	2438.38 (33.71%)
53	64 (82.05%)	16703.16 (87.15%)	2423.82 (32.52%)

Continued on the next page

Table B.9 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
54	66 (84.62%)	15950.46 (83.22%)	2319.51 (24.01%)
55	67 (85.9%)	16910.82 (88.24%)	2280.82 (20.86%)
56	67 (85.9%)	16946.52 (88.42%)	2279.68 (20.77%)
57	67 (85.9%)	16964.21 (88.51%)	2279.33 (20.74%)
58	68 (87.18%)	17109.24 (89.27%)	2278.08 (20.64%)
59	68 (87.18%)	17127.27 (89.37%)	2274.89 (20.38%)
60	68 (87.18%)	17190.65 (89.7%)	2196.41 (13.98%)
61	68 (87.18%)	18277.57 (95.37%)	2192.67 (13.67%)
62	69 (88.46%)	15127.89 (78.93%)	2401.03 (30.66%)
63	69 (88.46%)	16875.14 (88.05%)	2274.9 (20.38%)
64	69 (88.46%)	17080.13 (89.12%)	2274.46 (20.34%)
65	69 (88.46%)	17261.13 (90.06%)	2195.8 (13.93%)
66	69 (88.46%)	18258.13 (95.27%)	2192.66 (13.67%)
67	70 (89.74%)	14729.14 (76.85%)	2442.21 (34.02%)
68	70 (89.74%)	15769.93 (82.28%)	2222.26 (16.08%)
69	70 (89.74%)	15808.48 (82.48%)	2211.29 (15.19%)
70	70 (89.74%)	16984.29 (88.62%)	2127.57 (8.36%)
71	71 (91.03%)	15584.91 (81.32%)	2234.7 (17.1%)
72	71 (91.03%)	16908.15 (88.22%)	2127.57 (8.36%)
73	72 (92.31%)	16853.22 (87.94%)	2127.57 (8.36%)
74	73 (93.59%)	16578.76 (86.5%)	2130.74 (8.62%)
75	73 (93.59%)	16632.99 (86.79%)	2119.77 (7.73%)
76	73 (93.59%)	16808.08 (87.7%)	2118.71 (7.64%)
77	73 (93.59%)	16848.34 (87.91%)	2117.03 (7.5%)
78	73 (93.59%)	19165.47 (100%)	2028.06 (0.25%)
79	74 (94.87%)	18996.34 (99.12%)	2029.24 (0.34%)
80	75 (96.15%)	19099.63 (99.66%)	2025.02 (0%)
81	78 (100%)	18902.97 (98.63%)	2115.7 (7.39%)

Table B.10: Experimental results for instance 10 - rc201-B2 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
3173.7	0	0	3358.42
Solution S/N	**f_1**	**f_2**	**f_3**
1	4 (0%)	504.6 (2.45%)	3315.79 (100%)
2	5 (1.54%)	337.98 (1.4%)	3312.94 (99.72%)
3	5 (1.54%)	462.23 (2.18%)	3305.53 (99.01%)

Continued on the next page

Table B.10 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
4	6 (3.08%)	332.55 (1.37%)	3311.78 (99.61%)
5	6 (3.08%)	345.18 (1.45%)	3308.68 (99.31%)
6	6 (3.08%)	363.94 (1.57%)	3297.75 (98.26%)
7	6 (3.08%)	598.45 (3.03%)	3293.65 (97.86%)
8	6 (3.08%)	652.96 (3.38%)	3290.46 (97.55%)
9	8 (6.15%)	113.82 (0%)	3281.94 (96.73%)
10	10 (9.23%)	2227.91 (13.24%)	3235.2 (92.21%)
11	10 (9.23%)	2228.41 (13.24%)	3235.05 (92.19%)
12	10 (9.23%)	2246.24 (13.35%)	3230.44 (91.75%)
13	11 (10.77%)	1572.76 (9.14%)	3277.32 (96.28%)
14	11 (10.77%)	1613.34 (9.39%)	3274.03 (95.96%)
15	11 (10.77%)	1702.45 (9.95%)	3254.21 (94.05%)
16	11 (10.77%)	2352.43 (14.02%)	3228.98 (91.61%)
17	12 (12.31%)	2038 (12.05%)	3240.91 (92.76%)
18	12 (12.31%)	2431.32 (14.51%)	3226.21 (91.34%)
19	13 (13.85%)	1578.77 (9.17%)	3272.46 (95.81%)
20	13 (13.85%)	2426 (14.48%)	3226.3 (91.35%)
21	19 (23.08%)	1596.77 (9.29%)	3270.56 (95.63%)
22	25 (32.31%)	3852.64 (23.41%)	3181.79 (87.05%)
23	26 (33.85%)	3577.04 (21.69%)	3193.5 (88.18%)
24	26 (33.85%)	4088.65 (24.89%)	3177.78 (86.66%)
25	27 (35.38%)	4758.81 (29.09%)	3166.81 (85.6%)
26	28 (36.92%)	2787.65 (16.74%)	3165.82 (85.5%)
27	28 (36.92%)	2788 (16.75%)	3165.82 (85.5%)
28	29 (38.46%)	2953.46 (17.78%)	3085.89 (77.77%)
29	30 (40%)	2255.52 (13.41%)	3112.52 (80.35%)
30	30 (40%)	2795.23 (16.79%)	3095.15 (78.67%)
31	30 (40%)	2827.02 (16.99%)	3085.89 (77.77%)
32	31 (41.54%)	2318.1 (13.8%)	3111.84 (80.28%)
33	31 (41.54%)	2330.8 (13.88%)	3110.13 (80.12%)
34	31 (41.54%)	4641.08 (28.35%)	3079.48 (77.15%)
35	31 (41.54%)	4834.18 (29.56%)	3062.15 (75.48%)
36	31 (41.54%)	6033.06 (37.07%)	2959.53 (65.56%)
37	32 (43.08%)	2766.39 (16.61%)	3095.15 (78.67%)
38	32 (43.08%)	5648.6 (34.66%)	3046.81 (74%)
39	32 (43.08%)	6883.91 (42.4%)	2957.2 (65.33%)
40	33 (44.62%)	2254.04 (13.4%)	3112.52 (80.35%)
41	33 (44.62%)	5537.7 (33.97%)	3052.94 (74.59%)

Continued on the next page

Table B.10 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
42	33 (44.62%)	6255.85 (38.46%)	2957.83 (65.39%)
43	33 (44.62%)	6893.42 (42.46%)	2955.68 (65.19%)
44	34 (46.15%)	6597.14 (40.6%)	2955.96 (65.21%)
45	46 (64.62%)	6469.8 (39.8%)	2703.15 (40.77%)
46	47 (66.15%)	6771.29 (41.69%)	2702.96 (40.76%)
47	48 (67.69%)	6545.16 (40.27%)	2691.26 (39.62%)
48	50 (70.77%)	6665.81 (41.03%)	2620.59 (32.79%)
49	50 (70.77%)	6678.82 (41.11%)	2607.58 (31.53%)
50	58 (83.08%)	15201.86 (94.48%)	2467.86 (18.03%)
51	59 (84.62%)	15317.32 (95.21%)	2466.94 (17.94%)
52	60 (86.15%)	15475.93 (96.2%)	2353.47 (6.97%)
53	61 (87.69%)	15291.96 (95.05%)	2455.68 (16.85%)
54	61 (87.69%)	15448.39 (96.03%)	2354.71 (7.09%)
55	62 (89.23%)	15556.63 (96.71%)	2344.56 (6.11%)
56	63 (90.77%)	11247.11 (69.72%)	2480.18 (19.22%)
57	63 (90.77%)	13517.6 (83.94%)	2360.21 (7.62%)
58	63 (90.77%)	13539.11 (84.07%)	2357.82 (7.39%)
59	64 (92.31%)	13265.59 (82.36%)	2364.77 (8.06%)
60	64 (92.31%)	13397.72 (83.19%)	2362.26 (7.82%)
61	64 (92.31%)	15938 (99.09%)	2309.79 (2.75%)
62	65 (93.85%)	12661.36 (78.58%)	2478.79 (19.08%)
63	65 (93.85%)	14509.78 (90.15%)	2351.32 (6.76%)
64	66 (95.38%)	14651.95 (91.04%)	2349.42 (6.58%)
65	66 (95.38%)	14921.01 (92.73%)	2342.24 (5.88%)
66	66 (95.38%)	16082.58 (100%)	2289.23 (0.76%)
67	67 (96.92%)	15331.82 (95.3%)	2338.58 (5.53%)
68	67 (96.92%)	15556.73 (96.71%)	2335.11 (5.19%)
69	67 (96.92%)	15560.73 (96.73%)	2334.31 (5.12%)
70	67 (96.92%)	15580.68 (96.86%)	2329.32 (4.63%)
71	67 (96.92%)	15591.75 (96.93%)	2328.66 (4.57%)
72	67 (96.92%)	15607.43 (97.02%)	2326.62 (4.37%)
73	67 (96.92%)	15614.33 (97.07%)	2326.12 (4.32%)
74	67 (96.92%)	15939.11 (99.1%)	2289.62 (0.8%)
75	67 (96.92%)	15968.73 (99.29%)	2281.39 (0%)
76	68 (98.46%)	15389.11 (95.66%)	2335.18 (5.2%)
77	68 (98.46%)	15622.46 (97.12%)	2321.03 (3.83%)
78	68 (98.46%)	15659.62 (97.35%)	2319.52 (3.69%)
79	68 (98.46%)	15778.68 (98.1%)	2316.95 (3.44%)

Continued on the next page

Table B.10 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
80	68 (98.46%)	15878.85 (98.72%)	2311.34 (2.9%)
81	69 (100%)	13236.8 (82.18%)	2475.58 (18.77%)

Table B.11: Experimental results for instance 11 - rc201-B3 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
3451.5	0	0	3358.42
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	3 (0%)	39.8 (0%)	3285.71 (100%)
2	4 (1.35%)	156.72 (0.63%)	3274.52 (99.14%)
3	9 (8.11%)	1478.06 (7.81%)	3259.73 (98.01%)
4	10 (9.46%)	1741.47 (9.24%)	3195.37 (93.08%)
5	11 (10.81%)	463.52 (2.3%)	3192.17 (92.84%)
6	11 (10.81%)	1418.65 (7.49%)	3145.64 (89.28%)
7	12 (12.16%)	234.64 (1.06%)	3206.63 (93.95%)
8	12 (12.16%)	2215.36 (11.81%)	3144.24 (89.17%)
9	13 (13.51%)	243.09 (1.1%)	3179.35 (91.86%)
10	13 (13.51%)	243.19 (1.1%)	3179.25 (91.85%)
11	13 (13.51%)	245.56 (1.12%)	3179.1 (91.84%)
12	13 (13.51%)	279.82 (1.3%)	3160.52 (90.41%)
13	13 (13.51%)	280.38 (1.31%)	3159.96 (90.37%)
14	13 (13.51%)	281.43 (1.31%)	3158.91 (90.29%)
15	13 (13.51%)	281.77 (1.31%)	3158.57 (90.27%)
16	13 (13.51%)	2134.21 (11.37%)	3144.54 (89.19%)
17	14 (14.86%)	284.96 (1.33%)	3158.52 (90.26%)
18	25 (29.73%)	2759.93 (14.77%)	3124.62 (87.67%)
19	26 (31.08%)	2743.05 (14.68%)	3114.57 (86.9%)
20	32 (39.19%)	1266.97 (6.66%)	2993.7 (77.64%)
21	33 (40.54%)	1397.52 (7.37%)	2976.55 (76.33%)
22	33 (40.54%)	2913.71 (15.6%)	2794.97 (62.43%)
23	34 (41.89%)	1304.68 (6.87%)	2912.54 (71.43%)
24	34 (41.89%)	1324.1 (6.97%)	2897.87 (70.31%)
25	34 (41.89%)	2909.6 (15.58%)	2795.06 (62.43%)
26	34 (41.89%)	2912.63 (15.6%)	2794.73 (62.41%)
27	34 (41.89%)	2945.11 (15.78%)	2794.5 (62.39%)
28	34 (41.89%)	3592.39 (19.29%)	2792.78 (62.26%)
29	35 (43.24%)	1301.16 (6.85%)	2912.74 (71.44%)

Continued on the next page

Table B.11 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
30	35 (43.24%)	1304.16 (6.87%)	2896.45 (70.2%)
31	35 (43.24%)	1323.8 (6.97%)	2895.51 (70.12%)
32	35 (43.24%)	2909.4 (15.58%)	2795.06 (62.43%)
33	35 (43.24%)	2939.46 (15.74%)	2793.75 (62.33%)
34	35 (43.24%)	6442.17 (34.76%)	2763.58 (60.02%)
35	36 (44.59%)	5891.29 (31.77%)	2757.28 (59.54%)
36	40 (50%)	2448.63 (13.08%)	2865.68 (67.84%)
37	41 (51.35%)	2527.88 (13.51%)	2863.07 (67.64%)
38	42 (52.7%)	2374.84 (12.68%)	2863.46 (67.67%)
39	46 (58.11%)	2133.47 (11.37%)	2772.38 (60.7%)
40	47 (59.46%)	1917.37 (10.19%)	2786.29 (61.76%)
41	47 (59.46%)	1929.32 (10.26%)	2785.3 (61.69%)
42	47 (59.46%)	1985.08 (10.56%)	2775.47 (60.93%)
43	47 (59.46%)	2042.1 (10.87%)	2774.9 (60.89%)
44	48 (60.81%)	2041.75 (10.87%)	2767.95 (60.36%)
45	48 (60.81%)	6109.45 (32.96%)	2656.9 (51.86%)
46	49 (62.16%)	2012.22 (10.71%)	2765.74 (60.19%)
47	49 (62.16%)	6140.15 (33.12%)	2550.6 (43.72%)
48	49 (62.16%)	6176.99 (33.32%)	2549.61 (43.64%)
49	49 (62.16%)	6183.73 (33.36%)	2549.48 (43.63%)
50	50 (63.51%)	2002.47 (10.66%)	2766.73 (60.26%)
51	50 (63.51%)	2031.22 (10.81%)	2764.75 (60.11%)
52	50 (63.51%)	2653.18 (14.19%)	2763.86 (60.04%)
53	50 (63.51%)	2720.03 (14.55%)	2755.17 (59.38%)
54	50 (63.51%)	6183.02 (33.36%)	2548.06 (43.52%)
55	51 (64.86%)	3069.76 (16.45%)	2716.89 (56.45%)
56	51 (64.86%)	6149.78 (33.18%)	2549.93 (43.67%)
57	51 (64.86%)	6150.74 (33.18%)	2549.45 (43.63%)
58	51 (64.86%)	6315.67 (34.08%)	2547.76 (43.5%)
59	51 (64.86%)	6566.76 (35.44%)	2545.74 (43.34%)
60	51 (64.86%)	6573.37 (35.48%)	2545.61 (43.33%)
61	51 (64.86%)	6682.53 (36.07%)	2533.19 (42.38%)
62	51 (64.86%)	6688.84 (36.1%)	2533.06 (42.37%)
63	51 (64.86%)	6815.43 (36.79%)	2531.58 (42.26%)
64	51 (64.86%)	6841.79 (36.93%)	2528.15 (42%)
65	52 (66.22%)	3728.43 (20.03%)	2674.48 (53.2%)
66	52 (66.22%)	8081.47 (43.66%)	2502.84 (40.06%)
67	52 (66.22%)	9577.41 (51.79%)	2446.1 (35.72%)

Continued on the next page

Table B.11 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
68	53 (67.57%)	3904.55 (20.98%)	2652.52 (51.52%)
69	53 (67.57%)	4672.05 (25.15%)	2381.01 (30.73%)
70	54 (68.92%)	4169.08 (22.42%)	2422.45 (33.9%)
71	54 (68.92%)	4181.54 (22.49%)	2412.13 (33.11%)
72	55 (70.27%)	3955.56 (21.26%)	2503.03 (40.07%)
73	55 (70.27%)	4253.21 (22.88%)	2397.72 (32.01%)
74	55 (70.27%)	4310.86 (23.19%)	2392.28 (31.59%)
75	55 (70.27%)	4631.75 (24.93%)	2390.23 (31.44%)
76	56 (71.62%)	4135.96 (22.24%)	2409.82 (32.94%)
77	56 (71.62%)	4243.18 (22.82%)	2402.53 (32.38%)
78	56 (71.62%)	4248.37 (22.85%)	2397.72 (32.01%)
79	56 (71.62%)	5370.65 (28.95%)	2370.91 (29.96%)
80	56 (71.62%)	5781.26 (31.18%)	2359.29 (29.07%)
81	57 (72.97%)	4167.01 (22.41%)	2409.55 (32.92%)
82	57 (72.97%)	5265.93 (28.38%)	2379.71 (30.63%)
83	58 (74.32%)	4166.46 (22.41%)	2407.11 (32.73%)
84	58 (74.32%)	4190.53 (22.54%)	2406.84 (32.71%)
85	58 (74.32%)	4201.94 (22.6%)	2406.15 (32.66%)
86	58 (74.32%)	4667.88 (25.13%)	2387.24 (31.21%)
87	58 (74.32%)	5214.46 (28.1%)	2378.51 (30.54%)
88	59 (75.68%)	3845.53 (20.66%)	2378.94 (30.57%)
89	59 (75.68%)	4244.35 (22.83%)	2377.69 (30.48%)
90	59 (75.68%)	5166.58 (27.84%)	2369.47 (29.85%)
91	60 (77.03%)	3855.64 (20.72%)	2378.62 (30.55%)
92	61 (78.38%)	3843.74 (20.65%)	2380.57 (30.7%)
93	62 (79.73%)	4152.02 (22.33%)	2373.47 (30.15%)
94	62 (79.73%)	4379.59 (23.56%)	2362.29 (29.3%)
95	62 (79.73%)	4853.27 (26.14%)	2349.64 (28.33%)
96	62 (79.73%)	5057.08 (27.24%)	2337.77 (27.42%)
97	62 (79.73%)	5507.08 (29.69%)	2327.88 (26.66%)
98	63 (81.08%)	6306.98 (34.03%)	2318.95 (25.98%)
99	63 (81.08%)	6674.82 (36.03%)	2309.1 (25.23%)
100	63 (81.08%)	6884.48 (37.17%)	2308.59 (25.19%)
101	65 (83.78%)	3927.38 (21.11%)	2378.37 (30.53%)
102	66 (85.14%)	11569.08 (62.6%)	2224.65 (18.76%)
103	67 (86.49%)	10123.31 (54.75%)	2305.73 (24.97%)
104	67 (86.49%)	10131.33 (54.8%)	2265.55 (21.89%)
105	67 (86.49%)	11314.19 (61.22%)	2250.78 (20.76%)

Continued on the next page

Table B.11 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
106	68 (87.84%)	9403.15 (50.84%)	2271.52 (22.35%)
107	68 (87.84%)	10965.69 (59.33%)	2258.62 (21.36%)
108	68 (87.84%)	12537 (67.86%)	2163.91 (14.11%)
109	69 (89.19%)	10058.38 (54.4%)	2268.65 (22.13%)
110	69 (89.19%)	11235.58 (60.79%)	2235.3 (19.58%)
111	69 (89.19%)	11359.54 (61.46%)	2201.11 (16.96%)
112	69 (89.19%)	14037.57 (76.01%)	2135.97 (11.97%)
113	70 (90.54%)	10069.9 (54.46%)	2254.49 (21.04%)
114	70 (90.54%)	10446.43 (56.51%)	2233.02 (19.4%)
115	70 (90.54%)	10570.15 (57.18%)	2202.77 (17.08%)
116	70 (90.54%)	13145.61 (71.16%)	2147.67 (12.87%)
117	70 (90.54%)	13156.37 (71.22%)	2146.04 (12.74%)
118	70 (90.54%)	13559.82 (73.41%)	2144.52 (12.62%)
119	70 (90.54%)	14469.01 (78.35%)	2131.09 (11.6%)
120	70 (90.54%)	15671.85 (84.88%)	2093.41 (8.71%)
121	71 (91.89%)	9990.67 (54.03%)	2254.49 (21.04%)
122	71 (91.89%)	15127.97 (81.93%)	2127.81 (11.35%)
123	71 (91.89%)	17234.89 (93.37%)	2068.23 (6.78%)
124	71 (91.89%)	17292.4 (93.68%)	2067.49 (6.73%)
125	72 (93.24%)	16063.35 (87.01%)	2088.81 (8.36%)
126	72 (93.24%)	17336.61 (93.92%)	2064.19 (6.47%)
127	72 (93.24%)	17463.43 (94.61%)	2063.16 (6.4%)
128	73 (94.59%)	16981.72 (91.99%)	2007.35 (2.12%)
129	74 (95.95%)	15218.49 (82.42%)	2049.73 (5.37%)
130	74 (95.95%)	15331.63 (83.03%)	2049.07 (5.32%)
131	75 (97.3%)	15107.47 (81.81%)	1996.61 (1.3%)
132	75 (97.3%)	15475.2 (83.81%)	1996.48 (1.29%)
133	75 (97.3%)	16044.7 (86.9%)	1993.24 (1.04%)
134	75 (97.3%)	16787.13 (90.94%)	1990.41 (0.83%)
135	76 (98.65%)	15018.27 (81.33%)	1997.92 (1.4%)
136	76 (98.65%)	15126.91 (81.92%)	1996.02 (1.25%)
137	76 (98.65%)	15140.29 (81.99%)	1995.76 (1.23%)
138	76 (98.65%)	15448.24 (83.67%)	1993.35 (1.05%)
139	76 (98.65%)	17761.95 (96.23%)	1990.16 (0.81%)
140	76 (98.65%)	18456.59 (100%)	1984.37 (0.36%)
141	77 (100%)	18398.14 (99.68%)	1979.63 (0%)

Table B.12: Experimental results for instance 12 - rc201-B4 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
2563	0	0	3358.42
Solution S/N	**f_1**	**f_2**	**f_3**
1	3 (0%)	40.42 (0%)	3708.07 (100%)
2	4 (1.27%)	244.96 (1.09%)	3312.11 (73.14%)
3	4 (1.27%)	1025 (5.23%)	3248.52 (68.82%)
4	5 (2.53%)	226.37 (0.99%)	3307.59 (72.83%)
5	6 (3.8%)	182.1 (0.75%)	3315.06 (73.34%)
6	6 (3.8%)	187.62 (0.78%)	3314.74 (73.31%)
7	6 (3.8%)	222.22 (0.97%)	3314.23 (73.28%)
8	6 (3.8%)	236.99 (1.04%)	3279.01 (70.89%)
9	6 (3.8%)	910.97 (4.63%)	3248.52 (68.82%)
10	7 (5.06%)	249.17 (1.11%)	3277.67 (70.8%)
11	7 (5.06%)	638.62 (3.18%)	3256.07 (69.33%)
12	8 (6.33%)	180.58 (0.74%)	3272.1 (70.42%)
13	8 (6.33%)	508.87 (2.49%)	3256.07 (69.33%)
14	9 (7.59%)	199.53 (0.85%)	3234.07 (67.84%)
15	15 (15.19%)	1197.75 (6.15%)	3160.36 (62.84%)
16	19 (20.25%)	1727.26 (8.96%)	3155.15 (62.49%)
17	24 (26.58%)	1374.56 (7.09%)	3026.19 (53.74%)
18	28 (31.65%)	5868.22 (30.97%)	2935.89 (47.61%)
19	29 (32.91%)	1583.62 (8.2%)	2941.99 (48.02%)
20	29 (32.91%)	1585.4 (8.21%)	2939.22 (47.84%)
21	30 (34.18%)	1638.55 (8.49%)	2935.15 (47.56%)
22	30 (34.18%)	5586.6 (29.47%)	2930.27 (47.23%)
23	30 (34.18%)	6025.94 (31.81%)	2924.76 (46.86%)
24	52 (62.03%)	6047.23 (31.92%)	2911.12 (45.93%)
25	53 (63.29%)	5674.4 (29.94%)	2846.92 (41.57%)
26	53 (63.29%)	6383.03 (33.71%)	2706.16 (32.02%)
27	54 (64.56%)	5625.88 (29.68%)	2846.92 (41.57%)
28	54 (64.56%)	6211.87 (32.8%)	2706.16 (32.02%)
29	55 (65.82%)	6088.53 (32.14%)	2706.16 (32.02%)
30	55 (65.82%)	10831.6 (57.35%)	2696.42 (31.36%)
31	55 (65.82%)	11879.47 (62.92%)	2695.96 (31.33%)
32	56 (67.09%)	5895.5 (31.12%)	2716.68 (32.74%)
33	56 (67.09%)	6110.23 (32.26%)	2704.63 (31.92%)
34	56 (67.09%)	10722.99 (56.77%)	2701.11 (31.68%)

Continued on the next page

Table B.12 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
35	56 (67.09%)	10814.64 (57.26%)	2694.87 (31.26%)
36	56 (67.09%)	10836.26 (57.37%)	2694.48 (31.23%)
37	56 (67.09%)	11719.73 (62.07%)	2634.29 (27.15%)
38	57 (68.35%)	11659.82 (61.75%)	2635.03 (27.2%)
39	57 (68.35%)	11693.71 (61.93%)	2630.78 (26.91%)
40	58 (69.62%)	10176.01 (53.86%)	2703.43 (31.84%)
41	58 (69.62%)	10629.88 (56.28%)	2702.89 (31.8%)
42	67 (81.01%)	9388.86 (49.68%)	2696.37 (31.36%)
43	69 (83.54%)	10894.45 (57.68%)	2603.26 (25.04%)
44	70 (84.81%)	10478.37 (55.47%)	2609.07 (25.44%)
45	73 (88.61%)	13178.45 (69.82%)	2499.32 (17.99%)
46	73 (88.61%)	13280.31 (70.36%)	2484.76 (17%)
47	73 (88.61%)	15257.46 (80.87%)	2327.6 (6.34%)
48	73 (88.61%)	15520.96 (82.27%)	2313.01 (5.35%)
49	74 (89.87%)	9270.64 (49.05%)	2629.22 (26.8%)
50	74 (89.87%)	16466.58 (87.29%)	2305.57 (4.85%)
51	75 (91.14%)	9344.17 (49.44%)	2618.26 (26.06%)
52	75 (91.14%)	14134.07 (74.9%)	2418.72 (12.52%)
53	75 (91.14%)	15197.44 (80.55%)	2335.08 (6.85%)
54	75 (91.14%)	15615 (82.77%)	2300.73 (4.52%)
55	75 (91.14%)	16074.12 (85.21%)	2295.81 (4.18%)
56	76 (92.41%)	13877.45 (73.53%)	2457.97 (15.19%)
57	76 (92.41%)	14288.36 (75.72%)	2395.76 (10.97%)
58	77 (93.67%)	13473.52 (71.39%)	2453.31 (14.87%)
59	77 (93.67%)	13990.64 (74.14%)	2437.27 (13.78%)
60	77 (93.67%)	17461.02 (92.58%)	2274.61 (2.75%)
61	78 (94.94%)	15093.79 (80%)	2340.84 (7.24%)
62	79 (96.2%)	10065.62 (53.28%)	2617.13 (25.98%)
63	79 (96.2%)	11117.08 (58.87%)	2559.8 (22.09%)
64	80 (97.47%)	10690.56 (56.6%)	2536.41 (20.51%)
65	80 (97.47%)	10764.87 (56.99%)	2526.33 (19.82%)
66	80 (97.47%)	11168.77 (59.14%)	2525.78 (19.79%)
67	81 (98.73%)	11392.52 (60.33%)	2525.51 (19.77%)
68	81 (98.73%)	18857.43 (100%)	2234.14 (0%)
69	82 (100%)	10953.8 (58%)	2525.68 (19.78%)
70	82 (100%)	11682.15 (61.87%)	2521.94 (19.53%)

Table B.13: Experimental results for instance 13 - rc201-B5 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB$'_1$ for f$_1$**	**LB$'_2$ for f$_2$**	**Reference value for f$_3$**
2978.3	0	0	3358.42
Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
1	16 (0%)	2593.31 (7.43%)	3361.7 (98.59%)
2	19 (4.48%)	2347.93 (5.72%)	3374.91 (100%)
3	19 (4.48%)	2567.05 (7.25%)	3352.21 (97.57%)
4	20 (5.97%)	2557.32 (7.18%)	3372.51 (99.74%)
5	21 (7.46%)	1795.87 (1.85%)	3348.92 (97.22%)
6	21 (7.46%)	1822 (2.04%)	3346.5 (96.96%)
7	21 (7.46%)	2082.66 (3.86%)	3327.89 (94.98%)
8	22 (8.96%)	2566.27 (7.24%)	3288.3 (90.75%)
9	22 (8.96%)	2590.42 (7.41%)	3267.55 (88.53%)
10	22 (8.96%)	2684.11 (8.07%)	3246.63 (86.29%)
11	22 (8.96%)	2885.4 (9.47%)	3244.23 (86.04%)
12	23 (10.45%)	1977.27 (3.12%)	3318.53 (93.98%)
13	24 (11.94%)	1554.42 (0.17%)	3347.28 (97.05%)
14	24 (11.94%)	1951.36 (2.94%)	3319.03 (94.03%)
15	24 (11.94%)	1963.58 (3.03%)	3318.53 (93.98%)
16	24 (11.94%)	2023.35 (3.45%)	3307.07 (92.75%)
17	25 (13.43%)	1530.69 (0%)	3327.14 (94.9%)
18	25 (13.43%)	1844.03 (2.19%)	3306.91 (92.73%)
19	25 (13.43%)	1904.82 (2.62%)	3215.39 (82.96%)
20	27 (16.42%)	2062.71 (3.72%)	3210.8 (82.47%)
21	28 (17.91%)	1906.97 (2.63%)	3213.98 (82.81%)
22	29 (19.4%)	1911.27 (2.66%)	3213.49 (82.75%)
23	29 (19.4%)	1978.02 (3.13%)	3198.59 (81.16%)
24	36 (29.85%)	4744.98 (22.48%)	3122.9 (73.07%)
25	37 (31.34%)	4667.44 (21.94%)	3087.06 (69.24%)
26	39 (34.33%)	3336.52 (12.63%)	3096.99 (70.31%)
27	40 (35.82%)	3011.45 (10.36%)	3007.92 (60.79%)
28	41 (37.31%)	3188.99 (11.6%)	2988.97 (58.76%)
29	56 (59.7%)	8915.04 (51.64%)	2984.1 (58.24%)
30	56 (59.7%)	9432.52 (55.26%)	2931.82 (52.66%)
31	56 (59.7%)	9856.3 (58.23%)	2920.16 (51.41%)
32	57 (61.19%)	9448.43 (55.37%)	2925.77 (52.01%)
33	57 (61.19%)	9487.01 (55.64%)	2909.9 (50.32%)
34	58 (62.69%)	8793.31 (50.79%)	2961.2 (55.8%)

Continued on the next page

Table B.13 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
35	58 (62.69%)	8835.4 (51.09%)	2957.17 (55.37%)
36	58 (62.69%)	8970.37 (52.03%)	2943.23 (53.88%)
37	58 (62.69%)	11256.98 (68.02%)	2771.34 (35.51%)
38	59 (64.18%)	10864.39 (65.28%)	2773.53 (35.75%)
39	59 (64.18%)	11185.87 (67.52%)	2773.48 (35.74%)
40	60 (65.67%)	10800.8 (64.83%)	2773.53 (35.75%)
41	61 (67.16%)	9291.93 (54.28%)	2843.92 (43.27%)
42	61 (67.16%)	9801.34 (57.84%)	2757.77 (34.06%)
43	62 (68.66%)	9593.21 (56.39%)	2787.02 (37.19%)
44	62 (68.66%)	10085.9 (59.83%)	2756.47 (33.92%)
45	62 (68.66%)	10500.78 (62.73%)	2724.29 (30.48%)
46	62 (68.66%)	10930.21 (65.74%)	2720.97 (30.13%)
47	63 (70.15%)	9343.08 (54.64%)	2786.89 (37.17%)
48	63 (70.15%)	9343.82 (54.64%)	2786.52 (37.13%)
49	63 (70.15%)	9346.09 (54.66%)	2786.49 (37.13%)
50	63 (70.15%)	9417.17 (55.16%)	2785.1 (36.98%)
51	63 (70.15%)	11359.68 (68.74%)	2719.97 (30.02%)
52	63 (70.15%)	12053.19 (73.59%)	2717.81 (29.79%)
53	63 (70.15%)	12315.11 (75.42%)	2717.16 (29.72%)
54	63 (70.15%)	12987.6 (80.13%)	2714.08 (29.39%)
55	65 (73.13%)	12875.45 (79.34%)	2707.05 (28.64%)
56	66 (74.63%)	9532.91 (55.96%)	2784.27 (36.89%)
57	67 (76.12%)	8518.15 (48.87%)	2721.89 (30.23%)
58	68 (77.61%)	8076.59 (45.78%)	2748.47 (33.07%)
59	68 (77.61%)	8185.6 (46.54%)	2740.53 (32.22%)
60	68 (77.61%)	8310.1 (47.41%)	2721.89 (30.23%)
61	69 (79.1%)	7957.71 (44.95%)	2748.47 (33.07%)
62	70 (80.6%)	8131.09 (46.16%)	2739.61 (32.12%)
63	71 (82.09%)	7951.95 (44.91%)	2748.47 (33.07%)
64	71 (82.09%)	11474.13 (69.54%)	2670.23 (24.71%)
65	72 (83.58%)	8178.32 (46.49%)	2732.61 (31.37%)
66	72 (83.58%)	9070.1 (52.73%)	2707.73 (28.72%)
67	72 (83.58%)	10973.29 (66.04%)	2683.94 (26.17%)
68	73 (85.07%)	10671.81 (63.93%)	2683.58 (26.14%)
69	73 (85.07%)	11669.11 (70.9%)	2670.14 (24.7%)
70	73 (85.07%)	12977.61 (80.06%)	2545 (11.33%)
71	73 (85.07%)	13076.32 (80.75%)	2532.28 (9.97%)
72	74 (86.57%)	10080.47 (59.79%)	2671.28 (24.82%)

Continued on the next page

Table B.13 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
73	74 (86.57%)	12220.93 (74.76%)	2606.18 (17.87%)
74	74 (86.57%)	12732.76 (78.34%)	2600.01 (17.21%)
75	75 (88.06%)	10441.13 (62.32%)	2670.96 (24.79%)
76	75 (88.06%)	12677.59 (77.96%)	2589.03 (16.03%)
77	75 (88.06%)	12882.99 (79.39%)	2563.96 (13.35%)
78	75 (88.06%)	14420.94 (90.15%)	2465.63 (2.85%)
79	76 (89.55%)	12808.46 (78.87%)	2577.26 (14.78%)
80	76 (89.55%)	14522.48 (90.86%)	2464.77 (2.76%)
81	76 (89.55%)	14568.34 (91.18%)	2460.42 (2.29%)
82	76 (89.55%)	14658.08 (91.81%)	2458.27 (2.06%)
83	77 (91.04%)	14484.74 (90.6%)	2457.58 (1.99%)
84	78 (92.54%)	14614.3 (91.5%)	2456.88 (1.91%)
85	79 (94.03%)	14738.71 (92.37%)	2454.8 (1.69%)
86	79 (94.03%)	14953.01 (93.87%)	2453.32 (1.53%)
87	79 (94.03%)	15049.32 (94.54%)	2444.66 (0.61%)
88	79 (94.03%)	15123.31 (95.06%)	2440.46 (0.16%)
89	79 (94.03%)	15335.96 (96.55%)	2440.17 (0.13%)
90	80 (95.52%)	15165.87 (95.36%)	2439.72 (0.08%)
91	80 (95.52%)	15378.52 (96.85%)	2439.43 (0.05%)
92	80 (95.52%)	15445.33 (97.31%)	2439.39 (0.04%)
93	81 (97.01%)	15613.01 (98.49%)	2439.3 (0.04%)
94	83 (100%)	15666.1 (98.86%)	2439.2 (0.02%)
95	83 (100%)	15829.38 (100%)	2438.97 (0%)

Table B.14: Experimental results for instance 14 - rc201-B6 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
2466.1	14	855.33	3358.42
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	25 (0%)	4180.33 (9.58%)	3386.93 (97.93%)
2	26 (1.89%)	3540.19 (4.82%)	3396 (99.1%)
3	27 (3.77%)	3337.21 (3.31%)	3378.52 (96.85%)
4	28 (5.66%)	3183.46 (2.17%)	3353.81 (93.66%)
5	28 (5.66%)	3186.1 (2.19%)	3343.19 (92.29%)
6	31 (11.32%)	3130.62 (1.78%)	3357.73 (94.16%)
7	31 (11.32%)	3154.12 (1.95%)	3355.88 (93.93%)
8	31 (11.32%)	3183.91 (2.17%)	3342.49 (92.2%)

Continued on the next page

Table B.14 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
9	31 (11.32%)	3403.04 (3.8%)	3291.14 (85.58%)
10	31 (11.32%)	3432.63 (4.02%)	3290.54 (85.5%)
11	31 (11.32%)	3446.81 (4.13%)	3269.86 (82.83%)
12	31 (11.32%)	3734.89 (6.27%)	3248.58 (80.09%)
13	31 (11.32%)	4007.05 (8.29%)	3238.36 (78.77%)
14	32 (13.21%)	3106.24 (1.6%)	3338.36 (91.67%)
15	32 (13.21%)	3177.32 (2.12%)	3330.88 (90.7%)
16	32 (13.21%)	4004.87 (8.28%)	3236.75 (78.56%)
17	33 (15.09%)	3993.91 (8.2%)	3244.32 (79.54%)
18	33 (15.09%)	4794.22 (14.15%)	3228.12 (77.45%)
19	34 (16.98%)	2951.03 (0.44%)	3402.98 (100%)
20	34 (16.98%)	3283.42 (2.91%)	3317.23 (88.94%)
21	34 (16.98%)	4837.49 (14.47%)	3228.04 (77.44%)
22	34 (16.98%)	5385.31 (18.54%)	3223.01 (76.79%)
23	35 (18.87%)	2913.12 (0.16%)	3360.92 (94.58%)
24	35 (18.87%)	4985.23 (15.57%)	3227.32 (77.35%)
25	35 (18.87%)	5222.97 (17.34%)	3226.22 (77.21%)
26	35 (18.87%)	5233.84 (17.42%)	3224.81 (77.02%)
27	35 (18.87%)	5381.52 (18.52%)	3223.01 (76.79%)
28	36 (20.75%)	2896.16 (0.03%)	3375.94 (96.51%)
29	36 (20.75%)	2900.07 (0.06%)	3331.04 (90.72%)
30	36 (20.75%)	5217.57 (17.3%)	3226.22 (77.21%)
31	37 (22.64%)	2891.58 (0%)	3358.76 (94.3%)
32	37 (22.64%)	3623.89 (5.45%)	3236.58 (78.54%)
33	37 (22.64%)	3633.89 (5.52%)	3236.24 (78.5%)
34	37 (22.64%)	3899.84 (7.5%)	3223.61 (76.87%)
35	38 (24.53%)	3339.17 (3.33%)	3232.21 (77.98%)
36	38 (24.53%)	3359.17 (3.48%)	3221.11 (76.55%)
37	39 (26.42%)	3259 (2.73%)	3301.58 (86.92%)
38	39 (26.42%)	3351.16 (3.42%)	3231.87 (77.93%)
39	39 (26.42%)	3604.19 (5.3%)	3217.87 (76.13%)
40	40 (28.3%)	3201.59 (2.31%)	3310.39 (88.06%)
41	40 (28.3%)	3242.63 (2.61%)	3288.2 (85.2%)
42	40 (28.3%)	3335.36 (3.3%)	3232.21 (77.98%)
43	40 (28.3%)	3345.36 (3.37%)	3231.87 (77.93%)
44	40 (28.3%)	3352.33 (3.43%)	3218.54 (76.22%)
45	41 (30.19%)	3208.79 (2.36%)	3308.97 (87.88%)
46	41 (30.19%)	3218.79 (2.43%)	3308.63 (87.83%)

Continued on the next page

Table B.14 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
47	41 (30.19%)	3240.11 (2.59%)	3299.63 (86.67%)
48	41 (30.19%)	3326.53 (3.23%)	3230.6 (77.77%)
49	41 (30.19%)	3346.53 (3.38%)	3218.54 (76.22%)
50	42 (32.08%)	3223.33 (2.47%)	3299.97 (86.72%)
51	42 (32.08%)	3233.33 (2.54%)	3299.63 (86.67%)
52	42 (32.08%)	3242.03 (2.61%)	3295.63 (86.16%)
53	43 (33.96%)	3220.17 (2.44%)	3299.97 (86.72%)
54	43 (33.96%)	3341.92 (3.35%)	3230.26 (77.73%)
55	43 (33.96%)	3344.15 (3.37%)	3218.54 (76.22%)
56	44 (35.85%)	3228.39 (2.5%)	3294.67 (86.03%)
57	45 (37.74%)	3272.69 (2.83%)	3287.82 (85.15%)
58	47 (41.51%)	4810.23 (14.27%)	3094.8 (60.26%)
59	49 (45.28%)	4556.15 (12.38%)	3054.96 (55.12%)
60	49 (45.28%)	4559.45 (12.4%)	3054.09 (55.01%)
61	49 (45.28%)	4565.52 (12.45%)	3053.45 (54.93%)
62	50 (47.17%)	4569.59 (12.48%)	3053.06 (54.88%)
63	52 (50.94%)	4509.18 (12.03%)	3147.73 (67.08%)
64	54 (54.72%)	4539.58 (12.25%)	3086.95 (59.25%)
65	54 (54.72%)	4543.05 (12.28%)	3085.81 (59.1%)
66	55 (56.6%)	4531.25 (12.19%)	3086.95 (59.25%)
67	55 (56.6%)	4534.35 (12.22%)	3085.4 (59.05%)
68	55 (56.6%)	4679.01 (13.29%)	3043.87 (53.69%)
69	56 (58.49%)	4675.79 (13.27%)	3043.1 (53.59%)
70	56 (58.49%)	5204.34 (17.2%)	3018.01 (50.36%)
71	57 (60.38%)	4976.11 (15.5%)	3029.95 (51.9%)
72	58 (62.26%)	6143.31 (24.18%)	2927.69 (38.71%)
73	59 (64.15%)	5387.12 (18.56%)	2944.59 (40.89%)
74	59 (64.15%)	6854.08 (29.47%)	2912.51 (36.75%)
75	59 (64.15%)	7037.74 (30.83%)	2906.18 (35.93%)
76	60 (66.04%)	5885.49 (22.26%)	2934.2 (39.55%)
77	60 (66.04%)	7070.17 (31.07%)	2896.91 (34.74%)
78	61 (67.92%)	10135.05 (53.86%)	2772.88 (18.75%)
79	62 (69.81%)	7695.17 (35.72%)	2885.5 (33.27%)
80	62 (69.81%)	7706.75 (35.81%)	2880.54 (32.63%)
81	62 (69.81%)	9665.13 (50.37%)	2785.13 (20.32%)
82	62 (69.81%)	9698.01 (50.61%)	2784.94 (20.3%)
83	62 (69.81%)	9734.9 (50.89%)	2784.86 (20.29%)
84	62 (69.81%)	9804.97 (51.41%)	2784.25 (20.21%)

Continued on the next page

Table B.14 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
85	62 (69.81%)	9829.62 (51.59%)	2783.98 (20.18%)
86	62 (69.81%)	9863.02 (51.84%)	2783.84 (20.16%)
87	62 (69.81%)	9926.12 (52.31%)	2773.65 (18.84%)
88	63 (71.7%)	6038.67 (23.4%)	2932.77 (39.36%)
89	63 (71.7%)	6353.06 (25.74%)	2898.23 (34.91%)
90	63 (71.7%)	6836.77 (29.34%)	2891.82 (34.08%)
91	63 (71.7%)	7403.56 (33.55%)	2888.16 (33.61%)
92	63 (71.7%)	8773.54 (43.74%)	2846.4 (28.23%)
93	63 (71.7%)	8906.2 (44.72%)	2785.2 (20.33%)
94	63 (71.7%)	9028.25 (45.63%)	2785.19 (20.33%)
95	64 (73.58%)	6095.44 (23.82%)	2931.42 (39.19%)
96	64 (73.58%)	6141.54 (24.17%)	2921.62 (37.93%)
97	64 (73.58%)	6835.59 (29.33%)	2895.74 (34.59%)
98	64 (73.58%)	7288.06 (32.69%)	2889.33 (33.76%)
99	64 (73.58%)	8896.18 (44.65%)	2785.02 (20.31%)
100	64 (73.58%)	8946.43 (45.02%)	2774.28 (18.93%)
101	64 (73.58%)	8960.79 (45.13%)	2774.12 (18.9%)
102	64 (73.58%)	11616.71 (64.88%)	2771.64 (18.59%)
103	65 (75.47%)	6177.13 (24.43%)	2920.27 (37.75%)
104	65 (75.47%)	8508.93 (41.77%)	2776.66 (19.23%)
105	65 (75.47%)	8715.39 (43.31%)	2776.09 (19.16%)
106	65 (75.47%)	11573.52 (64.56%)	2765.9 (17.84%)
107	66 (77.36%)	7562.86 (34.74%)	2882.73 (32.91%)
108	66 (77.36%)	8179.18 (39.32%)	2853.48 (29.14%)
109	66 (77.36%)	8260.77 (39.93%)	2835.11 (26.77%)
110	66 (77.36%)	8325.29 (40.41%)	2833.37 (26.55%)
111	66 (77.36%)	8334.49 (40.47%)	2833.15 (26.52%)
112	66 (77.36%)	8505.73 (41.75%)	2774.1 (18.9%)
113	66 (77.36%)	8996.82 (45.4%)	2773.15 (18.78%)
114	66 (77.36%)	9053.71 (45.82%)	2769.71 (18.34%)
115	67 (79.25%)	7490.43 (34.2%)	2876.3 (32.08%)
116	67 (79.25%)	11807.47 (66.3%)	2760.81 (17.19%)
117	68 (81.13%)	11545.01 (64.35%)	2768.92 (18.23%)
118	68 (81.13%)	12634.59 (72.45%)	2735.99 (13.99%)
119	69 (83.02%)	7912.06 (37.33%)	2855.73 (29.43%)
120	69 (83.02%)	12740.11 (73.23%)	2732.89 (13.59%)
121	69 (83.02%)	12824.44 (73.86%)	2726.25 (12.73%)
122	70 (84.91%)	8207.55 (39.53%)	2848.47 (28.49%)

Continued on the next page

Table B.14 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
123	71 (86.79%)	14094.56 (83.31%)	2699.63 (9.3%)
124	71 (86.79%)	14329.69 (85.05%)	2698.14 (9.11%)
125	72 (88.68%)	13361.94 (77.86%)	2724.49 (12.5%)
126	72 (88.68%)	13842.65 (81.43%)	2711.84 (10.87%)
127	72 (88.68%)	13980.36 (82.46%)	2701.14 (9.49%)
128	72 (88.68%)	14755.75 (88.22%)	2695.85 (8.81%)
129	72 (88.68%)	14832.64 (88.79%)	2692.86 (8.43%)
130	72 (88.68%)	14926.24 (89.49%)	2691.68 (8.27%)
131	72 (88.68%)	14935.14 (89.56%)	2690.6 (8.13%)
132	73 (90.57%)	14862.93 (89.02%)	2687.99 (7.8%)
133	73 (90.57%)	15067.74 (90.54%)	2687.22 (7.7%)
134	73 (90.57%)	15919.65 (96.88%)	2646.84 (2.49%)
135	74 (92.45%)	13647.02 (79.98%)	2721.91 (12.17%)
136	74 (92.45%)	13742.43 (80.69%)	2720.39 (11.98%)
137	74 (92.45%)	14999.78 (90.04%)	2682.21 (7.05%)
138	74 (92.45%)	15010.88 (90.12%)	2681.7 (6.99%)
139	74 (92.45%)	15118.42 (90.92%)	2681.39 (6.95%)
140	75 (94.34%)	15170.87 (91.31%)	2681.14 (6.91%)
141	75 (94.34%)	15836.09 (96.26%)	2634.68 (0.92%)
142	77 (98.11%)	16020.3 (97.63%)	2634.31 (0.88%)
143	77 (98.11%)	16220.12 (99.11%)	2627.71 (0.02%)
144	78 (100%)	16122.67 (98.39%)	2627.68 (0.02%)
145	78 (100%)	16148.52 (98.58%)	2627.59 (0.01%)
146	78 (100%)	16339.69 (100%)	2627.52 (0%)

Table B.15: Experimental results for instance 15 - rc201-B7 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
3381.8	0	0	3358.42
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	13 (0%)	2595.75 (7.42%)	3314.19 (98.81%)
2	14 (1.47%)	1911.46 (3.4%)	3324.07 (99.97%)
3	15 (2.94%)	1722.75 (2.29%)	3324.37 (100%)
4	15 (2.94%)	2042.72 (4.17%)	3321.78 (99.7%)
5	16 (4.41%)	1454.86 (0.71%)	3323.88 (99.94%)
6	16 (4.41%)	1782.61 (2.64%)	3316.54 (99.09%)
7	17 (5.88%)	1334.11 (0%)	3298.14 (96.94%)

Continued on the next page

Table B.15 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
8	22 (13.24%)	1375.01 (0.24%)	3296.02 (96.7%)
9	25 (17.65%)	1673.56 (2%)	3293.5 (96.4%)
10	25 (17.65%)	5431.8 (24.11%)	3252.4 (91.61%)
11	26 (19.12%)	2404.15 (6.3%)	3289.53 (95.94%)
12	26 (19.12%)	3287 (11.49%)	3215.15 (87.27%)
13	27 (20.59%)	2470.39 (6.69%)	3229.25 (88.92%)
14	27 (20.59%)	2489.14 (6.8%)	3228.06 (88.78%)
15	27 (20.59%)	2579.29 (7.33%)	3215.15 (87.27%)
16	27 (20.59%)	3432.58 (12.35%)	3211.65 (86.87%)
17	27 (20.59%)	3454.01 (12.47%)	3211.27 (86.82%)
18	27 (20.59%)	4993.45 (21.53%)	3206.16 (86.23%)
19	28 (22.06%)	2430.62 (6.45%)	3231.56 (89.19%)
20	28 (22.06%)	2466.79 (6.67%)	3229.25 (88.92%)
21	28 (22.06%)	2478.79 (6.74%)	3229.04 (88.89%)
22	28 (22.06%)	2488.58 (6.79%)	3228.06 (88.78%)
23	29 (23.53%)	3639.26 (13.56%)	3196.98 (85.16%)
24	31 (26.47%)	2403.64 (6.29%)	3239.79 (90.15%)
25	31 (26.47%)	2423.99 (6.41%)	3239.79 (90.15%)
26	31 (26.47%)	2426 (6.43%)	3229.77 (88.98%)
27	31 (26.47%)	2426.27 (6.43%)	3229.5 (88.95%)
28	32 (27.94%)	2416.62 (6.37%)	3239.79 (90.15%)
29	32 (27.94%)	2418.81 (6.38%)	3237.61 (89.89%)
30	32 (27.94%)	2425.81 (6.42%)	3229.5 (88.95%)
31	32 (27.94%)	2432.65 (6.46%)	3227.53 (88.72%)
32	51 (55.88%)	6410.24 (29.87%)	3054.25 (68.53%)
33	52 (57.35%)	6269.6 (29.04%)	3054.25 (68.53%)
34	53 (58.82%)	6337.99 (29.44%)	3038.05 (66.64%)
35	53 (58.82%)	7084.29 (33.84%)	3031.9 (65.92%)
36	53 (58.82%)	7493.07 (36.24%)	3016.08 (64.08%)
37	53 (58.82%)	7493.66 (36.25%)	2990.48 (61.1%)
38	53 (58.82%)	8910.35 (44.58%)	2965.38 (58.17%)
39	54 (60.29%)	7662.8 (37.24%)	2983.75 (60.31%)
40	54 (60.29%)	8244.34 (40.66%)	2955.38 (57.01%)
41	55 (61.76%)	8096.4 (39.79%)	2970.18 (58.73%)
42	56 (63.24%)	7944.54 (38.9%)	2965.94 (58.24%)
43	58 (66.18%)	7355.16 (35.43%)	3026.06 (65.24%)
44	58 (66.18%)	7601.96 (36.88%)	2958.56 (57.38%)
45	58 (66.18%)	7768.87 (37.86%)	2947 (56.03%)

Continued on the next page

Table B.15 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
46	58 (66.18%)	8019.45 (39.34%)	2946.18 (55.94%)
47	58 (66.18%)	10101.48 (51.59%)	2788.07 (37.51%)
48	59 (67.65%)	7443.65 (35.95%)	2735.25 (31.36%)
49	60 (69.12%)	7399.21 (35.69%)	3000.6 (62.28%)
50	60 (69.12%)	7442.87 (35.95%)	2999.35 (62.13%)
51	61 (70.59%)	7272.39 (34.94%)	2724.74 (30.13%)
52	62 (72.06%)	6993.47 (33.3%)	2726.59 (30.35%)
53	62 (72.06%)	7007.18 (33.38%)	2725.93 (30.27%)
54	62 (72.06%)	7051.03 (33.64%)	2725.83 (30.26%)
55	62 (72.06%)	7083.81 (33.83%)	2724.51 (30.11%)
56	73 (88.24%)	16492.41 (89.2%)	2694.96 (26.66%)
57	74 (89.71%)	15013.65 (80.5%)	2694.25 (26.58%)
58	74 (89.71%)	15524.28 (83.5%)	2684.22 (25.41%)
59	75 (91.18%)	15189.45 (81.53%)	2689.34 (26.01%)
60	75 (91.18%)	15713.45 (84.61%)	2679.31 (24.84%)
61	81 (100%)	18328.23 (100%)	2466.11 (0%)

Table B.16: Experimental results for instance 16 - rc201-B8 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
3126.8	0	0	3358.42
Solution S/N	**f_1**	**f_2**	**f_3**
1	2 (0%)	141.25 (0.66%)	3263.47 (79.03%)
2	4 (2.67%)	31.13 (0.01%)	3275.11 (79.9%)
3	6 (5.33%)	50.77 (0.13%)	3274.81 (79.88%)
4	6 (5.33%)	754.9 (4.25%)	3243.27 (77.52%)
5	7 (6.67%)	28.84 (0%)	3543.8 (100%)
6	7 (6.67%)	124.6 (0.56%)	3174.86 (72.4%)
7	10 (10.67%)	1009.99 (5.74%)	3143.43 (70.05%)
8	11 (12%)	1416.43 (8.12%)	3125.26 (68.69%)
9	14 (16%)	219.37 (1.12%)	3151 (70.62%)
10	16 (18.67%)	1700.28 (9.78%)	3084.3 (65.63%)
11	17 (20%)	2691.52 (15.58%)	3075.67 (64.98%)
12	18 (21.33%)	1905.68 (10.98%)	3074.17 (64.87%)
13	20 (24%)	3045.83 (17.66%)	3043.18 (62.55%)
14	21 (25.33%)	2199.57 (12.7%)	2937.07 (54.62%)
15	21 (25.33%)	2259.57 (13.06%)	2936.49 (54.57%)

Continued on the next page

Table B.16 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
16	21 (25.33%)	2267.6 (13.1%)	2936.25 (54.56%)
17	21 (25.33%)	2270.02 (13.12%)	2927.08 (53.87%)
18	22 (26.67%)	2183.79 (12.61%)	2948.94 (55.51%)
19	22 (26.67%)	2245.4 (12.97%)	2936.76 (54.59%)
20	23 (28%)	1275.03 (7.29%)	2955.95 (56.03%)
21	23 (28%)	1279.13 (7.32%)	2953.9 (55.88%)
22	23 (28%)	1394.18 (7.99%)	2952.11 (55.74%)
23	23 (28%)	2192.72 (12.66%)	2937.34 (54.64%)
24	24 (29.33%)	1258.08 (7.19%)	2959.14 (56.27%)
25	24 (29.33%)	1259.6 (7.2%)	2958.38 (56.21%)
26	25 (30.67%)	1265.29 (7.24%)	2957.82 (56.17%)
27	28 (34.67%)	1160.93 (6.63%)	2952.84 (55.8%)
28	28 (34.67%)	3864.09 (22.45%)	2917.24 (53.13%)
29	29 (36%)	1066.45 (6.07%)	2942.87 (55.05%)
30	29 (36%)	3834.14 (22.27%)	2917.24 (53.13%)
31	30 (37.33%)	1060.9 (6.04%)	2942.87 (55.05%)
32	30 (37.33%)	1952.4 (11.26%)	2921.96 (53.49%)
33	30 (37.33%)	2017.98 (11.64%)	2910.66 (52.64%)
34	31 (38.67%)	1083.01 (6.17%)	2942.46 (55.02%)
35	32 (40%)	1959.17 (11.3%)	2920.76 (53.4%)
36	33 (41.33%)	1573.5 (9.04%)	2892.78 (51.31%)
37	40 (50.67%)	2426.24 (14.03%)	2879.64 (50.32%)
38	42 (53.33%)	2801.77 (16.23%)	2865.43 (49.26%)
39	45 (57.33%)	3489.6 (20.25%)	2852.18 (48.27%)
40	47 (60%)	4422.91 (25.72%)	2831.97 (46.76%)
41	49 (62.67%)	4935.35 (28.71%)	2808.18 (44.98%)
42	49 (62.67%)	7312.84 (42.63%)	2758.55 (41.27%)
43	50 (64%)	4183.34 (24.31%)	2828.39 (46.49%)
44	50 (64%)	5036.94 (29.31%)	2806.64 (44.86%)
45	50 (64%)	5770.76 (33.6%)	2800.23 (44.38%)
46	50 (64%)	6548.36 (38.15%)	2780.93 (42.94%)
47	51 (65.33%)	5234.62 (30.47%)	2798.69 (44.27%)
48	51 (65.33%)	5821.79 (33.9%)	2789.27 (43.56%)
49	51 (65.33%)	6349.99 (36.99%)	2771.46 (42.23%)
50	52 (66.67%)	5285.91 (30.77%)	2787.73 (43.45%)
51	53 (68%)	5750.07 (33.48%)	2781.73 (43%)
52	53 (68%)	6667.49 (38.85%)	2764.76 (41.73%)
53	53 (68%)	6842.6 (39.88%)	2746.03 (40.33%)

Continued on the next page

Table B.16 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
54	53 (68%)	11056.46 (64.54%)	2643.02 (32.62%)
55	54 (69.33%)	5812.95 (33.85%)	2606.88 (29.92%)
56	54 (69.33%)	9951.49 (58.07%)	2603.39 (29.66%)
57	54 (69.33%)	9976.63 (58.22%)	2602.39 (29.58%)
58	54 (69.33%)	10018.44 (58.46%)	2565.64 (26.84%)
59	54 (69.33%)	10663.92 (62.24%)	2543.49 (25.18%)
60	54 (69.33%)	10860.14 (63.39%)	2520.99 (23.5%)
61	55 (70.67%)	5715.78 (33.28%)	2603.51 (29.67%)
62	55 (70.67%)	5730.05 (33.37%)	2602.9 (29.62%)
63	55 (70.67%)	5753.62 (33.5%)	2602.64 (29.6%)
64	55 (70.67%)	5772.98 (33.62%)	2602.31 (29.58%)
65	55 (70.67%)	9977.24 (58.22%)	2551.68 (25.79%)
66	55 (70.67%)	10748.68 (62.74%)	2531.8 (24.3%)
67	55 (70.67%)	10758.68 (62.8%)	2521.8 (23.56%)
68	55 (70.67%)	10782.7 (62.94%)	2520.52 (23.46%)
69	56 (72%)	5819.77 (33.89%)	2601.52 (29.52%)
70	58 (74.67%)	5708.53 (33.24%)	2677.59 (35.21%)
71	58 (74.67%)	6213.33 (36.19%)	2600.63 (29.45%)
72	60 (77.33%)	6890.94 (40.16%)	2519.67 (23.4%)
73	61 (78.67%)	7799.53 (45.48%)	2519.06 (23.35%)
74	61 (78.67%)	8934.84 (52.12%)	2510.54 (22.71%)
75	62 (80%)	7621.41 (44.43%)	2518.03 (23.27%)
76	62 (80%)	8342.13 (48.65%)	2510.1 (22.68%)
77	62 (80%)	8475.46 (49.43%)	2509.18 (22.61%)
78	66 (85.33%)	6609.33 (38.51%)	2593.91 (28.95%)
79	67 (86.67%)	6546.06 (38.14%)	2584.01 (28.21%)
80	67 (86.67%)	6557.54 (38.21%)	2583.66 (28.18%)
81	67 (86.67%)	6565.28 (38.25%)	2581.96 (28.06%)
82	70 (90.67%)	6983.32 (40.7%)	2516.26 (23.14%)
83	70 (90.67%)	7005.17 (40.83%)	2515.77 (23.11%)
84	72 (93.33%)	6794.24 (39.59%)	2515.86 (23.11%)
85	72 (93.33%)	7033.5 (40.99%)	2513.06 (22.9%)
86	74 (96%)	16581.69 (96.87%)	2223.43 (1.24%)
87	75 (97.33%)	16582.73 (96.88%)	2222.83 (1.19%)
88	75 (97.33%)	16596.59 (96.96%)	2221.79 (1.12%)
89	75 (97.33%)	16605.15 (97.01%)	2220.55 (1.02%)
90	75 (97.33%)	16724.88 (97.71%)	2219.83 (0.97%)
91	76 (98.67%)	16908.31 (98.79%)	2218.91 (0.9%)

Continued on the next page

Table B.16 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
92	77 (100%)	17115.78 (100%)	2206.86 (0%)

Table B.17: Experimental results for instance 17 - rc202-C1 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
877.9	1	54	2683.88
Solution S/N	**f_1**	**f_2**	**f_3**
1	4 (0%)	382.44 (0.16%)	2685.58 (99.43%)
2	5 (2.13%)	370.27 (0%)	2688.69 (100%)
3	6 (4.26%)	371.1 (0.01%)	2683.94 (99.13%)
4	7 (6.38%)	374.08 (0.05%)	2680.06 (98.42%)
5	7 (6.38%)	1533.45 (15.55%)	2667.93 (96.2%)
6	8 (8.51%)	385.53 (0.2%)	2669.95 (96.57%)
7	11 (14.89%)	1219.7 (11.36%)	2660.21 (94.79%)
8	12 (17.02%)	1345.79 (13.04%)	2650.63 (93.04%)
9	12 (17.02%)	2125.49 (23.46%)	2650.33 (92.99%)
10	13 (19.15%)	1325.56 (12.77%)	2614.89 (86.5%)
11	14 (21.28%)	972.29 (8.05%)	2594.99 (82.87%)
12	15 (23.4%)	975.5 (8.09%)	2578.77 (79.9%)
13	15 (23.4%)	988.16 (8.26%)	2577.51 (79.67%)
14	15 (23.4%)	995.39 (8.36%)	2570.28 (78.35%)
15	18 (29.79%)	3550.95 (42.52%)	2553.39 (75.26%)
16	18 (29.79%)	3974.33 (48.18%)	2541.35 (73.06%)
17	20 (34.04%)	1178.81 (10.81%)	2563.83 (77.17%)
18	20 (34.04%)	1707.61 (17.88%)	2549.25 (74.5%)
19	20 (34.04%)	2626.31 (30.16%)	2453.67 (57.02%)
20	20 (34.04%)	2629.76 (30.21%)	2452.52 (56.81%)
21	20 (34.04%)	2648.33 (30.45%)	2443.44 (55.15%)
22	20 (34.04%)	2679.99 (30.88%)	2441.56 (54.81%)
23	21 (36.17%)	2645.92 (30.42%)	2443.85 (55.23%)
24	21 (36.17%)	2646.75 (30.43%)	2443.44 (55.15%)
25	21 (36.17%)	2650.2 (30.48%)	2442.7 (55.02%)
26	21 (36.17%)	2662.92 (30.65%)	2440.8 (54.67%)
27	21 (36.17%)	2690.89 (31.02%)	2428.15 (52.36%)
28	21 (36.17%)	2713.61 (31.33%)	2427.42 (52.22%)
29	21 (36.17%)	2714.37 (31.34%)	2427.01 (52.15%)
30	22 (38.3%)	1577.47 (16.14%)	2547.53 (74.19%)

Continued on the next page

Table B.17 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
31	22 (38.3%)	3271.73 (38.79%)	2413.21 (49.62%)
32	23 (40.43%)	1678.03 (17.48%)	2524.2 (69.92%)
33	23 (40.43%)	2139.01 (23.64%)	2521.83 (69.49%)
34	23 (40.43%)	3244.79 (38.43%)	2403.57 (47.86%)
35	24 (42.55%)	1988.09 (21.63%)	2514 (68.06%)
36	24 (42.55%)	2239.57 (24.99%)	2498.5 (65.22%)
37	24 (42.55%)	2353.65 (26.51%)	2423.85 (51.57%)
38	25 (44.68%)	2336.83 (26.29%)	2426.42 (52.04%)
39	25 (44.68%)	2347.64 (26.43%)	2425.91 (51.95%)
40	25 (44.68%)	2348.83 (26.45%)	2422.42 (51.31%)
41	25 (44.68%)	2358.84 (26.58%)	2421.91 (51.22%)
42	25 (44.68%)	2372.4 (26.76%)	2419.01 (50.68%)
43	27 (48.94%)	2903.8 (33.87%)	2413.97 (49.76%)
44	27 (48.94%)	2951.7 (34.51%)	2413.22 (49.63%)
45	27 (48.94%)	2951.91 (34.51%)	2413.01 (49.59%)
46	28 (51.06%)	2902.35 (33.85%)	2416.95 (50.31%)
47	28 (51.06%)	2905.46 (33.89%)	2413.84 (49.74%)
48	28 (51.06%)	2950.91 (34.5%)	2413.06 (49.6%)
49	28 (51.06%)	2951.75 (34.51%)	2412.64 (49.52%)
50	28 (51.06%)	2952.93 (34.53%)	2409.26 (48.9%)
51	31 (57.45%)	3638.18 (43.69%)	2392.38 (45.82%)
52	31 (57.45%)	3674.12 (44.17%)	2383.15 (44.13%)
53	31 (57.45%)	3865.33 (46.72%)	2379.76 (43.51%)
54	32 (59.57%)	4364.82 (53.4%)	2363.97 (40.62%)
55	33 (61.7%)	4280.95 (52.28%)	2379.7 (43.5%)
56	34 (63.83%)	4352.66 (53.24%)	2376.64 (42.94%)
57	35 (65.96%)	3387.61 (40.34%)	2370.63 (41.84%)
58	35 (65.96%)	4959.69 (61.35%)	2361.42 (40.15%)
59	35 (65.96%)	6821 (86.23%)	2276.33 (24.59%)
60	35 (65.96%)	6829.3 (86.35%)	2274.05 (24.18%)
61	36 (68.09%)	3362.03 (39.99%)	2373.5 (42.36%)
62	36 (68.09%)	3363.17 (40.01%)	2373.12 (42.29%)
63	36 (68.09%)	3387.53 (40.34%)	2372 (42.09%)
64	36 (68.09%)	3403.99 (40.56%)	2370.48 (41.81%)
65	36 (68.09%)	3416.21 (40.72%)	2369.95 (41.71%)
66	36 (68.09%)	5123.98 (63.55%)	2356.16 (39.19%)
67	36 (68.09%)	6842.79 (86.53%)	2272.09 (23.82%)
68	36 (68.09%)	6851.09 (86.64%)	2269.81 (23.4%)

Continued on the next page

Table B.17 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
69	37 (70.21%)	7349.65 (93.3%)	2269.68 (23.38%)
70	38 (72.34%)	6394.63 (80.53%)	2289.42 (26.99%)
71	39 (74.47%)	6028.38 (75.64%)	2275.35 (24.41%)
72	40 (76.6%)	4108.48 (49.97%)	2309.24 (30.61%)
73	40 (76.6%)	7671.19 (97.6%)	2268.08 (23.08%)
74	41 (78.72%)	4469.35 (54.8%)	2275.45 (24.43%)
75	41 (78.72%)	4470.27 (54.81%)	2274.53 (24.26%)
76	41 (78.72%)	4537.71 (55.71%)	2272.5 (23.89%)
77	41 (78.72%)	6518.92 (82.2%)	2270.9 (23.6%)
78	42 (80.85%)	3856.95 (46.61%)	2318.27 (32.26%)
79	42 (80.85%)	3858.78 (46.63%)	2317.66 (32.15%)
80	42 (80.85%)	3872.99 (46.82%)	2316.3 (31.9%)
81	42 (80.85%)	4011.37 (48.67%)	2312.6 (31.23%)
82	42 (80.85%)	4515.66 (55.42%)	2256.4 (20.95%)
83	43 (82.98%)	3789.73 (45.71%)	2332.97 (34.95%)
84	43 (82.98%)	3802.68 (45.89%)	2320.02 (32.58%)
85	43 (82.98%)	3804.34 (45.91%)	2315.94 (31.84%)
86	43 (82.98%)	3817.29 (46.08%)	2302.99 (29.47%)
87	43 (82.98%)	3818.61 (46.1%)	2302.15 (29.32%)
88	43 (82.98%)	6810.59 (86.1%)	2250.99 (19.96%)
89	44 (85.11%)	3774.96 (45.51%)	2320.58 (32.69%)
90	44 (85.11%)	3810.52 (45.99%)	2314.6 (31.59%)
91	44 (85.11%)	3823.47 (46.16%)	2301.65 (29.22%)
92	44 (85.11%)	4418.25 (54.11%)	2260.06 (21.62%)
93	44 (85.11%)	4658.84 (57.33%)	2245.86 (19.02%)
94	45 (87.23%)	4619.51 (56.8%)	2252.99 (20.33%)
95	45 (87.23%)	5203.13 (64.61%)	2240.07 (17.96%)
96	46 (89.36%)	5854.4 (73.31%)	2172.08 (5.53%)
97	46 (89.36%)	6374.28 (80.26%)	2167.17 (4.63%)
98	46 (89.36%)	6550.19 (82.61%)	2161.47 (3.59%)
99	47 (91.49%)	5378.8 (66.95%)	2234.22 (16.89%)
100	47 (91.49%)	6587.29 (83.11%)	2157.23 (2.81%)
101	47 (91.49%)	7009.36 (88.75%)	2155.34 (2.47%)
102	47 (91.49%)	7042.94 (89.2%)	2154.63 (2.34%)
103	47 (91.49%)	7043.75 (89.21%)	2153.82 (2.19%)
104	48 (93.62%)	5733.81 (71.7%)	2229 (15.94%)
105	48 (93.62%)	5745.42 (71.86%)	2225.11 (15.23%)
106	48 (93.62%)	5770.58 (72.19%)	2175 (6.06%)

Continued on the next page

Table B.17 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
107	48 (93.62%)	6953.93 (88.01%)	2156.13 (2.61%)
108	48 (93.62%)	7161.42 (90.79%)	2148.76 (1.27%)
109	50 (97.87%)	7561.54 (96.13%)	2142.13 (0.05%)
110	51 (100%)	7850.73 (100%)	2141.84 (0%)

Table B.18: Experimental results for instance 18 - rc202-C2 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
914.8	1	54	2683.88
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	6 (0%)	661.59 (2.35%)	2745.57 (100%)
2	6 (0%)	771.29 (3.66%)	2722.2 (95.98%)
3	7 (2.08%)	662.4 (2.36%)	2743.6 (99.66%)
4	8 (4.17%)	508.99 (0.53%)	2739.44 (98.95%)
5	9 (6.25%)	476.35 (0.15%)	2722.2 (95.98%)
6	10 (8.33%)	464.18 (0%)	2722.05 (95.96%)
7	11 (10.42%)	545.84 (0.97%)	2681.44 (88.98%)
8	13 (14.58%)	675.85 (2.52%)	2628.55 (79.88%)
9	14 (16.67%)	585.2 (1.44%)	2678.98 (88.55%)
10	14 (16.67%)	643.48 (2.14%)	2612.2 (77.07%)
11	14 (16.67%)	656.56 (2.29%)	2611.99 (77.04%)
12	15 (18.75%)	644.1 (2.14%)	2610.18 (76.73%)
13	15 (18.75%)	657.18 (2.3%)	2609.97 (76.69%)
14	15 (18.75%)	2187.58 (20.55%)	2609.59 (76.62%)
15	16 (20.83%)	649.23 (2.21%)	2609.13 (76.55%)
16	17 (22.92%)	659.49 (2.33%)	2607.35 (76.24%)
17	17 (22.92%)	683.46 (2.61%)	2606.18 (76.04%)
18	19 (27.08%)	814.09 (4.17%)	2594.82 (74.09%)
19	22 (33.33%)	1234.49 (9.18%)	2579.03 (71.37%)
20	22 (33.33%)	2315.27 (22.07%)	2556 (67.41%)
21	23 (35.42%)	1617.96 (13.76%)	2573.34 (70.39%)
22	23 (35.42%)	1687.96 (14.59%)	2559.73 (68.05%)
23	23 (35.42%)	1991.05 (18.2%)	2548.06 (66.05%)
24	27 (43.75%)	2490.41 (24.16%)	2546.3 (65.75%)
25	28 (45.83%)	3029.37 (30.58%)	2533.98 (63.63%)
26	28 (45.83%)	3046.94 (30.79%)	2533.66 (63.57%)
27	29 (47.92%)	2999.13 (30.22%)	2545.9 (65.68%)

Continued on the next page

Table B.18 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
28	29 (47.92%)	3040.08 (30.71%)	2532.17 (63.32%)
29	30 (50%)	2318.66 (22.11%)	2546.75 (65.82%)
30	30 (50%)	2572.34 (25.13%)	2540.94 (64.82%)
31	30 (50%)	4010.11 (42.27%)	2462.25 (51.3%)
32	31 (52.08%)	2330.16 (22.25%)	2543.02 (65.18%)
33	31 (52.08%)	2354.06 (22.53%)	2538.31 (64.37%)
34	31 (52.08%)	3665.14 (38.16%)	2524.7 (62.03%)
35	31 (52.08%)	4893.78 (52.81%)	2461.55 (51.18%)
36	32 (54.17%)	2329.32 (22.24%)	2544.27 (65.4%)
37	32 (54.17%)	2329.37 (22.24%)	2544.22 (65.39%)
38	32 (54.17%)	2657.71 (26.15%)	2519.16 (61.08%)
39	32 (54.17%)	5113.83 (55.43%)	2459.24 (50.78%)
40	33 (56.25%)	2352.57 (22.51%)	2542.18 (65.04%)
41	33 (56.25%)	4689.71 (50.38%)	2405.8 (41.59%)
42	33 (56.25%)	4766.62 (51.29%)	2311.22 (25.33%)
43	34 (58.33%)	2747.01 (27.22%)	2507.43 (59.06%)
44	34 (58.33%)	3615.91 (37.57%)	2435.93 (46.77%)
45	34 (58.33%)	3689.75 (38.45%)	2426.33 (45.12%)
46	34 (58.33%)	4698.75 (50.48%)	2311.57 (25.39%)
47	34 (58.33%)	4706.13 (50.57%)	2309.42 (25.03%)
48	35 (60.42%)	3566.36 (36.98%)	2436.61 (46.89%)
49	35 (60.42%)	3590.98 (37.28%)	2423.81 (44.69%)
50	35 (60.42%)	3597.86 (37.36%)	2422.09 (44.39%)
51	35 (60.42%)	3786.7 (39.61%)	2417.09 (43.53%)
52	35 (60.42%)	4261.74 (45.27%)	2311.09 (25.31%)
53	35 (60.42%)	4300.32 (45.73%)	2310.85 (25.27%)
54	36 (62.5%)	3761.15 (39.31%)	2417.14 (43.54%)
55	36 (62.5%)	3766.87 (39.37%)	2415.71 (43.3%)
56	36 (62.5%)	4070.54 (42.99%)	2308.63 (24.89%)
57	36 (62.5%)	4406.25 (47%)	2305.18 (24.3%)
58	38 (66.67%)	3810.51 (39.89%)	2372.75 (35.91%)
59	38 (66.67%)	3862.51 (40.51%)	2362.23 (34.1%)
60	38 (66.67%)	3922.06 (41.22%)	2350.44 (32.08%)
61	38 (66.67%)	3923.86 (41.25%)	2349.52 (31.92%)
62	39 (68.75%)	3799.79 (39.77%)	2377.67 (36.76%)
63	39 (68.75%)	3851.79 (40.39%)	2367.15 (34.95%)
64	40 (70.83%)	4893.02 (52.8%)	2304.12 (24.11%)
65	48 (87.5%)	6288.39 (69.43%)	2255.21 (15.71%)

Continued on the next page

Table B.18 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
66	49 (89.58%)	6243.85 (68.9%)	2253.11 (15.35%)
67	50 (91.67%)	6209.03 (68.49%)	2251.18 (15.01%)
68	50 (91.67%)	6598.35 (73.13%)	2249.86 (14.79%)
69	50 (91.67%)	7264.96 (81.08%)	2248.82 (14.61%)
70	50 (91.67%)	7285.68 (81.32%)	2248.5 (14.55%)
71	50 (91.67%)	8852.23 (100%)	2163.84 (0%)
72	51 (93.75%)	6963.86 (77.49%)	2248.76 (14.6%)
73	51 (93.75%)	6984.9 (77.74%)	2248.44 (14.54%)
74	51 (93.75%)	7121.73 (79.37%)	2248.34 (14.53%)
75	51 (93.75%)	7151.73 (79.73%)	2248.25 (14.51%)
76	51 (93.75%)	7368.19 (82.31%)	2239.72 (13.04%)
77	51 (93.75%)	8416.41 (94.8%)	2211.22 (8.14%)
78	52 (95.83%)	7928.7 (88.99%)	2238.58 (12.85%)
79	52 (95.83%)	8334.78 (93.83%)	2214.81 (8.76%)
80	52 (95.83%)	8405.02 (94.67%)	2210 (7.93%)
81	53 (97.92%)	7695.27 (86.21%)	2213.99 (8.62%)
82	53 (97.92%)	8267.73 (93.03%)	2209.33 (7.82%)
83	54 (100%)	8052.66 (90.47%)	2212.67 (8.39%)
84	54 (100%)	8263.1 (92.98%)	2211.77 (8.24%)

Table B.19: Experimental results for instance 19 - rc202-C3 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
987.2	1	100	2683.88
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	7 (0%)	1017.41 (5.2%)	2751.51 (91.39%)
2	9 (4.65%)	864.81 (3.33%)	2751.51 (91.39%)
3	11 (9.3%)	774.29 (2.22%)	2753.64 (91.76%)
4	11 (9.3%)	919.42 (4%)	2704.25 (83.21%)
5	12 (11.63%)	771.4 (2.18%)	2747.96 (90.78%)
6	12 (11.63%)	810.03 (2.66%)	2737.33 (88.94%)
7	12 (11.63%)	850.81 (3.16%)	2735.52 (88.62%)
8	12 (11.63%)	883.84 (3.56%)	2692.52 (81.17%)
9	12 (11.63%)	889.92 (3.64%)	2690.29 (80.79%)
10	13 (13.95%)	800.39 (2.54%)	2734.75 (88.49%)
11	13 (13.95%)	872.92 (3.43%)	2689.93 (80.73%)
12	13 (13.95%)	896.9 (3.72%)	2683.28 (79.57%)

Continued on the next page

Table B.19 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
13	14 (16.28%)	674 (0.99%)	2751.51 (91.39%)
14	14 (16.28%)	809.12 (2.64%)	2717.55 (85.51%)
15	15 (18.6%)	655.73 (0.76%)	2801.19 (100%)
16	15 (18.6%)	871.51 (3.41%)	2683.38 (79.59%)
17	16 (20.93%)	593.54 (0%)	2731.42 (87.91%)
18	17 (23.26%)	650.53 (0.7%)	2690.05 (80.75%)
19	18 (25.58%)	652.75 (0.73%)	2685.56 (79.97%)
20	19 (27.91%)	667.24 (0.9%)	2684.92 (79.86%)
21	20 (30.23%)	793.12 (2.45%)	2676.27 (78.36%)
22	22 (34.88%)	1305.66 (8.73%)	2669.96 (77.27%)
23	22 (34.88%)	2176.48 (19.42%)	2667.74 (76.88%)
24	23 (37.21%)	1668.7 (13.19%)	2659.85 (75.51%)
25	25 (41.86%)	1573.64 (12.02%)	2665.65 (76.52%)
26	25 (41.86%)	1583.06 (12.14%)	2664.73 (76.36%)
27	25 (41.86%)	2461.43 (22.91%)	2643.56 (72.69%)
28	26 (44.19%)	1630.42 (12.72%)	2655.64 (74.79%)
29	26 (44.19%)	3140.39 (31.24%)	2533.59 (53.64%)
30	27 (46.51%)	1613.35 (12.51%)	2657.02 (75.02%)
31	28 (48.84%)	1625.83 (12.66%)	2656.91 (75.01%)
32	28 (48.84%)	1878.25 (15.76%)	2528.78 (52.81%)
33	28 (48.84%)	1882.18 (15.81%)	2526.39 (52.39%)
34	28 (48.84%)	1892.25 (15.93%)	2526.17 (52.36%)
35	28 (48.84%)	1909.04 (16.14%)	2525.6 (52.26%)
36	29 (51.16%)	1858.88 (15.52%)	2526.44 (52.4%)
37	30 (53.49%)	1931.56 (16.41%)	2525.41 (52.22%)
38	30 (53.49%)	1944.72 (16.57%)	2524.13 (52%)
39	31 (55.81%)	6069.46 (67.17%)	2363.35 (24.15%)
40	31 (55.81%)	6531.87 (72.84%)	2362.85 (24.06%)
41	32 (58.14%)	1952.06 (16.66%)	2523.26 (51.85%)
42	32 (58.14%)	2354.33 (21.6%)	2516.2 (50.63%)
43	32 (58.14%)	5329.4 (58.09%)	2363.28 (24.14%)
44	32 (58.14%)	5653.11 (62.06%)	2358.62 (23.33%)
45	32 (58.14%)	6081.46 (67.32%)	2358.45 (23.3%)
46	33 (60.47%)	5265.07 (57.3%)	2361.81 (23.88%)
47	33 (60.47%)	5286.55 (57.57%)	2361.18 (23.77%)
48	47 (93.02%)	7187.2 (80.88%)	2247.71 (4.12%)
49	48 (95.35%)	7210.75 (81.17%)	2246.65 (3.93%)
50	48 (95.35%)	8714.43 (99.61%)	2224.11 (0.03%)

Continued on the next page

Table B.19 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
51	48 (95.35%)	8746.07 (100%)	2223.95 (0%)
52	49 (97.67%)	7489.8 (84.59%)	2241.98 (3.12%)
53	49 (97.67%)	7796.89 (88.36%)	2241.13 (2.98%)
54	49 (97.67%)	7982.26 (90.63%)	2239.53 (2.7%)
55	49 (97.67%)	8719.46 (99.67%)	2223.99 (0.01%)
56	50 (100%)	7395.52 (83.43%)	2242.07 (3.14%)
57	50 (100%)	7416.56 (83.69%)	2241.75 (3.08%)
58	50 (100%)	7448.04 (84.08%)	2241.59 (3.06%)

Table B.20: Experimental results for instance 20 - rc202-C4 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
993.3	3	177.95	2683.88
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	13 (0%)	1393.46 (6.79%)	2806.78 (90.22%)
2	15 (5.41%)	1477.67 (8.06%)	2800.07 (89.06%)
3	16 (8.11%)	1452.43 (7.68%)	2801.43 (89.29%)
4	17 (10.81%)	1150.82 (3.14%)	2794.83 (88.15%)
5	18 (13.51%)	1139.7 (2.97%)	2713.89 (74.17%)
6	18 (13.51%)	1746.02 (12.1%)	2685.59 (69.29%)
7	19 (16.22%)	1131.58 (2.85%)	2709.89 (73.48%)
8	19 (16.22%)	1153 (3.17%)	2706.16 (72.84%)
9	19 (16.22%)	1171.72 (3.46%)	2696.98 (71.25%)
10	20 (18.92%)	998.57 (0.85%)	2829.52 (94.14%)
11	20 (18.92%)	1111.74 (2.55%)	2798.54 (88.79%)
12	21 (21.62%)	989.99 (0.72%)	2786.52 (86.72%)
13	21 (21.62%)	4202.16 (49.04%)	2667.63 (66.19%)
14	22 (24.32%)	941.94 (0%)	2863.43 (100%)
15	22 (24.32%)	1113.6 (2.58%)	2714.27 (74.24%)
16	22 (24.32%)	1114.82 (2.6%)	2712.6 (73.95%)
17	22 (24.32%)	1123.78 (2.74%)	2712.5 (73.93%)
18	22 (24.32%)	1129.63 (2.82%)	2707.81 (73.12%)
19	22 (24.32%)	1148.35 (3.1%)	2701.09 (71.96%)
20	23 (27.03%)	1112.21 (2.56%)	2712.16 (73.88%)
21	23 (27.03%)	1118.36 (2.65%)	2711.72 (73.8%)
22	23 (27.03%)	1118.78 (2.66%)	2707.88 (73.14%)
23	23 (27.03%)	1125.05 (2.75%)	2705.25 (72.68%)

Continued on the next page

Table B.20 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
24	24 (29.73%)	1115.5 (2.61%)	2710.54 (73.6%)
25	24 (29.73%)	1116.73 (2.63%)	2709.98 (73.5%)
26	26 (35.14%)	2563.01 (24.38%)	2576.35 (50.42%)
27	28 (40.54%)	1878.81 (14.09%)	2683.1 (68.86%)
28	28 (40.54%)	2339.73 (21.03%)	2654.08 (63.85%)
29	28 (40.54%)	2555.11 (24.27%)	2587.08 (52.27%)
30	29 (43.24%)	2605.99 (25.03%)	2574.73 (50.14%)
31	30 (45.95%)	2693.54 (26.35%)	2566.94 (48.8%)
32	30 (45.95%)	5556.32 (69.41%)	2562.81 (48.08%)
33	31 (48.65%)	5015.53 (61.28%)	2559.73 (47.55%)
34	33 (54.05%)	5452.35 (67.85%)	2556.29 (46.96%)
35	33 (54.05%)	5544.08 (69.23%)	2555.3 (46.79%)
36	38 (67.57%)	4519.15 (53.81%)	2552.02 (46.22%)
37	39 (70.27%)	4354.44 (51.33%)	2552.2 (46.25%)
38	39 (70.27%)	4781.54 (57.76%)	2550.31 (45.92%)
39	39 (70.27%)	6988.76 (90.96%)	2411.94 (22.03%)
40	40 (72.97%)	4259.14 (49.9%)	2533.65 (43.05%)
41	40 (72.97%)	4276.99 (50.17%)	2532.61 (42.87%)
42	40 (72.97%)	4330.43 (50.97%)	2527.53 (41.99%)
43	40 (72.97%)	4343.23 (51.16%)	2523.18 (41.24%)
44	40 (72.97%)	5898.28 (74.55%)	2413.33 (22.27%)
45	40 (72.97%)	5898.64 (74.56%)	2413.15 (22.24%)
46	40 (72.97%)	5904.3 (74.64%)	2410.32 (21.75%)
47	40 (72.97%)	5996.15 (76.03%)	2407.96 (21.34%)
48	41 (75.68%)	4246.76 (49.71%)	2537.65 (43.74%)
49	41 (75.68%)	4247.21 (49.72%)	2537.5 (43.71%)
50	41 (75.68%)	4323.12 (50.86%)	2523.41 (41.28%)
51	41 (75.68%)	5006.47 (61.14%)	2496.87 (36.7%)
52	41 (75.68%)	5881.48 (74.3%)	2413.3 (22.26%)
53	42 (78.38%)	4750.27 (57.29%)	2497.39 (36.79%)
54	42 (78.38%)	7270.8 (95.2%)	2406.71 (21.12%)
55	43 (81.08%)	4569.86 (54.57%)	2521.42 (40.93%)
56	43 (81.08%)	6157.12 (78.45%)	2406.99 (21.17%)
57	43 (81.08%)	6517.88 (83.87%)	2405.59 (20.93%)
58	44 (83.78%)	5148.63 (63.28%)	2401.42 (20.21%)
59	45 (86.49%)	5458.07 (67.93%)	2375.26 (15.69%)
60	45 (86.49%)	5709.95 (71.72%)	2370.42 (14.86%)
61	45 (86.49%)	6049.18 (76.82%)	2366.88 (14.25%)

Continued on the next page

Table B.20 – *Continued from the previous page*

Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
62	45 (86.49%)	7406.1 (97.24%)	2301.72 (2.99%)
63	45 (86.49%)	7437.93 (97.71%)	2301.19 (2.9%)
64	46 (89.19%)	6713.94 (86.82%)	2360.32 (13.11%)
65	46 (89.19%)	7437.28 (97.7%)	2301.06 (2.88%)
66	46 (89.19%)	7447.4 (97.86%)	2300.97 (2.86%)
67	46 (89.19%)	7558.36 (99.53%)	2284.55 (0.03%)
68	47 (91.89%)	6550.33 (84.36%)	2363.41 (13.65%)
69	47 (91.89%)	6954.5 (90.44%)	2353.26 (11.89%)
70	47 (91.89%)	7589.89 (100%)	2284.39 (0%)
71	48 (94.59%)	6873.55 (89.22%)	2350.08 (11.34%)
72	48 (94.59%)	7364.11 (96.6%)	2316.28 (5.51%)
73	49 (97.3%)	7164.7 (93.6%)	2347.29 (10.86%)
74	49 (97.3%)	7204.5 (94.2%)	2344.87 (10.44%)
75	50 (100%)	7267.01 (95.14%)	2343.1 (10.14%)
76	50 (100%)	7310.48 (95.8%)	2318.72 (5.93%)
77	50 (100%)	7332.59 (96.13%)	2318.09 (5.82%)

Table B.21: Experimental results for instance 21 - rc202-C5 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB$'_1$ for f$_1$**	**LB$'_2$ for f$_2$**	**Reference value for f$_3$**
893.7	8	387.28	2683.88
Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
1	25 (0%)	3054.04 (19.39%)	2880.8 (94.75%)
2	25 (0%)	3067.65 (19.62%)	2876.97 (94.06%)
3	25 (0%)	3083.39 (19.89%)	2794.52 (79.21%)
4	25 (0%)	3123.71 (20.58%)	2779.57 (76.52%)
5	25 (0%)	3168.25 (21.34%)	2778.68 (76.36%)
6	25 (0%)	3271.09 (23.09%)	2771.01 (74.98%)
7	27 (7.14%)	2654.76 (12.57%)	2794.51 (79.21%)
8	27 (7.14%)	2706.6 (13.45%)	2783.66 (77.26%)
9	27 (7.14%)	2994.01 (18.36%)	2764.26 (73.76%)
10	29 (14.29%)	1990.84 (1.23%)	2798.95 (80.01%)
11	30 (17.86%)	2136.47 (3.71%)	2795.78 (79.44%)
12	30 (17.86%)	2326.82 (6.96%)	2780.93 (76.77%)
13	30 (17.86%)	2408.82 (8.37%)	2773.19 (75.37%)
14	30 (17.86%)	2436.35 (8.84%)	2754.15 (71.94%)
15	30 (17.86%)	2518.35 (10.24%)	2750.77 (71.34%)

Continued on the next page

Table B.21 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
16	31 (21.43%)	1995.96 (1.31%)	2798.2 (79.88%)
17	31 (21.43%)	2036.7 (2.01%)	2795.69 (79.42%)
18	31 (21.43%)	2107.78 (3.22%)	2787.1 (77.88%)
19	31 (21.43%)	2938.32 (17.41%)	2733.32 (68.19%)
20	32 (25%)	1957.06 (0.65%)	2891.44 (96.67%)
21	32 (25%)	1958.82 (0.68%)	2889.55 (96.33%)
22	32 (25%)	1971.83 (0.9%)	2883.88 (95.31%)
23	32 (25%)	2024.88 (1.81%)	2798.11 (79.86%)
24	33 (28.57%)	1945.86 (0.46%)	2895.83 (97.46%)
25	33 (28.57%)	2594.73 (11.54%)	2685.08 (59.51%)
26	33 (28.57%)	2632.78 (12.19%)	2676.89 (58.03%)
27	33 (28.57%)	2633.27 (12.2%)	2655.93 (54.26%)
28	33 (28.57%)	4079.06 (36.89%)	2650.62 (53.3%)
29	33 (28.57%)	4177.51 (38.57%)	2647.65 (52.76%)
30	34 (32.14%)	1931.84 (0.22%)	2909.94 (100%)
31	34 (32.14%)	2585.46 (11.38%)	2672.3 (57.2%)
32	34 (32.14%)	2659.48 (12.65%)	2651.63 (53.48%)
33	34 (32.14%)	2667.42 (12.78%)	2649.19 (53.04%)
34	35 (35.71%)	2584.48 (11.37%)	2720.82 (65.94%)
35	35 (35.71%)	2611.03 (11.82%)	2667.49 (56.34%)
36	35 (35.71%)	2629.73 (12.14%)	2650.8 (53.33%)
37	37 (42.86%)	1919.03 (0%)	2899.34 (98.09%)
38	37 (42.86%)	3022.17 (18.84%)	2614.74 (46.84%)
39	37 (42.86%)	3364.87 (24.69%)	2604.52 (45%)
40	38 (46.43%)	4242.03 (39.68%)	2598.13 (43.85%)
41	40 (53.57%)	4392.53 (42.25%)	2597.62 (43.75%)
42	41 (57.14%)	4053.1 (36.45%)	2592.91 (42.91%)
43	41 (57.14%)	4292.42 (40.54%)	2585.9 (41.64%)
44	41 (57.14%)	6008.38 (69.84%)	2497.31 (25.69%)
45	42 (60.71%)	4968.95 (52.09%)	2502.33 (26.59%)
46	42 (60.71%)	4969.38 (52.1%)	2501.9 (26.52%)
47	42 (60.71%)	5070.71 (53.83%)	2498.93 (25.98%)
48	43 (64.29%)	4908.48 (51.06%)	2501.05 (26.36%)
49	43 (64.29%)	4908.61 (51.06%)	2500.92 (26.34%)
50	43 (64.29%)	4919.62 (51.25%)	2500.53 (26.27%)
51	43 (64.29%)	4998.51 (52.6%)	2499.41 (26.07%)
52	43 (64.29%)	5070.97 (53.83%)	2498.27 (25.86%)
53	43 (64.29%)	5266.68 (57.18%)	2495.59 (25.38%)

Continued on the next page

Table B.21 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
54	43 (64.29%)	5549.64 (62.01%)	2492.43 (24.81%)
55	44 (67.86%)	4819.28 (49.53%)	2502.3 (26.59%)
56	44 (67.86%)	4828.92 (49.7%)	2501.03 (26.36%)
57	45 (71.43%)	4882.4 (50.61%)	2498.05 (25.82%)
58	45 (71.43%)	6191.24 (72.97%)	2473.06 (21.32%)
59	46 (75%)	5214.29 (56.28%)	2497.79 (25.78%)
60	46 (75%)	5233.49 (56.61%)	2490.87 (24.53%)
61	48 (82.14%)	5401.04 (59.47%)	2481.02 (22.76%)
62	49 (85.71%)	5667.59 (64.02%)	2472.98 (21.31%)
63	49 (85.71%)	6302.72 (74.87%)	2456.71 (18.38%)
64	50 (89.29%)	5914.39 (68.24%)	2466.11 (20.07%)
65	50 (89.29%)	6046 (70.49%)	2460.81 (19.12%)
66	50 (89.29%)	6357.05 (75.8%)	2452.82 (17.68%)
67	50 (89.29%)	6470.06 (77.73%)	2402.71 (8.65%)
68	50 (89.29%)	6620.93 (80.31%)	2401.37 (8.41%)
69	50 (89.29%)	6665.23 (81.06%)	2400.07 (8.18%)
70	50 (89.29%)	6687.84 (81.45%)	2398.94 (7.97%)
71	50 (89.29%)	6762.5 (82.72%)	2397.98 (7.8%)
72	53 (100%)	7774.03 (100%)	2354.66 (0%)

Table B.22: Experimental results for instance 22 - rc202-C6 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
1103	9	555.06	2683.88
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	26 (0%)	3787.1 (16.38%)	2912.69 (100%)
2	28 (8%)	3623.93 (13.64%)	2906.9 (98.65%)
3	28 (8%)	3627.02 (13.7%)	2892.52 (95.28%)
4	28 (8%)	3646.13 (14.02%)	2886.36 (93.84%)
5	28 (8%)	3656.32 (14.19%)	2866.12 (89.11%)
6	29 (12%)	3232.56 (7.08%)	2853 (86.04%)
7	29 (12%)	4171.67 (22.83%)	2844.12 (83.97%)
8	30 (16%)	3099.21 (4.84%)	2897.72 (96.5%)
9	30 (16%)	3220.62 (6.88%)	2880.16 (92.39%)
10	30 (16%)	3222.96 (6.92%)	2809.36 (75.84%)
11	30 (16%)	3231.24 (7.06%)	2807.45 (75.39%)
12	30 (16%)	3232.98 (7.08%)	2806.22 (75.1%)

Continued on the next page

Table B.22 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
13	30 (16%)	3246.48 (7.31%)	2802.69 (74.28%)
14	31 (20%)	2990.1 (3.01%)	2880.29 (92.42%)
15	31 (20%)	3028.06 (3.65%)	2880.25 (92.41%)
16	31 (20%)	3243.47 (7.26%)	2802.99 (74.35%)
17	32 (24%)	2950.77 (2.35%)	2897.49 (96.45%)
18	32 (24%)	2960.96 (2.52%)	2877.25 (91.71%)
19	32 (24%)	3129.45 (5.35%)	2864.61 (88.76%)
20	32 (24%)	3625.63 (13.67%)	2798.22 (73.23%)
21	33 (28%)	2950.17 (2.34%)	2892.39 (95.25%)
22	33 (28%)	3006.55 (3.29%)	2861.4 (88.01%)
23	33 (28%)	3200.06 (6.53%)	2835.02 (81.84%)
24	33 (28%)	3777.86 (16.23%)	2796.32 (72.79%)
25	33 (28%)	3936.24 (18.88%)	2781.91 (69.42%)
26	33 (28%)	4081.57 (21.32%)	2750.28 (62.02%)
27	33 (28%)	4408.63 (26.81%)	2743.27 (60.38%)
28	33 (28%)	4837.95 (34.01%)	2706.57 (51.8%)
29	34 (32%)	2954.76 (2.42%)	2890.03 (94.7%)
30	34 (32%)	3186.3 (6.3%)	2846.37 (84.49%)
31	34 (32%)	3858.41 (17.58%)	2783.92 (69.89%)
32	34 (32%)	4005.72 (20.05%)	2761.16 (64.57%)
33	34 (32%)	4385.9 (26.43%)	2739.04 (59.39%)
34	34 (32%)	4537.63 (28.97%)	2732.92 (57.96%)
35	34 (32%)	4639.64 (30.68%)	2707.64 (52.05%)
36	34 (32%)	4654.29 (30.93%)	2705.73 (51.61%)
37	34 (32%)	4695.66 (31.62%)	2700.23 (50.32%)
38	34 (32%)	5131.76 (38.94%)	2699.99 (50.26%)
39	35 (36%)	2947.15 (2.29%)	2911.57 (99.74%)
40	35 (36%)	4332.65 (25.53%)	2742.13 (60.12%)
41	36 (40%)	2810.7 (0%)	2900.24 (97.09%)
42	36 (40%)	4641.09 (30.71%)	2707.21 (51.95%)
43	36 (40%)	4649.72 (30.85%)	2707.2 (51.95%)
44	36 (40%)	4657.65 (30.99%)	2705.3 (51.5%)
45	36 (40%)	4660.04 (31.03%)	2704.72 (51.37%)
46	38 (48%)	4671.69 (31.22%)	2701.34 (50.58%)
47	38 (48%)	4685.07 (31.45%)	2700.47 (50.38%)
48	38 (48%)	5761.24 (49.5%)	2671.87 (43.69%)
49	38 (48%)	5777.34 (49.77%)	2671.08 (43.5%)
50	39 (52%)	5759.43 (49.47%)	2685.32 (46.83%)

Continued on the next page

Table B.22 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
51	40 (56%)	4635.34 (30.61%)	2710.68 (52.76%)
52	40 (56%)	5217.01 (40.37%)	2698.28 (49.86%)
53	40 (56%)	5949.46 (52.66%)	2667.66 (42.7%)
54	40 (56%)	6303.38 (58.6%)	2601.87 (27.32%)
55	40 (56%)	6542.75 (62.61%)	2598.84 (26.61%)
56	41 (60%)	4615.19 (30.27%)	2717.04 (54.25%)
57	41 (60%)	4634.84 (30.6%)	2710.68 (52.76%)
58	41 (60%)	5129.31 (38.9%)	2693.05 (48.64%)
59	41 (60%)	6013.51 (53.73%)	2625.83 (32.92%)
60	42 (64%)	5972.37 (53.04%)	2654.33 (39.59%)
61	43 (68%)	5760.92 (49.5%)	2669.64 (43.17%)
62	43 (68%)	5778.6 (49.79%)	2668.85 (42.98%)
63	43 (68%)	6138.58 (55.83%)	2613.79 (30.11%)
64	43 (68%)	6236.15 (57.47%)	2604.61 (27.96%)
65	44 (72%)	5459.96 (44.45%)	2650.44 (38.68%)
66	44 (72%)	6114.35 (55.43%)	2603.16 (27.62%)
67	44 (72%)	6117.7 (55.48%)	2602.86 (27.55%)
68	44 (72%)	6182.26 (56.56%)	2600.5 (27%)
69	44 (72%)	6470.3 (61.4%)	2586.25 (23.67%)
70	44 (72%)	6498.25 (61.87%)	2573.19 (20.61%)
71	44 (72%)	6630.04 (64.08%)	2502.22 (4.02%)
72	44 (72%)	6653.74 (64.47%)	2501.82 (3.92%)
73	44 (72%)	6676.39 (64.85%)	2495.16 (2.37%)
74	44 (72%)	7182.83 (73.35%)	2492.22 (1.68%)
75	45 (76%)	5759.45 (49.47%)	2643.43 (37.04%)
76	45 (76%)	5937.75 (52.46%)	2642.16 (36.74%)
77	46 (80%)	6300.79 (58.55%)	2587.74 (24.01%)
78	47 (84%)	5914.82 (52.08%)	2631.94 (34.35%)
79	47 (84%)	5955.81 (52.77%)	2631.57 (34.26%)
80	47 (84%)	5970.3 (53.01%)	2623.6 (32.4%)
81	47 (84%)	6078.26 (54.82%)	2618.08 (31.11%)
82	48 (88%)	5905.45 (51.92%)	2630.58 (34.03%)
83	48 (88%)	5923.13 (52.22%)	2629.79 (33.85%)
84	48 (88%)	6341.55 (59.24%)	2587.13 (23.87%)
85	48 (88%)	6428.47 (60.7%)	2582.9 (22.88%)
86	48 (88%)	8771.26 (100%)	2485.04 (0%)
87	49 (92%)	5899.53 (51.82%)	2642.48 (36.82%)
88	49 (92%)	6290.31 (58.38%)	2567.17 (19.2%)

Continued on the next page

Table B.22 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
89	49 (92%)	8070.82 (88.25%)	2488.37 (0.78%)
90	50 (96%)	6212.5 (57.07%)	2600.27 (26.94%)
91	51 (100%)	6584.35 (63.31%)	2555.63 (16.51%)

Table B.23: Experimental results for instance 23 - rc202-C7 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
920.1	3	203.27	2683.88
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	27 (0%)	5038.14 (21.69%)	2922.35 (100%)
2	30 (9.38%)	4875.36 (19.19%)	2902.53 (95.26%)
3	31 (12.5%)	4428.91 (12.33%)	2898.93 (94.4%)
4	31 (12.5%)	4443 (12.55%)	2891.81 (92.7%)
5	31 (12.5%)	4456.95 (12.77%)	2890.78 (92.45%)
6	32 (15.63%)	4411.14 (12.06%)	2907.87 (96.54%)
7	32 (15.63%)	4589.1 (14.79%)	2883.38 (90.68%)
8	32 (15.63%)	4655.9 (15.82%)	2862.67 (85.73%)
9	33 (18.75%)	5177.72 (23.83%)	2859.66 (85.01%)
10	34 (21.88%)	4070.11 (6.83%)	2886.39 (91.4%)
11	34 (21.88%)	4937.79 (20.15%)	2851.61 (83.08%)
12	34 (21.88%)	5163.16 (23.61%)	2848.58 (82.36%)
13	34 (21.88%)	5666.66 (31.34%)	2817.29 (74.87%)
14	35 (25%)	4461.98 (12.84%)	2816.56 (74.7%)
15	35 (25%)	5720.61 (32.17%)	2816.28 (74.63%)
16	35 (25%)	6004.64 (36.53%)	2815.9 (74.54%)
17	36 (28.13%)	3761.61 (2.09%)	2915.43 (98.34%)
18	36 (28.13%)	3957.28 (5.09%)	2897.27 (94%)
19	36 (28.13%)	4478.2 (13.09%)	2812.65 (73.76%)
20	36 (28.13%)	4920.65 (19.88%)	2811.15 (73.4%)
21	37 (31.25%)	3779 (2.36%)	2911.8 (97.48%)
22	38 (34.38%)	3778.66 (2.35%)	2914.92 (98.22%)
23	38 (34.38%)	5842.14 (34.03%)	2808.18 (72.69%)
24	39 (37.5%)	3625.48 (0%)	2918.25 (99.02%)
25	40 (40.63%)	5399.03 (27.23%)	2740.19 (56.43%)
26	40 (40.63%)	5418.35 (27.52%)	2739.26 (56.21%)
27	42 (46.88%)	5411 (27.41%)	2739.7 (56.32%)
28	42 (46.88%)	5432.42 (27.74%)	2729.67 (53.92%)

Continued on the next page

Table B.23 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
29	42 (46.88%)	5452.42 (28.05%)	2727.36 (53.36%)
30	44 (53.13%)	6647.34 (46.39%)	2714.99 (50.41%)
31	44 (53.13%)	7164.4 (54.33%)	2714.89 (50.38%)
32	45 (56.25%)	7230.86 (55.35%)	2712.39 (49.78%)
33	46 (59.38%)	6759.11 (48.11%)	2712.14 (49.72%)
34	46 (59.38%)	7083.15 (53.08%)	2708.75 (48.91%)
35	47 (62.5%)	6965.22 (51.27%)	2710.32 (49.29%)
36	49 (68.75%)	8149.31 (69.45%)	2708.54 (48.86%)
37	50 (71.88%)	7291.85 (56.29%)	2666.02 (38.69%)
38	51 (75%)	7319.14 (56.71%)	2660.07 (37.27%)
39	52 (78.13%)	7820.4 (64.4%)	2656.21 (36.35%)
40	52 (78.13%)	9324.48 (87.49%)	2522.03 (4.25%)
41	53 (81.25%)	7266.34 (55.9%)	2608.54 (24.95%)
42	53 (81.25%)	7317.66 (56.68%)	2606.36 (24.42%)
43	53 (81.25%)	9271.55 (86.68%)	2522.1 (4.27%)
44	53 (81.25%)	9304.59 (87.19%)	2521.72 (4.18%)
45	53 (81.25%)	9718.76 (93.55%)	2521.56 (4.14%)
46	54 (84.38%)	7266.01 (55.89%)	2608.95 (25.04%)
47	54 (84.38%)	7457.73 (58.83%)	2586.74 (19.73%)
48	54 (84.38%)	7521.72 (59.82%)	2575.43 (17.03%)
49	54 (84.38%)	7562.64 (60.44%)	2568.39 (15.34%)
50	54 (84.38%)	9270.17 (86.66%)	2521.47 (4.12%)
51	54 (84.38%)	9310.69 (87.28%)	2521.3 (4.08%)
52	54 (84.38%)	9616.46 (91.98%)	2506.73 (0.6%)
53	55 (87.5%)	9582.4 (91.45%)	2506.89 (0.63%)
54	55 (87.5%)	9595.82 (91.66%)	2506.45 (0.53%)
55	56 (90.63%)	8380.34 (73%)	2545.16 (9.79%)
56	56 (90.63%)	8407.75 (73.42%)	2542.65 (9.19%)
57	56 (90.63%)	9535.59 (90.73%)	2506.78 (0.61%)
58	56 (90.63%)	9561.6 (91.13%)	2506.61 (0.57%)
59	56 (90.63%)	10139.14 (100%)	2504.24 (0%)
60	57 (93.75%)	7764.81 (63.55%)	2567.29 (15.08%)
61	57 (93.75%)	7775.04 (63.71%)	2560.25 (13.4%)
62	57 (93.75%)	8035.15 (67.7%)	2554.13 (11.93%)
63	57 (93.75%)	8115.96 (68.94%)	2549.11 (10.73%)
64	57 (93.75%)	8739.76 (78.52%)	2532.19 (6.68%)
65	57 (93.75%)	9493.65 (90.09%)	2506.57 (0.56%)
66	57 (93.75%)	9506.31 (90.28%)	2505.94 (0.41%)

Continued on the next page

Table B.23 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
67	57 (93.75%)	9521.09 (90.51%)	2505.72 (0.35%)
68	57 (93.75%)	9642.09 (92.37%)	2505.49 (0.3%)
69	57 (93.75%)	9663.58 (92.7%)	2505.48 (0.3%)
70	57 (93.75%)	9667.93 (92.77%)	2505.32 (0.26%)
71	58 (96.88%)	8618.74 (76.66%)	2540.89 (8.77%)
72	58 (96.88%)	8646.15 (77.08%)	2538.38 (8.17%)
73	58 (96.88%)	8680.27 (77.6%)	2536.77 (7.78%)
74	58 (96.88%)	8690.47 (77.76%)	2535.19 (7.4%)
75	58 (96.88%)	8709.75 (78.06%)	2532.56 (6.77%)
76	58 (96.88%)	9233.6 (86.1%)	2531.95 (6.63%)
77	58 (96.88%)	9493.08 (90.08%)	2506.79 (0.61%)
78	58 (96.88%)	9517.57 (90.46%)	2505.85 (0.39%)
79	58 (96.88%)	9532.35 (90.68%)	2505.63 (0.33%)
80	58 (96.88%)	10116.02 (99.65%)	2504.31 (0.02%)
81	59 (100%)	9640.46 (92.34%)	2505.55 (0.31%)

Table B.24: Experimental results for instance 24 - c203-D1 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
243.5	4	665.36	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	4 (0%)	1632.95 (12.05%)	10125.36 (97.2%)
2	5 (5.56%)	1324.48 (2.69%)	10127.92 (100%)
3	5 (5.56%)	1838.71 (18.29%)	10106.54 (76.65%)
4	5 (5.56%)	2119.9 (26.81%)	10105.27 (75.26%)
5	6 (11.11%)	1235.63 (0%)	10121.69 (93.19%)
6	6 (11.11%)	1730.37 (15%)	10104.79 (74.74%)
7	7 (16.67%)	1550.79 (9.56%)	10111.23 (81.77%)
8	7 (16.67%)	1640.96 (12.29%)	10104.23 (74.12%)
9	7 (16.67%)	1829.02 (17.99%)	10101.48 (71.12%)
10	8 (22.22%)	1453.46 (6.61%)	10097.11 (66.35%)
11	8 (22.22%)	1872.81 (19.32%)	10093.05 (61.91%)
12	8 (22.22%)	2878.21 (49.81%)	10073.09 (40.11%)
13	9 (27.78%)	1989.91 (22.87%)	10073.07 (40.09%)
14	9 (27.78%)	1990.98 (22.9%)	10071.92 (38.83%)
15	9 (27.78%)	2265.55 (31.23%)	10067.62 (34.13%)
16	9 (27.78%)	2481.59 (37.78%)	10052.43 (17.54%)

Continued on the next page

Table B.24 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
17	9 (27.78%)	2726.11 (45.2%)	10047.22 (11.85%)
18	10 (33.33%)	1762.95 (15.99%)	10086.01 (54.22%)
19	10 (33.33%)	2006.23 (23.37%)	10071.28 (38.13%)
20	10 (33.33%)	2006.44 (23.37%)	10071.03 (37.86%)
21	10 (33.33%)	3022.76 (54.19%)	10042.69 (6.9%)
22	11 (38.89%)	2179 (28.61%)	10069.17 (35.83%)
23	11 (38.89%)	3038.01 (54.65%)	10040.01 (3.98%)
24	13 (50%)	3540.15 (69.88%)	10039.8 (3.75%)
25	15 (61.11%)	4241.53 (91.15%)	10039.4 (3.31%)
26	19 (83.33%)	3819.6 (78.35%)	10037.82 (1.58%)
27	20 (88.89%)	3739.94 (75.94%)	10036.81 (0.48%)
28	20 (88.89%)	3832.11 (78.73%)	10036.77 (0.44%)
29	22 (100%)	4533.49 (100%)	10036.37 (0%)

Table B.25: Experimental results for instance 25 - c203-D2 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
207.5	4	665.36	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	4 (0%)	4282.22 (100%)	10397.65 (100%)
2	5 (6.25%)	2747 (47.74%)	10327.33 (67.2%)
3	5 (6.25%)	2748.23 (47.78%)	10326.42 (66.78%)
4	6 (12.5%)	2641.43 (44.14%)	10301.82 (55.31%)
5	6 (12.5%)	2650.44 (44.45%)	10277.57 (44%)
6	7 (18.75%)	1439.85 (3.24%)	10282.56 (46.32%)
7	8 (25%)	1344.77 (0%)	10276.33 (43.42%)
8	13 (56.25%)	1740.7 (13.48%)	10237.45 (25.28%)
9	13 (56.25%)	1807.31 (15.75%)	10235.23 (24.25%)
10	13 (56.25%)	2521.85 (40.07%)	10230.48 (22.03%)
11	14 (62.5%)	1761.87 (14.2%)	10231.26 (22.4%)
12	14 (62.5%)	1838.15 (16.8%)	10231.23 (22.38%)
13	14 (62.5%)	2293.67 (32.3%)	10217.24 (15.86%)
14	15 (68.75%)	3631.39 (77.84%)	10187.24 (1.87%)
15	16 (75%)	1683.67 (11.54%)	10239.96 (26.45%)
16	16 (75%)	1745.98 (13.66%)	10236.26 (24.73%)
17	16 (75%)	2915.32 (53.47%)	10187.67 (2.07%)
18	16 (75%)	2919.77 (53.62%)	10186.88 (1.7%)

Continued on the next page

Table B.25 – *Continued from the previous page*

Solution S/N	f_1	f_2	f_3
19	16 (75%)	3794.78 (83.41%)	10183.95 (0.33%)
20	17 (81.25%)	1689.74 (11.74%)	10235.41 (24.33%)
21	17 (81.25%)	2071.54 (24.74%)	10222.5 (18.31%)
22	17 (81.25%)	2200.89 (29.15%)	10207.94 (11.52%)
23	17 (81.25%)	2368 (34.83%)	10200.21 (7.91%)
24	17 (81.25%)	2803.32 (49.65%)	10195 (5.48%)
25	17 (81.25%)	2931.85 (54.03%)	10185.63 (1.11%)
26	17 (81.25%)	3547.16 (74.98%)	10184.19 (0.44%)
27	18 (87.5%)	2392.32 (35.66%)	10199.14 (7.42%)
28	18 (87.5%)	4131.44 (94.87%)	10183.24 (0%)
29	19 (93.75%)	3092.16 (59.49%)	10184.44 (0.56%)
30	20 (100%)	2564.89 (41.54%)	10191.41 (3.81%)

Table B.26: Experimental results for instance 26 - c203-D3 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	LB'_1 for f_1	LB'_2 for f_2	Reference value for f_3
216.4	4	665.36	9601.72
Solution S/N	f_1	f_2	f_3
1	5 (0%)	7813.69 (100%)	10561.84 (100%)
2	7 (10.53%)	4478.03 (46.63%)	10441.59 (54.48%)
3	8 (15.79%)	3279.45 (27.45%)	10416.84 (45.12%)
4	10 (26.32%)	1779.95 (3.46%)	10410.86 (42.85%)
5	12 (36.84%)	1564 (0%)	10393.16 (36.15%)
6	13 (42.11%)	3956.54 (38.28%)	10365.14 (25.55%)
7	13 (42.11%)	4048.88 (39.76%)	10364.12 (25.16%)
8	13 (42.11%)	4899.92 (53.38%)	10359.86 (23.55%)
9	14 (47.37%)	2999.24 (22.96%)	10366.18 (25.94%)
10	15 (52.63%)	2393.66 (13.28%)	10364.81 (25.42%)
11	15 (52.63%)	3021.93 (23.33%)	10356.56 (22.3%)
12	15 (52.63%)	3872.62 (36.94%)	10356.21 (22.17%)
13	16 (57.89%)	1959.22 (6.32%)	10364.67 (25.37%)
14	17 (63.16%)	2816.25 (20.04%)	10353.15 (21.01%)
15	18 (68.42%)	1837.06 (4.37%)	10361.62 (24.21%)
16	18 (68.42%)	4712.28 (50.37%)	10304.24 (2.49%)
17	18 (68.42%)	4776.28 (51.4%)	10297.65 (0%)
18	19 (73.68%)	2224.93 (10.58%)	10344.16 (17.6%)
19	19 (73.68%)	2349.45 (12.57%)	10329.6 (12.09%)

Continued on the next page

Table B.26 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
20	19 (73.68%)	2516.56 (15.24%)	10321.87 (9.17%)
21	19 (73.68%)	3187.22 (25.97%)	10302.9 (1.99%)
22	19 (73.68%)	3358.36 (28.71%)	10301.86 (1.59%)
23	20 (78.95%)	3134.55 (25.13%)	10304.5 (2.59%)
24	20 (78.95%)	3139 (25.2%)	10303.71 (2.29%)
25	21 (84.21%)	1908.97 (5.52%)	10352.24 (20.66%)
26	21 (84.21%)	2058.58 (7.91%)	10336.18 (14.58%)
27	21 (84.21%)	3244.39 (26.89%)	10299.97 (0.88%)
28	22 (89.47%)	2430.74 (13.87%)	10320.22 (8.54%)
29	22 (89.47%)	2597.85 (16.54%)	10312.49 (5.62%)
30	22 (89.47%)	2946.23 (22.12%)	10304.68 (2.66%)
31	22 (89.47%)	2954.33 (22.25%)	10304.62 (2.64%)
32	22 (89.47%)	3189 (26%)	10299.12 (0.56%)
33	23 (94.74%)	3577.81 (32.22%)	10298.49 (0.32%)
34	24 (100%)	2784.12 (19.52%)	10308.24 (4.01%)
35	24 (100%)	2977.99 (22.62%)	10304.45 (2.57%)
36	24 (100%)	3146.87 (25.33%)	10301.93 (1.62%)

Table B.27: Experimental results for instance 27 - c203-D4 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
395.6	5	1304.36	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	6 (0%)	9529.03 (100%)	10658.95 (100%)
2	7 (5%)	8009.26 (77.46%)	10657.89 (99.5%)
3	7 (5%)	8055.53 (78.15%)	10632.76 (87.58%)
4	8 (10%)	7083.56 (63.73%)	10638.84 (90.46%)
5	9 (15%)	5477.41 (39.91%)	10622.68 (82.8%)
6	9 (15%)	5481.61 (39.98%)	10604.52 (74.18%)
7	9 (15%)	5511.49 (40.42%)	10598.03 (71.1%)
8	9 (15%)	5511.75 (40.42%)	10597.64 (70.92%)
9	10 (20%)	5315.88 (37.52%)	10567.27 (56.51%)
10	11 (25%)	3874.89 (16.15%)	10545.29 (46.09%)
11	11 (25%)	4028.1 (18.42%)	10540.65 (43.89%)
12	12 (30%)	3034.17 (3.68%)	10644.73 (93.25%)
13	13 (35%)	2999.35 (3.16%)	10629.47 (86.02%)
14	13 (35%)	2999.56 (3.17%)	10629.24 (85.91%)

Continued on the next page

Table B.27 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
15	13 (35%)	3027.9 (3.59%)	10619.73 (81.4%)
16	13 (35%)	3661.82 (12.99%)	10554.34 (50.38%)
17	13 (35%)	6666.67 (57.55%)	10502.57 (25.82%)
18	14 (40%)	2843.16 (0.85%)	10605.54 (74.67%)
19	14 (40%)	2849.08 (0.93%)	10605.49 (74.64%)
20	14 (40%)	3499.49 (10.58%)	10549.34 (48.01%)
21	14 (40%)	3580.24 (11.78%)	10544.15 (45.55%)
22	14 (40%)	4626.58 (27.3%)	10503.6 (26.31%)
23	15 (45%)	6617.65 (56.82%)	10502.27 (25.68%)
24	15 (45%)	6737.66 (58.6%)	10501.28 (25.21%)
25	16 (50%)	2849.81 (0.95%)	10554.27 (50.35%)
26	16 (50%)	3011.52 (3.34%)	10550.14 (48.39%)
27	16 (50%)	3021.82 (3.5%)	10548.55 (47.63%)
28	16 (50%)	5970.82 (47.23%)	10502.29 (25.69%)
29	16 (50%)	6822.89 (59.87%)	10499.73 (24.48%)
30	17 (55%)	2786.05 (0%)	10563.26 (54.61%)
31	17 (55%)	4266.9 (21.96%)	10507.58 (28.2%)
32	17 (55%)	6158.49 (50.01%)	10458.08 (4.72%)
33	17 (55%)	6720.32 (58.35%)	10457.12 (4.26%)
34	18 (60%)	4063.02 (18.94%)	10507.65 (28.23%)
35	18 (60%)	4064.74 (18.96%)	10507.19 (28.01%)
36	18 (60%)	4072.98 (19.09%)	10505.53 (27.23%)
37	18 (60%)	4202.23 (21%)	10494.65 (22.07%)
38	18 (60%)	4268.32 (21.98%)	10484.56 (17.28%)
39	18 (60%)	4526.07 (25.8%)	10468.41 (9.62%)
40	18 (60%)	6106.36 (49.24%)	10457.32 (4.36%)
41	18 (60%)	6193.73 (50.54%)	10454.68 (3.11%)
42	19 (65%)	4027.2 (18.41%)	10507.42 (28.12%)
43	19 (65%)	4632.54 (27.38%)	10466.37 (8.65%)
44	19 (65%)	4748.7 (29.11%)	10452.45 (2.05%)
45	19 (65%)	6726.58 (58.44%)	10451.37 (1.54%)
46	20 (70%)	3696.83 (13.51%)	10507.83 (28.32%)
47	20 (70%)	3705.07 (13.63%)	10506.17 (27.53%)
48	20 (70%)	3773.04 (14.64%)	10497.74 (23.53%)
49	20 (70%)	4040.91 (18.61%)	10481.59 (15.87%)
50	20 (70%)	4281.19 (22.17%)	10472.05 (11.35%)
51	20 (70%)	4358.89 (23.33%)	10456.46 (3.95%)
52	20 (70%)	4435.17 (24.46%)	10456.43 (3.94%)

Continued on the next page

Table B.27 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
53	20 (70%)	4558.24 (26.28%)	10455.07 (3.29%)
54	20 (70%)	4613.55 (27.1%)	10448.13 (0%)
55	22 (80%)	3405.07 (9.18%)	10507.34 (28.09%)
56	22 (80%)	3413.31 (9.3%)	10505.68 (27.3%)
57	22 (80%)	3418.79 (9.38%)	10505.5 (27.21%)
58	22 (80%)	3497.43 (10.55%)	10497.25 (23.3%)
59	22 (80%)	3781.45 (14.76%)	10481.1 (15.64%)
60	22 (80%)	4020.68 (18.31%)	10471.35 (11.01%)
61	22 (80%)	4172.33 (20.56%)	10463.62 (7.35%)
62	22 (80%)	4310.26 (22.6%)	10456.24 (3.85%)
63	22 (80%)	4571.2 (26.47%)	10454.25 (2.9%)
64	24 (90%)	3448.76 (9.83%)	10501.96 (25.53%)
65	25 (95%)	3170.26 (5.7%)	10521.55 (34.83%)
66	26 (100%)	3701.75 (13.58%)	10495.61 (22.52%)

Table B.28: Experimental results for instance 28 - c203-D5 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
234	4	665.36	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	5 (0%)	8141.03 (100%)	10712.68 (100%)
2	6 (6.25%)	4619.62 (45.98%)	10543.8 (56.36%)
3	7 (12.5%)	3095.95 (22.61%)	10485.46 (41.28%)
4	8 (18.75%)	2882.88 (19.34%)	10416.33 (23.42%)
5	8 (18.75%)	2915.97 (19.84%)	10397.95 (18.67%)
6	8 (18.75%)	3004.73 (21.21%)	10396.26 (18.23%)
7	9 (25%)	2065.56 (6.8%)	10417.28 (23.67%)
8	10 (31.25%)	1717.72 (1.46%)	10448.85 (31.82%)
9	10 (31.25%)	1736.06 (1.74%)	10411.8 (22.25%)
10	11 (37.5%)	1622.39 (0%)	10442.71 (30.24%)
11	11 (37.5%)	1622.64 (0%)	10442.62 (30.21%)
12	11 (37.5%)	1669.33 (0.72%)	10424.31 (25.48%)
13	11 (37.5%)	2108.22 (7.45%)	10395.84 (18.12%)
14	11 (37.5%)	2275.33 (10.02%)	10388.11 (16.13%)
15	12 (43.75%)	2041.49 (6.43%)	10408.35 (21.36%)
16	12 (43.75%)	2495.9 (13.4%)	10380.56 (14.18%)
17	12 (43.75%)	2931.22 (20.08%)	10375.35 (12.83%)

Continued on the next page

Table B.28 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
18	12 (43.75%)	3093.25 (22.56%)	10367.9 (10.9%)
19	12 (43.75%)	3231.03 (24.68%)	10355.86 (7.79%)
20	13 (50%)	3447.19 (27.99%)	10351.9 (6.77%)
21	14 (56.25%)	3631.59 (30.82%)	10341.98 (4.21%)
22	15 (62.5%)	3641.4 (30.97%)	10337.94 (3.16%)
23	16 (68.75%)	3846.08 (34.11%)	10336.33 (2.75%)
24	16 (68.75%)	4013.24 (36.68%)	10334.26 (2.21%)
25	17 (75%)	3770.22 (32.95%)	10333.29 (1.96%)
26	17 (75%)	4028.49 (36.91%)	10331.58 (1.52%)
27	18 (81.25%)	4361.25 (42.02%)	10331.01 (1.37%)
28	18 (81.25%)	5062.74 (52.78%)	10330.72 (1.3%)
29	20 (93.75%)	4094.25 (37.92%)	10330.37 (1.21%)
30	20 (93.75%)	4730.98 (47.69%)	10325.7 (0%)
31	21 (100%)	3903.82 (35%)	10332.02 (1.63%)

Table B.29: Experimental results for instance 29 - c203-D6 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
555.4	8	2978.6	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	14 (0%)	14993.44 (97.95%)	11386.08 (76.64%)
2	14 (0%)	14993.73 (97.95%)	11385.37 (76.38%)
3	14 (0%)	15074.91 (99.01%)	11379.16 (74.1%)
4	14 (0%)	15075.2 (99.01%)	11378.45 (73.84%)
5	14 (0%)	15086.76 (99.17%)	11373.46 (72.01%)
6	15 (4%)	13703.3 (81.08%)	11381.88 (75.1%)
7	15 (4%)	13703.59 (81.08%)	11381.17 (74.84%)
8	15 (4%)	14890.58 (96.6%)	11379.03 (74.05%)
9	16 (8%)	13573.09 (79.38%)	11393.82 (79.49%)
10	16 (8%)	13573.38 (79.38%)	11393.11 (79.22%)
11	16 (8%)	13617.1 (79.95%)	11374.46 (72.37%)
12	16 (8%)	13617.39 (79.96%)	11373.75 (72.11%)
13	16 (8%)	14508.86 (91.61%)	11350.12 (63.43%)
14	16 (8%)	14780.31 (95.16%)	11335.41 (58.03%)
15	17 (12%)	12692.93 (67.87%)	11394.42 (79.71%)
16	17 (12%)	14639.98 (93.33%)	11325.84 (54.51%)
17	17 (12%)	14980.8 (97.78%)	11318.42 (51.79%)

Continued on the next page

Table B.29 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
18	17 (12%)	15150.59 (100%)	11312.5 (49.61%)
19	18 (16%)	11463.36 (51.8%)	11411.83 (86.1%)
20	18 (16%)	11466.21 (51.84%)	11411.76 (86.08%)
21	18 (16%)	11536.76 (52.76%)	11402.04 (82.51%)
22	18 (16%)	11548.91 (52.92%)	11396.82 (80.59%)
23	18 (16%)	12512.53 (65.51%)	11380.25 (74.5%)
24	18 (16%)	12512.76 (65.52%)	11372.5 (71.65%)
25	18 (16%)	12605.75 (66.73%)	11314.98 (50.52%)
26	18 (16%)	12617.43 (66.89%)	11314.86 (50.48%)
27	18 (16%)	12633.99 (67.1%)	11313.41 (49.94%)
28	18 (16%)	12647.81 (67.28%)	11310.87 (49.01%)
29	18 (16%)	13021.17 (72.16%)	11307.59 (47.81%)
30	19 (20%)	9953.15 (32.06%)	11445.78 (98.57%)
31	19 (20%)	10401.91 (37.92%)	11378.17 (73.74%)
32	19 (20%)	10414.09 (38.08%)	11377.19 (73.38%)
33	19 (20%)	10458.48 (38.66%)	11368.18 (70.07%)
34	19 (20%)	11228.16 (48.73%)	11342.52 (60.64%)
35	19 (20%)	11503.02 (52.32%)	11327.81 (55.24%)
36	19 (20%)	11710.91 (55.04%)	11314.89 (50.49%)
37	19 (20%)	11712.99 (55.06%)	11314.28 (50.26%)
38	19 (20%)	11782.69 (55.97%)	11313.3 (49.9%)
39	19 (20%)	11987.53 (58.65%)	11313.22 (49.88%)
40	19 (20%)	12309.94 (62.87%)	11310.52 (48.88%)
41	19 (20%)	12601.32 (66.68%)	11310.12 (48.74%)
42	19 (20%)	12663.4 (67.49%)	11306.69 (47.48%)
43	20 (24%)	9468.31 (25.72%)	11424.3 (90.68%)
44	20 (24%)	9495.27 (26.07%)	11424.01 (90.58%)
45	20 (24%)	9510.93 (26.28%)	11421.92 (89.81%)
46	20 (24%)	10990.53 (45.62%)	11312.06 (49.45%)
47	20 (24%)	11009.63 (45.87%)	11311.25 (49.15%)
48	20 (24%)	11039.03 (46.25%)	11310.66 (48.93%)
49	20 (24%)	11932.86 (57.94%)	11272.75 (35.01%)
50	20 (24%)	12150.93 (60.79%)	11266.52 (32.72%)
51	20 (24%)	13017.47 (72.12%)	11259.64 (30.19%)
52	21 (28%)	9060.81 (20.39%)	11431.42 (93.3%)
53	21 (28%)	9984.87 (32.47%)	11313.63 (50.03%)
54	21 (28%)	10875.16 (44.11%)	11295.7 (43.44%)
55	21 (28%)	10991.2 (45.63%)	11288.54 (40.81%)

Continued on the next page

Table B.29 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
56	21 (28%)	11395.38 (50.91%)	11281.78 (38.32%)
57	21 (28%)	11613.45 (53.76%)	11275.55 (36.04%)
58	21 (28%)	13830.16 (82.74%)	11246.62 (25.41%)
59	22 (32%)	7695.97 (2.55%)	11449.66 (100%)
60	22 (32%)	7813.85 (4.09%)	11415.43 (87.42%)
61	22 (32%)	8166.49 (8.7%)	11380.54 (74.61%)
62	22 (32%)	8167.92 (8.72%)	11379.17 (74.1%)
63	22 (32%)	8170.15 (8.75%)	11377.14 (73.36%)
64	22 (32%)	8202.72 (9.18%)	11375.16 (72.63%)
65	22 (32%)	8264.24 (9.98%)	11372.28 (71.57%)
66	22 (32%)	8310.31 (10.58%)	11361.45 (67.59%)
67	22 (32%)	8697.12 (15.64%)	11343.88 (61.14%)
68	22 (32%)	9262.15 (23.03%)	11330.68 (56.29%)
69	22 (32%)	9714.19 (28.93%)	11313.27 (49.89%)
70	22 (32%)	9722.43 (29.04%)	11311.61 (49.28%)
71	22 (32%)	9760.86 (29.54%)	11304.52 (46.68%)
72	22 (32%)	11398.65 (50.95%)	11246.7 (25.44%)
73	22 (32%)	11401 (50.98%)	11245.88 (25.14%)
74	22 (32%)	11402.82 (51.01%)	11242.87 (24.03%)
75	22 (32%)	11637.75 (54.08%)	11240.42 (23.13%)
76	22 (32%)	12819.65 (69.53%)	11237.21 (21.95%)
77	23 (36%)	7715.28 (2.8%)	11424.95 (90.92%)
78	23 (36%)	7727.35 (2.96%)	11418.17 (88.43%)
79	23 (36%)	7731.08 (3.01%)	11416.25 (87.73%)
80	23 (36%)	8438.04 (12.25%)	11359.55 (66.9%)
81	23 (36%)	8461.47 (12.56%)	11354.82 (65.16%)
82	23 (36%)	9114.1 (21.09%)	11313.7 (50.05%)
83	23 (36%)	9121.31 (21.18%)	11304.53 (46.68%)
84	23 (36%)	9372.13 (24.46%)	11289.64 (41.21%)
85	23 (36%)	9590.2 (27.31%)	11283.41 (38.92%)
86	23 (36%)	10456.74 (38.64%)	11276.53 (36.4%)
87	23 (36%)	14084.49 (86.06%)	11235.96 (21.49%)
88	24 (40%)	7729.19 (2.99%)	11352.94 (64.47%)
89	24 (40%)	7730.62 (3%)	11351.57 (63.96%)
90	24 (40%)	7732.85 (3.03%)	11349.54 (63.22%)
91	24 (40%)	7757.04 (3.35%)	11340.43 (59.87%)
92	24 (40%)	11009.13 (45.86%)	11244.25 (24.54%)
93	24 (40%)	11231.26 (48.77%)	11239.65 (22.85%)

Continued on the next page

Table B.29 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
94	24 (40%)	12097.8 (60.09%)	11232.77 (20.32%)
95	26 (48%)	10512.21 (39.37%)	11267.49 (33.07%)
96	26 (48%)	10730.28 (42.22%)	11261.26 (30.79%)
97	27 (52%)	8805.08 (17.05%)	11312.51 (49.61%)
98	27 (52%)	9914.28 (31.55%)	11273.47 (35.27%)
99	27 (52%)	10150.19 (34.63%)	11247.65 (25.79%)
100	27 (52%)	10365.51 (37.45%)	11245.96 (25.17%)
101	27 (52%)	10368.26 (37.48%)	11241.42 (23.5%)
102	27 (52%)	10952.1 (45.12%)	11239.58 (22.82%)
103	28 (56%)	7699.84 (2.6%)	11404.14 (83.28%)
104	28 (56%)	7715.33 (2.8%)	11403.11 (82.9%)
105	28 (56%)	7715.46 (2.81%)	11400.81 (82.05%)
106	28 (56%)	8614.99 (14.57%)	11312.89 (49.75%)
107	28 (56%)	8639.53 (14.89%)	11302.06 (45.78%)
108	28 (56%)	8889.55 (18.15%)	11286.38 (40.01%)
109	28 (56%)	9107.62 (21.01%)	11280.15 (37.73%)
110	28 (56%)	9974.16 (32.33%)	11273.27 (35.2%)
111	28 (56%)	10082.38 (33.75%)	11245.97 (25.17%)
112	28 (56%)	10085.33 (33.79%)	11245.59 (25.03%)
113	28 (56%)	10308.79 (36.71%)	11245.06 (24.83%)
114	29 (60%)	7742.67 (3.16%)	11323.84 (53.78%)
115	29 (60%)	7766.53 (3.47%)	11313.01 (49.8%)
116	29 (60%)	7767.96 (3.49%)	11311.64 (49.29%)
117	29 (60%)	7954.1 (5.93%)	11311.56 (49.27%)
118	29 (60%)	7955.53 (5.94%)	11310.19 (48.76%)
119	29 (60%)	8027.48 (6.89%)	11297.31 (44.03%)
120	29 (60%)	8985.58 (19.41%)	11279.55 (37.51%)
121	29 (60%)	9212.62 (22.38%)	11263.4 (31.57%)
122	29 (60%)	9431.01 (25.23%)	11255.11 (28.53%)
123	29 (60%)	9649.08 (28.08%)	11248.88 (26.24%)
124	29 (60%)	10003.35 (32.71%)	11244.46 (24.61%)
125	29 (60%)	10270.47 (36.21%)	11244.44 (24.61%)
126	30 (64%)	9681.94 (28.51%)	11247 (25.55%)
127	30 (64%)	9772.89 (29.7%)	11246.39 (25.32%)
128	30 (64%)	9893.58 (31.28%)	11243.71 (24.34%)
129	30 (64%)	9905.09 (31.43%)	11242.09 (23.74%)
130	30 (64%)	10505.79 (39.28%)	11237.04 (21.89%)
131	31 (68%)	8785.46 (16.79%)	11288.95 (40.96%)

Continued on the next page

Table B.29 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
132	31 (68%)	8996.88 (19.56%)	11272.8 (35.03%)
133	31 (68%)	9049.75 (20.25%)	11260.29 (30.43%)
134	31 (68%)	9592.1 (27.34%)	11247.37 (25.68%)
135	31 (68%)	9834.5 (30.51%)	11236.57 (21.72%)
136	31 (68%)	10701.04 (41.83%)	11229.69 (19.19%)
137	31 (68%)	10914.3 (44.62%)	11224.48 (17.27%)
138	32 (72%)	9614.27 (27.63%)	11245.76 (25.09%)
139	32 (72%)	11132.71 (47.48%)	11221.07 (16.02%)
140	33 (76%)	11314.66 (49.86%)	11211.51 (12.51%)
141	33 (76%)	11841.3 (56.74%)	11208.11 (11.26%)
142	33 (76%)	12371.74 (63.67%)	11204.55 (9.95%)
143	33 (76%)	13872.73 (83.3%)	11186.06 (3.16%)
144	33 (76%)	14299.11 (88.87%)	11180.55 (1.14%)
145	34 (80%)	9643.48 (28.01%)	11244.39 (24.59%)
146	34 (80%)	12579.72 (66.39%)	11202.56 (9.22%)
147	34 (80%)	13712.37 (81.2%)	11182.47 (1.84%)
148	34 (80%)	13715.53 (81.24%)	11181.46 (1.47%)
149	34 (80%)	13800.32 (82.35%)	11179.25 (0.66%)
150	35 (84%)	12725.36 (68.3%)	11193.3 (5.82%)
151	35 (84%)	13633.36 (80.17%)	11181.94 (1.65%)
152	35 (84%)	14255.39 (88.3%)	11177.86 (0.15%)
153	35 (84%)	14369.47 (89.79%)	11177.46 (0%)
154	36 (88%)	12919.01 (70.83%)	11191.32 (5.09%)
155	36 (88%)	13016.9 (72.11%)	11188.71 (4.13%)
156	37 (92%)	7500.76 (0%)	11431.4 (93.29%)
157	37 (92%)	7503.18 (0.03%)	11424.37 (90.71%)
158	37 (92%)	7508.55 (0.1%)	11417.4 (88.15%)
159	37 (92%)	9646.75 (28.05%)	11244.29 (24.55%)
160	37 (92%)	10840.01 (43.65%)	11228.93 (18.91%)
161	37 (92%)	12836.09 (69.74%)	11188.33 (3.99%)
162	37 (92%)	12994.01 (71.81%)	11186.21 (3.21%)
163	39 (100%)	7557.13 (0.74%)	11374.67 (72.45%)
164	39 (100%)	7557.51 (0.74%)	11368.83 (70.3%)
165	39 (100%)	7559.93 (0.77%)	11368.13 (70.05%)

Table B.30: Experimental results for instance 30 - c203-D7 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB$'_1$ for f$_1$**	**LB$'_2$ for f$_2$**	**Reference value for f$_3$**
549.5	4	665.36	9601.72
Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
1	13 (0%)	11108.69 (83.9%)	11182.01 (73.96%)
2	14 (5.26%)	9711.48 (65.88%)	11181.3 (73.67%)
3	14 (5.26%)	9722.44 (66.03%)	11170.01 (69.08%)
4	14 (5.26%)	9939.38 (68.82%)	11166.5 (67.66%)
5	15 (10.53%)	8602.74 (51.59%)	11194.28 (78.94%)
6	15 (10.53%)	8618.75 (51.8%)	11184 (74.77%)
7	15 (10.53%)	8640.3 (52.07%)	11181.07 (73.58%)
8	15 (10.53%)	8651.26 (52.22%)	11169.78 (68.99%)
9	15 (10.53%)	9647.99 (65.07%)	11150.31 (61.08%)
10	15 (10.53%)	10904.13 (81.26%)	11146.07 (59.36%)
11	15 (10.53%)	10910.57 (81.34%)	11145.58 (59.16%)
12	15 (10.53%)	12357.63 (100%)	11145.28 (59.04%)
13	16 (15.79%)	8127.98 (45.47%)	11140.38 (57.05%)
14	17 (21.05%)	7787.2 (41.08%)	11141.56 (57.53%)
15	17 (21.05%)	7787.3 (41.08%)	11140.88 (57.25%)
16	17 (21.05%)	7802.36 (41.27%)	11140.38 (57.05%)
17	17 (21.05%)	7804.2 (41.3%)	11139.94 (56.87%)
18	17 (21.05%)	7824.76 (41.56%)	11128.35 (52.17%)
19	17 (21.05%)	8821.49 (54.41%)	11108.88 (44.26%)
20	18 (26.32%)	6852.67 (29.03%)	11143.21 (58.2%)
21	18 (26.32%)	6852.77 (29.03%)	11142.53 (57.92%)
22	18 (26.32%)	6890.23 (29.51%)	11130 (52.84%)
23	18 (26.32%)	8493.42 (50.18%)	11119.99 (48.77%)
24	18 (26.32%)	8599.94 (51.55%)	11112.82 (45.86%)
25	18 (26.32%)	8716.56 (53.06%)	11104.24 (42.38%)
26	18 (26.32%)	9089.54 (57.87%)	11096.42 (39.2%)
27	18 (26.32%)	10332.74 (73.89%)	11081.96 (33.33%)
28	18 (26.32%)	10332.84 (73.9%)	11081.28 (33.05%)
29	18 (26.32%)	10452.41 (75.44%)	11081.21 (33.02%)
30	19 (31.58%)	5967.05 (17.61%)	11246.14 (100%)
31	19 (31.58%)	5970.93 (17.66%)	11203.26 (82.59%)
32	19 (31.58%)	7558.89 (38.13%)	11121.64 (49.44%)
33	19 (31.58%)	7665.41 (39.51%)	11114.47 (46.53%)
34	19 (31.58%)	8495.01 (50.2%)	11108.18 (43.98%)

Continued on the next page

Table B.30 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
35	19 (31.58%)	8984.61 (56.51%)	11091.78 (37.32%)
36	19 (31.58%)	10446.19 (75.36%)	11076.97 (31.3%)
37	20 (36.84%)	9233.76 (59.73%)	11090.64 (36.85%)
38	20 (36.84%)	10896.48 (81.16%)	11073.91 (30.06%)
39	21 (42.11%)	5950.44 (17.4%)	11237.59 (96.53%)
40	21 (42.11%)	5954.32 (17.45%)	11194.71 (79.11%)
41	21 (42.11%)	6845.64 (28.94%)	11149.01 (60.56%)
42	21 (42.11%)	6889.75 (29.51%)	11137.11 (55.72%)
43	21 (42.11%)	8057.98 (44.57%)	11113.96 (46.32%)
44	21 (42.11%)	8080.16 (44.85%)	11083.19 (33.83%)
45	21 (42.11%)	9514.08 (63.34%)	11082.23 (33.44%)
46	21 (42.11%)	10217.13 (72.4%)	11079.57 (32.36%)
47	21 (42.11%)	10815.48 (80.12%)	11076.07 (30.94%)
48	22 (47.37%)	4601.15 (0%)	11208.47 (84.7%)
49	22 (47.37%)	4606.44 (0.07%)	11206.33 (83.83%)
50	22 (47.37%)	5792.88 (15.36%)	11148.86 (60.5%)
51	22 (47.37%)	5830.44 (15.85%)	11135.65 (55.13%)
52	22 (47.37%)	6261.81 (21.41%)	11127.49 (51.82%)
53	23 (52.63%)	6930.47 (30.03%)	11119.13 (48.42%)
54	23 (52.63%)	6990.47 (30.8%)	11117.58 (47.79%)
55	23 (52.63%)	7036.99 (31.4%)	11111.96 (45.51%)
56	23 (52.63%)	7843.87 (41.81%)	11105.23 (42.78%)
57	23 (52.63%)	9508.68 (63.27%)	11083.1 (33.79%)
58	24 (57.89%)	6768.92 (27.95%)	11121.52 (49.39%)
59	24 (57.89%)	7258.52 (34.26%)	11105.12 (42.73%)
60	24 (57.89%)	7713.43 (40.12%)	11100.12 (40.7%)
61	24 (57.89%)	9402.14 (61.9%)	11082.56 (33.57%)
62	24 (57.89%)	9560.67 (63.94%)	11080.38 (32.69%)
63	25 (63.16%)	4695.56 (1.22%)	11145.68 (59.2%)
64	25 (63.16%)	4795.47 (2.51%)	11144.1 (58.56%)
65	25 (63.16%)	5728.83 (14.54%)	11136.5 (55.48%)
66	25 (63.16%)	8751.9 (53.51%)	11067.89 (27.61%)
67	25 (63.16%)	8758.13 (53.59%)	11064.55 (26.26%)
68	25 (63.16%)	8947.53 (56.04%)	11041.68 (16.97%)
69	26 (68.42%)	6723.92 (27.37%)	11127.07 (51.65%)
70	26 (68.42%)	7419.91 (36.34%)	11084.53 (34.37%)
71	26 (68.42%)	8308.26 (47.79%)	11071.45 (29.06%)
72	26 (68.42%)	9247.97 (59.91%)	11039.3 (16%)

Continued on the next page

Table B.30 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
73	27 (73.68%)	4632.86 (0.41%)	11186.35 (75.72%)
74	27 (73.68%)	4636.74 (0.46%)	11184.21 (74.85%)
75	27 (73.68%)	4649.96 (0.63%)	11145.65 (59.19%)
76	27 (73.68%)	5035.01 (5.59%)	11129.09 (52.47%)
77	27 (73.68%)	5130.92 (6.83%)	11129.06 (52.45%)
78	27 (73.68%)	5168.48 (7.31%)	11115.85 (47.09%)
79	27 (73.68%)	6165.21 (20.16%)	11096.38 (39.18%)
80	27 (73.68%)	7458.56 (36.84%)	11084.05 (34.18%)
81	27 (73.68%)	7530.87 (37.77%)	11081.35 (33.08%)
82	27 (73.68%)	7532.01 (37.79%)	11080.34 (32.67%)
83	27 (73.68%)	7785.54 (41.05%)	11074.85 (30.44%)
84	27 (73.68%)	8462.94 (49.79%)	11018.74 (7.65%)
85	27 (73.68%)	8853.99 (54.83%)	11008.61 (3.54%)
86	28 (78.95%)	5837.14 (15.93%)	11107.49 (43.7%)
87	28 (78.95%)	5943.66 (17.31%)	11100.32 (40.78%)
88	28 (78.95%)	6433.26 (23.62%)	11083.92 (34.12%)
89	28 (78.95%)	7080.92 (31.97%)	11083.09 (33.79%)
90	28 (78.95%)	8192.87 (46.31%)	11019.28 (7.87%)
91	28 (78.95%)	8205.15 (46.46%)	11018.52 (7.57%)
92	28 (78.95%)	8418.48 (49.21%)	11012.17 (4.99%)
93	29 (84.21%)	4664.99 (0.82%)	11141.81 (57.63%)
94	29 (84.21%)	5068.06 (6.02%)	11119.2 (48.45%)
95	29 (84.21%)	5202.16 (7.75%)	11111.41 (45.29%)
96	29 (84.21%)	7294.42 (34.72%)	11081.38 (33.09%)
97	29 (84.21%)	7902.69 (42.56%)	11014.56 (5.96%)
98	29 (84.21%)	10123.02 (71.19%)	10999.89 (0%)
99	30 (89.47%)	5870.82 (16.37%)	11103.05 (41.89%)
100	30 (89.47%)	5977.34 (17.74%)	11095.88 (38.98%)
101	30 (89.47%)	6350.79 (22.56%)	11088.84 (36.12%)
102	30 (89.47%)	6945.46 (30.22%)	11083.15 (33.81%)
103	30 (89.47%)	7021.87 (31.21%)	11082.57 (33.58%)
104	30 (89.47%)	7025.55 (31.26%)	11082.21 (33.43%)
105	30 (89.47%)	7102.1 (32.24%)	11080.69 (32.81%)
106	30 (89.47%)	7125.88 (32.55%)	11044.42 (18.08%)
107	30 (89.47%)	7370.4 (35.7%)	11039.21 (15.97%)
108	30 (89.47%)	7549.74 (38.01%)	11018.75 (7.66%)
109	31 (94.74%)	6129.24 (19.7%)	11092.78 (37.72%)
110	31 (94.74%)	6328.06 (22.26%)	11088.34 (35.92%)

Continued on the next page

Table B.30 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
111	31 (94.74%)	7574.11 (38.33%)	11018.58 (7.59%)
112	31 (94.74%)	7764.08 (40.78%)	11017.97 (7.34%)
113	32 (100%)	6975.84 (30.62%)	11082.94 (33.73%)
114	32 (100%)	7057.9 (31.67%)	11080.22 (32.62%)
115	32 (100%)	7098.97 (32.2%)	11080.1 (32.57%)
116	32 (100%)	7569.07 (38.26%)	11015.03 (6.15%)

Table B.31: Experimental results for instance 31 - c203-D8 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
342.1	4	959.31	9601.72
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	15 (0%)	16514.86 (59.6%)	11543.81 (55.4%)
2	16 (5%)	15535.87 (50.37%)	11534.83 (53.2%)
3	18 (15%)	15208.37 (47.28%)	11470.38 (37.4%)
4	19 (20%)	14322.25 (38.92%)	11468.14 (36.85%)
5	20 (25%)	13544.51 (31.58%)	11466.99 (36.57%)
6	21 (30%)	15153.16 (46.76%)	11432.63 (28.15%)
7	21 (30%)	15153.33 (46.76%)	11432.29 (28.07%)
8	22 (35%)	13251.13 (28.81%)	11482.58 (40.39%)
9	22 (35%)	14834.41 (43.75%)	11429.71 (27.44%)
10	22 (35%)	15286.78 (48.02%)	11429.35 (27.35%)
11	23 (40%)	12374.81 (20.54%)	11498.6 (44.32%)
12	23 (40%)	13345.37 (29.7%)	11467.07 (36.59%)
13	23 (40%)	13479.99 (30.97%)	11455.69 (33.8%)
14	23 (40%)	13879.6 (34.74%)	11428.91 (27.24%)
15	23 (40%)	14081.25 (36.64%)	11428.02 (27.02%)
16	23 (40%)	14660.32 (42.11%)	11428 (27.02%)
17	23 (40%)	15028.71 (45.58%)	11422.34 (25.63%)
18	24 (45%)	11554.4 (12.8%)	11494.9 (43.41%)
19	24 (45%)	12469.05 (21.43%)	11483.09 (40.52%)
20	24 (45%)	12603.67 (22.7%)	11471.71 (37.73%)
21	24 (45%)	13474.97 (30.92%)	11430.15 (27.54%)
22	24 (45%)	13546.62 (31.6%)	11429.83 (27.46%)
23	24 (45%)	18970.05 (82.77%)	11348.08 (7.43%)
24	24 (45%)	20524.34 (97.44%)	11342.06 (5.95%)
25	24 (45%)	20610.89 (98.25%)	11341.38 (5.78%)

Continued on the next page

Table B.31 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
26	25 (50%)	11648.64 (13.69%)	11479.39 (39.61%)
27	25 (50%)	11783.26 (14.96%)	11468.01 (36.82%)
28	25 (50%)	12744.65 (24.03%)	11462.54 (35.48%)
29	25 (50%)	12845.01 (24.98%)	11437.48 (29.34%)
30	25 (50%)	13857.26 (34.53%)	11426.89 (26.74%)
31	25 (50%)	13922.08 (35.14%)	11424.31 (26.11%)
32	25 (50%)	18811.04 (81.27%)	11349.77 (7.84%)
33	25 (50%)	18966.36 (82.73%)	11346.95 (7.15%)
34	25 (50%)	19289.99 (85.79%)	11339.44 (5.31%)
35	25 (50%)	20796.15 (100%)	11338.43 (5.06%)
36	26 (55%)	11904.11 (16.1%)	11458.84 (34.58%)
37	26 (55%)	12238.89 (19.26%)	11443.97 (30.93%)
38	26 (55%)	15004.06 (45.35%)	11397.63 (19.57%)
39	26 (55%)	15473.58 (49.78%)	11393.76 (18.62%)
40	26 (55%)	16532.17 (59.77%)	11386.61 (16.87%)
41	26 (55%)	16912.47 (63.36%)	11334.66 (4.14%)
42	26 (55%)	16999.64 (64.18%)	11334.48 (4.09%)
43	26 (55%)	17014.18 (64.32%)	11333.9 (3.95%)
44	27 (60%)	11059.62 (8.13%)	11524.67 (50.71%)
45	27 (60%)	12215.75 (19.04%)	11429.92 (27.49%)
46	27 (60%)	12226.91 (19.14%)	11426.22 (26.58%)
47	27 (60%)	12338.15 (20.19%)	11420.25 (25.12%)
48	27 (60%)	12790 (24.46%)	11415.12 (23.86%)
49	27 (60%)	13423.85 (30.44%)	11407.08 (21.89%)
50	27 (60%)	14955 (44.89%)	11401.04 (20.41%)
51	27 (60%)	16750.68 (61.83%)	11335.39 (4.32%)
52	27 (60%)	16754.36 (61.86%)	11335.03 (4.23%)
53	27 (60%)	16754.78 (61.87%)	11334.81 (4.17%)
54	27 (60%)	16767.92 (61.99%)	11333.98 (3.97%)
55	27 (60%)	16770.68 (62.02%)	11333.53 (3.86%)
56	28 (65%)	10197.92 (0%)	11725.76 (100%)
57	28 (65%)	10759.21 (5.3%)	11484.67 (40.91%)
58	28 (65%)	11219.47 (9.64%)	11471.74 (37.74%)
59	28 (65%)	11575.4 (13%)	11431.18 (27.8%)
60	28 (65%)	11575.46 (13%)	11429.81 (27.46%)
61	28 (65%)	11576.47 (13.01%)	11428.59 (27.16%)
62	28 (65%)	14656.27 (42.07%)	11360.71 (10.52%)
63	28 (65%)	15125.79 (46.5%)	11356.84 (9.57%)

Continued on the next page

Table B.31 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
64	28 (65%)	16184.38 (56.49%)	11349.69 (7.82%)
65	28 (65%)	17446.29 (68.39%)	11317.78 (0%)
66	29 (70%)	10664.67 (4.4%)	11500.18 (44.71%)
67	29 (70%)	10666.1 (4.42%)	11498.81 (44.37%)
68	29 (70%)	10668.33 (4.44%)	11496.78 (43.87%)
69	29 (70%)	12809.24 (24.64%)	11414.69 (23.75%)
70	29 (70%)	13123.58 (27.61%)	11407.24 (21.93%)
71	29 (70%)	13433.56 (30.53%)	11401.29 (20.47%)
72	29 (70%)	13521.51 (31.36%)	11377.29 (14.59%)
73	29 (70%)	13973.36 (35.62%)	11372.16 (13.33%)
74	29 (70%)	14607.21 (41.6%)	11364.12 (11.36%)
75	29 (70%)	15080.14 (46.07%)	11360.25 (10.41%)
76	29 (70%)	16992.83 (64.11%)	11332.78 (3.68%)
77	30 (75%)	11899.82 (16.06%)	11424.71 (26.21%)
78	30 (75%)	13320.04 (29.46%)	11387.41 (17.07%)
79	30 (75%)	16217.81 (56.8%)	11334.81 (4.17%)
80	30 (75%)	16220.27 (56.82%)	11331.63 (3.39%)
81	30 (75%)	16232.18 (56.94%)	11329.87 (2.96%)
82	30 (75%)	16533.99 (59.78%)	11325.63 (1.92%)
83	35 (100%)	10199.84 (0.02%)	11658.35 (83.48%)
84	35 (100%)	10201.27 (0.03%)	11656.98 (83.14%)

Table B.32: Experimental results for instance 32 - c101-E1 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
1981.6	2	215.06	9828.93
Solution S/N	**f_1**	**f_2**	**f_3**
1	6 (0%)	4061.46 (84.15%)	10487.53 (84.38%)
2	6 (0%)	4158.63 (91.22%)	10455.03 (62.09%)
3	6 (0%)	4167.43 (91.86%)	10454.38 (61.65%)
4	6 (0%)	4277.78 (99.89%)	10419.92 (38.02%)
5	7 (2.86%)	4151.35 (90.69%)	10461.85 (66.77%)
6	8 (5.71%)	3993.93 (79.24%)	10473.35 (74.65%)
7	8 (5.71%)	4066.01 (84.49%)	10447.18 (56.71%)
8	8 (5.71%)	4188.51 (93.4%)	10415.06 (34.69%)
9	10 (11.43%)	3924.41 (74.19%)	10473.22 (74.56%)
10	10 (11.43%)	4135.87 (89.57%)	10424.12 (40.9%)

Continued on the next page

Table B.32 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
11	11 (14.29%)	3788.04 (64.27%)	10482.08 (80.64%)
12	11 (14.29%)	3892.73 (71.88%)	10474.87 (75.7%)
13	11 (14.29%)	3990 (78.96%)	10448.77 (57.8%)
14	11 (14.29%)	4002.54 (79.87%)	10440.37 (52.04%)
15	11 (14.29%)	4075.15 (85.15%)	10424.09 (40.88%)
16	12 (17.14%)	4130.58 (89.18%)	10394.2 (20.39%)
17	12 (17.14%)	4134.3 (89.45%)	10393.72 (20.06%)
18	13 (20%)	3946.04 (75.76%)	10432.84 (46.88%)
19	13 (20%)	4062.2 (84.21%)	10390.68 (17.98%)
20	13 (20%)	4269.35 (99.28%)	10389.81 (17.38%)
21	13 (20%)	4279.3 (100%)	10385.98 (14.75%)
22	15 (25.71%)	3925.01 (74.23%)	10432.3 (46.51%)
23	15 (25.71%)	3925.87 (74.29%)	10427.23 (43.03%)
24	16 (28.57%)	3768.4 (62.84%)	10464.15 (68.35%)
25	16 (28.57%)	3785.09 (64.05%)	10391.11 (18.27%)
26	16 (28.57%)	3975.91 (77.93%)	10391.07 (18.24%)
27	16 (28.57%)	3978.93 (78.15%)	10387.86 (16.04%)
28	17 (31.43%)	3711.23 (58.68%)	10457.08 (63.5%)
29	18 (34.29%)	3572.26 (48.57%)	10479.99 (79.21%)
30	18 (34.29%)	3673.67 (55.95%)	10452.69 (60.49%)
31	18 (34.29%)	4070.48 (84.81%)	10383.03 (12.73%)
32	19 (37.14%)	3642.2 (53.66%)	10464.54 (68.61%)
33	19 (37.14%)	3691.16 (57.22%)	10391.91 (18.82%)
34	19 (37.14%)	3800.89 (65.2%)	10390.62 (17.94%)
35	19 (37.14%)	3816.64 (66.35%)	10389.64 (17.26%)
36	19 (37.14%)	3818.3 (66.47%)	10386.47 (15.09%)
37	19 (37.14%)	3995.95 (79.39%)	10378.13 (9.37%)
38	20 (40%)	3628.37 (52.65%)	10467.85 (70.88%)
39	20 (40%)	3639.61 (53.47%)	10393.44 (19.87%)
40	20 (40%)	3653.45 (54.48%)	10390.47 (17.83%)
41	21 (42.86%)	3581.1 (49.21%)	10391.54 (18.57%)
42	21 (42.86%)	3582.76 (49.33%)	10388.75 (16.65%)
43	23 (48.57%)	3456.85 (40.18%)	10456.19 (62.89%)
44	23 (48.57%)	3678.12 (56.27%)	10384.88 (14%)
45	24 (51.43%)	3421.2 (37.58%)	10477.94 (77.8%)
46	24 (51.43%)	3425.93 (37.93%)	10456.4 (63.03%)
47	24 (51.43%)	3428.61 (38.12%)	10455.57 (62.46%)
48	24 (51.43%)	3508.58 (43.94%)	10394.39 (20.52%)

Continued on the next page

Table B.32 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
49	24 (51.43%)	3636.34 (53.23%)	10382.49 (12.36%)
50	24 (51.43%)	3638.89 (53.42%)	10382.15 (12.13%)
51	25 (54.29%)	3474.24 (41.44%)	10394.39 (20.52%)
52	26 (57.14%)	3376.16 (34.31%)	10386.51 (15.12%)
53	27 (60%)	3311.9 (29.63%)	10461.02 (66.2%)
54	27 (60%)	3317.06 (30.01%)	10459.64 (65.25%)
55	27 (60%)	3339.82 (31.66%)	10389.36 (17.07%)
56	27 (60%)	3496.61 (43.07%)	10385.68 (14.55%)
57	27 (60%)	3547.3 (46.75%)	10364.46 (0%)
58	28 (62.86%)	3515.62 (44.45%)	10366.11 (1.13%)
59	29 (65.71%)	3261.56 (25.97%)	10461.03 (66.21%)
60	29 (65.71%)	3479.28 (41.81%)	10368.96 (3.09%)
61	31 (71.43%)	3169.59 (19.28%)	10482.22 (80.73%)
62	32 (74.29%)	3137.91 (16.98%)	10483.87 (81.87%)
63	32 (74.29%)	3148.68 (17.76%)	10457.2 (63.58%)
64	32 (74.29%)	3168.36 (19.19%)	10455.93 (62.71%)
65	32 (74.29%)	3282.12 (27.47%)	10444.57 (54.92%)
66	32 (74.29%)	3370.4 (33.89%)	10384.85 (13.98%)
67	33 (77.14%)	3077.25 (12.56%)	10429.21 (44.39%)
68	34 (80%)	3034.29 (9.44%)	10426.44 (42.49%)
69	34 (80%)	3188.35 (20.64%)	10416.1 (35.4%)
70	35 (82.86%)	2998.36 (6.83%)	10419.9 (38.01%)
71	35 (82.86%)	3006.79 (7.44%)	10418.7 (37.19%)
72	35 (82.86%)	3286.7 (27.8%)	10384.33 (13.62%)
73	35 (82.86%)	3295.13 (28.41%)	10383.13 (12.8%)
74	36 (85.71%)	3275.12 (26.96%)	10395.29 (21.14%)
75	36 (85.71%)	3289.92 (28.03%)	10372.43 (5.46%)
76	36 (85.71%)	3397.76 (35.88%)	10367.25 (1.91%)
77	37 (88.57%)	2979.88 (5.48%)	10448.76 (57.8%)
78	37 (88.57%)	3099.06 (14.15%)	10394.74 (20.76%)
79	37 (88.57%)	3286.25 (27.77%)	10393.77 (20.09%)
80	38 (91.43%)	2920.02 (1.13%)	10463.06 (67.6%)
81	38 (91.43%)	2920.53 (1.16%)	10414.42 (34.25%)
82	38 (91.43%)	3001.59 (7.06%)	10386.44 (15.07%)
83	39 (94.29%)	2904.53 (0%)	10510.32 (100%)
84	39 (94.29%)	2910.54 (0.44%)	10481.52 (80.26%)
85	39 (94.29%)	2931.66 (1.97%)	10410.12 (31.3%)
86	41 (100%)	3199.28 (21.44%)	10367.78 (2.28%)

Table B.33: Experimental results for instance 33 - c101-E2 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB$'_1$ for f$_1$**	**LB$'_2$ for f$_2$**	**Reference value for f$_3$**
2098.2	2	215.06	9828.93
Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
1	8 (0%)	5209.53 (81.24%)	10647.11 (100%)
2	8 (0%)	5250.97 (83.41%)	10628.37 (82.33%)
3	8 (0%)	5318.43 (86.95%)	10622.32 (76.62%)
4	8 (0%)	5387.8 (90.6%)	10610.94 (65.89%)
5	8 (0%)	5459.26 (94.35%)	10606.83 (62.01%)
6	9 (2.22%)	5154.43 (78.34%)	10625.67 (79.78%)
7	10 (4.44%)	5130.52 (77.09%)	10633.87 (87.51%)
8	10 (4.44%)	5191.67 (80.3%)	10615.44 (70.13%)
9	10 (4.44%)	5566.9 (100%)	10589.34 (45.52%)
10	11 (6.67%)	5070.96 (73.96%)	10635.03 (88.61%)
11	11 (6.67%)	5079.76 (74.42%)	10634.38 (87.99%)
12	11 (6.67%)	5089.53 (74.93%)	10633.09 (86.78%)
13	11 (6.67%)	5123.31 (76.71%)	10617.25 (71.84%)
14	11 (6.67%)	5132.11 (77.17%)	10616.6 (71.23%)
15	11 (6.67%)	5139.96 (77.58%)	10607.61 (62.75%)
16	11 (6.67%)	5506.18 (96.81%)	10589.31 (45.49%)
17	11 (6.67%)	5548.18 (99.02%)	10583.99 (40.47%)
18	12 (8.89%)	5065.35 (73.67%)	10639.79 (93.1%)
19	12 (8.89%)	5200.12 (80.74%)	10589.11 (45.3%)
20	12 (8.89%)	5360.23 (89.15%)	10589.04 (45.23%)
21	12 (8.89%)	5477.58 (95.31%)	10588.51 (44.73%)
22	13 (11.11%)	4936.11 (66.88%)	10641.66 (94.86%)
23	13 (11.11%)	5025.11 (71.55%)	10640.43 (93.7%)
24	13 (11.11%)	5126.19 (76.86%)	10614.19 (68.95%)
25	13 (11.11%)	5443.26 (93.51%)	10584.13 (40.6%)
26	14 (13.33%)	4951.07 (67.66%)	10636.34 (89.84%)
27	14 (13.33%)	4970.39 (68.68%)	10635.82 (89.35%)
28	14 (13.33%)	5004.97 (70.49%)	10632.12 (85.86%)
29	14 (13.33%)	5018.78 (71.22%)	10615.61 (70.29%)
30	14 (13.33%)	5070.88 (73.96%)	10612.51 (67.37%)
31	14 (13.33%)	5119.78 (76.52%)	10579.59 (36.32%)
32	15 (15.56%)	5095.87 (75.27%)	10587.79 (44.05%)
33	16 (17.78%)	4944.63 (67.33%)	10624.99 (79.14%)
34	16 (17.78%)	5036.31 (72.14%)	10588.95 (45.15%)

Continued on the next page

Table B.33 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
35	17 (20%)	4904.06 (65.2%)	10628.48 (82.43%)
36	17 (20%)	4925.1 (66.3%)	10616.68 (71.3%)
37	17 (20%)	4954.38 (67.84%)	10613.6 (68.4%)
38	18 (22.22%)	4830.35 (61.33%)	10621.54 (75.88%)
39	18 (22.22%)	4855.26 (62.63%)	10612.27 (67.14%)
40	18 (22.22%)	4860.17 (62.89%)	10611.26 (66.19%)
41	18 (22.22%)	4876.38 (63.74%)	10611.07 (66.01%)
42	18 (22.22%)	4882.7 (64.07%)	10603.76 (59.12%)
43	18 (22.22%)	4887.61 (64.33%)	10602.75 (58.16%)
44	18 (22.22%)	4915.81 (65.81%)	10598.66 (54.31%)
45	18 (22.22%)	4950.16 (67.62%)	10597.71 (53.41%)
46	18 (22.22%)	5006.88 (70.6%)	10582.77 (39.32%)
47	18 (22.22%)	5192.01 (80.32%)	10578.93 (35.7%)
48	19 (24.44%)	4747.25 (56.96%)	10617.48 (72.06%)
49	19 (24.44%)	4986.24 (69.51%)	10583.76 (40.25%)
50	20 (26.67%)	4742.19 (56.7%)	10613.29 (68.1%)
51	20 (26.67%)	4840.96 (61.88%)	10606.72 (61.91%)
52	20 (26.67%)	4874.69 (63.65%)	10584.9 (41.33%)
53	21 (28.89%)	4695.69 (54.26%)	10617.54 (72.11%)
54	21 (28.89%)	4704.89 (54.74%)	10611.66 (66.57%)
55	21 (28.89%)	4709.8 (55%)	10610.65 (65.61%)
56	21 (28.89%)	4753.04 (57.27%)	10588.51 (44.73%)
57	21 (28.89%)	4835.99 (61.62%)	10582.04 (38.63%)
58	21 (28.89%)	4925.71 (66.33%)	10581.68 (38.29%)
59	22 (31.11%)	4722.51 (55.66%)	10588.21 (44.45%)
60	23 (33.33%)	4628.1 (50.71%)	10634.14 (87.77%)
61	23 (33.33%)	4768.26 (58.07%)	10583.97 (40.45%)
62	24 (35.56%)	4547.52 (46.48%)	10611.19 (66.12%)
63	24 (35.56%)	4611.41 (49.83%)	10598.35 (54.01%)
64	24 (35.56%)	4912.79 (65.65%)	10557.96 (15.92%)
65	24 (35.56%)	4969.94 (68.66%)	10555.8 (13.88%)
66	25 (37.78%)	4502 (44.09%)	10603.34 (58.72%)
67	25 (37.78%)	4554.35 (46.83%)	10585.56 (41.95%)
68	25 (37.78%)	4621.81 (50.38%)	10579.51 (36.24%)
69	25 (37.78%)	4626.82 (50.64%)	10571.76 (28.94%)
70	25 (37.78%)	4739.65 (56.56%)	10561.59 (19.34%)
71	25 (37.78%)	4875.49 (63.7%)	10556.33 (14.38%)
72	25 (37.78%)	4878.04 (63.83%)	10555.99 (14.06%)

Continued on the next page

Table B.33 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
73	25 (37.78%)	4885.5 (64.22%)	10555.16 (13.28%)
74	25 (37.78%)	4885.89 (64.24%)	10553.8 (12%)
75	26 (40%)	4531.66 (45.64%)	10599.43 (55.03%)
76	27 (42.22%)	4551.02 (46.66%)	10577.5 (34.35%)
77	28 (44.44%)	4477.01 (42.77%)	10615.26 (69.96%)
78	28 (44.44%)	4486.64 (43.28%)	10577.09 (33.96%)
79	28 (44.44%)	4486.82 (43.29%)	10576.3 (33.22%)
80	28 (44.44%)	4489.36 (43.42%)	10574.84 (31.84%)
81	28 (44.44%)	4720.31 (55.55%)	10559.98 (17.83%)
82	28 (44.44%)	4782.18 (58.8%)	10556.53 (14.57%)
83	28 (44.44%)	4784.72 (58.93%)	10555.07 (13.19%)
84	28 (44.44%)	4785.16 (58.95%)	10552.16 (10.45%)
85	29 (46.67%)	4439.45 (40.8%)	10610.87 (65.82%)
86	29 (46.67%)	4465.62 (42.18%)	10588.51 (44.73%)
87	29 (46.67%)	4559.97 (47.13%)	10558.58 (16.5%)
88	30 (48.89%)	4428.32 (40.22%)	10586.88 (43.2%)
89	30 (48.89%)	4467.35 (42.27%)	10557.26 (15.26%)
90	31 (51.11%)	4381 (37.73%)	10644.75 (97.77%)
91	31 (51.11%)	4396.64 (38.55%)	10588.53 (44.75%)
92	31 (51.11%)	4403.15 (38.9%)	10571.09 (28.3%)
93	32 (53.33%)	4296.84 (33.31%)	10584.87 (41.3%)
94	32 (53.33%)	4353.74 (36.3%)	10583.75 (40.24%)
95	32 (53.33%)	4354.62 (36.35%)	10569.31 (26.62%)
96	32 (53.33%)	4403.19 (38.9%)	10558.94 (16.84%)
97	32 (53.33%)	4578.96 (48.13%)	10553.06 (11.3%)
98	33 (55.56%)	4393.49 (38.39%)	10559.64 (17.5%)
99	33 (55.56%)	4566.44 (47.47%)	10553.31 (11.53%)
100	34 (57.78%)	4353.73 (36.3%)	10556.15 (14.21%)
101	34 (57.78%)	4428.63 (40.23%)	10555.98 (14.05%)
102	35 (60%)	4293.56 (33.14%)	10587.19 (43.49%)
103	35 (60%)	4295.54 (33.24%)	10586.1 (42.46%)
104	35 (60%)	4320.47 (34.55%)	10558.61 (16.53%)
105	35 (60%)	4322.83 (34.68%)	10555.37 (13.48%)
106	35 (60%)	4557.53 (47%)	10553.65 (11.86%)
107	37 (64.44%)	4258.75 (31.31%)	10631.04 (84.84%)
108	38 (66.67%)	4193.5 (27.89%)	10585.69 (42.07%)
109	38 (66.67%)	4195.48 (27.99%)	10583.6 (40.1%)
110	38 (66.67%)	4258.35 (31.29%)	10577.87 (34.7%)

Continued on the next page

Table B.33 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
111	38 (66.67%)	4353.03 (36.26%)	10555.35 (13.46%)
112	38 (66.67%)	4374.6 (37.4%)	10554.85 (12.99%)
113	38 (66.67%)	4405.21 (39%)	10548.54 (7.04%)
114	41 (73.33%)	4008.37 (18.17%)	10615.65 (70.33%)
115	41 (73.33%)	4023.11 (18.94%)	10573.84 (30.9%)
116	41 (73.33%)	4079.09 (21.88%)	10565.78 (23.3%)
117	42 (75.56%)	3968.13 (16.05%)	10614.74 (69.47%)
118	42 (75.56%)	3975.11 (16.42%)	10613.75 (68.54%)
119	42 (75.56%)	3978.87 (16.62%)	10577.46 (34.31%)
120	42 (75.56%)	4043.72 (20.02%)	10569.64 (26.94%)
121	45 (82.22%)	3899.24 (12.44%)	10581.38 (38.01%)
122	45 (82.22%)	3964.09 (15.84%)	10573.56 (30.63%)
123	45 (82.22%)	4135.42 (24.84%)	10560.16 (17.99%)
124	45 (82.22%)	4199.49 (28.2%)	10552.34 (10.62%)
125	46 (84.44%)	4071.34 (21.47%)	10559.57 (17.44%)
126	47 (86.67%)	3881.84 (11.52%)	10625.97 (80.06%)
127	47 (86.67%)	4034.73 (19.55%)	10559.81 (17.66%)
128	48 (88.89%)	3813.48 (7.93%)	10573.69 (30.76%)
129	48 (88.89%)	4000.39 (17.75%)	10559.81 (17.66%)
130	49 (91.11%)	3762.63 (5.26%)	10615.93 (70.59%)
131	49 (91.11%)	3784.09 (6.39%)	10573.69 (30.76%)
132	49 (91.11%)	3961.71 (15.72%)	10556.47 (14.51%)
133	50 (93.33%)	3783.03 (6.33%)	10589.19 (45.37%)
134	51 (95.56%)	3712.39 (2.63%)	10619.42 (73.88%)
135	51 (95.56%)	3723.53 (3.21%)	10569.3 (26.62%)
136	51 (95.56%)	3914.28 (13.23%)	10558.83 (16.74%)
137	51 (95.56%)	3915.92 (13.31%)	10554.2 (12.37%)
138	52 (97.78%)	3662.39 (0%)	10610.76 (65.72%)
139	52 (97.78%)	3677.34 (0.78%)	10602.28 (57.72%)
140	52 (97.78%)	3871.76 (10.99%)	10552.08 (10.37%)
141	52 (97.78%)	3906 (12.79%)	10550.57 (8.95%)
142	53 (100%)	3862.99 (10.53%)	10548.9 (7.38%)
143	53 (100%)	3927.06 (13.9%)	10541.08 (0%)

Table B.34: Experimental results for instance 34 - c101-E3 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB$'_1$ for f$_1$**	**LB$'_2$ for f$_2$**	**Reference value for f$_3$**
2040.9	2	215.06	9828.93
Solution S/N	**f$_1$**	**f$_2$**	**f$_3$**
1	8 (0%)	5377.42 (67.2%)	10660.57 (100%)
2	8 (0%)	5439.73 (71.46%)	10638.31 (71.02%)
3	8 (0%)	5492.08 (75.04%)	10628.7 (58.5%)
4	8 (0%)	5562.15 (79.82%)	10622.55 (50.49%)
5	9 (2.86%)	5297.06 (61.71%)	10649.76 (85.92%)
6	9 (2.86%)	5857.38 (100%)	10605.34 (28.09%)
7	10 (5.71%)	5270.64 (59.9%)	10652.5 (89.49%)
8	10 (5.71%)	5277.17 (60.35%)	10642.28 (76.18%)
9	10 (5.71%)	5320.82 (63.33%)	10632.45 (63.39%)
10	10 (5.71%)	5397.03 (68.54%)	10617.68 (44.15%)
11	10 (5.71%)	5498.36 (75.47%)	10606.04 (29%)
12	10 (5.71%)	5499.84 (75.57%)	10604.84 (27.43%)
13	10 (5.71%)	5576.05 (80.77%)	10590.07 (8.2%)
14	11 (8.57%)	5270.35 (59.88%)	10650.13 (86.41%)
15	12 (11.43%)	5182.44 (53.88%)	10633.73 (65.05%)
16	12 (11.43%)	5182.76 (53.9%)	10633.44 (64.67%)
17	12 (11.43%)	5258.65 (59.08%)	10618.96 (45.82%)
18	12 (11.43%)	5258.97 (59.11%)	10618.67 (45.44%)
19	13 (14.29%)	5132.53 (50.47%)	10635.3 (67.1%)
20	13 (14.29%)	5234.33 (57.42%)	10612.73 (37.71%)
21	14 (17.14%)	5127.6 (50.13%)	10650.04 (86.29%)
22	15 (20%)	5071.45 (46.29%)	10643.86 (78.24%)
23	15 (20%)	5073.47 (46.43%)	10641.4 (75.04%)
24	15 (20%)	5079.92 (46.87%)	10633.5 (64.75%)
25	15 (20%)	5084.4 (47.18%)	10631.04 (61.55%)
26	15 (20%)	5152.63 (51.84%)	10630.66 (61.05%)
27	15 (20%)	5164.69 (52.66%)	10618.73 (45.52%)
28	15 (20%)	5165.04 (52.69%)	10615.41 (41.2%)
29	16 (22.86%)	5015.45 (42.46%)	10646.4 (81.55%)
30	17 (25.71%)	4954.11 (38.27%)	10660.18 (99.49%)
31	17 (25.71%)	4963.31 (38.9%)	10641.05 (74.58%)
32	17 (25.71%)	4973.01 (39.56%)	10640.92 (74.41%)
33	17 (25.71%)	4988.17 (40.6%)	10633.26 (64.44%)
34	17 (25.71%)	5009.65 (42.07%)	10631.36 (61.97%)

Continued on the next page

Table B.34 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
35	17 (25.71%)	5037.25 (43.95%)	10629.86 (60.01%)
36	17 (25.71%)	5049.92 (44.82%)	10628.4 (58.11%)
37	17 (25.71%)	5079.02 (46.81%)	10616.25 (42.29%)
38	18 (28.57%)	4924.87 (36.27%)	10636.62 (68.82%)
39	18 (28.57%)	4998.81 (41.33%)	10625.43 (54.24%)
40	19 (31.43%)	4875.35 (32.89%)	10654.84 (92.54%)
41	19 (31.43%)	4888.53 (33.79%)	10639.47 (72.53%)
42	19 (31.43%)	4889.7 (33.87%)	10626.64 (55.82%)
43	19 (31.43%)	4963.64 (38.92%)	10615.45 (41.25%)
44	19 (31.43%)	4993.88 (40.99%)	10614.1 (39.49%)
45	19 (31.43%)	5015.36 (42.46%)	10612.2 (37.02%)
46	20 (34.29%)	4850.4 (31.18%)	10650.47 (86.85%)
47	20 (34.29%)	5151.11 (51.73%)	10604.79 (27.37%)
48	21 (37.14%)	4809.01 (28.36%)	10642.47 (76.43%)
49	21 (37.14%)	4822.19 (29.26%)	10627.1 (56.42%)
50	21 (37.14%)	4826.04 (29.52%)	10622.59 (50.55%)
51	21 (37.14%)	4896.13 (34.31%)	10615.91 (41.85%)
52	21 (37.14%)	5052.83 (45.02%)	10606.19 (29.19%)
53	21 (37.14%)	5120.28 (49.63%)	10604.79 (27.37%)
54	22 (40%)	4772.67 (25.87%)	10645.32 (80.14%)
55	22 (40%)	4789.7 (27.04%)	10625.44 (54.26%)
56	22 (40%)	4831.51 (29.89%)	10618.9 (45.74%)
57	22 (40%)	4863.64 (32.09%)	10614.25 (39.69%)
58	22 (40%)	4905.45 (34.95%)	10607.71 (31.17%)
59	22 (40%)	5009.77 (42.08%)	10605.91 (28.83%)
60	22 (40%)	5015.23 (42.45%)	10605.05 (27.71%)
61	22 (40%)	5056.97 (45.3%)	10599.18 (20.07%)
62	22 (40%)	5097.38 (48.06%)	10594.43 (13.88%)
63	22 (40%)	5098.64 (48.15%)	10593.12 (12.17%)
64	22 (40%)	5140.37 (51%)	10592.28 (11.08%)
65	23 (42.86%)	4762.29 (25.16%)	10639.82 (72.98%)
66	23 (42.86%)	5020.63 (42.82%)	10602.03 (23.78%)
67	23 (42.86%)	5098.56 (48.14%)	10594.31 (13.72%)
68	24 (45.71%)	4731.46 (23.06%)	10639.82 (72.98%)
69	24 (45.71%)	4767.76 (25.54%)	10636.13 (68.18%)
70	24 (45.71%)	4985.2 (40.4%)	10605.83 (28.72%)
71	24 (45.71%)	4989.8 (40.71%)	10602.03 (23.78%)
72	24 (45.71%)	5066.07 (45.92%)	10592.65 (11.56%)

Continued on the next page

Table B.34 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
73	25 (48.57%)	4707.86 (21.44%)	10638.4 (71.13%)
74	25 (48.57%)	4971.81 (39.48%)	10605.97 (28.91%)
75	25 (48.57%)	4978.99 (39.97%)	10605.88 (28.79%)
76	26 (51.43%)	4674.68 (19.18%)	10640.21 (73.49%)
77	26 (51.43%)	4677.81 (19.39%)	10634.57 (66.15%)
78	26 (51.43%)	4746.33 (24.07%)	10633.08 (64.21%)
79	26 (51.43%)	4771.23 (25.77%)	10630.19 (60.44%)
80	26 (51.43%)	4776.26 (26.12%)	10630.1 (60.33%)
81	26 (51.43%)	4791.39 (27.15%)	10618.2 (44.83%)
82	26 (51.43%)	4838.18 (30.35%)	10605.92 (28.84%)
83	26 (51.43%)	4978.29 (39.92%)	10604.19 (26.59%)
84	27 (54.29%)	4843.65 (30.72%)	10601.56 (23.16%)
85	28 (57.14%)	4755.38 (24.69%)	10617.03 (43.31%)
86	28 (57.14%)	4807.31 (28.24%)	10604.41 (26.87%)
87	29 (60%)	4617.91 (15.3%)	10640.71 (74.14%)
88	29 (60%)	4650.66 (17.53%)	10636.28 (68.37%)
89	29 (60%)	4725.85 (22.67%)	10620.19 (47.42%)
90	29 (60%)	4782.09 (26.52%)	10604.06 (26.42%)
91	29 (60%)	4794.65 (27.37%)	10603.49 (25.68%)
92	30 (62.86%)	4676.54 (19.3%)	10632.69 (63.7%)
93	30 (62.86%)	4706.84 (21.37%)	10614.3 (39.75%)
94	30 (62.86%)	4758.31 (24.89%)	10606.34 (29.39%)
95	30 (62.86%)	4766.02 (25.42%)	10605.89 (28.8%)
96	31 (65.71%)	4607.1 (14.56%)	10634.93 (66.61%)
97	31 (65.71%)	4631.08 (16.2%)	10624.96 (53.63%)
98	31 (65.71%)	4681.7 (19.66%)	10617.18 (43.5%)
99	31 (65.71%)	4684.18 (19.83%)	10605.33 (28.07%)
100	31 (65.71%)	4724.99 (22.61%)	10605.01 (27.66%)
101	31 (65.71%)	4760.94 (25.07%)	10594.59 (14.09%)
102	32 (68.57%)	4646.26 (17.23%)	10610.67 (35.03%)
103	32 (68.57%)	4704.3 (21.2%)	10601.08 (22.54%)
104	32 (68.57%)	4728.61 (22.86%)	10600.17 (21.35%)
105	32 (68.57%)	4778.24 (26.25%)	10589.89 (7.97%)
106	33 (71.43%)	4546.54 (10.42%)	10630.54 (60.9%)
107	33 (71.43%)	4627.26 (15.94%)	10616.46 (42.57%)
108	33 (71.43%)	4651.73 (17.61%)	10594.27 (13.67%)
109	33 (71.43%)	4734.08 (23.24%)	10583.77 (0%)
110	34 (74.29%)	4571.97 (12.16%)	10625.82 (54.75%)

Continued on the next page

Table B.34 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
111	35 (77.14%)	4554.41 (10.96%)	10628.9 (58.76%)
112	35 (77.14%)	4643.76 (17.06%)	10614.26 (39.7%)
113	36 (80%)	4500.34 (7.26%)	10658.91 (97.84%)
114	36 (80%)	4520.34 (8.63%)	10647.89 (83.49%)
115	36 (80%)	4521.72 (8.72%)	10633.62 (64.91%)
116	36 (80%)	4523.67 (8.86%)	10633.15 (64.3%)
117	36 (80%)	4537.24 (9.78%)	10633.12 (64.26%)
118	36 (80%)	4597.93 (13.93%)	10611.58 (36.21%)
119	37 (82.86%)	4511.68 (8.04%)	10629.56 (59.62%)
120	38 (85.71%)	4460.45 (4.54%)	10631.15 (61.69%)
121	38 (85.71%)	4531.48 (9.39%)	10629.28 (59.26%)
122	40 (91.43%)	4445.27 (3.5%)	10633.94 (65.33%)
123	40 (91.43%)	4476.89 (5.66%)	10624.8 (53.42%)
124	41 (94.29%)	4394.07 (0%)	10651.21 (87.81%)
125	41 (94.29%)	4414.08 (1.37%)	10624.01 (52.4%)
126	41 (94.29%)	4597.74 (13.92%)	10614.24 (39.67%)
127	41 (94.29%)	4613.35 (14.99%)	10610.45 (34.74%)
128	42 (97.14%)	4556.89 (11.13%)	10619.13 (46.04%)
129	42 (97.14%)	4582.46 (12.87%)	10615.22 (40.95%)
130	43 (100%)	4545.58 (10.35%)	10619.01 (45.89%)

Table B.35: Experimental results for instance 35 - c101-E4 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	**LB'_1 for f_1**	**LB'_2 for f_2**	**Reference value for f_3**
2186.1	10	867.18	9828.93
Solution S/N	**f_1**	**f_2**	**f_3**
1	17 (0%)	10863.57 (86.73%)	11485.55 (100%)
2	17 (0%)	11000.54 (92.97%)	11478.51 (95.24%)
3	18 (1.69%)	10800.76 (83.86%)	11485.31 (99.84%)
4	19 (3.39%)	10726.05 (80.46%)	11478.83 (95.46%)
5	19 (3.39%)	11051.26 (95.28%)	11450.31 (76.17%)
6	20 (5.08%)	10820.26 (84.75%)	11473.26 (91.69%)
7	20 (5.08%)	10826.53 (85.04%)	11449.99 (75.96%)
8	21 (6.78%)	10571.83 (73.43%)	11471.5 (90.5%)
9	21 (6.78%)	10578.61 (73.74%)	11470.48 (89.81%)
10	21 (6.78%)	10593.49 (74.41%)	11463.46 (85.06%)
11	21 (6.78%)	10733.19 (80.78%)	11451.83 (77.2%)

Continued on the next page

Table B.35 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
12	21 (6.78%)	10738.96 (81.05%)	11445.09 (72.64%)
13	21 (6.78%)	10761.02 (82.05%)	11441.73 (70.37%)
14	22 (8.47%)	10544.55 (72.18%)	11465.74 (86.61%)
15	22 (8.47%)	10651.81 (77.07%)	11453.75 (78.5%)
16	22 (8.47%)	10679.64 (78.34%)	11443.65 (71.67%)
17	24 (11.86%)	10479.99 (69.24%)	11485.3 (99.83%)
18	24 (11.86%)	10497.67 (70.05%)	11481.62 (97.34%)
19	25 (13.56%)	10393.61 (65.3%)	11476.55 (93.91%)
20	25 (13.56%)	10423.84 (66.68%)	11474.63 (92.62%)
21	25 (13.56%)	10450.01 (67.87%)	11465.85 (86.68%)
22	25 (13.56%)	10612.5 (75.28%)	11449.87 (75.88%)
23	25 (13.56%)	10627.46 (75.96%)	11449.52 (75.64%)
24	25 (13.56%)	10634.56 (76.29%)	11446.51 (73.6%)
25	25 (13.56%)	10639.65 (76.52%)	11441.86 (70.46%)
26	26 (15.25%)	10584.5 (74%)	11422.84 (57.6%)
27	27 (16.95%)	10303.28 (61.18%)	11458.03 (81.39%)
28	28 (18.64%)	10348.6 (63.25%)	11450.09 (76.02%)
29	29 (20.34%)	10274.21 (59.86%)	11447.23 (74.09%)
30	29 (20.34%)	10275.28 (59.91%)	11446.26 (73.43%)
31	29 (20.34%)	10330.61 (62.43%)	11436.53 (66.86%)
32	29 (20.34%)	11153.77 (99.96%)	11414.78 (52.15%)
33	29 (20.34%)	11154.69 (100%)	11413.25 (51.12%)
34	30 (22.03%)	10458.04 (68.24%)	11427.62 (60.83%)
35	32 (25.42%)	10101.56 (51.99%)	11443.3 (71.43%)
36	33 (27.12%)	10955.26 (90.91%)	11399.29 (41.68%)
37	36 (32.2%)	10829.8 (85.19%)	11389.41 (35%)
38	37 (33.9%)	10055.99 (49.91%)	11431.67 (63.57%)
39	37 (33.9%)	10059.86 (50.09%)	11430.08 (62.49%)
40	40 (38.98%)	9932.99 (44.3%)	11434.82 (65.7%)
41	40 (38.98%)	10104.66 (52.13%)	11381.47 (29.63%)
42	40 (38.98%)	10324.4 (62.15%)	11378.01 (27.29%)
43	40 (38.98%)	10386.29 (64.97%)	11377.08 (26.66%)
44	41 (40.68%)	10038.48 (49.11%)	11380.78 (29.16%)
45	41 (40.68%)	10039.96 (49.18%)	11379.58 (28.35%)
46	41 (40.68%)	10045.65 (49.44%)	11375.89 (25.86%)
47	41 (40.68%)	10110.1 (52.38%)	11375.19 (25.38%)
48	41 (40.68%)	10266.52 (59.51%)	11367.44 (20.14%)
49	41 (40.68%)	10511.26 (70.67%)	11362.2 (16.6%)

Continued on the next page

Table B.35 – *Continued from the previous page*

Solution S/N	**f_1**	**f_2**	**f_3**
50	44 (45.76%)	9777.83 (37.23%)	11439.27 (68.71%)
51	45 (47.46%)	9884.74 (42.1%)	11437.8 (67.71%)
52	46 (49.15%)	9832.55 (39.72%)	11432.67 (64.25%)
53	49 (54.24%)	9643.4 (31.1%)	11450.13 (76.05%)
54	50 (55.93%)	9597.66 (29.01%)	11447.32 (74.15%)
55	50 (55.93%)	9601.61 (29.19%)	11447.09 (74%)
56	50 (55.93%)	9605.74 (29.38%)	11445.87 (73.17%)
57	50 (55.93%)	9610.25 (29.59%)	11445.03 (72.6%)
58	50 (55.93%)	10017.81 (48.17%)	11377.93 (27.23%)
59	50 (55.93%)	10032.05 (48.82%)	11375.74 (25.75%)
60	52 (59.32%)	9644.43 (31.15%)	11442.15 (70.66%)
61	53 (61.02%)	9541.14 (26.44%)	11437.96 (67.82%)
62	53 (61.02%)	9619.24 (30%)	11416.61 (53.39%)
63	53 (61.02%)	9907.22 (43.13%)	11362.11 (16.54%)
64	54 (62.71%)	9486.73 (23.96%)	11434.63 (65.57%)
65	54 (62.71%)	9490.86 (24.15%)	11433.41 (64.75%)
66	54 (62.71%)	9539.75 (26.37%)	11405.93 (46.17%)
67	54 (62.71%)	9540.03 (26.39%)	11404.23 (45.02%)
68	54 (62.71%)	9548.2 (26.76%)	11402.62 (43.93%)
69	58 (69.49%)	9385.4 (19.34%)	11444.34 (72.14%)
70	58 (69.49%)	9391.14 (19.6%)	11442.02 (70.57%)
71	58 (69.49%)	9395.77 (19.81%)	11441.47 (70.2%)
72	58 (69.49%)	9490.02 (24.11%)	11378.14 (27.38%)
73	58 (69.49%)	9490.3 (24.12%)	11376.44 (26.23%)
74	58 (69.49%)	9497.25 (24.44%)	11375.67 (25.71%)
75	58 (69.49%)	9517.13 (25.34%)	11375.11 (25.33%)
76	58 (69.49%)	9521.6 (25.55%)	11374.61 (24.99%)
77	58 (69.49%)	9567.49 (27.64%)	11374.48 (24.9%)
78	61 (74.58%)	9245.67 (12.97%)	11431.74 (63.62%)
79	62 (76.27%)	9202.13 (10.98%)	11423.14 (57.8%)
80	62 (76.27%)	9226.17 (12.08%)	11412.85 (50.85%)
81	62 (76.27%)	9239.68 (12.69%)	11412.11 (50.34%)
82	62 (76.27%)	9265.45 (13.87%)	11409.47 (48.56%)
83	62 (76.27%)	9269.46 (14.05%)	11383.31 (30.87%)
84	65 (81.36%)	9129.92 (7.69%)	11414.75 (52.13%)
85	65 (81.36%)	9132.49 (7.81%)	11413.77 (51.47%)
86	65 (81.36%)	9226.08 (12.07%)	11380.8 (29.18%)
87	65 (81.36%)	9300.4 (15.46%)	11378.75 (27.79%)

Continued on the next page

Table B.35 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
88	66 (83.05%)	9104.53 (6.53%)	11412.57 (50.66%)
89	66 (83.05%)	9114.38 (6.98%)	11411.64 (50.03%)
90	66 (83.05%)	9115.88 (7.05%)	11410.78 (49.45%)
91	66 (83.05%)	9118.46 (7.17%)	11410.37 (49.17%)
92	66 (83.05%)	9138.47 (8.08%)	11410.15 (49.02%)
93	66 (83.05%)	9205.55 (11.14%)	11368.22 (20.67%)
94	66 (83.05%)	9208.07 (11.25%)	11367.51 (20.19%)
95	66 (83.05%)	10130.69 (53.32%)	11347.07 (6.37%)
96	69 (88.14%)	9062.28 (4.61%)	11408.72 (48.05%)
97	70 (89.83%)	9033.63 (3.3%)	11412.79 (50.8%)
98	70 (89.83%)	9039.42 (3.56%)	11411.57 (49.98%)
99	70 (89.83%)	9050.28 (4.06%)	11410.42 (49.2%)
100	70 (89.83%)	9056.07 (4.32%)	11409.2 (48.38%)
101	70 (89.83%)	9121.19 (7.29%)	11340.96 (2.24%)
102	70 (89.83%)	9126.48 (7.53%)	11340.56 (1.97%)
103	70 (89.83%)	9272.17 (14.18%)	11337.65 (0%)
104	73 (94.92%)	8976.12 (0.68%)	11404.74 (45.36%)
105	74 (96.61%)	8961.96 (0.03%)	11407.35 (47.13%)
106	74 (96.61%)	8969.04 (0.36%)	11405.22 (45.69%)
107	74 (96.61%)	8971.73 (0.48%)	11398.24 (40.97%)
108	74 (96.61%)	9016.2 (2.51%)	11361.78 (16.32%)
109	75 (98.31%)	8961.24 (0%)	11407.46 (47.2%)
110	75 (98.31%)	8965.37 (0.19%)	11406.24 (46.38%)
111	75 (98.31%)	8970.74 (0.43%)	11404.06 (44.9%)
112	75 (98.31%)	8980.57 (0.88%)	11395.55 (39.15%)
113	75 (98.31%)	9000.56 (1.79%)	11364.6 (18.22%)
114	75 (98.31%)	9006.35 (2.06%)	11363.38 (17.4%)
115	75 (98.31%)	9011.72 (2.3%)	11361.2 (15.92%)
116	75 (98.31%)	9089.14 (5.83%)	11346.95 (6.29%)
117	75 (98.31%)	9089.92 (5.87%)	11346.6 (6.05%)
118	75 (98.31%)	9090.44 (5.89%)	11346.51 (5.99%)
119	75 (98.31%)	9093.27 (6.02%)	11345.73 (5.46%)
120	75 (98.31%)	9095.81 (6.14%)	11344.33 (4.52%)
121	76 (100%)	8991.53 (1.38%)	11379.15 (28.06%)
122	76 (100%)	8999.95 (1.76%)	11364.71 (18.3%)
123	76 (100%)	9091.01 (5.92%)	11345.35 (5.21%)
124	76 (100%)	9092.58 (5.99%)	11344.44 (4.59%)
125	76 (100%)	9095.14 (6.1%)	11344.13 (4.38%)

Continued on the next page

Table B.35 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
126	76 (100%)	9100.51 (6.35%)	11341.95 (2.91%)
127	76 (100%)	9128.92 (7.64%)	11340.08 (1.64%)

Table B.36: Experimental results for instance 36 - c101-E5 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
1613.2	2	215.06	9828.93
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	17 (0%)	7517.24 (96.04%)	11100.82 (100%)
2	19 (6.06%)	7327.59 (83.77%)	10997.36 (49.98%)
3	19 (6.06%)	7349.13 (85.16%)	10995.98 (49.32%)
4	19 (6.06%)	7356.25 (85.62%)	10992.65 (47.71%)
5	20 (9.09%)	7085.38 (68.09%)	11031.96 (66.71%)
6	21 (12.12%)	7047.1 (65.61%)	11030.47 (65.99%)
7	22 (15.15%)	6986.44 (61.69%)	11028.25 (64.92%)
8	22 (15.15%)	7094.86 (68.7%)	10957.96 (30.94%)
9	23 (18.18%)	6936.53 (58.46%)	11027.57 (64.59%)
10	24 (21.21%)	6907.83 (56.6%)	11005.55 (53.94%)
11	25 (24.24%)	6893.7 (55.69%)	11017.35 (59.65%)
12	27 (30.3%)	6809.71 (50.25%)	11005.25 (53.8%)
13	27 (30.3%)	6818.49 (50.82%)	10949.46 (26.83%)
14	27 (30.3%)	6848.49 (52.76%)	10910.73 (8.1%)
15	27 (30.3%)	6970.03 (60.63%)	10897.14 (1.53%)
16	27 (30.3%)	7159.42 (72.88%)	10896.52 (1.23%)
17	27 (30.3%)	7578.45 (100%)	10896.25 (1.1%)
18	28 (33.33%)	6769.15 (47.63%)	11043.82 (72.44%)
19	29 (36.36%)	6748.76 (46.31%)	11028.88 (65.22%)
20	29 (36.36%)	6771.98 (47.81%)	11023.62 (62.68%)
21	31 (42.42%)	6624.77 (38.28%)	10941.17 (22.82%)
22	31 (42.42%)	6825.74 (51.29%)	10905.23 (5.44%)
23	32 (45.45%)	6609.64 (37.3%)	10950.35 (27.26%)
24	32 (45.45%)	6610.8 (37.38%)	10906.51 (6.06%)
25	32 (45.45%)	6643.32 (39.48%)	10906.31 (5.97%)
26	32 (45.45%)	6788.12 (48.85%)	10894.54 (0.28%)
27	33 (48.48%)	6588.32 (35.92%)	10946.96 (25.62%)
28	34 (51.52%)	6556.73 (33.88%)	10942.96 (23.68%)
29	35 (54.55%)	6529.01 (32.08%)	10941.37 (22.92%)

Continued on the next page

Table B.36 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
30	36 (57.58%)	6425.8 (25.41%)	10934.3 (19.5%)
31	36 (57.58%)	6706.53 (43.57%)	10904.7 (5.19%)
32	37 (60.61%)	6397.14 (23.55%)	10936.23 (20.43%)
33	37 (60.61%)	6406.86 (24.18%)	10935.5 (20.08%)
34	37 (60.61%)	6499.06 (30.15%)	10908.21 (6.88%)
35	37 (60.61%)	6501.19 (30.28%)	10907.39 (6.49%)
36	37 (60.61%)	6649.2 (39.86%)	10903.2 (4.46%)
37	37 (60.61%)	6654.19 (40.19%)	10902.84 (4.29%)
38	38 (63.64%)	6392.96 (23.28%)	10909.3 (7.41%)
39	38 (63.64%)	6414.09 (24.65%)	10906.9 (6.25%)
40	39 (66.67%)	6338.54 (19.76%)	10931.73 (18.25%)
41	39 (66.67%)	6345.17 (20.19%)	10931.63 (18.21%)
42	39 (66.67%)	6373.63 (22.03%)	10898.01 (1.95%)
43	39 (66.67%)	6375.32 (22.14%)	10897.19 (1.56%)
44	41 (72.73%)	6325.82 (18.94%)	11020.77 (61.3%)
45	41 (72.73%)	6349.53 (20.47%)	10910.88 (8.18%)
46	42 (75.76%)	6212.28 (11.59%)	10922.73 (13.9%)
47	42 (75.76%)	6304.54 (17.56%)	10893.97 (0%)
48	46 (87.88%)	6187.78 (10%)	10972.78 (38.1%)
49	47 (90.91%)	6174.35 (9.13%)	10969.96 (36.74%)
50	47 (90.91%)	6182.69 (9.67%)	10942.93 (23.67%)
51	47 (90.91%)	6185.62 (9.86%)	10942.91 (23.66%)
52	48 (93.94%)	6033.23 (0%)	10925.95 (15.46%)
53	50 (100%)	6257.27 (14.5%)	10902.01 (3.89%)

Table B.37: Experimental results for instance 37 - c101-E6 (Heuristic Approach 3 for the SCD-VRPTW-3)

Runtime (sec)	$\mathbf{LB'_1}$ **for** $\mathbf{f_1}$	$\mathbf{LB'_2}$ **for** $\mathbf{f_2}$	**Reference value for** $\mathbf{f_3}$
2714.9	8	834.31	9828.93
Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
1	25 (0%)	14804.14 (84.18%)	12048.27 (99.58%)
2	25 (0%)	15315.92 (100%)	12032.76 (94.52%)
3	27 (5.13%)	14305.78 (68.78%)	12049.08 (99.85%)
4	28 (7.69%)	14036.46 (60.45%)	12049.55 (100%)
5	28 (7.69%)	14273.54 (67.78%)	12047.45 (99.31%)
6	28 (7.69%)	14275.42 (67.84%)	12046.21 (98.91%)
7	28 (7.69%)	14289.92 (68.29%)	12042.75 (97.78%)

Continued on the next page

Table B.37 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
8	28 (7.69%)	14323.82 (69.33%)	12039.39 (96.68%)
9	29 (10.26%)	14233.1 (66.53%)	12047.81 (99.43%)
10	30 (12.82%)	14014.66 (59.78%)	12040.86 (97.16%)
11	30 (12.82%)	14170.71 (64.6%)	12037.81 (96.17%)
12	31 (15.38%)	13913.12 (56.64%)	12043.49 (98.02%)
13	31 (15.38%)	13931.97 (57.22%)	12028.09 (92.99%)
14	31 (15.38%)	14196.33 (65.39%)	11973.98 (75.32%)
15	32 (17.95%)	13853.08 (54.78%)	12040.27 (96.97%)
16	32 (17.95%)	13856.15 (54.88%)	12040.26 (96.97%)
17	32 (17.95%)	13859.96 (54.99%)	12026.06 (92.33%)
18	32 (17.95%)	13863.05 (55.09%)	12025.91 (92.28%)
19	32 (17.95%)	13871.47 (55.35%)	12025.49 (92.14%)
20	33 (20.51%)	13732.99 (51.07%)	12046.66 (99.06%)
21	34 (23.08%)	13823.19 (53.86%)	12041.1 (97.24%)
22	35 (25.64%)	13594.55 (46.79%)	11937.04 (63.26%)
23	35 (25.64%)	13598.99 (46.93%)	11936.29 (63.01%)
24	35 (25.64%)	13693.67 (49.85%)	11928.84 (60.58%)
25	37 (30.77%)	13561.11 (45.76%)	12048.24 (99.57%)
26	37 (30.77%)	13677.77 (49.36%)	11892.53 (48.72%)
27	37 (30.77%)	13693.34 (49.84%)	11888.56 (47.43%)
28	37 (30.77%)	13697.78 (49.98%)	11887.81 (47.18%)
29	41 (41.03%)	13501.23 (43.91%)	11878.53 (44.15%)
30	42 (43.59%)	13354.27 (39.36%)	11947.54 (66.69%)
31	42 (43.59%)	13362.35 (39.61%)	11946.28 (66.28%)
32	42 (43.59%)	13363.51 (39.65%)	11940.04 (64.24%)
33	42 (43.59%)	13380.13 (40.16%)	11936.85 (63.2%)
34	42 (43.59%)	13408.82 (41.05%)	11898.31 (50.61%)
35	42 (43.59%)	13410.3 (41.09%)	11897.11 (50.22%)
36	42 (43.59%)	13413.42 (41.19%)	11896.36 (49.98%)
37	42 (43.59%)	13420.02 (41.4%)	11896.3 (49.96%)
38	42 (43.59%)	13435.59 (41.88%)	11878.14 (44.03%)
39	43 (46.15%)	13406.11 (40.97%)	11874.13 (42.72%)
40	45 (51.28%)	13241.25 (35.87%)	11885.76 (46.51%)
41	45 (51.28%)	13249.33 (36.12%)	11884.5 (46.1%)
42	46 (53.85%)	13212.84 (34.99%)	11934.21 (62.34%)
43	46 (53.85%)	13508.22 (44.12%)	11859.73 (38.01%)
44	47 (56.41%)	13093.45 (31.3%)	11925.82 (59.6%)
45	47 (56.41%)	13094.21 (31.32%)	11925.44 (59.47%)

Continued on the next page

Table B.37 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
46	47 (56.41%)	13094.99 (31.35%)	11925.09 (59.36%)
47	47 (56.41%)	13154.27 (33.18%)	11897.35 (50.3%)
48	48 (58.97%)	13072.51 (30.65%)	11921.9 (58.32%)
49	48 (58.97%)	13228.89 (35.49%)	11820.89 (25.33%)
50	49 (61.54%)	13034.53 (29.48%)	11920.14 (57.74%)
51	49 (61.54%)	13035.29 (29.5%)	11919.76 (57.62%)
52	49 (61.54%)	13036.07 (29.53%)	11919.41 (57.5%)
53	49 (61.54%)	13084.43 (31.02%)	11895.11 (49.57%)
54	49 (61.54%)	13085.19 (31.05%)	11894.73 (49.44%)
55	49 (61.54%)	13085.97 (31.07%)	11894.38 (49.33%)
56	49 (61.54%)	13117.16 (32.03%)	11821.68 (25.59%)
57	49 (61.54%)	13117.89 (32.06%)	11821.44 (25.51%)
58	49 (61.54%)	13120.33 (32.13%)	11821.09 (25.4%)
59	49 (61.54%)	13127.14 (32.34%)	11820.79 (25.3%)
60	49 (61.54%)	13212.96 (34.99%)	11815.05 (23.42%)
61	50 (64.1%)	13005.24 (28.57%)	12045.6 (98.71%)
62	51 (66.67%)	13502.51 (43.95%)	11809.73 (21.69%)
63	52 (69.23%)	12506.18 (13.15%)	11886.29 (46.69%)
64	52 (69.23%)	12506.4 (13.15%)	11884.69 (46.16%)
65	52 (69.23%)	12507.8 (13.2%)	11883.4 (45.74%)
66	52 (69.23%)	12587.9 (15.67%)	11883.36 (45.73%)
67	52 (69.23%)	12781.85 (21.67%)	11882.55 (45.47%)
68	52 (69.23%)	12818.82 (22.81%)	11881.03 (44.97%)
69	52 (69.23%)	12914.62 (25.77%)	11874.4 (42.8%)
70	54 (74.36%)	13256.09 (36.33%)	11814.7 (23.31%)
71	55 (76.92%)	13841.06 (54.41%)	11748.54 (1.7%)
72	56 (79.49%)	12887.47 (24.93%)	11822.63 (25.9%)
73	57 (82.05%)	12814.95 (22.69%)	11821.12 (25.41%)
74	57 (82.05%)	12815.23 (22.7%)	11819.42 (24.85%)
75	57 (82.05%)	13003.29 (28.51%)	11818.17 (24.44%)
76	58 (84.62%)	12516.04 (13.45%)	11822.66 (25.91%)
77	59 (87.18%)	12248.23 (5.17%)	11878 (43.98%)
78	59 (87.18%)	12249.01 (5.2%)	11877.65 (43.87%)
79	59 (87.18%)	12256.01 (5.41%)	11877.46 (43.8%)
80	59 (87.18%)	12445.08 (11.26%)	11822.79 (25.95%)
81	59 (87.18%)	12445.86 (11.28%)	11822.44 (25.84%)
82	59 (87.18%)	12445.86 (11.28%)	11821.24 (25.44%)
83	59 (87.18%)	12451.23 (11.45%)	11821.22 (25.44%)

Continued on the next page

Table B.37 – *Continued from the previous page*

Solution S/N	$\mathbf{f_1}$	$\mathbf{f_2}$	$\mathbf{f_3}$
84	59 (87.18%)	12638.75 (17.25%)	11799.5 (18.35%)
85	61 (92.31%)	12110.9 (0.93%)	11870.63 (41.57%)
86	61 (92.31%)	12147.25 (2.05%)	11863.95 (39.39%)
87	61 (92.31%)	12209.56 (3.98%)	11842.68 (32.45%)
88	61 (92.31%)	12219.56 (4.29%)	11837.94 (30.9%)
89	61 (92.31%)	12219.8 (4.3%)	11835.06 (29.96%)
90	61 (92.31%)	12235.21 (4.77%)	11833.77 (29.54%)
91	61 (92.31%)	12242.9 (5.01%)	11831.71 (28.86%)
92	62 (94.87%)	12080.85 (0%)	11890.81 (48.16%)
93	62 (94.87%)	12081.34 (0.02%)	11888.91 (47.54%)
94	62 (94.87%)	12083.94 (0.1%)	11888.76 (47.49%)
95	62 (94.87%)	12090.11 (0.29%)	11888.67 (47.46%)
96	62 (94.87%)	12335.39 (7.87%)	11815.07 (23.43%)
97	62 (94.87%)	13117.25 (32.04%)	11746.63 (1.08%)
98	62 (94.87%)	13118.23 (32.07%)	11743.32 (0%)
99	63 (97.44%)	12318.02 (7.33%)	11813.75 (23%)
100	64 (100%)	12586.31 (15.62%)	11787.72 (14.5%)