VEHICLE ROUTING ON REAL ROAD NETWORKS

by

Saeideh Dehghan Nasiri

SUBMITTED FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

STOR-I DOCTORAL TRAINING CENTER
LANCASTER UNIVERSITY
LANCASTER, UK, SEPTEMBER 2014

© Copyright by Saeideh Dehghan Nasiri, 2014

CONTENTS

DECLARATION

This thesis is my original work and has not been submitted, in whole or in part, for a degree at this or any other university. Nor does it contain, to the best of my knowledge and belief, any material published or written by another person, except as acknowledged in the text.

ABSTRACT

The *vehicle routing problem* (VRP) has received particular attention, in the field of transportation and logistics. Producing good solutions for the problem is of interest both commercially and theoretically. Reliable solutions to real life applications require an approach based on realistic assumptions that resemble real-world conditions. In that respects, this thesis studies vehicle routing problems on real road networks addressing aspects of the problem that need to be modelled on the original road network graph and aims to provide appropriate modelling techniques for solving them.

As a preliminary step, chapter 2 studies the *travelling salesman problem* (TSP) on real road networks, referred to as the *Steiner* TSP (STSP) and proposes alternative integer programming formulations for the problem and some other related routing problems. The performances of formulations is examined both theoretically and computationally. Chapter 3 highlights the fact that travel speeds on road networks are correlated and uses a real traffic dataset to explore the structure of this correlation. In conclusion, it is shown that there is still significant spatial correlations between speeds on roads that are up to twenty links apart, in our congested road network.

Chapter 4 extends chapter 2 and incorporates the findings of chapter 3 into a modelling framework for VRP. The STSP with correlated costs is defined as a potentially useful variant of VRP that considers the costs in the STSP to be stochastic random variables with correlation. The problem is then formulated as a single-objective problem with eight different integer programming formulations presented. It is then shown how to account for three different correlation structures in each of the formulations.

Chapter 5 considers the VRPs with time windows and shows how most of the exact algorithms proposed for them, might not be applicable if the problem is defined on the original road network graph due to the underlying assumption of these algorithms that the cheapest path between a pair of customers is the same as the quickest path. This assumption is not always true on a real road network. Instead some alternative pricing routines are proposed that can solve the problem directly on the original graph.

ACKNOWLEDGEMENTS

Many people supported me throughout the production of this thesis. I thank them all. I like to thank the people below in particular.

At the university I must first thank Professor Adam N. Letchford for giving me the opportunity to work under his inspiring supervision. He is a great scientist with an amazing eye for detail. Our meetings were invariably instructive, frequently yielding fruitful ideas. It has been most pleasant to work with him and without his unfailing support this thesis would probably not have been possible. The guidance and support from Professor Richard Eglese and Professor Jon Tawn has also provided invaluable perspective on several occasions.

At home I am endlessly grateful to my husband Alireza Bahraminasab and my son Shayan Bahraminasab for their patience and unconditional support. Special thanks go to my parents and other members of my family. I would not have been where I am now without their encouragement.

Finally, thanks to STOR-i Doctoral Training Centre for their financial support.

Lancaster, UK Saeideh Dehghan Nasiri

September, 2014

LIST OF PUBLICATIONS

Parts of the work in this thesis has been submitted or published in the following journals:

1. A. N. Letchford, S. D. Nasiri and D. O. Theis (2013) Compact formulations of the Steiner Traveling Salesman Problem and related problems, *Eur. J. Oper. Res.*, 228, 83–92.

2. A. N. Letchford, S. D. Nasiri and A. Oukil (2014) Pricing Routines for Vehicle Routing with Time Windows on Road Networks. *Comput. & Oper. Res.*, 51, 331–337.

3. S. D. Nasiri (2014) Correlation Between Speeds on a Congested Road Network in the City of London. *Submitted for publication.*

4. A. N. Letchford and S. D. Nasiri (2014) The Steiner Travelling Salesman Problem with Correlated Costs. *Submitted for publication.*

1. INTRODUCTION & BACKGROUND

1.1 Motivation

The *vehicle routing problem* (VRP) has drawn a lot of interest from many researchers, since its first proposal in 1959. In recent years, the development of E-commerce and popularisation of internet trading has led to the creation of many on-line businesses like Amazon, eBay, Bidz, Buy.com, among others. A similar sales approach is now offered by many high street retailers and major supermarkets. One of the common promises advertised by most internet businesses to compete for customers in the marketplace, is delivery time guarantees. Although several modes of transport may be used for transferring goods from the source depot, delivery to the end users almost always involves road transport and vehicle movements to a number of destinations in a local area. This increasing demand for satisfying or improving service levels for just-in-time deliveries, has signified the applicability of the problem in every-day life even more.

Moreover, issues and concerns about environmental impacts and the need to reduce carbon footprint are continually discussed by the media and periodically new government targets are issued such as the 80% reduction in greenhouse gas emissions from 1990 to 2050 in the UK government's 2011 "'carbon plan"'. Carbon dioxide accounts for 20% of the green house effect and road transport on its own currently accounts for about 22% of total UK emissions of CO_2 [47]. These cultural changes and the concerns about climate change, also appeals for vehicle usage reduction and cost savings by optimal vehicle routing strategies.

In practice, the reliability of a routing schedule depends highly on the extent to which its underlying assumptions resemble real-life conditions. In a real-life environment, travel times on road networks are stochastic in nature. Temporal and spatial fluctuations of traffic flow may occur due to congestion, road works, accidents, weather conditions, etc. For the same reasons, travel times are often positively correlated across road segments. That is, if at a certain time travel times are unexpectedly high on a link then the travel times on nearby links are likely to be unexpectedly high as well. Through a focus on correlation between travel times, this thesis will mainly look at the VRP on a real road network, correlation structures and how to address correlation in vehicle routing models.

1.2 Vehicle Routing Problems

The VRP seeks to find the optimal set of routes for a fleet of vehicles which are to operate from a single depot to serve a set of customers given the cost of travel between every pair of customers and every customer and depot [30]. The problem was first introduced by Dantzig and Ramser [30] more than 50 years ago as the *truck dispatching problem.* They considered a real-life application concerned with designing routes with minimum total mileage for a fleet of trucks that deliver gasoline to service stations.

Since then, the problem has attracted a lot of attention in the academic literature, due to its many applications in practice, but also because it is theoretically a challenging problem to solve. Several variants of VRP have been introduced that apply to real-life situations and many models and algorithms have been proposed for computing their solutions.

The simplest and probably most famous variant of VRP is the *travelling salesman problem* (TSP) that is concerned with finding the cheapest way for a salesman who wishes to travel around a given set of cities and return to the starting point [3]. Although it appears to be a simple problem, the TSP has many applications in real life. For example, in computer science, in order to increase the production rate in manufacturing of printed circuit boards, one wishes to minimise the hole drilling time which is a function of the number of holes and the order in which they are drilled. A similar problem arises in computational Genomics when one wishes to find the optimal gene order by minimising the total distance in a sequence of vectors each associated with a gene. One could enumerate many other underlying or explicit applications of the TSP in areas such as of transportation, logistics, manufacturing, telecommunications and neuroscience. For more details and other applications of the TSP, the interested reader is referred to [33]. In this thesis, the focus is on transportation on road networks.

While the general VRP and TSP do directly capture some real-life applications, in others extra practical limitations may apply. In reality, the service provider with a limited operation budget may have a fixed number of vehicles and be faced with an increasing customer demand base. So, the concern is not only to find the cheapest route but also select the customers that will be served. Some customers may be more profitable than others. Equally, the company's top priorities may be their service reliability, rather than expanding their customer base, and so they must trade their resources off between performance quality and overall profit, or they may not be able to operate at the desired level. By adding some constraints and restrictions to the VRP and TSP we can capture some of these complications.

Capacity constraints Often vehicles will have spatial capacity limits for a given commodity. The *Capacitated Vehicle Routing Problem* (CVRP) is a variant of VRP with the additional restriction that every vehicle has a fixed capacity of a single commodity and customers have known demands [124]. The fleet may be heterogeneous where vehicles have different characteristics, or homogeneous where they are all identical. The CVRP is a combination of the TSP and the classical bin packing problem. In a bin packing problem a set of items with fixed dimensions need to be packed into a number of bins with a certain capacity, such that the number of used bins is minimised and no bin's items exceed its capacity. Good surveys on bin packing problems and CVRP are given in [25] and [124].

Time windows Another common complication arises when individual customers may only be serviced at a particular time of day. Attended home delivery services are a good example, when customer must be present for the delivery. In the *vehicle routing problem with time windows* (VRPTW) every customer is associated with a service time and a time window within which they are available to be serviced [118]. In case of early arrival at the customer location, the vehicle must wait. In some applications only *soft* time window restrictions may be imposed in which case a penalty charge is incurred for every customer that is serviced outside their time window.

Also of note is the *multiple travelling salesman problem with time windows* (mTSPTW), a relaxation of the VRPTW in which the vehicle capacity is sufficiently large so that no capacity restrictions are imposed. A special case of the mTSPTW, is the *travelling salesman problem with time windows* (TSPTW) where the number of vehicles is restricted to one. Desrochers *et al.* [36] give an introduction to variants of routing problems with time windows and survey the literature prior to 1988. A more recent but short survey is presented by Kumar and Panneerselvam [81], who also provide a comparison of heuristics and meta-heuristic approaches for the problem.

Multiple depots The *multi-depot vehicle routing problem* (MDVRP) is another extension of the VRP where vehicles operate from one of several depots instead of only one [124]. The MDVRP may be encountered for large transportation companies that serve a huge number of customers at sparse geographical locations. In this case, each depot may be assigned with its own fleet of vehicles and set of customers to serve, or vehicles may use intermediate depots to replenish, along their route. A survey of recent literature on the MDVRP is presented in [110].

Pick-up and delivery In some applications of the VRP, customers in a route require deliveries and pick-ups simultaneously, or some customers may only expect deliveries while others only require collection services [124]. For example, a milkman delivering milk door-to-door or postal/courier services are applications of this problem that one may encounter on a daily basis. The *VRP with pick-up and delivery* (VRPPD) is becoming very popular because of the trend towards recycling and re-use of products. Often fluctuations in the load of the vehicle makes it challenging to satisfy the capacity restrictions on a route. Therefore, it is sometimes desirable to perform the pick-ups after all the deliveries have been made. This variant is referred to as *VRP with backhauls.* A good survey of VRPPD is given by Berbeglia *et al.* [15].

Real road networks Road network features and traffic restrictions, such as construction works, roundabouts and one way systems, may sometimes impose limitations on the vehicle routes. For example, if customer a is located at a dead-end-street and is only accessible though customer b, then in order to service customer a and return to the depot, inevitably the vehicle must visit customer b at least twice. The *Steiner* VRP (SVRP) and TSP (STSP) are variants that solve the problem directly on the original road network, unlike the standard VRP and TSP, that assign a cost for direct routes from every customer to every other customer and depot by calculating the cost of the cheapest route between every pair of locations [28]. In principle, any instance of the SVRP or STSP instance can be converted into an instance of the standard VRP or TSP, by computing shortest paths between every pair of customers. However, as will be discussed in chapters 4 and 5, in practice solving the original Steiner problem is sometimes necessary and can be much quicker. Moreover, we will show in chapter 5 that for problems with time window restrictions, converting the problem to standard form may even result in the wrong solution. A review of the literature on the SVRP, STSP and some other routing problems on real road networks is given in chapters 2 and 5.

In the definition of the classical VRP and its variants that have been introduced so far, all input parameters are deterministic and it is assumed that everything goes according to *a priori* determined static schedule. However, routing problems in real-life may contain a fairly high degree of uncertainty. For examples, the customer's presence at the time of service, customer's demand, road conditions and traffic on the road are often unpredictable before the start of the journey. The following two scenarios are examples of the most studied and applicable variants of the VRP with stochastic parameters.

Stochastic demands In the *VRP with stochastic demands* (VRPSD), the capacitated delivery vehicle will not have knowledge of the demands that it will encounter on a route. Therefore, there is a positive probability associated with the vehicle running out of the product along the route, before serving all the customers on that route. In a situation where the vehicle fails to meet the demand of one or more customers on a route, that route becomes a failure. These failures are discouraged with a penalty that can be unique to each customer for not satisfying their demand. Vehicle routing with stochastic customers, where even the presence of each customer at its location is uncertain, can also be seen as a special case of the VRPSD in which all the demand is binary, either 0 or 1. A common application for this scenario is the distribution of packages by the post office.

The VRPSD was first introduced by Tillman in 1969 [122], for designing routes in a multiple terminal delivery problem. He proposed penalties whenever vehicles were almost empty or filled over capacity. A huge amount of research has been attributed to VRPSD, since then. Most modelling techniques for VRPSD take a *static* approach where routes are designed before knowing the actual demand, and the original planned sequence of visits is not changed during the journey.

The alternative is *dynamic* planning, where information is updated at each step of process and decisions on the next course of action are made after the present customer's demand is observed. Excellent surveys on the VRPSD are given in [44, 61].

Stochastic travel times Vehicle speeds and travel times depend on the level of congestion on the roads, which may be influenced by different factors such as the number of vehicles, road capacity, road conditions and weather conditions, among others. In recent years, both the *VRP with stochastic travel times* (VRPSTT) and *TSP with stochastic travel times* (TSPSTT) have received more attention in the academic literature since novel advances in technology now ease the access to real-time traffic information. This makes it possible to plan vehicle journeys taking account of congestion that is predictable from the traffic patterns of the past.

In VRPSTT the objective is to find the set of routes with minimum expected cost, while both travel times and customer service times can be random variables. Stochastic travel times were first considered in a study by Leipälä [86] where he calculates the expected length of the optimal tour for a TSP with stochastic distances. Similar to the VRPSD, a static modelling approach is to let vehicles follow their *a priori* planned routes and charge penalties for the routes that exceed a pre-set deadline.

Another approach is to model the travel times as random variables with known distributions, and either minimise the variation in travel times or minimise the total expected travel time while imposing an upper bound on its variation. In the dynamic approach, one integrates traffic information obtained in real time and updates the route plan during service execution. A survey and review of literature on the VRPSTT and TSPSTT will be given in chapter 4.

1.3 Combinatorial Optimisation

The vehicle routing problem is classified as a *combinatorial optimisation problem* (COP) in the field of optimisation and mathematical modelling. COPs are optimisation problems that involve concepts from combinatorics, such as sets, subsets, permutations and combinations. One way to define them formally is as follows. We are given a ground set E, a function $f(S)$ that maps any subset $S \subset E$ to a real number, and a rule which, given any $S \subset E$, tells us whether S is *feasible*. The task is to find a feasible subset where the desired extermum of $f(S)$ is attained. Well known examples of COPs include the *assignment problem*, *knapsack problem* and the *shortest path problem* which may also appear as a subproblem in other more complicated problems.

The general form of a mathematical model for an optimisation problem is,

$$\begin{matrix}\min \\ \max\end{matrix} \left(f(x) : g_i(x) \begin{Bmatrix} \leq \\ = \\ \geq \end{Bmatrix} b_i, i \in M, x \in \mathbb{R}^n_+ \right) \tag{1.1}$$

where the objective is to minimise or maximise f that might represent cost or profit and is a function of the decision variables x. N is the set of variables that may take positive real values and M represents the set of constraints that impose the limitations or requirements that need to be fulfilled for the *feasible solutions* in the set of interest.

COPs and integer programs The general optimisation problem (1.1) is called a *linear program* (LP) when the objective function and constraints are linear and variables are continuous. If integrality is imposed on the variables by replacing the condition $x \in \mathbb{R}^n_+$ by the condition $x \in \mathbb{Z}^n_+$, it is denoted an *integer program* (IP) and a *mixed integer program* (MIP) is obtained when both types of real and integer variables may exist. When variables in a LP are constrained to take integer values the problem is called an *integer linear program* (ILP) and when they are restricted to take only binary values, i.e. $x \in \{0, 1\}^n$, we derive a 0-1 *linear* *program* (0-1 LP). A LP that has both types of real and integer variables is called a *mixed-integer linear program* (MILP).

Most COPs of interest can be formulated as MILPs. When attempting to formulate a COP as a MILP, the most natural starting point is to define a 0-1 variable x_e for each $e \in E$, taking the value 1 if $e \in S$, and 0 otherwise. The set function $f(S)$ then needs to be mapped to a function of the x variables, and the condition that S must be feasible needs to be mapped to constraints involving the x variables. The resulting MILP is then a 0-1 LP. In some cases, however, either finding such a 0-1 LP formulation is not at all easy, or one obtains a formulation with a huge (exponentially large) number of constraints. In such cases, it often helps to introduce additional variables and constraints, to obtain a simpler MILP formulation. In fact, for some COPs, it is possible to formulate them as MILPs in several different ways. This is the case for the TSP and the variants of it that we consider.

In principle one could find the optimum solution of a COP by enumeration of the set of feasible solutions. However, for practical applications this set mostly contains too many elements that computing times for enumeration process become unacceptably long. Therefore, an *efficient* algorithm is needed to solve the problem in a reasonable time.

Analysis of algorithms The *efficiency* of an algorithm is defined in terms of its running time. For a general function $t : \mathbb{N} \to \mathbb{R}$, an algorithm is said to run in time $\mathcal{O}(t(n))$, if for some constants α and n_0 and for any $n \geq n_0$, the algorithm takes at most $\alpha t(n)$ time to compute a solution. An efficient algorithm is one that runs in *polynomial time* in the size of input, i.e. $t : n \to n^c$, for some constant c, as opposed to *exponential time* algorithms where $t : n \to c^n$ for some constant $c > 1$.

All of the routing problems described in the previous subsection appear to be hard in the sense that no obvious efficient algorithm comes to mind for solving

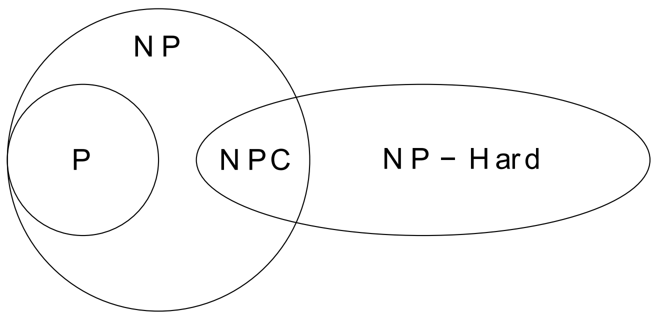


Fig. 1.1: Relationship of some complexity classes if $\mathcal{P} \neq \mathcal{NP}$.

them. However, it is not apparent precisely how difficult they are either. *Computational complexity theory* provides the foundation for systematically classifying the hardness of a problem.

Computational complexity Considering only the *decision problems*, where given an *instance*, the algorithm will either accept or reject it on the basis of whether or not it lies within the set of interest, the simplest class of problems are complexity class $\mathcal{P}$ (*Polynomial*). $\mathcal{P}$ is the set of all decision problems for which there exists an efficient algorithm that its running time is polynomial in its input size. On the other hand, $\mathcal{NP}$ (*Non-deterministic Polynomial*) class is the set of decision problems that are *verifiable* in polynomial time, meaning that given an instance of the decision, there is a polynomial algorithm in size of the instance that *certifies* the instance lies within the set of interest. Clearly, $\mathcal{P} \subseteq \mathcal{NP}$ since all decision problems in $\mathcal{P}$ can be solved in polynomial time without any certificate but whether the reverse is true or not is still an important open question amongst theoretical computer scientists.

Another set of interest are $\mathcal{NPC}$ (*NP Complete*) problems. A decision problem $A \in \mathcal{NP}$ becomes $\mathcal{NPC}$ if it is *polynomial reducible*, that is if it can be shown that for every problem $B \in \mathcal{NP}$ there exists a function $g : B \to A$ which maps acceptable instances of B to acceptable instances of A in polynomial time in its input size. The existence of a polynomial time reduction from B to A proves that,

B is no harder to solve than A. Finally, if we assume that $\mathcal{P} \neq \mathcal{NP}$, there is a class of $\mathcal{NP}$ *Hard* problems which consists of optimisation problems that are not polynomially solvable and are at least as hard as some $\mathcal{NPC}$ decision problem to solve. An optimisation problem A is $\mathcal{NP}$ *Hard* if there exists some decision problem $B \in \mathcal{NPC}$ that can be reduced to it.

The decision problem of the TSP (where, given a tour with cost f, the task is to find whether there exists any tour cheaper than f) belongs to the class of $\mathcal{NPC}$ problems [109] and TSP itself is a $\mathcal{NP} - Hard$ problem. The general VRP and all its other variations that have been mentioned in the previous subsection are at least as hard as the TSP, implying that they also belong to the complexity class $\mathcal{NP}$ Hard.

Although the simplified questions underlying practical instances of routing problems are amongst the hardest problems in computer science themselves, still additional complexities are often brought about by real world issues and untidiness - road traffic, weather conditions, employee rights and working hours - as well. To design an efficient algorithm that can inspect the structure of an instance and search the solution space to find a feasible solution with the best objective value, we need to characterize the input parameters and features of the problem. Discrete mathematics and *graph theory* is the language used for the characterization and formal description of these problems. In this representation, feasible solutions of the problem are expressed as subgraphs on an abstract graph that embodies the road network on which the vehicle is travelling.

1.4 Graph Theory

As mentioned, the mathematical theory of graphs is one of the main tools for describing routing problems and effective design of related algorithms, so I present some key ideas and terminology here. Graph $G(V, E)$ is a mathematical structure consisting of a set V of *nodes* or *vertices* and a set E of *edges* that connect the vertices. Set V has $|V|$ distinct vertices and edge set E contains $|E|$ distinct edges. An edge is identified by the pair of vertices at its end points and existence of an edge $e = \{i, j\} \in E$ indicates that vertices i and j are *adjacent* to each other.

In an *undirected* graph, there is no distinction between the two vertices associated with an edge and so edges $\{i, j\}$ and $\{j, i\}$ are identical, whereas in a *directed* graph, edges have direction. A directed edge is also called an *arc*. Arc $a = (i, j)$ has a direction that points from its initial vertex i, also referred to as the arc *tail*, to its end vertex j, also referred to as the arc *head*. The *degree* of a vertex is the number of edges that connect to that vertex; in a directed graph a vertex has *in-degrees* and *out-degrees* that are the number of arcs pointing to that vertex and away from it, respectively.

A *path* in a graph G is a consecutive sequence of edges $\big\langle\{v_1, v_2\}, \ldots, \{v_{n-1}, v_n\}\big\rangle$ from the edge set E that connect the distinct set of vertices $\{v_1, \ldots, v_n\} \in V$, at their end points. Vertex b is *reachable* from a if there exists a path in the graph that starts from a and ends at b and two vertices a and b are *connected* if they are both reachable from one another - clearly, connectivity is implied by reachability in an undirected graph. G is described as a *connected* graph if every pair of vertices in V are connected and it becomes a *complete* graph if every vertex in V is adjacent to every other vertex in the graph. Graph G is a *multi-graph* if it contains *parallel* edges or arcs, i.e. edges or arcs that have the same end nodes.

A *cycle* is a path that starts and ends at the same vertex. A path or cycle is said to be *Eulerian* if they traverse every edge exactly once and is described as *Hamiltonian* if it visits each vertex exactly once. Graph G is Eulerian if it contains an Eulerian cycle and is Hamiltonian if it possesses a Hamiltonian cycle - for example see figure 1.2. It was shown by Hierholzer in 1873 [70] that an undirected graph has an Eulerian cycle if and only if the graph is connected and

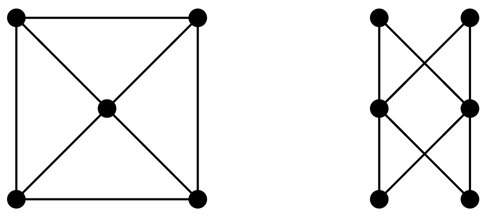


Fig. 1.2: On the left a graph which is Hamiltonian and not Eulerian and on the right a graph which is Eulerian and not Hamiltonian .

every vertex has even degree. We will refer to this result in the next chapter, for modelling the STSP.

A graph G is a *tree* if it is connected and does not have any cycles. A *forest* is a union of trees that are pairwise disjoint. A *spanning tree* of a connected undirected graph G, is a tree that contains every vertex of G and all of whose edges are in E. A *Spanning forest* of graph G is a set of spanning trees, such that any two reachable vertices in G are in the same tree.

Graph G is said to be *weighted* when a label (weight), which is usually a real number, is associated with every edge in E. The weight of a path or cycle in a weighted graph, often referred to as *cost* in this thesis, is the sum of all edge weights in that path or cycle.

The terminology and notation described in this section will be used in the next chapters for formal descriptions of the TSP, the VRP and some other routing problems. For these routing problems various solution methods have been proposed during the years that can mainly be divided into three categories:

- *Exact algorithms* guarantee to find an optimal solution, i.e. a solution with the best objective function value.

- *Heuristics* find solutions in reasonable computing times with no guarantee on the quality of the solution. They can be useful for large scale practical problems where computing times for exact methods are excessive or finding the optimal solution is not imperative.

- *Approximation algorithms* find an approximate solution with proven 'performance guarantees', usually in the form of an upper bound on the ratio between the solution and the optimum.

As discussed in the previous section, the routing problems in this thesis are $\mathcal{NP}-Hard$ and exact solution methods for them mostly have exponential running times. However, study of the exact methods can provide insight into the problem behaviour and improvements of exact methods in recent decades have extended the boundaries for expected computing times. Our focus is solely on exact methods in this thesis.

1.5 Exact Methods for Vehicle Routing

VRPs are typically formulated as MIPs. Many exact algorithms have been proposed for solving the VRP and its variants. The performance of algorithms for solving the problem depends strongly on the way that the problem is formulated and on the features of its formulation. In this section we describe five exact algorithms used to solve different classes of formulations.

Branch-and-bound was first proposed by Land and Doig [82] in 1960. It is an enumeration scheme that is based on the idea of successively partitioning the solution space and generating a sequence of subproblems until a (better) feasible solution is found or it is determined that the partition does not contain a (better) solution. Compact formulations, in which the number of variables is bounded by a polynomial in the number of nodes (e.g., the CVRP formulation of Gavish & Graves [60]) can be solved by this method.

The enumeration scheme is displayed by a branching tree where the *root node* of the tree corresponds to the original problem and has a solution space that contains all the feasible solutions of the problem. In order to construct a branch-and-bound algorithm, one needs a lower bound (LB) for the problem. Considering a MIP with minimising objective function, an efficient way of calculating a LB $\bar{x}^0$ for the problem is by solving its LP relaxation, i.e., the enlarged solution space that allows for integer variables to take on continuous values. If $\bar{x}_i^0$ is integer for all $i \in N$, then $\bar{x}^0$ is an optimal solution and the algorithm terminates. If for some $i \in N$, $\bar{x}_i^0$ is non-integral, the algorithm defines two new subproblems of the original problem with the additional constraints $x_i \geq \lceil \bar{x}_i^0 \rceil$ and $x_i \leq \lfloor \bar{x}_i^0 \rfloor$ by *branching* on the root node. Variable x_i is called the *branching variable* and the two new subproblems are called the *active nodes* of the tree.

In the next stage, the algorithm attempts to solve the LP relaxation of one of the active nodes. In case the LP relaxation is infeasible, that node is pruned from the tree. If the solution of the subproblem is integer feasible, then its objective value becomes an upper bound (UB) for the objective value of the original problem and if the solution is not integer feasible, then two new subproblems are defined and the branching procedure is repeated. Also, at each stage active nodes with LBs higher than an existing UB can be discarded from the tree since the objective value of an optimal solution cannot be greater than an UB. A typical example is a branching strategy based on choosing either a value 0 or 1 for a single binary variable (see, e.g., Figure 1.3). The algorithm terminates when there are no more active nodes in the tree and the optimal solution of the original problem is the best integer solution found in the tree.

Branch-and-cut algorithm embeds strong valid inequalities (*cutting planes*) into a branch-and-bound paradigm to cut off some of the current fractional solution, i.e. the non-integer solution obtained by the LP relaxation. This method performs particularly well on formulations that have an exponentially large number of constraints, but a polynomial number of variables (e.g., the CVRP formulation

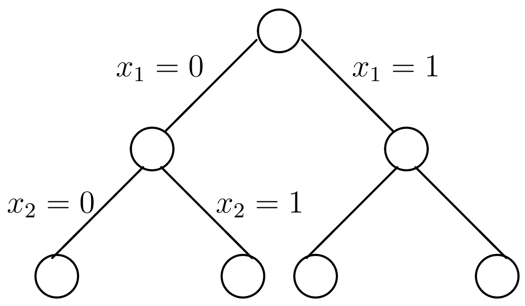


Fig. 1.3: Partial branch-and-bound tree example .

of Laporte & Nobert, [84]). Such MIP formulations have too many valid inequalities to be included in the LP from the start.

Formally, cutting planes are defined as follows. Let P_{MIP} denote the convex hull containing all the feasible solutions of the original MIP. If the optimal solution of its LP relaxation $\bar{x}^0$, is not integer and it does not satisfy all the constraints in MIP, then there exists a hyperplane $\{x \in \mathbb{R}^n : a^T x = \alpha\}$ such that $a^T \bar{x}^0 > \alpha$ and $P_{MIP} \subseteq \{x \in \mathbb{R}^n : a^T x \leq \alpha\}$. Such a hyperplane is called a cutting plane. A basic cutting plane algorithm consists of the following steps:

1. *Relaxation*: Solve the LP relaxation. If the solution of LP relaxation $\bar{x}^0$, is integer output the optimal integer solution and stop, else continue to step 2.

2. *Add cutting planes*: Search for cutting planes that are violated by $\bar{x}^0$ and remain valid for P_{MIP}; if any are found add them to LP formulation and return to step 1. If no inequalities have been found output the current bound and stop.

The problem of finding the violated cutting planes is called the *separation problem.* Adding cutting planes to the LP relaxation (generating rows) strengthens the LP relaxation and decreases the gap between UB and LB by raising the LB in the branch-and-bound tree. Lysgaard *et al.* [92] have implemented this method for solving a CVRP.

Branch-and-price incorporates *column generation* into the branch-and-bound paradigm to divide the problem into smaller subproblems that have their solutions combined in a master problem. Dantzig and Wolfe [31] proposed the column generation technique as a scheme for solving formulations with an exponentially large number of variables, but a polynomial number of constraints (e.g., the Set Partitioning formulation of Balinski & Quandt [11]). These problems typically have too many columns to handle efficiently.

Column generation is very similar to the cutting plane method but instead of adding *rows* to the LP relaxation of the problem, column generation adds *columns* to a restricted version of the LP relaxation. Initially, sets of columns are left out of the LP relaxation of the large MIP because there are too many columns to handle efficiently and since the basis has limited dimension, many of these columns will have their associated variables equal to zero in an optimal solution anyway. First the LP relaxation of *restricted master problem* (where many columns are left out) is solved to optimality. Then a *pricing algorithm* is used to verify the optimality of the solution of the restricted LP relaxation (RLP) for the LP relaxation of the problem. The solution will be optimal if all variables of the LP relaxation have non-negative *reduced cost.* If the solution is not optimal then at least one column is found with negative reduced cost. This column is added to the RLP and the LP is re-optimised. The procedure terminates when there are no more columns with negative reduced cost. Similar to branch-and-cut, the solution to the LP relaxation is often non-integer and branching is performed to find a feasible solution to the MIP. Desrochers *et al.* [37] have applied this algorithm for solving the VRPTW.

Branch-cut-and-price is a combination of branch-and-bound, column generation and the cutting plane method. This is a method of choice for solving formulations with an exponentially large number of both variables and constraints (e.g., the CVRP formulation of Fukasawa et al. [57]).

Analogous to branch-and-price, the algorithm initially considers the LP relaxation of a restricted master problem with many columns left out. In order to strengthen the LP relaxation, cutting planes are added to the master problem to cut off fractional solutions of the master problem. The master problem is then re-optimised but due to addition of new constraint, the variables already contained in the restricted master problem are not necessarily sufficient to construct an optimal solution for the new master problem. Therefore, column generation is used to search for columns with negative reduced cost again. The procedure is repeated until no more columns with negative reduced cost are found and no more violated cuts can be generated. As with the branch-and-cut and branch-and-price procedures, at this stage branching needs to be performed if the solution to the LP relaxation is non-integer.

Dynamic programming (DP) is an implicit enumeration scheme that first decomposes the problem into a nested family of subproblems. The solution to the original problem is then found by recursively solving the (generally easier) subproblems and working either backward from the end of the problem to the beginning (*backward DP*) or forward from the beginning to the end (*forward DP*). Optimisation problems for which a feasible solution can be viewed as a sequence of decisions can be tackled by DP [14]. The basic characteristics that are common to DP applications are as follows:

1. The problem can be divided into a series of stages i with a decision x_i required at each stage.

2. Each stage i has a set of states $\{s_i\}$ associated with it which can describe the current condition of the system.

3. The decision x_i made at any stage determines the next state s_{i+1} of the system at next stage, as well as the immediately earned award or cost $f_i(x_i, s_i)$.

4. At any stage, the optimal decision for each of the remaining stages must not depend on previous states and decisions made at previous stages. This is according to *Bellman's principle of optimality* for DP [14].

5. On each stage a *recursion* is available that relates the cost or reward achieved during each stage i to the cost or reward achieved in stage $i+1$. Considering a minimisation problem in a forward DP algorithm, the recursive relationship can be written as:

$$F_i(s_i) = \min_{x_i \in D_i} \{c_i(s_i, x_i) + F_{i-1}(s_{i-1}(s_i, x_i))\}, \tag{1.2}$$

   where $c_i(s_i, x_i)$ is the cost of moving from state $s_{i-1}(s_i, x_i)$ to state s_i according to decision x_i and $F_i(s_i)$ is the minimum accumulative cost incurred up to stage i.

Dynamic programming algorithms are computationally efficient, as long as the state space does not become too large (e.g., Mingozzi *et al.* [101] have used DP to solve the TSPTW). In this thesis we use DP to solve pricing problems for routing problems with time windows in chapter 5.

1.6 Contribution

The main contributions of the thesis, summarized in the points below, is to show

- how to adapt compact formulations of the TSP to the Steiner TSP and some other variants of VRP that are defined on the original road network graph and solve them by the use of the branch-and-bound algorithm.
- how correlation between travel times on roads depends on the amount of congestion and the type of road network.
- how to model and solve the Steiner TSP in the presence of correlated travel times. Furthermore, to show how to capture different correlation structures

that may be representative of different types of road network in practice, into the model and solve the problem using branch-and-bound algorithm.

- how exact methods proposed for solving VRPs with time windows can fail to give the correct optimal solution when the instance is defined on a real road network. Moreover, to describe natural ways to remedy the situation and present pricing routines that can be incorporated within a branch-and-price algorithm to solve the problems on the original network graph.

A more detailed description of the contributions of each chapter can be found in the following section.

1.7 Structure of Thesis

The next four chapters of this thesis consist of four journal articles that have been submitted for publication, two of which are published. Each chapter contains theory, methodology or application which is motivated by aspects of modelling routing problems on real road networks. The following is a brief description of each chapter.

Chapter 2: Compact Formulations of the Steiner TSP and Related Problems. This paper considers the STSP as a variant of the TSP and adapts some of the compact formulations of the TSP (i.e. formulations of polynomial size) to the STSP. The proposed formulations are compared both theoretically and computationally. It is shown that the best of the formulations can solve instances with over 200 nodes to proven optimality, on a standard branch-and-bound solver. It is also shown how to adapt the best of the formulations to some related problems. This work is co-authored with Adam Letchford and Dirk Oliver Theis and has been published in the European Journal of Operational Research, see Letchford *et al.* [88].

Chapter 3: Correlation Between Speeds on a Congested Road Network in the City of London. This paper explores the temporal and spatial correlations between travel speeds in a congested road network. Statistical analysis is carried out on real traffic data that has been collected over a period of two months, in the City of London. It is shown that the assumption of a first-order Markovian property for spatial correlations on a congested road network is rather naive. In fact, for our London data set the spatial correlations are still significant for roads up to twenty links apart. However, performing a principal component analysis proves that the correlation structure is not complex and only three components are needed to explain 89% of the temporal variation in speed, and only six components are needed to explain over 80% of the spatial variation. This paper has been submitted for publication.

Chapter 4: The Steiner Travelling Salesman Problem with Correlated Costs. This paper is an extension of our work in Letchford *et al.* [88] that introduces STSP with correlated costs as a potentially useful variant of the TSP, that considers the cost of traversing roads to be stochastic random variables with correlation. The problem is modelled as a bi-objective mean-variance problem and eight IP formulations are presented for the problem. It is also shown how to exploit special structures in the correlation matrix. Computational results are presented for the eight formulations and three different correlation structures, for instances with up to 100 nodes. This is joint work with Adam Letchford. The paper has been submitted for publication.

Chapter 5: Pricing Routines for Vehicle Routing with Time Windows on Road Networks. This paper studies the VRP with time windows and brings to attention an implicit assumption that is made in most of the exact algorithms developed for standard VRPs, which is that the cheapest path between each pair of customers is the same as the quickest path. This assumption is not valid on road networks and therefore, most of the existing exact algorithms cannot be applied to an instance that is defined on the original road network. Instead, some alternative pricing routines are proposed for these problems and their performance is tested on some experimental instances. We also propose two sets of formulations that can be used to solve the VRPTW directly on the original road network but due to their practical limitations they are not included in this article and I present them in the appendix. This work is co-authored with Adam Letchford and Amar Oukil and has been published in the journal Computers & Operations Research, see Letchford *et al.* [87].

Finally, *chapter 6* gives concluding remarks and discusses some potential areas for future research.

2. COMPACT FORMULATIONS OF THE STEINER TRAVELLING SALESMAN PROBLEM AND RELATED PROBLEMS

2.1 abstract

The Steiner Traveling Salesman Problem (STSP) is a variant of the TSP that is particularly suitable when routing on real-life road networks. The standard integer programming formulations of both the TSP and STSP have an exponential number of constraints. On the other hand, several *compact* formulations of the TSP, i.e., formulations of polynomial size, are known. In this paper, we adapt some of them to the STSP, and compare them both theoretically and computationally. It turns out that, just by putting the best of the formulations into the CPLEX branch-and-bound solver, one can solve instances with over 200 nodes. We also briefly discuss the adaptation of our formulations to some related problems.

Keywords: traveling salesman problem, integer programming.

2.2 Introduction

The *Traveling Salesman Problem* (TSP), in its undirected version, can be defined as follows. We are given a complete undirected graph $G = (V, E)$ and a positive integer cost c_e for each edge $e \in E$. The task is to find a Hamiltonian circuit, or *tour*, of minimum total cost. The best algorithms for solving the TSP to proven optimality, such as the ones described in [3, 103, 107], are based on a formulation of the TSP as a 0-1 linear program due to Dantzig *et al.* [29], which we present in Subsection 2.3.1 of this paper.

The Dantzig *et al.* formulation has only one variable per edge, but has an exponentially-large number of constraints, which makes cutting-plane methods necessary (see again [3, 103, 107]). If one wishes to avoid this complication, one can instead use a so-called *compact* formulation of the TSP, i.e., a formulation with a polynomial number of both variables and constraints. A variety of compact formulations are available (see the surveys [66, 105, 108] and also Subsection 2.3.2 of this paper).

When dealing with routing problems on real-life road networks, however, one is much more likely to encounter the following variant of the TSP. We are given a connected undirected graph $G = (V, E)$, a positive integer cost c_e for each $e \in E$, and a set $V_R \subseteq V$ of *required* nodes. The task is to find a minimum-cost closed walk, not necessarily Hamiltonian, that visits each required node at least once. Nodes may be visited more than once if desired, and edges may be traversed more than once if desired. This variant of the TSP was proposed, apparently independently, by three sets of authors [28, 53, 106]. (The special case in which all nodes must be visited was considered earlier in [69, 98].) We will follow Cornuéjols *et al.* [28] in calling this variant the *Steiner* TSP, or STSP for short.

As noted in [28, 53], it is possible to convert any instance of the STSP into an instance of the standard TSP, by computing shortest paths between every pair of required nodes. So, in principle, one could use any of the above-mentioned TSP formulations to solve the STSP. If, however, the original STSP instance is defined on a sparse graph, the conversion to a standard TSP instance increases the number of variables substantially, which is undesirable. So, in this paper, we provide and analyse some compact formulations for the STSP. Specifically, we introduce single-commodity flow, multi-commodity flow and time-staged formulations, and compare them both theoretically and computationally.

It turns out that, just by putting the best of our formulations into the CPLEX branch-and-bound solver, one can routinely solve instances with over 200 nodes to proven optimality. This is in sharp contrast to the situation with the standard TSP, where sophisticated cutting-plane techniques are needed to solve instances with more than about 50 nodes (see, e.g., [3, 103]).

The paper is structured as follows. The relevant literature is reviewed in Section 2.3. In Section 2.4, the so-called *commodity-flow* formulations of the TSP are adapted to the Steiner case. In Section 2.5, the same is done for the so-called *time-staged* formulation. In Section 2.6, some computational results are given. Then, in Section 2.7, we briefly discuss the possibility of adapting our approach to other variants of the TSP. Finally, some concluding remarks appear in Section 2.8.

2.3 Literature Review

We now review the relevant literature. We cover the classical formulation of the standard TSP in Subsection 2.3.1, compact formulations of the standard TSP in Subsection 2.3.2, and the classical formulation of the STSP in Subsection 2.3.3.

2.3.1 The classical formulation of the standard TSP

The classical and most commonly-used formulation of the standard TSP is the following one, due to Dantzig, Fulkerson and Johnson [29]:

$$\begin{array}{lll} \min & \sum_{e \in E} c_e x_e & \\ \text{s.t.} & \sum_{e \in \delta(\{i\})} x_e = 2 \quad (\forall i \in V) & (2.1) \\ & \sum_{e \in \delta(S)} x_e \geq 2 \quad (\forall S \subseteq V : 2 \leq |S| \leq |V|/2) & (2.2) \\ & x_e \in \{0,1\} \qquad (\forall e \in E). & \end{array}$$

Here, x_e is a binary variable, taking the value 1 if and only if the edge e belongs to the tour, and, for any $S \subset V$, $\delta(S)$ denotes the set of edges having exactly one end-node inside S. The constraints (2.1), called *degree* constraints, enforce that the tour uses exactly two of the edges incident on each node. The constraints (2.2), called *subtour elimination constraints*, ensure that the tour is connected.

We will call this formulation the *DFJ* formulation. A key feature of this formulation is that the subtour elimination constraints (2.2) are exponential in number.

2.3.2 Compact formulations of the standard TSP

As mentioned above, a wide variety of compact formulations exist for the standard TSP, and there are several surveys available (e.g., [66, 105, 108]). For the sake of brevity, we mention here only four of them. All of them start by setting $V = \{1, 2, \dots n\}$ and viewing node 1 as a 'depot', which the salesman must leave at the start of the tour and return to at the end of the tour. Moreover, all of them can be used for the asymmetric TSP as well as for the standard (symmetric) TSP.

We begin with the formulation of Miller, Tucker & Zemlin [99], which we call the *MTZ* formulation. For all node pairs (i, j), let $\tilde{x}_{ij}$ be a binary variable, taking the value 1 if and only if the salesman travels from node i to node j. Also, for $i = 2, \dots, n$, let u_i be a continuous variable representing the position of node i in the tour. The MTZ formulation is then:

$$\begin{array}{llll}
\min & \sum_{i,j=1}^{n} c_{ij}\tilde{x}_{ij} & & (2.3)\\
\text{s.t.} & \sum_{j=1}^{n} \tilde{x}_{ji} = 1 & (1 \le i \le n) & (2.4)\\
& \sum_{j=1}^{n} \tilde{x}_{ij} = 1 & (1 \le i \le n) & (2.5)\\
& \tilde{x}_{ij} \in \{0,1\} & (1 \le i, j \le n; i \neq j) & (2.6)\\
& u_i - u_j + (n-1)\tilde{x}_{ij} \le n-2 & (2 \le i, j \le n; i \neq j) & (2.7)\\
& 1 \le u_i \le n-1 & (2 \le i \le n). & (2.8)
\end{array}$$

The constraints (2.4) and (2.5) ensure that the salesman arrives at and departs from each node exactly once. The constraints (2.7) ensure that, if the salesman travels from i to j, then the position of node j is at least one more than that of node i. Together with the bounds (2.8), this ensures that each non-depot node is in a unique position.

The MTZ formulation is compact, having only $\mathcal{O}(n^2)$ variables and $\mathcal{O}(n^2)$ constraints. Unfortunately, Padberg & Sung [108] show that its LP relaxation yields an extremely weak lower bound, much weaker than that of the DFJ formulation.

The next compact formulation, historically, was the 'time-staged' (TS) formulation proposed by both Vajda [126] and Houck *et al.* [72] independently. For all $1 \leq i, j, k \leq n$ with $i \neq j$, let r_{ij}^k be a binary variable taking the value 1 if and only if the edge $\{i, j\}$ is the kth edge to be traversed in the tour, and is traversed in the direction going from i to j. We then have:

$$\min \quad \sum_{i=2}^n c_{1i} r_{1i}^1 + \sum_{k=2}^{n-1} \sum_{i,j=2}^n c_{ij} r_{ij}^k + \sum_{i=2}^n c_{i1} r_{i1}^n$$

$$\text{s.t.} \quad \sum_{j=2}^n r_{1j}^1 = 1 \tag{2.9}$$

$$\sum_{j=2}^n r_{j1}^n = 1 \tag{2.10}$$

$$\sum_{k=1}^{n-1} \sum_{j \neq i} r_{ji}^k = 1 \qquad (2 \leq i \leq n) \tag{2.11}$$

$$\sum_{j \neq i} r_{ji}^k = \sum_{j \neq i} r_{ij}^{k+1} \qquad (2 \leq i \leq n; 1 \leq k \leq n-1) \tag{2.12}$$

$$r_{ij}^k \in \{0, 1\} \qquad (1 \leq i, j, k \leq n; i \neq j).$$

The constraints (2.9) and (2.10) state that the salesman must leave the depot at the start of the tour and return to it at the end. The constraints (2.11) ensure that the salesman arrives at each non-depot node exactly once, and the constraints (2.12) ensure that the salesman departs from each node that he visits.

The TS formulation has $\mathcal{O}(n^3)$ variables and $\mathcal{O}(n^2)$ constraints. It follows from results in [66, 108] that the associated lower bound is intermediate in strength between the MTZ and DFJ bounds.

Next, we mention the *single-commodity flow* (SCF) formulation of Gavish & Graves [60]. Imagine that the salesman carries $n - 1$ units of a commodity when he leaves node 1, and delivers 1 unit of this commodity to each other node. Let the $\tilde{x}_{ij}$ variables be defined as above, and define additional continuous variables g_{ij}, representing the amount of the commodity (if any) passing directly from node i to node j. The formulation then consists of the objective function (2.3), the constraints (2.4)–(2.6), and the following constraints:

$$\textstyle\sum_{j=1}^{n} g_{ji} - \sum_{j=2}^{n} g_{ij} = 1 \quad (2 \leq i \leq n) \tag{2.13}$$

$$0 \leq g_{ij} \leq (n-1)\tilde{x}_{ij} \qquad (1 \leq i, j \leq n; j \neq i). \tag{2.14}$$

The constraints (2.13) ensure that one unit of the commodity is delivered to each non-depot node. The bounds (2.14) ensure that the commodity can flow only along edges that are in the tour.

The SCF formulation has $\mathcal{O}(n^2)$ variables and $\mathcal{O}(n^2)$ constraints. It is proved in [108] that the associated lower bound is intermediate in strength between the MTZ and DFJ bounds. Later on, in [66], it was shown that it is in fact intermediate in strength between the MTZ and TS bounds.

Finally, we mention the *multi-commodity flow* (MCF) formulation of Claus [24]. Here, we imagine that the salesman carries $n-1$ commodities, one unit of each for each customer. Let the $\tilde{x}_{ij}$ variables be defined as above. Also define, for all $1 \leq i, j \leq n$ with $i \neq j$ and all $2 \leq k \leq n$, the additional continuous variable f_{ij}^k, representing the amount of the kth commodity (if any) passing directly from node i to node j. The formulation then consists of the objective function (2.3), the constraints (2.4)–(2.6), and the following constraints:

$$0 \leq f_{ij}^k \leq \tilde{x}_{ij} \qquad (k = 2, \ldots, n; \{i, j\} \subset \{1, \ldots, n\}) \tag{2.15}$$

$$\textstyle\sum_{i=2}^{n} f_{1i}^k = 1 \qquad (k = 2, \ldots, n) \tag{2.16}$$

$$\textstyle\sum_{i=1}^{n} f_{ik}^k = 1 \qquad (k = 2, \ldots, n) \tag{2.17}$$

$$\textstyle\sum_{i=1}^{n} f_{ij}^k - \sum_{i=2}^{n} f_{ji}^k = 0 \quad (k = 2, \ldots, n; j \in \{2, \ldots, n\} \setminus \{k\}). \tag{2.18}$$

The constraints (2.15) state that a commodity cannot flow along an edge unless that edge belongs to the tour. The constraints (2.16) and (2.17) impose that each commodity leaves the depot and arrives at its destination. The constraints (2.18) ensure that, when a commodity arrives at a node that is not its final destination, then it also leaves that node.

The MCF formulation has $\mathcal{O}(n^3)$ variables and $\mathcal{O}(n^3)$ constraints. It is proved in [108] that the associated lower bound is equal to the DFJ bound. Therefore, this is the strongest of the four compact formulations mentioned.

2.3.3 The classical formulation of the STSP

The integer programming formulation given in [53] for the STSP is as follows:

$$\min \quad \textstyle\sum_{e\in E} c_e x_e \qquad (2.19)$$

$$\text{s.t.} \quad \textstyle\sum_{e\in\delta(i)} x_e \text{ even} \quad (i \in V) \qquad (2.20)$$

$$\textstyle\sum_{e\in\delta(S)} x_e \geq 2 \quad (S \subset V : S \cap V_R \neq \emptyset, V_R \setminus S \neq \emptyset) \qquad (2.21)$$

$$x_e \in \mathbb{Z}_+ \quad (e \in E). \qquad (2.22)$$

Note that, since edges may be traversed more than once if desired, the x variables are now general integers. Also, since nodes may be visited more than once if desired, the degree constraints (2.20) specify only that the degrees must be even. (Although these constraints are non-linear, they can be easily linearised, using one additional variable for each node.) The crucial point, however, is that the so-called *connectivity constraints* (2.21) are exponential in number.

2.4 Flow-Based Formulations of the STSP

In this section, we adapt the formulations SCF and MCF, mentioned in Subsection 2.3.2, to the Steiner case. We also give some results concerned with the strength of the LP relaxations of our formulations.

2.4.1 Some preliminaries

At this point, we present some additional notation. Let $\tilde{G} = (V, A)$ be a directed graph, where the set of directed arcs A is obtained from the edge set E by replacing each edge $\{i, j\}$ with two directed arcs (i, j) and (j, i). For each arc $a \in A$, the cost c_a is viewed as being equal to the cost of the corresponding edge. For any node set $S \subset V$, let $\delta^+(S)$ denote the set of arcs in A whose tail is in S and whose head is in $V \setminus S$, and let $\delta^-(S)$ denote the set of arcs in A for which the reverse holds. For readability, we write $\delta^+(i)$ and $\delta^-(i)$ in place of $\delta^+(\{i\})$ and $\delta^-(\{i\})$, respectively. Finally, let $n_R = |V_R|$ denote the number of required nodes.

We will find the following lemma useful:

Lemma 1. *In an optimal solution to the STSP, no edge will be traversed more than once in either direction.*

This lemma is part of the folklore, but an explicit proof can be found in the appendix of [89]. An immediate consequence is that one can define a binary variable $\tilde{x}_a$ for each arc $a \in A$, taking the value 1 if and only if the salesman travels along a.

We will also need the following classical result from network flow theory, due to Gale [58] and Hoffman [71]:

Theorem 1 (Gale–Hoffman)**.** *Let $\tilde{G} = (V, A)$ be a directed graph, $s \in V$ be a source node, $d \in \mathbb{Q}_+^{|V|}$ be a demand vector with $d_s = 0$, and $\ell, u \in \mathbb{Q}_+^{|A|}$ be lower and upper bound vectors with $\ell \leq u$. There exists a feasible flow $f \in \mathbb{Q}_+^{|A|}$ satisfying (i) $\ell \leq f \leq u$ and (ii) $\sum_{a \in \delta^-(i)} f_a = d_i + \sum_{a \in \delta^+(i)} f_a$ for all $i \in V$ if and only if:*

$$\sum_{a \in \delta^-(S)} u_a \geq \sum_{i \in S} d_i + \sum_{a \in \delta^+(S)} \ell_a$$

holds for all $S \subset V \setminus \{s\}$.

2.4.2 An initial single-commodity flow formulation

Without loss of generality, assume that node 1 is required. By analogy with the case of the standard TSP, we imagine that the salesman departs the depot with $n_R - 1$ units of the commodity, and delivers one unit of that commodity to each required node. So, for each arc $a \in A$, let the new variable f_a represent the amount of the commodity passing through a. The single-commodity flow formulation (SCF) may then be adapted to the sparse graph setting as follows:

$$\begin{aligned}
\min \quad & \textstyle\sum_{a\in A} c_a \tilde{x}_a && && (2.23)\\
\text{s.t.} \quad & \textstyle\sum_{a\in\delta^+(i)} \tilde{x}_a \geq 1 && (\forall i \in V_R) && (2.24)\\
& \textstyle\sum_{a\in\delta^+(i)} \tilde{x}_a = \sum_{a\in\delta^-(i)} \tilde{x}_a && (\forall i \in V) && (2.25)\\
& \textstyle\sum_{a\in\delta^-(i)} f_a - \sum_{a\in\delta^+(i)} f_a = 1 && (\forall i \in V_R \setminus \{1\}) && (2.26)\\
& \textstyle\sum_{a\in\delta^-(i)} f_a - \sum_{a\in\delta^+(i)} f_a = 0 && (\forall i \in V \setminus V_R) && (2.27)\\
& 0 \leq f_a \leq (n_R - 1)\tilde{x}_a && (\forall a \in A) && (2.28)\\
& \tilde{x}_a \in \{0,1\} && (\forall a \in A). && (2.29)
\end{aligned}$$

The constraints (2.24) ensure that the salesman departs from each required node at least once, and the constraints (2.25) ensure that the salesman departs from each node as many times as he arrives. The constraints (2.26) impose that one unit of the commodity is delivered to each required node, and the constraints (2.27) ensure that the amount of commodity on board when leaving a non-required node is equal to the amount when arriving. The bounds (2.28) ensure that, if any of the commodity passes along an arc, then that arc appears in the tour.

In what follows, we refer to this SCF formulation as 'SCF1'. Note that it contains $\mathcal{O}(|E|)$ variables and $\mathcal{O}(|E|)$ constraints.

The following theorem characterises the projection of the LP relaxation of SCF1 into the space of the x variables:

Theorem 2. *Let P^1 be the polytope in $(\tilde{x}, f)$-space defined by the constraints (2.24)–(2.28) and the trivial bounds $\tilde{x} \in [0,1]^{|A|}$, let P^2 be the projection of P^1 into $\tilde{x}$-space, and let P^3 be the projection of P^2 into x-space, under the linear mapping $x_{ij} = \tilde{x}_{ij} + \tilde{x}_{ji}$ for all $\{i,j\} \in E$. Then P^3 is completely described by the following linear inequalities:*

$$\begin{aligned}
\textstyle\sum_{e\in\delta(i)} x_e &\geq 2 && (i \in V_R) && (2.30)\\
\textstyle\sum_{e\in\delta(S)} x_e &\geq 2\tfrac{|S\cap V_R|}{n_R-1} && (\forall S \subset V \setminus \{1\} : S \cap V_R \neq \emptyset) && (2.31)\\
x &\in [0,2]^{|E|}. && && (2.32)
\end{aligned}$$

Proof. Setting $\ell_a = 0$ and $u_a = (n_R - 1)\tilde{x}_a$ for all $a \in A$, $s = 1$, $d_i = 0$ for all $i \in V \setminus V_R$ and $d_i = 1$ for all $i \in V_R \setminus \{1\}$ in Theorem 1, we see that P^2 is described by the constraints (2.24), (2.25), the trivial bounds $\tilde{x} \in [0,1]^{|A|}$, and the inequalities

$$(n_R - 1) \sum_{a \in \delta^+(S)} \tilde{x}_a \geq |S \cap V_R| \tag{2.33}$$

for all $S \subset V \setminus \{1\}$ with $S \cap V_R \neq \emptyset$. Then, using (2.25), we can re-write (2.24) as

$$\sum_{a \in \delta^+(i) \cup \delta^-(i)} \tilde{x}_a \geq 2 \qquad (\forall i \in V_R)$$

and re-write the inequalities (2.33) in the form:

$$(n_R - 1) \sum_{a \in \delta^+(S) \cup \delta^-(S)} \tilde{x}_a \geq 2|S \cap V_R|.$$

The result then follows from the stated mapping of $\tilde{x}$ onto x. □

A consequence of Theorem 2 is the following:

Corollary 1. *The lower bound from the LP relaxation of SCF1 is never stronger than the lower bound from Fleischmann's formulation (2.19)–(2.22).*

Proof. The inequalities (2.30) are a special case of the inequalities (2.21), and the inequalities (2.31) are weaker than the inequalities (2.21). Moreover, an optimal solution to the LP relaxation of Fleischmann's formulation will always satisfy $x_e \leq 2$ for all $e \in E$. □

2.4.3 Strengthened single-commodity flow formulation

It is possible to strengthen SCF1 using the following argument. One can assume that, if any required node is visited more than once by the salesman, then the commodity is delivered on the first visit. Accordingly, for each node $i \in V \setminus \{1\}$, let r_i be the minimum number of required nodes (not including the depot) that the salesman must have visited when he leaves i for the first time. Also, by convention, let $r_1 = 0$. (Note that one can compute r_i for all $i \in V \setminus \{1\}$ efficiently, using Dijkstra's single-source shortest-path algorithm [40]). Now, the constraints (2.28) can be replaced with the following stronger constraints:

$$0 \leq f_{ij} \leq (n_R - r_i - 1)\tilde{x}_{ij} \qquad (\forall (i,j) \in A). \tag{2.34}$$

We call this strengthened SCF formulation 'SCF2'. We do not have a complete characterisation of the projection of the LP relaxation of SCF2 into the space of the x variables, but the following theorem gives a partial description:

Theorem 3. *Let P^1 be the polytope in $(\tilde{x}, f)$-space defined by the constraints (2.24)–(2.27), the strengthened constraints (2.34), and the trivial bounds $\tilde{x} \in [0,1]^{|A|}$. Let P^2 be the projection of P^1 into $\tilde{x}$-space, and let P^3 be the projection of P^2 into x-space. Also, for any set $S \subseteq V \setminus \{1\}$ such that $S \cap V_R \neq \emptyset$, let $T(S)$ be the set of all nodes that are not in S but are adjacent to at least one node in S. Finally, define $L(S) = \min_{i \in T(S)} r_i$ and $U(S) = \max_{i \in T(S)} r_i$. Then P^3 satisfies the inequalities (2.30) and (2.32), together with the following inequality for all such sets S and for $k = L(S), \dots, U(S)$:*

$$(n_R - k - 1) \sum_{e \in \delta(S)} x_e + 2 \sum_{\{i,j\} \in \delta(S) : j \in S} \max\{0, k - r_i\}\, x_{ij} \geq 2|S \cap V_R|. \qquad (2.35)$$

Proof. The fact that P^3 satisfies (2.30) and (2.32) follows from Theorem 2 and the fact that SCF2 dominates SCF1. Now, setting $\ell_{ij} = 0$ and $u_{ij} = (n_R - r_i - 1)\tilde{x}_{ij}$ for all $(i,j) \in A$, $s = 1$, $d_i = 0$ for all $i \in V \setminus V_R$ and $d_i = 1$ for all $i \in V_R \setminus \{1\}$ in Theorem 1, we see that P^2 is described by the constraints (2.24), (2.25), the trivial bounds $\tilde{x} \in [0,1]^{|A|}$, and the inequalities

$$\sum_{(i,j) \in \delta^-(S)} (n_R - r_i - 1)\tilde{x}_{ij} \geq |S \cap V_R|$$

for all $S \subset V \setminus \{1\}$ with $S \cap V_R \neq \emptyset$. Together with non-negativity on $\tilde{x}$ and the equations (2.25), this implies that P^2 satisfies

$$(n_R - k - 1) \sum_{(i,j) \in \delta^+(S) \cup \delta^-(S)} \tilde{x}_{ij} + 2 \sum_{(i,j) \in \delta^-(S)} \max\{0, k - r_i\}(\tilde{x}_{ij} + \tilde{x}_{ji}) \geq 2|S \cap V_R|$$

for all such S. The result then follows from the mapping of $\tilde{x}$ onto x. □

Note that, as one would expect, the inequalities (2.35) generalise and dominate the inequalities (2.31). (To see this, just set $k = L(S)$ and note that $L(S)$ will never exceed $n_R - 1 - |S \cap V_R|$.) We conjecture that the constraints (2.30), (2.32) and (2.35) give a complete description of the projection. We also conjecture that the lower bound from formulation SCF2 always lies between the one from formulation SCF1 and the one from Fleischmann's formulation. (In practice, it is usually only slightly better than that of SCF1. See Table 2.3 in Section 2.6.)

2.4.4 Multi-commodity flow formulation

Now we move on to a multi-commodity flow (MCF) formulation. Similar to the MCF formulation for the standard TSP, we assume that the salesman leaves the depot (node 1) with one unit of commodity for each required node. Accordingly, let the binary variable g_a^k be 1 if and only if commodity k passes through arc a, for every $k \in V_R \setminus \{1\}$ and $a \in A$. The resulting formulation then consists of minimising (2.23) subject to the following constraints:

$$\sum_{a \in \delta^+(i)} \tilde{x}_a \geq 1 \qquad (\forall i \in V_R) \tag{2.36}$$

$$\sum_{a \in \delta^+(i)} \tilde{x}_a = \sum_{a \in \delta^-(i)} \tilde{x}_a \qquad (\forall i \in V) \tag{2.37}$$

$$\sum_{a \in \delta^-(i)} g_a^k - \sum_{a \in \delta^+(i)} g_a^k = 0 \qquad (\forall i \in V \setminus \{1\}; k \in V_R \setminus \{1, i\}) \tag{2.38}$$

$$\sum_{a \in \delta^-(k)} g_a^k - \sum_{a \in \delta^+(k)} g_a^k = 1 \qquad (\forall k \in V_R \setminus \{1\}) \tag{2.39}$$

$$\sum_{a \in \delta^-(1)} g_a^k - \sum_{a \in \delta^+(1)} g_a^k = -1 \qquad (\forall k \in V_R \setminus \{1\}) \tag{2.40}$$

$$\tilde{x}_a \geq g_a^k \qquad (\forall a \in A; k \in V_R \setminus \{1\}) \tag{2.41}$$

$$\tilde{x}_a \in \{0, 1\} \qquad (\forall a \in A) \tag{2.42}$$

$$g_a^k \in \{0, 1\} \qquad (\forall a \in A \& k \in V_R \setminus \{1\}). \tag{2.43}$$

The constraints are interpreted along similar lines to those of the formulations already seen.

This MCF formulation has $\mathcal{O}(n_R|E|)$ variables and $\mathcal{O}(n_R|E|)$ constraints. As for the projection into the space of x variables, we have the following result:

Theorem 4. *Let P^1 be the polytope in $(\tilde{x}, g)$-space defined by the constraints (2.37)–(2.41) and the trivial bounds $\tilde{x} \in [0,1]^{|A|}$, let P^2 be the projection of P^1 into $\tilde{x}$-space, and let P^3 be the projection of P^2 into x-space. Then P^3 is completely described by the inequalities (2.21), (2.30) and (2.32).*

Proof. Let k be an arbitrary node in $V_R \setminus \{1\}$. The classical *max-flow min-cut* theorem [55] implies that imposing the constraints (2.38)–(2.41) for the given k is equivalent to imposing the following inequalities:

$$\sum_{a \in \delta^+(S)} \tilde{x}_a^* \geq 1 \qquad (\forall S \subset V_R \setminus \{1\} : k \in S).$$

Therefore, P^2 is completely described by (2.37), the trivial bounds $\tilde{x} \in [0,1]^{|A|}$ and the constraints

$$\sum_{a \in \delta^+(S)} x_a \geq 1 \qquad (\forall S \subset V_R \setminus \{1\} : S \cap V_R \neq \emptyset).$$

Using (2.37), these last inequalities can be re-written in the form:

$$\sum_{a \in \delta^+(S) \cup \delta^-(S)} x_a \geq 2.$$

The result then follows from the mapping of $\tilde{x}$ onto x. □

An immediate consequence of Theorem 4 is the following:

Corollary 2. *The lower bound from the LP relaxation of MCF is equal to the lower bound from Fleischmann's formulation (2.19)–(2.22).*

2.5 Time-Staged Formulations of the STSP

In this section, we adapt the TS formulation for the standard TSP, mentioned in Subsection 2.3.2, to the Steiner case. A simple formulation is presented in the following subsection. A method to reduce the number of variables is presented in Subsection 2.5.2. Then, in Subsection 2.5.3, we evaluate the total number of variables and constraints in each of the formulations that we have considered.

2.5.1 An initial time-staged formulation

In this context, it is natural to have one time stage for each time that an edge of G is traversed (in either direction). In terms of the classical STSP formulation given in Subsection 2.3.3, the total number of time stages will then be equal to $\sum_{e \in E} x_e$. The problem here is that we do not know this value in advance. Observe, however, that Lemma 1 implies that it cannot exceed $2|E|$.

Now, let A be defined as in Subsection 2.4.1, and recall that $|A| = 2|E|$. For all $a \in A$ and all $1 \leq k \leq |A|$, let the binary variable r_a^k take the value 1 if and only if arc a is the kth arc to be traversed in the tour. Our TS formulation for the

STSP is as follows:

$$\begin{aligned}
\min \quad & \textstyle\sum_{k=1}^{|A|}\sum_{a\in A} c_a r_a^k && && (2.44)\\
\text{s.t.} \quad & \textstyle\sum_{a\in\delta^+(1)} r_a^1 = 1 && && (2.45)\\
& r_a^1 = 0 && (a \in A \setminus \delta^+(1)) && (2.46)\\
& \textstyle\sum_{k=1}^{|A|}\sum_{a\in\delta^+(1)} r_a^k = \sum_{k=1}^{|A|}\sum_{a\in\delta^-(1)} r_a^k && && (2.47)\\
& \textstyle\sum_{k=1}^{|A|}\sum_{a\in\delta^+(i)} r_a^k \geq 1 && (\forall i \in V_R) && (2.48)\\
& \textstyle\sum_{a\in\delta^-(i)} r_a^k = \sum_{a\in\delta^+(i)} r_a^{k+1} && (\forall i \in V\setminus\{1\}; 1\leq k\leq |A|-1) && (2.49)\\
& r_a^k \in \{0,1\} && (\forall a\in A, 1\leq k\leq |A|). && (2.50)
\end{aligned}$$

Constraints (2.45) and (2.46) ensure that the salesman departs from the depot in the first time stage, and constraint (2.47) ensures that he arrives at the depot as many times as he leaves it. Constraints (2.48) ensure that each required node is visited at least once. Constraints (2.49) ensure that, if the salesman arrives at a non-depot node in any given time stage, then he must depart from it in the subsequent time stage. Finally, constraints (2.50) are the usual binary conditions.

We call this TS formulation 'TS1'. Note that TS1 has $\mathcal{O}(|E|^2)$ variables and $\mathcal{O}(n|E|)$ constraints. We will see in Section 2.6 that the lower bound from this formulation is very weak in practice.

2.5.2 Bounding the number of edge traversals

Clearly, one could reduce the number of variables and constraints in the TS formulation if one had a better upper bound on the total number of times that the salesman traverses an edge of G. The following theorem provides such a bound:

Theorem 5. *There exists an optimal STSP solution in which the total number of edge traversals (in either direction) does not exceed* $2(n-1)$.

For the proof of this theorem, we will use the following lemma.

Lemma 2. *Let H be a connected graph with k nodes and ℓ edges. If $\ell > 2(k-1)$, then there exists a cycle C in H such that the graph arising when the edges of C are deleted from H is still connected.*

Proof. Let T be a spanning tree in H, and let H' be the graph resulting if the edges of T are deleted from H. For the number ℓ' of edges of H' we have $\ell' = \ell - (k-1)$, which, by the hypothesis in the lemma, is greater than $k-1$. Clearly, the number of nodes of H' is equal to k.

Now, let T' be a spanning forest in H'. Note that H' may fail to be connected. Firstly, if one of the connected components of H' contains an edge e other than those in T', then let C be the cycle defined by taking e and the path in T' connecting the end-nodes of e. Clearly, deleting the edges of C from H leaves a connected graph because connectivity is assured by the tree T.

But, secondly, it is impossible that all connected components of H' contain no other edges except those in T': In that case, H' would be a forest, and hence have at most $k-1$ edges. But the number of edges of H' is greater than $k-1$, a contradiction. $\square$

We can now complete the proof of the theorem.

Proof. (of Theorem 5) Let x be an optimal solution to the STSP, which has, among all optimal solutions, the smallest number of edge traversals.

Construct a graph H by starting with the node set V, and precisely x_e copies of the edge e, for all $e \in E$. Then delete every isolated node from H. The number of nodes k of H is at most n, and the number of edges is $\ell := \sum_{e \in E} x_e$.

For the sake of contradiction, we assume that $\ell > 2(n-1)$. If that is the case, then Lemma 2, is applicable. Let C be a cycle with the property given in the lemma, and let F be its edge set. For every $e \in E$, denote by y_e the number of times the edge e occurs in C. The fact that after deleting the edges of C from H, a connected graph remains, implies that $x - y$ is a solution to the STSP, whose total cost is at most that of x. Thus, $x - y$ is an optimal solution in which the total number of edge traversals is smaller than in x, contradicting the choice of x.

Thus, we conclude that $\sum_e x_e = \ell \leq 2(n-1)$. $\square$

An immediate consequence of this theorem is that one does not need to define the variables r_a^k in the TS formulation when $k > 2(n-1)$. The constraints in which $k > 2(n-1)$ can be dropped as well. As a result, the number of variables and constraints in the TS formulation can be reduced to $\mathcal{O}(n|E|)$ and $\mathcal{O}(n^2)$, respectively. We call this smaller TS formulation 'TS2'. Since TS2 is obtained from TS1 by eliminating variables, the lower bound from the LP relaxation of TS2 is no worse than the one from TS1. We will see in Section 2.6 that it is usually slightly better (though still poor).

Formulation	Classical	SCF	MCF	TS1	TS2
Variables	$\lvert E\rvert$	$\mathcal{O}(\lvert E\rvert)$	$\mathcal{O}(n_R\lvert E\rvert)$	$\mathcal{O}(\lvert E\rvert^2)$	$\mathcal{O}(n\lvert E\rvert)$
Constraints	$\mathcal{O}(2^{n_R})$	$\mathcal{O}(\lvert E\rvert)$	$\mathcal{O}(n_R\lvert E\rvert)$	$\mathcal{O}(n\lvert E\rvert)$	$\mathcal{O}(n^2)$

Tab. 2.1: Alternative STSP formulations and their size

2.5.3 Summary

Table 3.2 displays, for each of the STSP formulations that we have considered, bounds on the total number of variables and constraints. Here, 'classical' refers to the formulation of Fleischmann [53] mentioned in Subsection 2.3.3, 'SCF' refers to either of the single-commodity flow formulations SCF1 and SCF2, given in Subsections 2.4.2 and 2.4.3, 'MCF' refers to the multi-commodity flow formulation given in Subsection 2.4.4, 'TS1' refers to the time-staged formulation given in Subsection 2.5.1, and 'TS2' refers to the reduced time-staged formulation given in Subsection 2.5.2.

Observe that, in the case of real road networks, the graph G is typically very sparse, and we have $|E| = \mathcal{O}(n)$. Moreover, in many cases, n_R is small relative to n. So none of the formulations are particularly large. Therefore, the main thing determining whether or not the formulations are useful in practice is likely to be the amount of computing time and memory taken to solve them by branch-and-bound. This in turn is influenced by the quality of the LP relaxations and the time taken to solve those. The next section explores these computational issues.

We close this section with a remark on the MTZ formulation of the TSP. The MTZ formulation is based on the idea of determining the order in which the nodes are visited. Since nodes can be visited multiple times in the Steiner case, a unique order cannot be determined. As a result, it does not appear possible to adapt the MTZ formulation to the STSP. This is not a problem, though, given the extreme weakness of the MTZ formulation.

2.6 Computational Results

Some experiments were conducted, in order to compare empirically our compact formulations of the STSP. The starting point was the creation of ten sparse graphs, with $n \in \{25, 50, 75, \ldots, 250\}$. The following procedure was used, which is designed to lead to graphs that resemble real-life road networks:

1. Set $V = \{1, \ldots, n\}$ and $E = \emptyset$.
2. Place the n nodes at random in a circle of radius 100.
3. Define the set of *potential edges* $P = \{\{i, j\} : 1 \leq i < j \leq n\}$.
4. For each $\{i, j\} \in P$, let the cost c_{ij} be the Euclidean distance between the end-nodes, rounded to the nearest integer.
5. Sort the potential edges in non-decreasing order of cost.
6. Examine each edge in P in turn. Insert it into E if both of the following conditions are satisfied:
   (a) It does not cross one of the edges already inserted into E.
   (b) It does not form an angle of less than $60°$ with an edge that has already been inserted into E and with which it shares an end-node.
7. If there are no isolated nodes in the resulting graph, output the graph. Otherwise, repeat the procedure.

For each of the ten graphs created, three STSP instances were created, by varying the proportion of required nodes. In the first such instance, each node has a probability of $1/3$ of being required. In the second instance, the probability is $2/3$, and in the third instance, all nodes are required. In each case, one of the required nodes was selected at random to be the depot. Figure 2.1 shows the

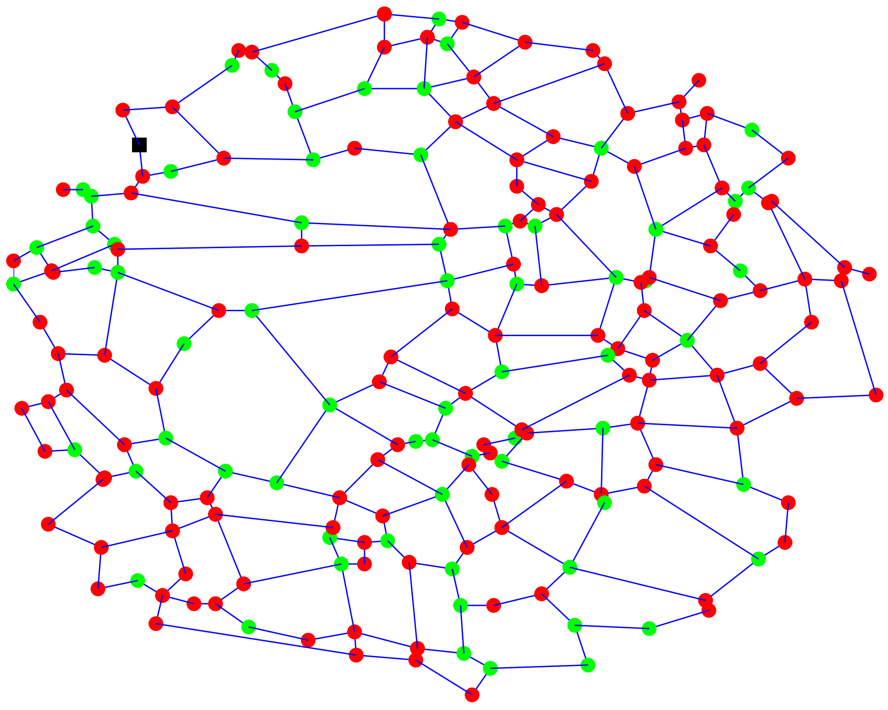


Fig. 2.1: STSP instance with $n = 200$ and $n_R = 133$.

instance that was obtained for $n = 200$ and a probability of 2/3. The required and non-required nodes are represented by small red and green circles, respectively, and the depot is represented by the small black square. The optimal solution for this instance can be deduced from Figure 2.2, in which nodes that are not visited and edges that are not traversed have been omitted.

A program was written in Microsoft Visual C that reads an instance from a file, and then outputs five integer programs, corresponding to the formulations SCF1, SCF2, MCF, TS1 and TS2. Each of the resulting 150 ($10 \times 3 \times 5$) integer programs was fed into the branch-and-bound solver of IBM CPLEX version 12.3. The computer used was a PC with a 1.6 GHz Intel Core i7 processor, with 8 GB of RAM, operating under Windows 7. Default CPLEX settings were used.

For each integer program, a time limit of 5000 seconds was imposed. We were able to find the optimal solutions to 28 out of the 30 instances within the time limit. The two that were not solved were the ones with $n \in \{225, 250\}$ and 2/3 of

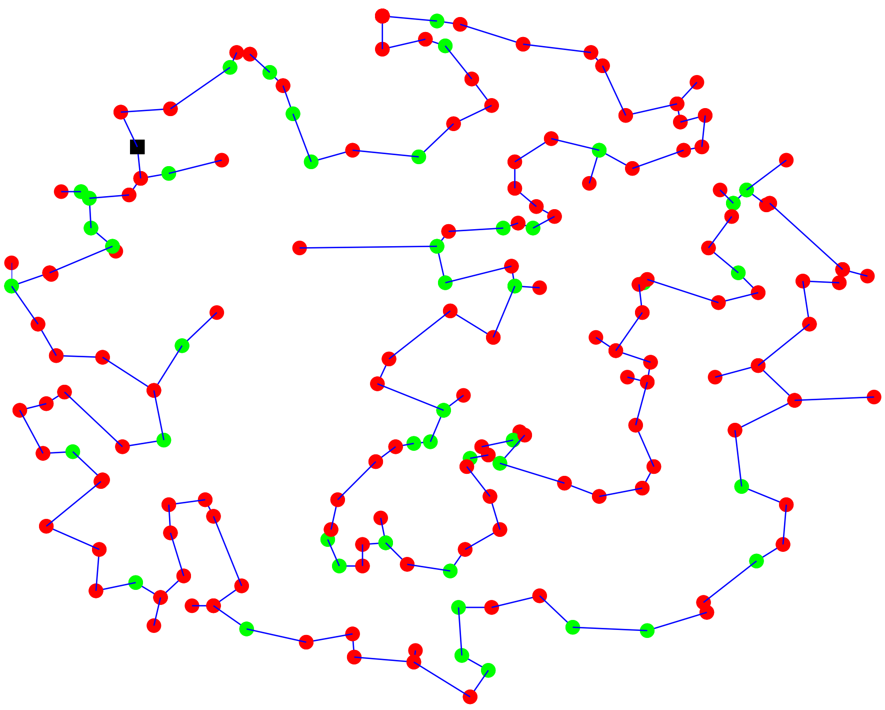


Fig. 2.2: Optimal tour for the STSP instance with $n = 200$ and $n_R = 133$.

the nodes being required.

Table 2.2 shows, for each of the thirty instances and each of the five formulations, the time taken to solve the LP relaxation. We see that the LPs associated with the SCF formulations are very easy to solve, whereas the LPs associated with the other formulations consume more time. This is probably due to the fact that the SCF formulations have the fewest variables (see again Table 3.2). For a similar reason, the LP relaxation of TS2 can be solved more quickly than that of TS1.

Table 2.3 shows, for the 28 instances that were solved to optimality, and for the same five formulations, the *percentage integrality gap*, i.e., the difference between the lower bound given by the LP solution and the cost of the true integer optimum, expressed as a percentage of the latter. We see that the gaps are far smaller for the MCF formulation than for the SCF and TS formulations. The superiority of the MCF formulation over the formulation SCF1 can be explained by comparing Corollaries 1 and 2. We also see that the gaps for TS2 are a little better than

Instance	SCF1	SCF2	MCF	TS1	TS2
25-1/3	0.01	0.02	0.01	0.19	0.12
50-1/3	0.02	0.01	0.11	1.00	0.73
75-1/3	0.02	0.03	0.45	3.63	1.90
100-1/3	0.03	0.03	1.19	9.25	5.15
125-1/3	0.03	0.03	2.93	21.86	11.48
150-1/3	0.03	0.05	9.38	33.66	18.80
175-1/3	0.05	0.05	30.33	66.27	38.80
200-1/3	0.05	0.06	20.09	184.16	93.90
225-1/3	0.06	0.06	65.81	240.76	136.52
250-1/3	0.08	0.08	52.80	326.57	195.77
25-2/3	0.02	0.01	0.01	0.23	0.16
50-2/3	0.02	0.02	0.44	1.44	0.94
75-2/3	0.02	0.02	2.09	5.58	2.54
100-2/3	0.02	0.03	7.33	10.30	6.33
125-2/3	0.03	0.03	12.74	39.78	17.71
150-2/3	0.03	0.05	17.30	51.92	28.58
175-2/3	0.06	0.06	190.31	106.21	58.6
200-2/3	0.06	0.06	379.52	181.63	94.72
225-2/3	0.06	0.06	707.65	286.29	149.31
250-2/3	0.09	0.08	5521.67	516.30	327.87
25-3/3	0.02	0.02	0.06	0.34	0.30
50-3/3	0.01	0.02	1.34	1.48	1.15
75-3/3	0.02	0.03	3.96	5.43	3.51
100-3/3	0.03	0.03	8.80	14.10	8.19
125-3/3	0.05	0.03	18.49	35.51	22.53
150-3/3	0.05	0.05	33.20	76.24	40.89
175-3/3	0.06	0.05	634.25	151.26	78.83
200-3/3	0.06	0.06	2238.60	219.71	114.58
225-3/3	0.06	0.08	2839.66	454.76	237.78
250-3/3	0.08	0.09	5566.54	727.42	436.74

Tab. 2.2: Time taken to solve LP relaxation, in seconds.

Instance	SCF1	SCF2	MCF	TS1	TS2
25-1/3	24.42	22.51	0.00	43.91	42.01
50-1/3	22.81	22.01	0.38	41.70	40.20
75-1/3	24.50	23.82	0.41	43.64	41.76
100-1/3	30.33	30.12	0.00	51.61	49.72
125-1/3	34.51	34.20	0.09	44.80	43.71
150-1/3	33.42	33.21	1.70	45.60	44.43
175-1/3	35.44	35.32	0.04	44.84	43.83
200-1/3	32.20	31.90	0.00	40.05	39.22
225-1/3	35.51	35.40	1.20	45.01	44.12
250-1/3	33.40	33.22	0.54	41.82	40.98
25-2/3	18.91	17.21	0.00	33.33	31.31
50-2/3	19.60	19.01	0.66	30.36	28.76
75-2/3	20.81	20.50	1.91	25.59	24.57
100-2/3	24.13	23.82	0.23	31.29	30.19
125-2/3	26.42	26.21	1.59	33.18	32.10
150-2/3	27.44	27.20	1.48	33.48	32.39
175-2/3	24.62	24.53	2.01	28.19	27.68
200-2/3	28.61	28.41	0.87	32.08	31.40
25-3/3	14.21	13.71	1.87	19.40	18.44
50-3/3	18.52	18.19	0.95	24.16	22.64
75-3/3	14.03	13.82	0.00	17.99	16.83
100-3/3	15.42	15.20	1.00	19.31	18.17
125-3/3	18.24	17.87	1.26	22.09	21.23
150-3/3	17.41	17.30	0.75	20.18	19.46
175-3/3	16.63	16.54	0.99	17.93	17.61
200-3/3	17.10	16.91	0.35	17.68	17.46
225-3/3	14.32	14.19	0.50	15.28	14.98
250-3/3	17.92	17.88	0.82	19.27	18.87

Tab. 2.3: Percentage integrality gaps.

those for TS1. This is due to the fact that TS2 can be obtained from TS1 by fixing some variables to zero. Finally, observe that, as the proportion of required nodes increases, the gaps decrease for the SCF and TS formulations. The reason for this phenomenon is not clear. Table 2.4 shows, for each of the 28 instances solved, the number of branch-and-bound nodes needed to solve the integer programs to proven optimality. A dash (−) indicates that the given formulation could not be solved within the given time limit. We see that, for smaller values of n, the MCF formulation gives the true integer optimal solution without any branch-and-bound. For the SCF formulations, quite a few branch-and-bound nodes are needed in general. For the TS formulations, an excessive number of nodes is needed even

Instance	SCF1	SCF2	MCF	TS1	TS2
25-1/3	32	0	0	12370	8733
50-1/3	650	529	0	–	–
75-1/3	2416	2530	0	–	–
100-1/3	4331	1414	0	–	–
125-1/3	4133	5588	0	–	–
150-1/3	20931	139198	73	–	–
175-1/3	64354	21491	0	–	–
200-1/3	12040	37628	0	–	–
225-1/3	786162	442307	210	–	–
250-1/3	96728	–	34	–	–
25-2/3	0	0	0	6576	9908
50-2/3	655	606	0	–	–
75-2/3	955	3354	37	–	–
100-2/3	861	2383	0	–	–
125-2/3	19505	63716	753	–	–
150-2/3	143990	72724	–	–	–
175-2/3	76311	153114	–	–	–
200-2/3	237057	2296863	–	–	–
25-3/3	53	199	0	78302	38508
50-3/3	1812	623	0	–	–
75-3/3	613	637	0	–	–
100-3/3	5590	6884	234	–	–
125-3/3	19675	21128	–	–	–
150-3/3	220850	134777	–	–	–
175-3/3	238167	603673	–	–	–
200-3/3	78741	44442	–	–	–
225-3/3	75065	420801	–	–	–
250-3/3	942746	–	–	–	–

Tab. 2.4: Number of branch-and-bound nodes.

for small values of n.

It should be noted that, in a few cases, the number of branch-and-bound nodes is zero even though the integrality gap is positive. The explanation for this anomalous behaviour is that, in some cases, CPLEX automatically appended some of its own internal cutting planes to the LP relaxation. If one were to switch off all internal cutting plane generation in CPLEX, then branching would become necessary whenever there is a positive integrality gap.

Finally, Table 2.5 shows the total time, in seconds, taken by the branch-and-bound procedure. We see that the SCF formulations perform remarkably well, enabling one to find a provably optimal solution even for quite large values of n.

Instance	SCF1	SCF2	MCF	TS1	TS2
25-1/3	0.25	0.00	0.00	79.68	58.02
50-1/3	2.12	0.20	0.00	–	–
75-1/3	4.01	4.98	0.00	–	–
100-1/3	10.03	7.07	0.00	–	–
125-1/3	11.53	13.74	0.00	–	–
150-1/3	38.03	162.40	102.10	–	–
175-1/3	147.12	64.52	0.00	–	–
200-1/3	33.80	76.46	0.00	–	–
225-1/3	2221.34	1032.31	1420.6	–	–
250-1/3	438.72	–	290.8	–	–
25-2/3	0.00	0.00	0.00	104.96	71.84
50-2/3	1.48	1.86	0.00	–	–
75-2/3	3.46	4.81	12.15	–	–
100-2/3	6.74	6.65	0.00	–	–
125-2/3	26.76	79.94	3676.48	–	–
150-2/3	202.33	97.02	–	–	–
175-2/3	162.58	223.85	–	–	–
200-2/3	457.07	4235.12	–	–	–
25-3/3	0.22	0.08	0.00	623.69	271.42
50-3/3	1.47	0.23	0.00	–	–
75-3/3	0.42	2.96	0.00	–	–
100-3/3	9.19	9.14	608.90	–	–
125-3/3	45.41	39.67	–	–	–
150-3/3	211.23	149.15	–	–	–
175-3/3	393.43	1066.05	–	–	–
200-3/3	144.32	125.46	–	–	–
225-3/3	282.00	1064.5	–	–	–
250-3/3	4147.20	–	–	–	–

Tab. 2.5: Branch-and-bound time, in seconds.

This is presumably due to the fact that, although the number of nodes is high, the time taken to process each node is small, due to the small number of variables.

For the MCF formulation, the time is small or even zero in many cases. Nevertheless, as the instances grow in size, MCF starts to struggle. This is probably due to the comparatively large number of variables. The TS formulations, on the other hand, perform consistently very poorly.

All things considered, we would recommend using the MCF formulation for n up to 100 or so, and SCF2 for n between 100 and 250.

2.7 Some Related Problems

Many variants and extensions of the TSP have appeared in the literature, such as the *Orienteering Problem* (e.g., [50, 52, 64]), the *Prize-Collecting* TSP (e.g., [7, 8, 50]), the *Capacitated Profitable Tour Problem* (e.g., [50, 75]), the TSP with *Time Windows* (e.g., [4, 45]) and the *Sequential Ordering Problem* [48]. For each of these problems, it is easy to define a 'Steiner' version. It suffices to define the problem on a general graph $G = (V, E)$, designate node 1 as the 'depot', define a set $V_R \subset V \setminus \{1\}$ of 'customer' nodes, permit edges to be traversed more than once if desired, and permit nodes to be visited more than once if desired.

In this section, we explore possible ways to formulate these other problems of 'Steiner' type. For the sake of brevity, however, we restrict attention to three specific problems, which we call the *Steiner Orienteering Problem*, the *Steiner Capacitated Profitable Tour Problem*, and the *Steiner TSP with Time Windows*. These are considered in the following three subsections.

2.7.1 The Steiner Orienteering Problem

We define the *Steiner Orienteering Problem* (SOP) as follows. For each $e \in E$, we are given a non-negative cost c_e. For each $i \in V_R$, we are given a positive *revenue* (or 'prize') p_i. The nodes in V_R do not all have to be visited, but the revenue can only be collected from such a node if that node is visited at least once. We are also given an upper bound U on the total route cost. The task is to maximise the sum of the prizes collected, subject to the upper bound.

Observe that Lemma 1 applies to the SOP. To see this, let $V^* \subset V_R$ be the set of nodes whose prizes are collected in the optimal solution. The optimal solution is then also optimal for a STSP instance defined on the same graph, but with V_R set to V^*.

Knowing that Lemma 1 applies, it is easy to adapt the classical (non-compact) formulation of the STSP, presented in Subsection 2.3.3, to the SOP. For each $i \in V_R$, we define a new binary variable y_i, taking the value 1 if and only if the salesman collects a prize from node i. We then change the objective function from (2.19) to:

$$\max \sum_{i \in V_R} p_i y_i, \tag{2.51}$$

replace the connectivity constraints (2.21) with:

$$\sum_{e \in \delta(S)} x_e \geq 2y_i \qquad (i \in V_R, S \subseteq V \setminus \{1\} : i \in S), \tag{2.52}$$

and add the route-cost constraint

$$\sum_{e \in E} c_e x_e \leq U. \tag{2.53}$$

It is also easy to adapt the TS formulation of the STSP (Subsection 2.5.1) to the SOP. It suffices to add the y_i variables mentioned above, change the objective function from (2.45) to (2.51), add the 'expanded' route-cost constraint

$$\sum_{k=1}^{|A|} \sum_{a \in A} c_a r_a^k \leq U,$$

and replace the constraints (2.48) with the constraints

$$\sum_{k=1}^{|A|} \sum_{a \in \delta^+(i)} r_a^k \geq y_i \qquad (\forall i \in V_R). \tag{2.54}$$

Moreover, Theorem 5, given in Subsection 2.5.1, applies to the SOP as well (for the same reason that Lemma 1 applies). So one can reduce the number of stages to $2(|V| - 1)$, without losing any optimal solutions.

It is also easy to adapt the MCF formulation of the STSP (Subsection 2.4.4) to the SOP. It suffices to add the same y_i variables, change the objective function from (2.23) to (2.51), change the right-hand sides of constraints (2.36) and (2.39) from 1 to y_i, change the right-hand sides of constraints (2.40) from -1 to $-y_i$, and add the 'directed' route-cost constraint:

$$\sum_{a \in A} c_a \tilde{x}_a \leq U. \tag{2.55}$$

As for the SCF formulation of the STSP (Subsection 2.4.2), there is an elegant way to adapt it to the SOP, which leads to an LP relaxation with desirable properties. The key is to redefine the variables f_a, so that:

- if arc a is traversed (i.e., $\tilde{x}_a = 1$), then f_a represents the total cost accumulated so far when the salesman begins to traverse the arc;
- if arc a is not traversed (i.e., $\tilde{x}_a = 0$), then $f_a = 0$.

Once this is done, one can introduce the same additional y_i variables, and use the objective function (2.51), along with the following constraints:

$$\textstyle\sum_{a\in\delta^+(1)} \tilde{x}_a \geq 1 \tag{2.56}$$

$$\textstyle\sum_{a\in\delta^+(i)} \tilde{x}_a \geq y_i \qquad (\forall i \in V_R) \tag{2.57}$$

$$\textstyle\sum_{a\in\delta^+(i)} \tilde{x}_a = \sum_{a\in\delta^-(i)} \tilde{x}_a \qquad (\forall i \in V) \tag{2.58}$$

$$\textstyle\sum_{a\in\delta^-(1)} (f_a + c_a\tilde{x}_a) - \sum_{a\in\delta^+(1)} f_a \leq U \tag{2.59}$$

$$\textstyle\sum_{a\in\delta^+(i)} f_a = \sum_{a\in\delta^-(i)} (f_a + c_a\tilde{x}_a) \qquad (\forall i \in V \setminus \{1\}) \tag{2.60}$$

$$0 \leq f_a \leq (U - c_a)\tilde{x}_a \qquad (\forall a \in A) \tag{2.61}$$

$$\tilde{x}_a \in \{0,1\} \qquad (\forall a \in A) \tag{2.62}$$

$$y_i \in \{0,1\} \qquad (\forall i \in V_R). \tag{2.63}$$

We then have the following analogue of Theorem 2:

Proposition 1. *Let P^1 be the polytope in $(\tilde{x}, y, f)$-space defined by the constraints (2.56)–(2.61) and the trivial bounds $\tilde{x} \in [0,1]^{|A|}$ and $y \in [0,1]^{|A|}$. Let P^2 be the projection of P^1 into $(\tilde{x}, y)$-space, and let P^3 be the projection of P^2 into (x,y)-space. Then P^3 is described by the inequality (2.53) and the following linear inequalities:*

$$\begin{array}{rl}
\sum_{e\in\delta(1)} x_e \geq 2 & \\
\sum_{e\in\delta(i)} x_e \geq 2y_i & (i \in V_R) \\
\sum_{e\in\delta(S)} x_e \geq \frac{2}{U} \sum_{\{i,j\}\in E:\{i,j\}\cap S\neq\emptyset} c_e x_e & (\forall S \subset V \setminus \{1\}) \\
(x,y) \in [0,2]^{|E|} \times [0,1]^{|E|}. &
\end{array}$$

Proof. Similar to the proof of Theorem 2. One small addition is that one must sum together the constraint (2.59) and all of the constraints (2.60), and then replace pairs of $\tilde{x}$ variables with their corresponding x variables, to obtain the inequality (2.53). $\square$

As in the case of the STSP (Subsection 2.4.3), it is possible to strengthen this SCF formulation of the SOP. Indeed, if a given arc (i, j) is traversed, then the smallest value that f_{ij} can take is equal to the cost of the shortest path from the depot to node i. Similarly, the largest value that f_{ij} can take is equal to $U - c_{ij}$ minus the cost of the shortest path from node j to the depot. One can adjust the constraints (2.61) accordingly, and then derive a stronger projection result, analogous to Theorem 3. We omit details, for the sake of brevity.

2.7.2 The Steiner Capacitated Profitable Tour Problem

The *Steiner Capacitated Profitable Tour Problem* (SCPTP) is similar to the SOP, but with the following differences:

- We are given a positive *demand* q_i for each $i \in V_R$, in addition to the revenue p_i.
- If we wish to gain the revenue for a given $i \in V_R$, then we have to deliver the demand q_i.
- Instead of an upper bound U on the route cost, we are given a vehicle capacity Q, which does not exceed the sum of the demands. The total demand of the serviced customers must not exceed Q.
- The task is to find a tour of maximum total profit, where the profit is defined as the sum of the revenues gained, minus the cost of the edges traversed.

Observe that Lemma 1 applies to the SCPTP, for the same reason that it applies to the SOP. Then, one can easily adapt the classical formulation of the

STSP to the SCPTP. We use the same binary variables y_i as used in the previous subsection, change the objective function from (2.19) to

$$\max \sum_{i \in V_R} p_i y_i - \sum_{e \in E} c_e x_e,$$

replace the connectivity constraints (2.21) with the constraints (2.52), and add the capacity constraint

$$\sum_{i \in V_R} q_i y_i \leq Q. \tag{2.64}$$

One can adapt the TS formulation in a similar way. It suffices to add the same y_i variables, add the capacity constraint (2.64), change the objective function (2.45) to

$$\max \sum_{i \in V_R} p_i y_i - \sum_{k=1}^{|A|} \sum_{a \in A} c_a r_a^k,$$

and replace the constraints (2.48) with the constraints (2.54). Moreover, Theorem 5 is again applicable.

As for the SCF formulation, we propose again to redefine the variables f_a. Now, f_a represents the total load (if any) that is carried along the arc a. Then, again using the additional y_i variables, it suffices to:

$$\max \sum_{i \in V_R} p_i y_i - \sum_{a \in A} c_a \tilde{x}_a \tag{2.65}$$

subject to the following constraints:

$$\textstyle\sum_{a \in \delta^+(1)} \tilde{x}_a \geq 1 \tag{2.66}$$

$$\textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a \geq y_i \qquad (\forall i \in V_R) \tag{2.67}$$

$$\textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a = \sum_{a \in \delta^-(i)} \tilde{x}_a \qquad (\forall i \in V) \tag{2.68}$$

$$\textstyle\sum_{a \in \delta^+(1)} f_a - \sum_{a \in \delta^-(1)} f_a \leq Q \tag{2.69}$$

$$\textstyle\sum_{a \in \delta^-(i)} f_a - \sum_{a \in \delta^+(i)} f_a = q_i y_i \qquad (\forall i \in V_R) \tag{2.70}$$

$$\textstyle\sum_{a \in \delta^-(i)} f_a - \sum_{a \in \delta^+(i)} f_a = 0 \qquad (\forall i \in V \setminus (V_R \cup \{1\})) \tag{2.71}$$

$$0 \le f_a \le Q\tilde{x}_a \quad (\forall a \in A) \tag{2.72}$$

$$\tilde{x}_a \in \{0,1\} \quad (\forall a \in A) \tag{2.73}$$

$$y_i \in \{0,1\} \quad (\forall i \in V_R). \tag{2.74}$$

The analogue of Theorem 2 is now as follows:

Proposition 2. *Let P^1 be the polytope in $(\tilde{x}, y, f)$-space defined by the constraints (2.64) and (2.66)–(2.71), and the trivial bounds $\tilde{x} \in [0,1]^{|A|}$ and $y \in [0,1]^{|A|}$. Let P^2 be the projection of P^1 into $(\tilde{x}, y)$-space, and let P^3 be the projection of P^2 into (x, y)-space. Then P^3 is described by the inequality (2.64), together with the following inequalities:*

$$\begin{array}{rl}
\sum_{e\in\delta(1)} x_e \ge 2 & \\
\sum_{e\in\delta(i)} x_e \ge 2y_i & (i \in V_R) \\
\sum_{e\in\delta(S)} x_e \ge \frac{2}{Q}\sum_{i\in S\cap V_R} q_i y_i & (\forall S \subseteq V \setminus \{1\} : S \cap V_R \ne \emptyset) \\
(x,y) \in [0,2]^{|E|} \times [0,1]^{|E|}. &
\end{array}$$

Proof. Similar to that of Theorem 2. One small addition is that one must sum together the constraints (2.69)–(2.71), and then replace pairs of $\tilde{x}$ variables with their corresponding x variables, to obtain the inequality (2.64). □

As for the MCF formulation described in Subsection 2.4.4, one can adapt it to the SCPTP by redefining the binary variables g_a^k, so that they take the value 1 if and only if q_k units of commodity k pass through arc a. We omit the details for brevity.

2.7.3 The Steiner TSP with Time Windows

Finally, we define the *Steiner Traveling Salesman Problem with Time Windows* (STSPTW) as follows. For each $e \in E$, we are given, not only a non-negative cost c_e, but also a non-negative *traversal time* t_e. Moreover, for each $i \in V_R$, we are given a non-negative *servicing time* s_i, along with a *time window* $[a_i, b_i]$. Finally, we are given a positive time T by which the vehicle must return to the depot. All nodes in V_R must be visited at least once. On one such visit, the customer must receive service. The time at which service begins must lie between a_i and b_i. The task is to minimise the cost of the tour. We assume without loss of generality that the vehicle departs from the depot at time zero. We also assume that the vehicle is permitted to wait at any time.

Perhaps surprisingly, the situation here is completely different from those of the previous two subsections. To be specific:

- Lemma 1 does not apply. To see this, set $V = \{1, \ldots, 4\}$, $V_R = \{2, 3, 4\}$ and $E = \{\{1, 2\}, \{1, 3\}, \{2, 4\}\}$, set $c_e = t_e = 1$ for all $e \in E$, set $s_i = 1$ for $i \in \{2, 3, 4\}$, and set $a_2 = b_2 = 1$, $a_3 = b_3 = 3$, and $a_4 = b_4 = 6$. The unique optimal solution is for the salesman to service nodes $2, 3$ and 4 in that order, and then return to the depot. In this solution, the edge $\{1, 2\}$ is traversed 4 times.

- Theorem 5 does not apply either. In the same example, the total number of edge traversals is 8, whereas $2(|V| - 1)$ is only 6.

- In fact it is not even true that the total number of edge traversals is bounded by $2|E|$, as the same example shows.

- The only thing that one can say in general seems to be that the total number of edge traversals is bounded by $(n_R + 1)(|V| - 1)$. (This is so since the maximum number of edge traversals between two successive occasions of service, or between a service and the vehicle leaving or returning to the depot, will never exceed $|V| - 1$ in an optimal solution.)

For these reasons, it does not seem possible to adapt the classical, SCF or MCF formulations to the STSPTW. It may be possible to adapt the TS formulation, but this would not be desirable, since (a) one would appear to need $(n_R + 1)(|V| - 1)$ time stages and (b) we already saw in the last section that time-staged formulations perform very poorly.

With some work, it is possible to formulate the STSPTW using only $\mathcal{O}(n_R|E|)$ variables and constraints. The key idea is to keep track of the number of required nodes that have been serviced so far each time one visits a node or traverses an edge. For the sake of brevity, we present the details in the appendix.

2.8 Discussion

Since many real-life vehicle routing problems are defined on sparse networks, such as road networks, continued research on the Steiner TSP and its variants is of practical importance. We have seen that, if the Steiner TSP is formulated intelligently, then one can solve instances with over 200 nodes to proven optimality, without invoking sophisticated machinery such as branch-and-cut. (As mentioned in Section 2.6, however, CPLEX did sometimes add cutting planes of its own at the root node.) We have also shown how to adapt our best formulations to some related problems.

Possible topics for future research would be the derivation of smaller and/or stronger compact formulations for the problems mentioned, the derivation of useful compact formulations for the Steiner version of other routing problems, or testing whether the addition of cutting planes could enable even larger instances to be solved using our formulations.

Acknowledgement: The authors thank Juan-José Salazar-González for bringing Theorem 1 to our attention. Thanks are also due to an anonymous referee for helpful comments that led to significant improvements in the paper.

3. CORRELATION BETWEEN SPEEDS ON A CONGESTED ROAD NETWORK IN THE CITY OF LONDON

3.1 abstract

This article uses real traffic data, from the City of London, to explore temporal and spatial correlations between travel speeds in a congested road network. It is shown that, in contrast with results found in other studies on non-congested networks, the first-order Markovian property does not hold for spatial correlations. Indeed, for our data set the spatial correlations are still significant for roads up to twenty links apart. On the other hand, if one analyses the correlations using principal component analysis, it turns out that only three components are needed to explain 89% of the temporal variation in speed, and only six components are needed to explain over 80% of the spatial variation.

Keywords: traffic congestion; spatial correlation; temporal correlation.

3.2 Introduction

Traffic congestion on road networks is inevitable, particularly in larger cities. In the field of Operational Research and Transportation, the two major areas of research that involve modelling of road networks and traffic are traffic forecasting and route optimization. Over the last few decades, a lot of research has been done towards improvement of traffic flow [117, 120, 130] or enhancing service efficiency [20, 21, 54] on road networks. When it comes to modelling the road networks, correlation in speeds or travel times is an important factor that needs to be considered.

In recent years, an increasing number of studies [41, 49, 100] consider the correlation in travel times on road networks. Most of the studies in the relevant current literature either use relatively simplistic models to represent the dependence structure in speeds or travel times, or use traffic flow simulators to generate data. This is mainly due to inaccessibility of real traffic data. However, unrealistic representations of the correlations could affect the legitimacy of any proposed planning on these networks.

This paper studies correlations between speeds on road networks explicitly, based on real road traffic data taken from an urban network with heavy congestion. We compute correlations in speeds across different times of day (temporal correlation) and across different locations (spatial correlation). Perhaps surprisingly, it turns out that there are significant spatial correlations even for pairs of roads that are geographically quite distant. On the other hand, a Principal Components Analysis (PCA) shows that only three components are needed to explain 89% of the temporal variation in speed, and only six components are needed to explain over 80% of the spatial variation. Thus, although both the temporal and spatial correlation matrices are of extremely large dimension, one can capture most of their information in matrices of much smaller dimension. This fact could be useful in road network models.

The rest of this paper is organized as follows. We review the relevant literature in section 3.3. In Section 3.4, we present some descriptive statistics for the data. In Section 3.5, we derive the temporal correlation structure for speeds and carry out a PCA to reduce the dimensionality of the correlation matrix. We also look at the partial temporal correlations to find the conditionally independent elements. In Section 3.6, we derive the spatial correlations and partial spatial correlations in speeds for a range of impact areas $0 - 20$ and carry out a PCA on the correlation matrix. Finally, some concluding remarks appear in Section 3.7.

3.3 Literature review

We now review the relevant literature. Amongst the research studies on *travel time reliability* and *reliable shortest path*, spatial dependences in travel times have been considered in stochastic routing problems by a number of researchers.

Chen *et al.* [22] derive the spatial correlation in link travel time using real-time travel data in an urban area in Hong Kong. The data for that study, are collected at a weekday morning peak hour, for the period of one hour. Travel time estimates are generated by traffic flow simulators such that data are available for every 5 minute interval. The authors introduce a topological distance measure called *impact area*, that is defined to be the number of links between any two links. Chen *et al.*'s analysis on these data reveals that the spatial correlation decreases from 0.29 to 0.04 when the size of impact area increases from 1 to 4. Therefore, the authors conclude that the correlation in travel times is only considerable for roads that are spatially very close. Considering the flow of traffic in this data set (one vehicle movement every 5 minutes), this conclusion seems reasonable. However, this flow might not be a good representation for a typical busy urban area.

Dong *et al.* [41], Fan *et al.* [49] and Samaranayake *et al.* [113] incorporate limited spatial correlation by considering a Markovian property for the link states and using the conditional probability distribution of link travel times to account for the correlation between travel times on neighbouring links. Xing and Zhou [129] construct a sampling-based solution algorithm, and use a set of individual historical measurements to capture the inherent spatial correlation, in their study of path travel time reliability. The result of our analysis in Section 3.6 shows that the first order Markovian property does not hold on real road networks and fourth-order dependencies are statistically significant between travel times.

Min and Wynter [100], in their study of *road traffic prediction*, introduce a transient VARMA model in which both temporal and spatial interactions in speed on a network can be represented by an adjacency matrix. They test their prediction model on a set of traffic data for a medium-sized road network with 502 links. Links are assigned to five different classes, ranging from express-ways to small local roads. Their prediction model of speed for each link at any given time interval captures spatial dependencies of up to the link's 15th nearest neighbours, and temporal dependencies of up to the previous six 5-minute time periods. The resulting predictions for speed have accuracies of up to 93% on major roads and 83% for small local roads, for a forecasting horizon of 15 minutes. It is shown clearly that the present average speed on a link may be influenced by the average speeds on its neighbouring links, in the previous time periods. Therefore, a univariate time series model that ignores spatial dependencies would be inadequate and could not capture all of the dependencies. Asif *et al.* [5] also incorporate spatial temporal patterns in their study of large-scale traffic speed prediction. They use unsupervised learning methods to exploit spatial and temporal trends in their prediction model.

Finally, Cheng *et al.* [23] consider the dynamics of autocorrelation in time and space and examine the spatio-temporal autocorrelation structure of road network in order to determine likely requirements for a suitable space-time forecasting model. They use travel time data collected for 22 individual links in City of London on Tuesdays over a period of 6 months. Considering three time intervals of AM peak ($7:00--10:00$), inter-peak ($10:00--16:00$) and PM peak ($16:00--19:00$) they calculate the global space-time autocorrelations between all links in impact areas of zero to three and up to 150 temporal lags. As reported by Min and Wynter [100], the result of this analysis also indicates that there is significant space-time autocorrelation in the network. Moreover, autocorrelations are dynamic in time and heterogeneous in space which is a reflection of dynamics and heterogeneity of network complexity.

In this paper the focus is on exploring both the temporal and spatial correlation structures on a static road network using real traffic data.

3.4 Data Preparation & Descriptive Statistics

The data was collected in the City of London over a period of two months in Spring 2009, by a traffic information company called ITIS Holdings. The speed measurements were collected by attaching tracking devices to the operating vehicles in service of the company. The operation was carried out on a daily basis Monday-Friday. The vehicles used were all 3.5 tonne GVW box vans, so there were no restrictions on the roads on which they may travel. The accuracy of measurements were expected to be the same across the whole dataset since the same type of device was used for all vehicles.

In this section, we first describe the format of the data sets available for the network map and traffic data in London, and then provide the descriptive statistics for the traffic data.

The road network map data consists of link identifiers (ids) which are categorized into 5 different network levels, where network level 1 represents the motorways with highest speed limit and network level 5 represents city links with lowest speed limit.

Table 3.1 shows the speed limits on these 5 different network levels. Note that the speed limits on network levels 2 and 3 and 4 may sometimes fluctuate between 65, 81 and 97 kph and the values in Table 3.1 correspond to the majority of cases.

The traffic data consists of speed measurements on $51,368$ unique links. In terms of link categories, 1.2% of the links are motorways (network level 1), 54.5% are single carriageways (network levels 2, 3 and 4) and 44.3% are in built-up areas (network level 5).

Average speed measurements on each link are recorded for 15 different time

Network Level	Speed Limit (kph)
1	113
2	97
3	81
4	65
5	49

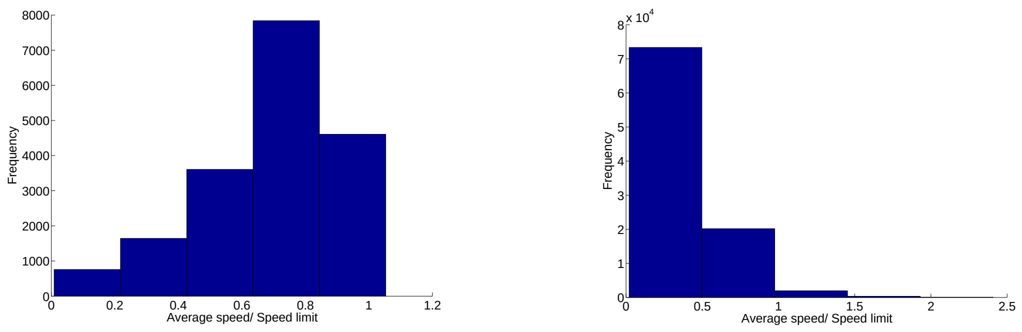
Tab. 3.1: Speed limits on the 5 network levels.

Interval	Time	$\overline{S}(kph)$	$\sigma(kph)$
1	6:00-7:30	34.9	22.1
2	7:30-8:00	32.1	21.6
3	8:00-8:30	30.4	21.3
4	8:30-9:00	29.9	21.4
5	9:00-9:30	31.4	21.7
6	9:30-10:00	32.7	22.0
7	10:00-10:30	33.2	22.3
8	10:30-15:00	30.9	19.6
9	15:00-16:00	31.9	21.5
10	16:00-18:00	31.2	20.2
11	18:00-18:30	31.0	21.7
12	18:30-19:30	31.5	21.4
13	19:30-20:00	32.4	22.6
14	20:00-22:00	32.4	21.7
15	22:00-6:00	32.3	21.9

Tab. 3.2: Average speed and standard deviations in the 15 time intervals.

intervals during the day. Time intervals vary in length and are selected so as to ensure approximate homogeneity in each interval. For example, in the morning rush hour from 7.30 to 8.30, we have the two smallest time intervals of 30 minutes, and at midnight, when there is not much traffic, we have the longest time interval of 8 hours from 10 pm to 6 am.

Table 3.2 shows the average speeds $\overline{S_i}$ and the standard deviations σ_i of speed in time intervals $i \in \{1, 2, \ldots 15\}$, in kilometres per hour. Observe that speeds have low averages of around 30 kph and high standard deviations of around 20 kph. This is typical for an urban area, due to high congestion, traffic lights, roundabouts, and so on.



(a) *(b)*

Fig. 3.1: Histogram of speed over all time intervals, for (a) network level 1 and (b) network level 5.

Figures 3.1(a) and 3.1(b) show the frequency distributions for speed ratios with respect to each level's speed limit, in network levels 1 and 5 respectively. It is interesting to see that the distribution for network level 1 is heavily skewed to the left while the distribution for network level 5 is heavily skewed to the right. In fact, congestion on network level 5 is so heavy that more than 70% of vehicles drive below half the speed limit.

Figures 3.2(a) and 3.2(b) show the frequency distributions of speed ratios in network level 5, for the quiet time interval 1 and the most congested time interval 4, respectively. Both distributions have positive skewness, which is expected due to heavy congestion on this network level, with the distribution in Figure 3.2(a) having a slightly longer tail. Note that network level 5 represents small local roads where typically there are a lot of traffic lights present. Hence, we still see quite low speed ratios in Figure 3.2(a) for quiet time periods.

The low means and large variances in Table 3.2 and the low medians for the frequency distributions in both Figures 3.1 and 3.2 suggest that the London traffic data are likely to be reasonably typical for a congested urban area.

Please note that the descriptive statistics provided in this section are representative for parts of the City of London in the season for which the data has been collected. Considering the impact of seasonality in traffic on road networks,

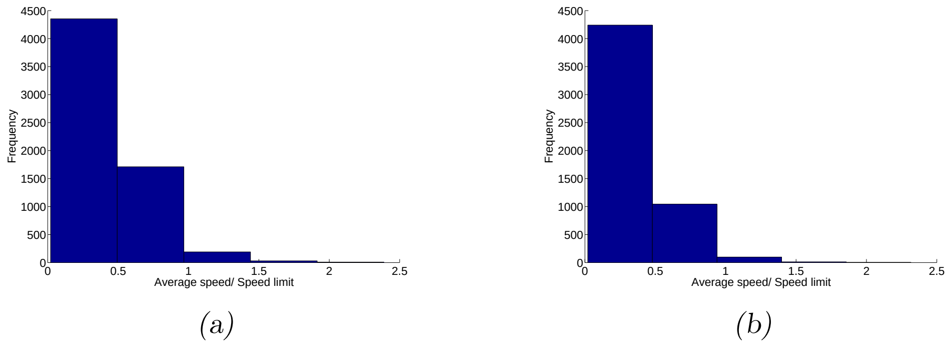


Fig. 3.2: Histogram of speed for network level 5, for (a) quiet time interval 1 and (b) busy time interval 4.

different patterns might emerge for other times of year. However, the results of our correlation analysis in the following sections are valid for any network that has the same speed characteristics as described in this section.

The speed measurements in our traffic data have been recorded for all vehicles from ITIS Holdings that had a detecting device attached to them. Note that if no observable vehicle travelled along a link at a certain time interval, the corresponding speed measurements will be missing for that link. Since our objective is to evaluate the correlations in speeds, using real traffic data, we do not substitute simulated values for the missing intervals. However, it is important to ensure, as well as possible, that the missing data will not have an undue influence on the correlation analyses in Sections 3.5 and 3.6.

The proportions of missing data on each network level are quite similar and vary in the range of $8-10\%$. The proportion of data missing for the 15 time intervals is between 30% and 70%, with time intervals $3, 6$ and 7 having the minimum missing data and time intervals $9, 10$ and 15 having the maximum missing data. After an inspection of the proportions of missing data over different network levels and time intervals, we conjecture that the missing data are unlikely to affect the correlation analysis. This conjecture will be supported in Section 3.5, when we compare the temporal pairwise correlations calculated using all of the available data in each pair of time intervals with the temporal correlations calculated using the complete data set.

In order to prepare the data for the correlation analysis, firstly all the links with complete traffic data were identified. Then the observed average speeds on these links were divided by the speed limits on each link, so that roads in different network levels have the same scale for comparison. In the following two sections, we explore the temporal and spatial correlations in speeds using this complete traffic data set.

3.5 Temporal Correlation

In this section, we look at correlations over time. We begin by defining the sample *temporal covariance* u_{ij} between the speeds in two time intervals i and j as:

$$u_{ij} = \frac{1}{N}\sum_{r=1}^{N} s_{ri}s_{rj} - \bar{s_i}\bar{s_j} \qquad i, j \in T, \tag{3.1}$$

where s_{ri} is the average speed on link r at time interval i, $\bar{s_i}$ is the average speed over all links at time interval i, $T = \{1, 2, \ldots, 15\}$ is the set containing all time intervals, and $N = 10,255$ is the total number of links for which data are available on all T time intervals. The sample *temporal correlation coefficient* λ_{ij} between the speeds in two time intervals i and j is then defined as:

$$\lambda_{ij} = \frac{u_{ij}}{\sqrt{u_{ii}u_{jj}}}. \tag{3.2}$$

Once the temporal correlations were computed, we tested the significance of each correlation at a 95% significance level. All of the correlation values reported in the rest of the paper were significant at this level.

Figure 3.3 shows a colour-coded representation of the temporal correlation matrix. As expected, the correlation decays as the time intervals become farther apart. The pairwise temporal correlations between the speeds in time intervals i and j were also evaluated for all $i, j \in T$, using all the links with available data for time intervals i and j. Further significance tests then showed that there is

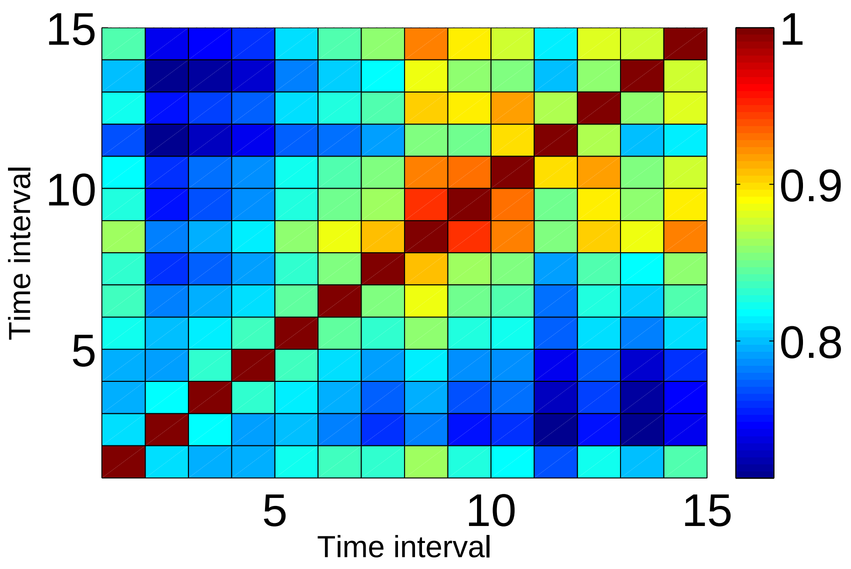


Fig. 3.3: Colour coded presentation of the temporal correlation for time intervals $1-15$.

no significant difference between the original correlations and these modified ones. The largest difference was only 0.07, which was between the original and modified correlations between time intervals 14 and 15. This result supports our conjecture, made in Section 3.4, that the missing data are non-informative. For brevity, we do not present the modified correlations here.

In order to find a more efficient representation of the temporal correlation matrix, we carried out a PCA. As is well known, any correlation matrix $C_{p \times p}$ has p linearly independent eigenvectors $q_i, i = (1, \ldots, p)$ corresponding to p (not necessarily distinct) non-negative eigenvalues. One can express C, in terms of its eigendecomposition components $A_{p \times p}$, $Q_{p \times p}$ and $Q^{-1}_{p \times p}$, in the following way:

$$C = QAQ^{-1}, \tag{3.3}$$

where component A is a diagonal matrix containing all the eigenvalues of C, in descending order, component Q is the square matrix whose i^{th} column is the eigenvector q_i of C and component Q^{-1} is the inverse matrix of Q. Now, in order to reduce the dimensionality of Q, A and Q^{-1}, we can set the c smallest eigenvalues in A to zero and eliminate the corresponding c columns in Q and c

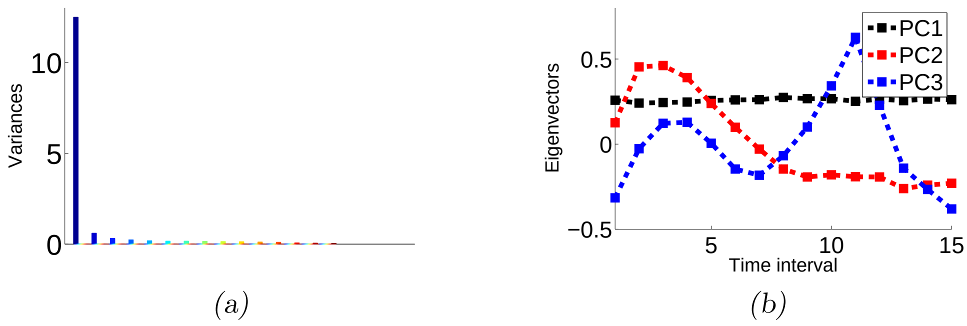


Fig. 3.4: (a) Eigenvalues of temporal correlation Λ, (b) First three principal components of Λ that overall can explain 89% of total variation.

rows in Q^{-1}. This will reduce the dimensions of Q, A and Q^{-1} to $(p \times (p-c))$, $((p-c) \times (p-c))$ and $((p-c) \times p)$, respectively. For more details we refer the reader to [93].

The rule of thumb is to retain enough eigenvalues to make up around $80-90\%$ of variation in A. If most of the variation is due to the first few eigenvectors (principal components), this technique is expected to reduce the dimensionality of the eigendecomposition components significantly.

Now, for our temporal correlation matrix Λ, an eigendecomposition shows that the three highest variances are 12.51, 0.60 and 0.31, as shown in Figure 3.4(a). The corresponding principal components are in the three directions shown in Figure 3.4(b). The first principal component gives a positive weight to all variables and thus represents an 'average' speed, the second component represents a contrast between speeds in the busier time intervals of 3 and 4 and other times of the day, and the third component indicates a contrast in speeds between the uncongested time intervals of 3, 4, 11 and other times of day.

Figure 3.5 shows the relative contribution of various principal components to the variation, where the first three principal components explain 83%, 4% and 2% of the total variation in the data, respectively. The same correlation analysis was carried out on each network level separately. The result for every network level is quite similar where the first three principal components explain 91% of the

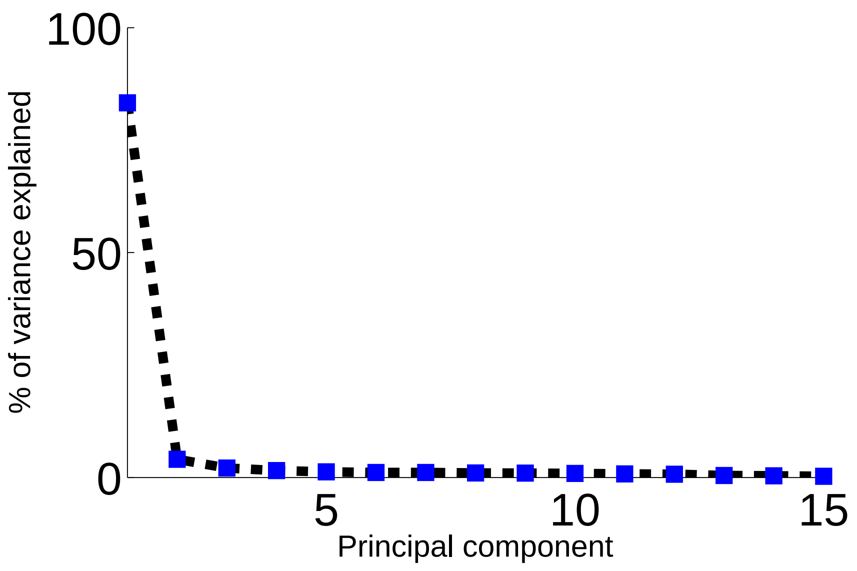


Fig. 3.5: Percentage of variance explained by each eigenvalue of Λ.

variation for network level 1 and 93% of the variation for network level 5.

An approximate correlation matrix can be constructed by using only the three largest eigenvalues of Λ and their corresponding principal components, using (3.3). Figure 3.6 shows a colour coded representation of the difference between this approximate correlation matrix and the actual temporal correlation matrix Λ. Note that storing more dimensions for the principal components and the corresponding eigenvalues results in even smaller differences and more accurate representations of the correlation matrix.

Finally, we carry out a partial correlation analysis on the sample temporal correlation matrix. Partial correlation measures the degree of dependence between two variables, after removing the effect of other variables. This analysis helps in detecting any conditional dependencies or any correlations that can be explained by the effect of other variables. There are different methods for computing partial correlations. The interested reader is referred to [127] for further details.

We calculate the sample temporal partial correlations by inverting the sample temporal correlation matrix Λ. This analysis will reveal if the speeds between any of the two time intervals may become conditionally uncorrelated. Element ij in

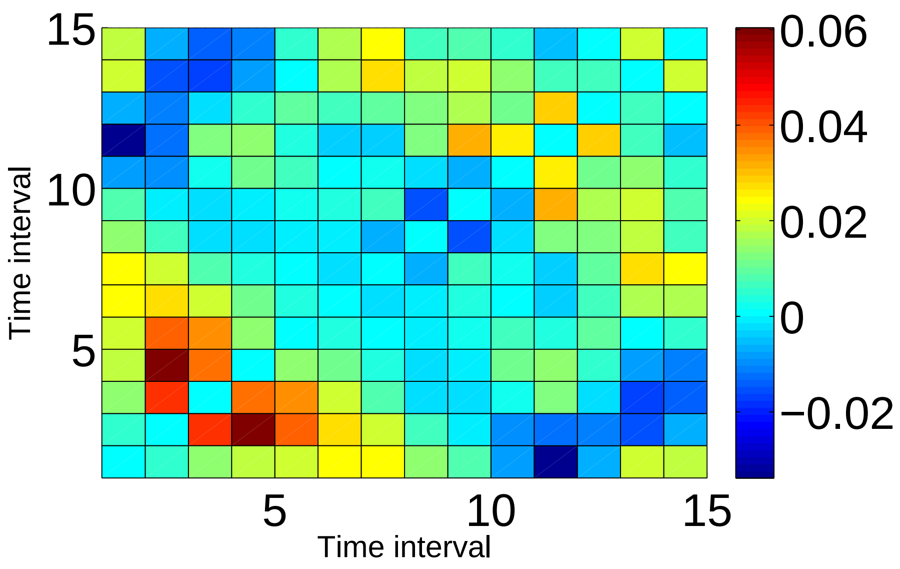


Fig. 3.6: Colour coded presentation of the difference between the simplified temporal correlation, obtained from 3 principal components,and the true temporal correlation matrix.

the temporal partial correlation matrix is calculated as

$$\lambda_{ij.T\backslash\{ij\}} = \frac{[\Lambda^{-1}]_{ij}}{\sqrt{[\Lambda^{-1}]_{ii}[\Lambda^{-1}]_{jj}}}, \qquad (3.4)$$

where $\lambda_{ij.T\backslash\{ij\}}$ gives the value of temporal correlation in speeds between time intervals i and j, with the effect of speeds in all other time intervals, removed. Matrix Λ^{-1} is the inverse of Λ and $[\Lambda^{-1}]_{ij}$ is element ij in Λ^{-1}.

Figure 3.7 shows a colour-coded representation of the temporal partial correlation matrix. The corresponding values for every pair of adjacent time intervals are statistically significant, while the partial correlations between pairs of time intervals that are more than two intervals apart are insignificant. The significant temporal partial correlations between adjacent time intervals are in range of 0.14 to 0.41. Lower partial correlations of around 0.1 are between time interval pairs of $(5, 6)$, $(6, 7)$ and $(12, 13)$ and higher partial correlations of around 0.4 correspond

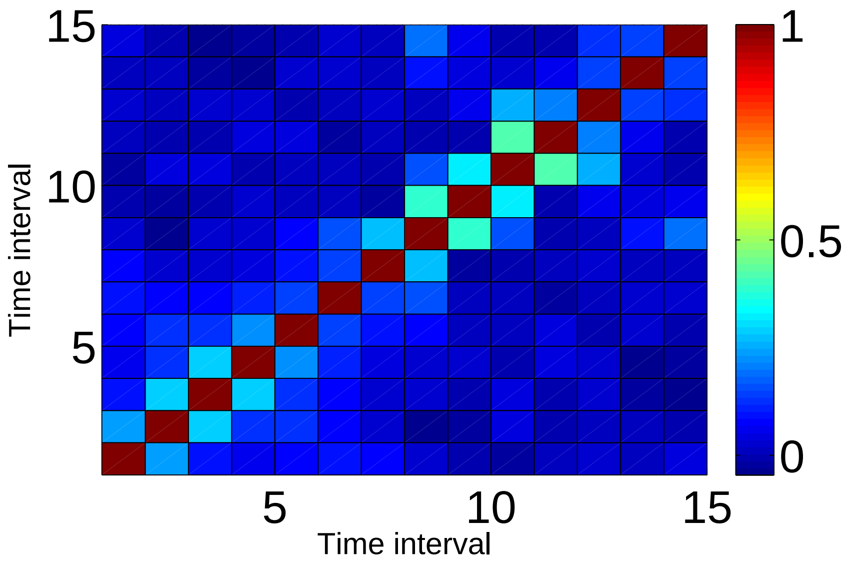


Fig. 3.7: Colour coded presentation of the temporal partial correlation for time intervals $1 - 15$.

to time interval pairs of $(10, 11)$ and $(8, 9)$. Also, the correlations between time interval pairs $(1, 3), (1, 14), (5, 7), (9, 7), (9, 11), (11, 13)$, that are two intervals apart, are not significant. As expected, generally, the corresponding partial correlations are larger between adjacent time intervals with more congestion.

3.6 Spatial Correlation

A topological measure for adjacency of links on a road network was introduced by Chen *et al.* [22], called the *impact area*. This measure represents the minimum number of links between any two links. Following this terminology, in this section, we study the spatial correlations for impact areas $0 - 20$.

In the traffic data for the City of London, there are 737 unique links that have data available for up to their $20th$ connected neighbouring links (i.e. impact area 20). We call this set of unique links the *basis set*, and build a new data set with 21 columns, such that the first column contains the average speeds for the basis set, over all time intervals. Then the next columns are filled such that column $i \in \{2, 3, \ldots, 21\}$ contains the average speeds for links that are in impact area

$i-1$ with respect to the links in the corresponding row of the basis set.

The sample *spatial covariance* v_{ij} between the speeds in columns i and j is calculated as

$$v_{ij} = \frac{1}{B}\sum_{r=1}^{B} s_{ri}s_{rj} - \bar{s}_i\bar{s}_j \qquad i,j \in \{1,2,\ldots,m\}, \tag{3.5}$$

where s_{ri} is the average speed on link r in column i, $\bar{s}_i$ is the average speed over all links in column i, $m-1$ is the maximum impact area under consideration and $B = 737$ is the total number of links in the basis set. Hence, element v_{1j} in the matrix represents the covariance in speeds between links that are in impact area $j-1$ with respect to the basis set. The sample *spatial correlation coefficient* ρ_{ij} between the speeds in two columns i and j is then calculated as

$$\rho_{ij} = \frac{v_{ij}}{\sqrt{v_{ii}v_{jj}}}. \tag{3.6}$$

Figure 3.8 shows a colour-coded representation of the spatial correlation matrix P, for impact areas $0-20$, with the diagonal elements representing self-correlations (i.e. impact area 0). The correlations decay slowly from 0.72 to 0.3 as the size of the impact area increases from 1 to 20.

This result is in contrast with Chen *et al.*'s findings in [22], where they conclude that spatial correlations are only significant within impact area 4. Chen *et al.*'s analysis is based on a traffic data set in Hong Kong with flow of one vehicle movement in every five minute. This traffic flow represents a remote road network area without much congestion. However, as we have shown here, the same result does not apply for a typical congested urban road network.

An eigendecomposition of the spatial correlation matrix P shows that the first six principal components explain 82.2% of total variation. The six highest variances are 10.8, 2.2, 1.5, 1.2, 0.9 and 0.7, as shown in Figure 3.9(a). The corresponding principal components are in the six directions shown in Figure 3.9(b). The first principal component gives a positive weight to all variables and thus

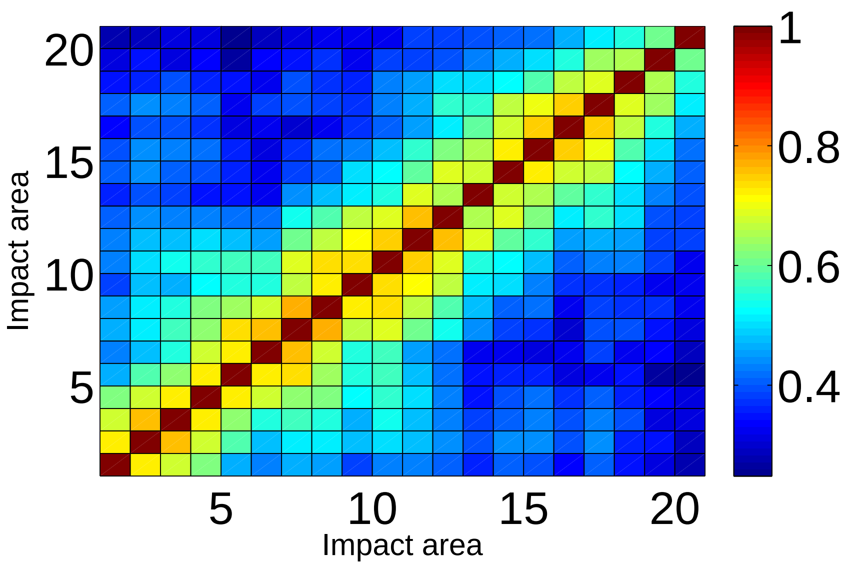


Fig. 3.8: Colour coded presentation of the spatial correlation for impact areas $0 - 20$.

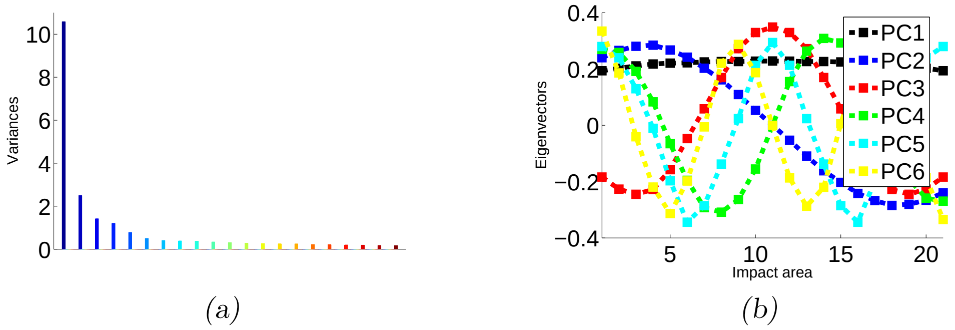


Fig. 3.9: (a) Eigenvalues of spatial correlation P, (b) First six principal components of P that overall can explain 82% of total variation.

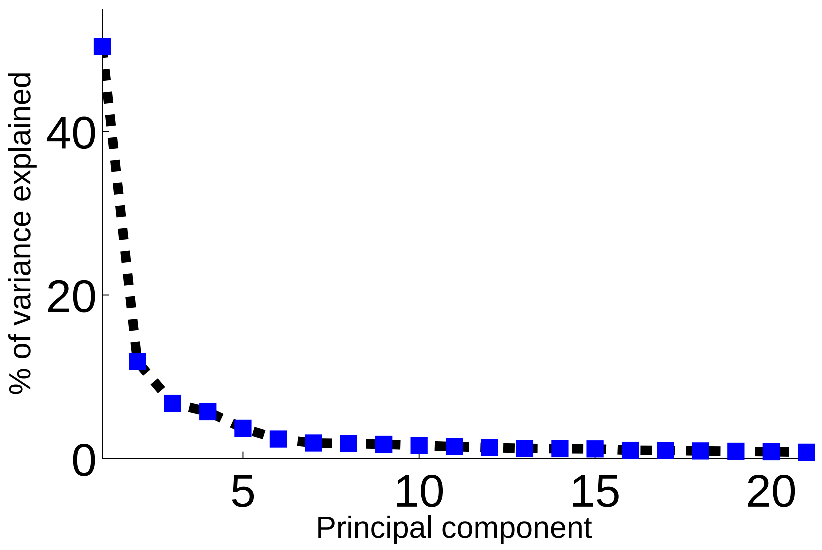


Fig. 3.10: Percentage of variance explained by each eigen value of P.

represents an 'average' speed, while the second component gives a positive weight for impact areas $1 - 10$ and negative weight for farther links with impact areas $11 - 21$. The next four components show an alternating pattern. For example component three gives more weight for impact areas $2 - 4$, $9 - 13$ and $18 - 20$, where component four gives more weight for impact areas $1-3$, $6-9$ and $13-16$.

Figure 3.10 shows the relative contribution of various principal components in variation where the first six principal components explain 51.2%, 10.6%, 7.1%, 5.8%, 4.2% and 3.3% of total variation in the data respectively. Now, using these six principal components and their corresponding eigenvalues, we an construct an approximate representation of the spatial correlation matrix P, using (3.3).

Figure 3.11 shows a colour-coded representation of the difference between this approximate spatial correlation matrix and the actual spatial correlation matrix P. As can be seen, the differences between corresponding values are very small. However, depending on the requirements, one can choose a more accurate presentation by choosing eigendecomposition components of higher dimensionality.

Finally, we calculate the sample spatial partial correlations by inverting the sample spatial correlation matrix P, following the same procedure as explained in

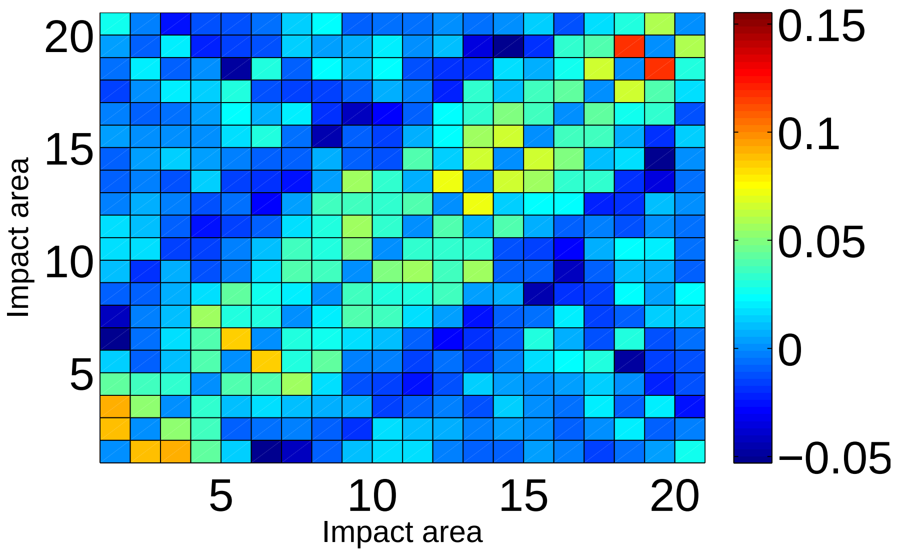


Fig. 3.11: Colour coded presentation of the difference between the simplified spatial correlation, obtained from 6 principal components and the true spatial correlation matrix.

Section 3.5. Figure 3.12 shows a colour-coded representation of the spatial partial correlation matrix.

The spatial partial correlations are significant at 95% level, only for links within impact area 4. The corresponding values for impact areas $1-4$ are 0.39, 0.21, 0.15 and -0.10. Note that the sample spatial partial correlations derived for City of London are nearly the same as the values that Chen *et al.* [22] report for the actual spatial correlations. On the other hand, the actual spatial correlation values for our data are significant at long range, which was not the case in [22]. As explained before, this is because the traffic flow in the data that Chen *et al.* use was low and not at all comparable to the traffic flow for the congested road network of London.

3.7 Discussion

We have studied the temporal and spatial correlations in speeds on road networks, using real traffic data for the congested city of London. It was shown that, despite what is normally assumed in the literature (e.g. [22, 41, 49]), speeds in a congested urban area can have significant spatial correlations even between roads that are

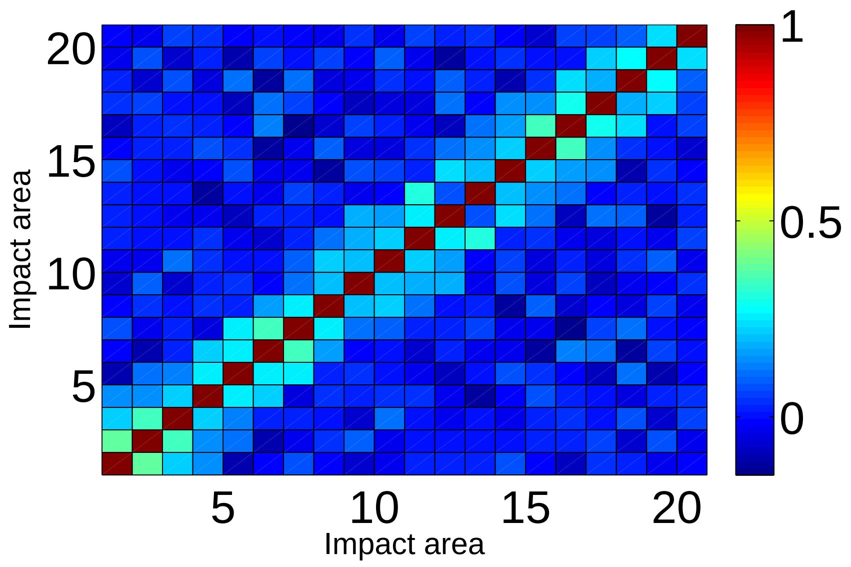


Fig. 3.12: Colour coded presentation of the spatial partial correlation for impact areas $0-20$.

up to 20 links apart.

Both temporal and spatial correlations in speeds were calculated and analysed in detail. A principal component analysis of the temporal correlation revealed that up to 89% of the variation can be explained in 3 dimensions, whereas for spatial correlation 82% of the variation can be explained in 6 dimensions. This could be useful for data compression. It also shows that, although speeds on congested road networks have long-range correlations, the correlation structure itself is not very complex.

Evaluation of partial correlations shows that in most cases temporal partial correlations are significant only for adjacent time intervals, whereas spatial partial correlations become insignificant for impact areas greater than 4. This indicates that the significance of correlations over a long range is due to the conditional dependencies between speeds in neighbouring roads.

Acknowledgement: The author would like to thank Prof. Adam Letchford, Prof. Jon Tawn and Dr. Alireza Bahraminasab for providing invaluable suggestions and guidelines. Also thanks to Prof. Richard Eglese for providing the data for this research.

4. THE STEINER TRAVELLING SALESMAN PROBLEM WITH CORRELATED COSTS

4.1 abstract

The Steiner Travelling Salesman Problem (STSP) is a variant of the Travelling Salesman Problem that is suitable for instances defined on road networks. We consider an extension of the STSP in which the costs of traversing roads are both stochastic and correlated. This model is suitable, for example, when vehicles are prone to unexpected delays due to road works or accidents. Following the work of Markowitz on portfolio selection, we model the problem as a bi-objective mean-variance problem. Then, we show how to find efficient solutions via integer programming. We also show how to exploit certain special structures in the correlation matrices. Computational results are presented for instances with up to 100 nodes.

Keywords: travelling salesman problem, mean-variance optimisation, mixed-integer nonlinear programming.

4.2 Introduction

In the *Travelling Salesman Problem* (TSP), one is given a complete graph, and a cost for each edge of the graph, and one seeks a Hamiltonian circuit of minimum total cost. The TSP is a classical and much-studied problem in combinatorial optimisation, and several books have been written on it (e.g., [3, 67, 85]). Somewhat less well known is the *Steiner* TSP (STSP), which is a variant of the TSP that is suitable for instances defined on road networks (see, e.g., [28, 53, 88, 106]). In the

STSP, one is given a sparse graph, representing a road network, a cost for each edge of the graph, and a subset of nodes that represent customers. The task is to find a minimum-cost tour that passes through each customer node. Edges may be traversed more than once, and nodes visited more than once, if desired.

In real-life STSP applications, the edge-costs are likely to be stochastic rather than deterministic. (They are also likely to be time-varying rather than static, but we ignore that possibility in this paper.) For example, if the costs represent fuel consumption or travel times, then events such as road works or accidents can cause them to be larger than expected. In addition, the random costs are likely to be correlated, rather than statistically independent. Indeed, if an accident takes place, it is likely to disrupt travel on several roads simultaneously.

These considerations lead us to consider an extension of the STSP, which we call the *Steiner TSP with Correlated Costs* (STSPCC). This is like the STSP, except that the costs follow a multi-variate distribution whose first and second moments are known. Following the classical Markowitz mean-variance approach to portfolio selection [94], one can view the STSPCC as a *bi-objective* optimisation problem, in which there is a trade-off between minimising the expected cost of a tour and minimising the variance of the cost. We consider however two single-objective versions of the problem, obtained via the well-known *weighting* and *ϵ-constraint* methods for multi-objective problems [97].

Our approach to these problems is based on integer programming. We begin by presenting four mixed 0-1 convex quadratic programming formulations of polynomial size, based on results in our earlier paper [88]. After that, we present four strong mixed 0-1 linear programming formulations of polynomial size. Next, we show how the various formulations can be improved if the cost correlations have either of two special structures that may arise in practice. The first is when the correlation matrix has low rank, and the second is when the correlations are significant only for pairs of roads that are close to each other in the network. Finally, we present extensive computational results, which show the relative performance of the formulations under certain conditions.

The paper is structured as follows. The relevant literature is reviewed in Section 4.3. In Section 4.4, we present the formulations for the case in which the correlation matrix has no special structure. In Section 4.5, we show how to deal with the two special correlation structures mentioned. The computational results are presented in Section 4.6, and some concluding remarks appear in Section 4.7.

We use the following notation in the remainder of the paper. The road network is denoted by $G = (V, E)$, where V is the vertex set and E is the edge set. The cost of traversing the edge (i.e., road) $e \in E$ is denoted by c_e. The set of customer nodes, which is a subset of V, is denoted by C. It is assumed that G is undirected, i.e., that all edges $e \in E$ represent two-way streets. We do however make some remarks about the possibility of adapting our approach to the 'asymmetric' case, in which some or all of the roads are one-way (see Subsections 4.3.1 and 4.4.2).

4.3 Literature Review

We now review the relevant literature. We cover formulations of the STSP in Subsection 4.3.1, mean-variance and multi-objective optimisation in Subsection 4.3.2, and vehicle routing with correlated costs in Subsection 4.3.3.

4.3.1 Formulations of the STSP

In [28, 53], it is shown how to formulate the STSP as an integer program with $|E|$ variables and an exponential number of constraints. In our own paper [88], we presented some alternative formulations of polynomial size. Of those, the two which performed best computationally were the *single-commodity flow* (SCF) and *multi-commodity flow* (MCF) formulations, obtained by adapting formulations of the standard TSP due to Gavish & Graves [60] and Claus [24], respectively.

Before presenting the SCF and MCF formulations in detail, we need some additional notation. Let $\tilde{G} = (V, A)$ be a directed graph, where the set of directed arcs A is obtained from the edge set E by replacing each edge $\{i, j\}$ with two directed arcs (i, j) and (j, i). For each arc $a \in A$, the cost c_a is set to the cost of the corresponding edge. For any $S \subset V$, let $\delta^+(S)$ denote the set of arcs in A whose tail is in S and whose head is in $V \setminus S$, and let $\delta^-(S)$ denote the set of arcs in A for which the reverse holds. We also assume, without loss of generality, that vertex 1 is a customer.

In the SCF formulation, we imagine that the salesman departs from vertex 1 with $|C| - 1$ units of a commodity, and delivers one unit of that commodity to each vertex in $C \setminus \{1\}$. There are two variables for each arc $a \in A$. The binary variable x_a takes the value 1 if and only if arc a is traversed, and the continuous variable f_a represents the amount of the commodity passing through arc a. The formulation is then:

$$
\begin{array}{llll}
\min & \sum_{a \in A} c_a x_a & & (4.1) \\
\text{s.t.} & \sum_{a \in \delta^+(i)} x_a \geq 1 & (\forall i \in C) & (4.2) \\
& \sum_{a \in \delta^+(i)} x_a = \sum_{a \in \delta^-(i)} x_a & (\forall i \in V \setminus \{1\}) & (4.3) \\
& \sum_{a \in \delta^-(i)} f_a - \sum_{a \in \delta^+(i)} f_a = 1 & (\forall i \in C \setminus \{1\}) & (4.4) \\
& \sum_{a \in \delta^-(i)} f_a - \sum_{a \in \delta^+(i)} f_a = 0 & (\forall i \in V \setminus C) & (4.5) \\
& 0 \leq f_a \leq (|C| - 1) x_a & (\forall a \in A) & (4.6) \\
& x_a \in \{0, 1\} & (\forall a \in A). & (4.7)
\end{array}
$$

Note that the SCF formulation is a mixed 0-1 linear program (mixed 0-1 LP) with $\mathcal{O}(|E|)$ variables and constraints.

In the MCF formulation, we imagine that the salesman departs from vertex 1 with a distinct commodity for each vertex in $C \setminus \{1\}$. Accordingly, for all $a \in A$ and all $k \in C \setminus \{1\}$, we define the continuous variable f_a^k, representing the proportion of commodity k passing through arc a. (One could declare f_a^k to be binary, but there is no need to.) We then minimise (4.1) subject to (4.3), (4.7) and the following constraints:

$$\sum_{a \in \delta^-(i)} f_a^k - \sum_{a \in \delta^+(i)} f_a^k = 0 \quad (\forall i \in V \setminus \{1\}; k \in C \setminus \{1, i\}) \tag{4.8}$$

$$\sum_{a \in \delta^-(k)} f_a^k - \sum_{a \in \delta^+(k)} f_a^k = 1 \quad (\forall k \in C \setminus \{1\}) \tag{4.9}$$

$$0 \leq f_a^k \leq x_a \quad (\forall a \in A,\ k \in C \setminus \{1\}). \tag{4.10}$$

Note that the MCF formulation is a mixed 0-1 LP with $\mathcal{O}(|C||E|)$ variables and constraints.

We remark that the x variables are binary in both formulations, because there always exists an optimal STSP solution such that no arc is traversed more than once. This is not true in the case of asymmetric costs, so if one wished to adapt the formulations to the asymmetric STSP, one would have to change the x variables from binary to general-integer.

4.3.2 Mean-variance and multi-objective optimisation

Mean-variance optimisation was first presented by Markowitz [94], in the context of portfolio selection. In his model, there are n stocks, and we are given a vector $r \in \mathbb{R}^n$ of expected returns and a covariance matrix $Q \in \mathbb{R}^{n \times n}$. For $i = 1, \ldots, n$, the continuous variable x_i represents the proportion of money invested in stock i. Then, for a given solution x, the mean (expected) return is $r^T x$, and the variance of return is $x^T Q x$.

It is desirable to have a high mean, but a low variance. Hence, the mean-variance problem is a *bi-objective* problem. Markowitz calls a solution x *efficient* if all solutions with higher mean than x have higher variance, and all solutions with smaller variance have smaller mean. He then defines the *efficient frontier* as the curve in $\mathbb{R}^2_+$ described by all mean-variance pairs that are achieved by at least one efficient solution.

Computing the efficient frontier is rather demanding. On the other hand, computing a single efficient solution is relatively easy. One simple way to do it is to solve the convex quadratic program (CQP)

$$\max\left\{\alpha(r^T x) - (1-\alpha)(x^T Q x) : \sum_{i=1}^{n} x_i = 1,\ x \geq 0\right\}$$

for some $0 < \alpha < 1$. This CQP can be solved, e.g., with simplex methods [13, 128]. Another way is to solve the CQP

$$\min\left\{x^T Q x : \sum_{i=1}^{n} x_i = 1,\ r^T x \geq \epsilon,\ x \geq 0\right\}$$

for some $\epsilon > 0$. These two approaches can be viewed as applications of the well-known *weighting* and *ϵ-constraint* methods for multi-objective optimisation [97]. These methods also have the nice property that when applied to portfolio selection problems, they lead to convex quadratic programmes which are amenable to being solved by our available software package (CPLEX). A more complicated method for multi-objective optimisation might lead to sub-problems that were not of that type, and therefore not solvable using CPLEX (or indeed any existing software package).

In this paper, we wish to apply the mean-variance approach to a combinatorial optimisation problem. Although there does not seem to be a literature on this specific topic, there is a considerable literature on multi-objective combinatorial optimisation problems with *linear* objectives (e.g., [46, 96, 119, 125]). Even for that apparently simpler case, computing the entire efficient set is extremely challenging. Moreover, a subtle complication arises that is not present in the case of continuous multi-objective problems: it can happen that there exist efficient solutions that cannot be obtained using the weighting method. This is because an efficient solution could be dominated by a *convex combination* of other efficient solutions, but not by any single efficient solution (see, e.g., [46, 125]).

4.3.3 Vehicle routing with correlated costs

For an overview of stochastic vehicle routing, we refer the reader to Gendreau *et al.* [61]. Here, we concentrate on the case in which the travel times and/or costs are stochastic.

Three papers are concerned with TSPs with stochastic travel times. Kao [78] proposed a dynamic programming heuristic for the variant in which the objective is to maximise the probability of finishing the route by a deadline. Jula *et al.* [76] applied dynamic programming to a different variant, in which the goal is to minimise expected cost, subject to a lower bound on the service level. Teng *et al.* [121] used 2-stage stochastic integer programming to solve a third variant, in which a recourse action is available.

Three other papers are concerned with problems with more than one vehicle. Laporte *et al.* [83] presented a branch-and-cut algorithm for a variant in which vehicles incur a penalty for tardiness. Kenyon and Morton [79] described another branch-and-cut algorithm for a second variant, in which the objective is to minimise the expected total completion time of service. Li *et al.* [90] considered an extension with time windows, and used tabu search to solve both chance-constrained and recourse variants heuristically.

Another variant is the *dynamic* TSP with stochastic costs / times, in which the vehicle can be re-routed in real-time, as information on congestion becomes available. Secomandi [115] and Toriello *et al.* [123] propose a "rollout" heuristic and an approximate dynamic programming heuristic, respectively, for this variant.

A different though related stream of literature is concerned with modelling spatial and/or temporal correlations between travel times on real-life road networks. Fan *et al.* [49], Samaranayake *et al.* [113] and Dong *et al.* [41] argue that, in rural areas, spatial correlations satisfies a certain Markovian property, so that only the conditional probability distributions of travel times on adjacent pairs of links need to be modelled. Chen *et al.* [22] examine real travel time data from an urban area in Hong Kong with low traffic flow, and find that spatial correlations decrease from 0.29 to 0.04 as one moves from adjacent links to links that are up to 4 links apart. They conclude that it is enough to model correlations only for pairs of roads that are spatially very close. On the other hand, Nasiri [104] analysed both temporal and spatial correlations in real travel time data from a highly congested road network in the City of London, and found that spatial correlations between roads that are up to 20 links apart are still significant. The ways in which we handle these different correlation structures are explained in the following three sections.

4.4 Formulations for the General Case

In this section and the next, we present several different integer programming formulations of the STSPCC. This section is concerned with the general case, in which the cost correlations have no special structure. In the next section, two special correlation structures are considered.

From now on, we use the following notation. For all $e \in E$, we let $\bar{c}_e$ denote the expected cost of edge e. For all pairs $e, f \in E$, we let Q_{ef} denote the covariances between the costs on edges e and f. When considering $\tilde{G}$, the directed graph derived from G (see Subsection 4.3.1), we let $\bar{c}_a$ denote the expected cost of the edge in E that corresponds to the arc a, and let Q_{ab} denote the covariance between the costs of the edges in E that corresponds to the arcs $a, b \in A$.

4.4.1 Mixed 0-1 CQP formulations

In Subsection 4.3.1, we presented two known formulations for the STSP: the SCF formulation, (4.1)-(4.7), and the MCF formulation (4.1), (4.3), (4.7), (4.8)-(4.10). In order to obtain efficient solutions of the STSPCC, it suffices to modify those formulations. To do this, we apply the weighting and ϵ-constraint methods mentioned in Subsection 4.3.2.

Observe that, in the case of the STSPCC, it is desirable to have a low mean and a low variance. Then, to apply the weighting method, one should take either the SCF or MCF formulation, and change the objective function (4.1) to

$$\sum_{a\in A} \bar{c}_a x_a + w \sum_{a\in A} \sum_{b\in A} Q_{ab} x_a x_b,$$

for some positive weight w. To apply the ϵ-constraint method, one should change the objective function to

$$\sum_{a\in A} \sum_{b\in A} Q_{ab} x_a x_b,$$

and add the constraint

$$\sum_{a\in A} \bar{c}_a x_a \le \epsilon. \tag{4.11}$$

This leads to four mixed 0-1 CQP formulations in total, which we call SCF-w, SCF-ϵ, MCF-w and MCF-ϵ.

4.4.2 Mixed 0-1 LP formulations

It is possible nowadays to solve mixed 0-1 CQPs directly, with commercial software packages. Nevertheless, software for mixed 0-1 LPs is much more sophisticated. For this reason, it is desirable also to derive and test mixed 0-1 LP formulations. Fortunately, using a standard trick [56], one can linearise any of the four formulations mentioned in the previous subsection, as follows:

- For all pairs $a, b \in A$, create a new continuous variable, say y_{ab}, representing the product $x_a x_b$. (One could declare y_{ab} to be binary, but there is no need to.)
- Replace each quadratic term of the form $Q_{ab} x_a x_b$ with the linear term $Q_{ab} y_{ab}$.
- For all pairs $a, b \in A$, add the constraints $y_{ab} \ge 0$, $y_{ab} \le x_a$, $y_{ab} \le x_b$ and $y_{ab} \ge x_a + x_b - 1$ to the formulation.

The resulting four mixed 0-1 LP formulations are valid, but tend to have a weak continuous relaxation. In order to strengthen them, we propose to add the following valid linear equations:

$$\sum_{b \in \delta^+(i)} y_{ab} = \sum_{b \in \delta^-(i)} y_{ab} \qquad (i \in V \setminus \{1\},\ a \in A). \tag{4.12}$$

These can be derived by multiplying the equations (4.3) by individual x variables, and then linearising them by replacing products of x variables with the corresponding y variables.

We also propose to further strengthen the two mixed 0-1 LP formulations arising from the ϵ-constraint method, by adding the following valid inequalities:

$$\begin{array}{rl} \sum_{b \in A} \bar{c}_b y_{ab} \leq \epsilon x_a & (a \in A) \\ \sum_{b \in A} \bar{c}_b (x_b - y_{ab}) \leq \epsilon (1 - x_a) & (a \in A). \end{array}$$

These inequalities are obtained by multiplying the inequality (4.11) by each x variable and its complement in turn, and then linearising as above.

The idea of multiplying constraints by binary variables to derive new constraints was first proposed in [1], and forms the basis of the well-known *Reformulation-Linearization Technique* (RLT) for global optimisation problems [116]. Accordingly, we call our four mixed 0-1 LP formulations RLT-SCF-w, RLT-SCF-ϵ, RLT-MCF-w and RLT-MCF-ϵ. We remark, however, that a full application of the RLT to the mixed 0-1 CQP formulations would lead to mixed 0-1 LP formulations with even more variables and constraints. Specifically, there would be additional variables representing products of x and f variables, and additional constraints obtained by multiplying the flow conservation constraints by x variables. We do not bother to do that here, since none of the formulations presented in the previous subsection have quadratic terms involving f variables.

Observe that each of the four formulations presented in this subsection has $\mathcal{O}(|E|)$ binary variables, $\mathcal{O}(|E|^2)$ continuous variables and $\mathcal{O}(|E|^2)$ constraints.

Provided the graph G is sparse, as is usually the case in practice, this is likely to be manageable.

We remark that the version of the RLT used in this subsection cannot be applied to the asymmetric version of the STSPCC, since it relies on the assumption that the x variables are binary.

4.5 Exploiting Special Correlation Structures

An instance of the STSPCC is given by a graph $G = (V, E)$, an expected cost vector $c \in \mathbb{R}^{|E|}$ and a covariance matrix $Q \in \mathbb{R}^{|E|\times|E|}$. Up to now, we have assumed that Q is simply given. In real-life applications, however, Q will be unknown initially, and will have to be estimated, based on empirical measurements of road costs. Realistically speaking, it is unlikely that enough measurements will have been made to estimate all of the entries of Q with any accuracy. In such a situation, it may be acceptable (or even essential) to work with a rough approximation of Q.

In our opinion, there are two possible approximations of Q that are likely to be acceptable in practice. The first is a low-rank approximation, based on principal components analysis. The other is a sparse approximation, in which it is assumed that the covariance will be zero whenever two roads are far from each other. In the following two subsections, we explain these two approximation strategies in detail, and show how they can be used to derive improved formulations of the STSPCC.

4.5.1 Covariance matrices with low rank

We now recall some well-known facts from linear algebra (see, e.g., [63], ch. 6). Given a covariance matrix $Q \in \mathbb{R}^{n\times n}$, let $\lambda_1, \ldots, \lambda_n \in \mathbb{R}_+$ be its eigenvalues, sorted in non-increasing order of value, and let $v^1, \ldots, v^n \in \mathbb{R}^n$ be the corresponding (normalised) eigenvectors. Then Q can be decomposed as:

$$\sum_{i=1}^{n} \lambda_i \, v^i (v^i)^T. \tag{4.13}$$

The *rank* of Q is the least integer k such that $\lambda_i > 0$ for $i \leq k$, but $\lambda_i = 0$ for $i > k$. If we let $M \in \mathbb{R}^{k \times n}$ be the matrix whose ith row is $\sqrt{\lambda_i} v^i$, we have $Q = M^T M$.

In our context, Q is a matrix of order $|E|$, where E is the set of roads in a road network. Given raw data on the road costs, there are basically two ways to compute a low-rank approximation of Q. One simple way is to perform a *principal components analysis* (see, e.g., [2]). This means computing the sample covariance matrix, computing the decomposition (4.13), and then selecting a value k such that the sum of the first k eigenvalues is large compared to the sum of the remaining $|E| - k$ eigenvalues. Then, the desired low-rank approximation can be obtained by summing only the first k terms in (4.13). Another way, popular in the finance literature (e.g., [18]), is to assume *a priori* that Q has rank k, for some small integer k, and then estimate the matrix M directly.

In any case, suppose that we are given an instance of the STSPCC such that Q has rank k, where k is a small integer. We can improve the mixed 0-1 CQP formulations presented in Subsection 4.4.1 in the following way. Let $M \in \mathbb{R}^{k \times |E|}$ be such that $Q = M^T M$. Add k new continuous variables $z_1, \ldots, z_k$ to the formulation, along with the following k linear equations:

$$z_r = \sum_{e=\{i,j\} \in E} M_{re}(x_{ij} + x_{ji}) \qquad (r = 1, \ldots, k).$$

Then, in the objective function, replace the expression

$$\sum_{a \in A} \sum_{b \in A} Q_{ab} x_a x_b$$

with the simpler expression $\sum_{r=1}^{k} z_r^2$.

The above modification of the mixed 0-1 CQP formulations does not affect the lower bound obtained by solving the continuous relaxation, but it dramatically decreases the number of non-zero terms in the objective function, at the cost of introducing only k continuous variables and k linear constraints. If k is small compared to n, this may reduce the amount of work done at each node of the branch-and-bound tree.

We remark that it is not possible to modify the mixed 0-1 LP formulations presented in Subsection 4.4.1 in an analogous way. This is because the y variables are continuous, rather than binary.

4.5.2 Covariances only between nearby links

Another way to simplify the covariance matrix Q is to assume that the covariances are significant only for pairs of roads that are 'close' in the road network. To be more precise, let us define the *distance* between edges e and f to be the least integer p such that there exists a path in G, from one of the end-vertices of e to one of the end-vertices of f, that uses p intermediate edges. Then, the assumption under consideration in this subsection is that there exists a small non-negative integer k such that, for all pairs $e, f \in E$, the covariance Q_{ef} is close to zero whenever the distance between e and f exceeds k.

As mentioned in Subsection 4.3.3, there is empirical evidence that the above-mentioned property holds for small values of k in road networks that have only mild congestion. Moreover, there now exist efficient computational methods for estimating sparse covariance matrices with specified sparsity patterns (see, e.g., [19]). Such methods can be used to compute good approximations of Q for any specified value of k.

So, suppose that we are given an instance of the STSPCC such that Q has a zero entry whenever the distance between the corresponding pair of edges is larger than k, for some small non-negative integer k. If we consider the mixed 0-1 CQP formulations presented in Subsection 4.4.1, we see that the only benefit is that the objective function is sparse. On the other hand, if we consider the mixed 0-1 LP formulations presented in Subsection 4.4.2, a much more significant benefit arises: for all pairs $a, b \in A$ such that the distance between the corresponding edges $e, f \in E$ exceeds k, we can delete the variable y_{ab} from the formulation, along with all constraints in which it appears. If k is small, this will result in a dramatic decrease in the number of variables and constraints.

4.6 Computational Experiments

In this section, we report on some computational experiments that we conducted with the formulations presented in the previous two sections. Section 4.6.1 explains how we created our test instances. The computational results are given in Subsection 4.6.3.

4.6.1 Construction of test instances

We began by taking eight STSP instances that were described in our earlier paper [88]. These were constructed as follows. For $V \in \{25, 50, 75, 100\}$, a random planar graph was created that was designed to resemble a real road network. For each of these four graphs, a random node was selected to be the depot, and the cost c_e of traversing an edge was then set to be proportional to the Euclidean distance between the end-nodes. Then, for each of the four graphs, two STSP instances were created. In one, each non-depot node was selected to be a customer with probability 1/3. In the other, the probability was 2/3. Table 4.1 shows the number of edges and customers in each of the eight sparse graphs.

Next, we converted each of the eight STSP instances into STSPCC instances with no special covariance structure. The expected cost vector $\bar{c}$ for every edge e was just set to c_e. The covariance matrix Q was created as follows:

- Generate $|E| \times |E|$ non-negative random numbers from the standard normal distribution.
- Scale each of the $|E|$ vector coordinates to have length one. This results in having $|E|$ uniform random vectors in the non-negative orthant of the unit

Instance	$\|V\|$	$\|E\|$	$\|C\|$
25-1/3	25	31	8
25-2/3			16
50-1/3	50	69	16
50-2/3			32
75-1/3	75	105	25
75-2/3			50
100-1/3	100	139	33
100-2/3			66

Tab. 4.1: Number of nodes, edges and customers for the 8 instances.

sphere in $|E|$ dimensions [95, 102].

- Multiply each of the $|E|$ vectors by a uniform random number in range $[0, 10]$ representing the standard deviation of the cost of that edge and round the components of every vector to the nearest integer.
- Let $M \in \mathbb{Z}_+^{|E|\times|E|}$ be the matrix whose columns are the $|E|$ dimensional vectors, in any order.
- Set Q to $M^T M$.

By construction, the entries in Q are in the range $[0, 100]$ for all eight instances. We chose to create covariance matrices with non-negative entries since negative covariances seem unlikely in real life (see, e.g. [104]).

Next, in order to explore the potential of the formulations that we gave in Subsection 4.5.1, we created another eight STSPCC instances, with a *low-rank* covariance structure. In fact, we considered the situation that ought to be most favourable to those formulations: the case in which the rank of Q is only one. To create Q in this case, we just set Q to vv^T, where $v \in \mathbb{Z}$ is a vector in which each entry is a random integer between 0 and 10. The entries in Q are then again between 0 and 100.

Finally, in order to explore the potential of the formulations that we gave in Subsection 4.5.2, we created another eight STSPCC instances in which covariances are non-zero only for links that are close in the road network. Again, we considered the situation that ought to be most favourable to those formulations: the case in which the covariance between two edges is non-zero only if those edges are adjacent in G. To create Q in this case, we set Q to MM^T, where $M \in \mathbb{Z}_+^{|E| \times |V|}$ is created as follows. For a given $e \in E$ and a given $i \in |V|$, we set M_{ei} to:

- A random integer between 1 and 10 if edge e has node i as an end-node;
- zero otherwise.

For this case, Q will be a sparse matrix with entries between 0 and 200.

4.6.2 Selection of the weights and epsilon

As mentioned above, the weighting and ϵ-constraint methods are two standard methods for computing efficient solutions to multi-objective problems. In the case of continuous problems, one can call them repeatedly to construct the entire efficient frontier to any desired degree of accuracy. In the case of discrete problems, the situation is a little more complicated, as mentioned in Section 4.3.2. In this paper, for simplicity, we experiment with only one value of weight and ϵ for each instance and each formulation, in such a way that the solutions obtained are guaranteed to be efficient.

For our experiments, in order to ensure that the comparison between the weighting and ϵ-constraint methods were as fair as possible, we used the following procedure for any given instance:

1. Solve the weighted problem via $0-1$ linear or quadratic programming using a weight of 10.
2. Let ϵ be the expected cost of the solution found in point 1.

3. Solve the ϵ-constrained version, for this value of ϵ, via $0-1$ linear or quadratic programming.

This way, both the weighting and ϵ methods will find tours of around the same length. The weighting method is well known to always give efficient solutions, even for problems with binary variables. Due to the way we have selected ϵ, the ϵ-constrained method is guaranteed to give the same solution as the weighting method. Therefore, we conclude that the solution obtained by the ϵ-constrained method will also be efficient.

4.6.3 Experimental results

A program was written in Microsoft Visual C that reads an instance from a file, and then outputs eight integer programs, corresponding to the formulations SCF-w, SCF-ϵ, MCF-w, MCF-ϵ, RLT-SCF-w, RLT-SCF-ϵ, RLT-MCF-w and RLT-MCF-ϵ. Each of the resulting 192 ($4 \times 3 \times 2 \times 8$) integer programs was fed into the branch-and-bound solver of IBM CPLEX version 12.3. The computer used was a PC with a 1.6 GHz Intel Core i7 processor, with 8 GB of RAM, operating under Windows 7. Default CPLEX settings were used.

For each integer program, a time limit of 5000 seconds was imposed. We were able to find the optimal solutions to 23 out of the 24 instances within the time limit. The one that was not solved was the instance with 100 nodes where 2/3 of the nodes are required and the covariance matrix has non-zero elements only for adjacent links.

Table 4.2 shows, for each of the twenty three instances and each of the eight formulations, the time taken to solve the continuous relaxation. Indices 'G', 'L' and 'N' refer to the general, low rank and nearby covariance structures, respectively. For brevity we use these indices to refer to the corresponding instances, throughout the rest of this paper. We see that the continuous relaxations of the mixed 0-1 CQP formulations are very easy to solve for the G and L instances, whereas for the

Instance	SCF-w	SCF-ϵ	MCF-w	MCF-ϵ	RLT-SCF-w	RLT-SCF-ϵ	RLT-MCF-w	RLT-MCF-ϵ
25-1/3-G	0.08	0.06	0.13	0.13	0.20	0.41	0.37	0.86
50-1/3-G	0.16	0.08	0.27	0.30	3.15	6.33	3.45	6.91
75-1/3-G	0.13	0.16	0.94	1.12	11.08	24.77	12.37	30.72
100-1/3-G	0.14	0.16	2.48	2.59	28.59	96.03	25.46	86.24
25-2/3-G	0.11	0.08	0.23	0.30	0.31	0.67	0.50	0.78
50-2/3-G	0.09	0.11	0.90	0.83	2.89	5.23	4.02	7.69
75-2/3-G	0.14	0.22	2.57	3.12	14.56	32.18	12.53	27.85
100-2/3-G	0.14	0.16	10.08	8.74	27.07	106.78	26.05	111.15
25-1/3-L	0.09	0.05	0.08	0.08	0.28	0.36	0.38	0.83
50-1/3-L	0.05	0.06	0.23	0.25	2.15	5.74	2.84	8.47
75-1/3-L	0.06	0.09	0.73	0.75	10.90	19.00	13.03	43.46
100-1/3-L	0.13	0.13	1.92	1.78	30.97	62.96	29.00	88.39
25-2/3-L	0.09	0.09	0.19	0.19	0.33	0.73	0.59	0.89
50-2/3-L	0.08	0.06	0.92	0.80	3.23	5.88	5.02	7.88
75-2/3-L	0.09	0.13	2.75	2.43	11.45	21.34	13.24	35.41
100-2/3-L	0.14	0.11	7.43	5.46	25.80	54.24	21.00	72.70
25-1/3-N	0.11	0.05	0.11	0.13	0.01	0.03	0.03	0.09
50-1/3-N	0.08	0.09	0.31	0.27	0.06	0.05	0.20	0.58
75-1/3-N	0.08	0.14	0.83	1.00	0.08	0.08	0.80	2.82
100-1/3-N	0.08	0.13	1.68	2.00	0.09	0.06	2.06	6.10
25-2/3-N	0.06	0.08	0.19	0.22	0.03	0.03	0.05	0.11
50-2/3-N	0.06	0.08	0.86	0.87	0.05	0.05	0.41	1.06
75-2/3-N	0.09	0.09	2.61	3.46	0.09	0.06	1.95	2.95

Tab. 4.2: Time taken to solve continuous relaxation, in seconds.

Instance	SCF-w	SCF-ϵ	MCF-w	MCF-ϵ	RLT-SCF-w	RLT-SCF-ϵ	RLT-MCF-w	RLT-MCF-ϵ
25-1/3-G	51.9	53.0	0.0	0.0	71.8	72.5	58.7	57.6
50-1/3-G	51.5	51.9	0.0	0.0	88.7	88.4	85.8	78.8
75-1/3-G	63.5	63.8	0.4	0.4	95.2	95.5	93.5	92.4
100-1/3-G	67.4	68.0	0.0	0.0	99.6	100.0	99.5	98.4
25-2/3-G	41.7	40.4	0.0	0.0	70.5	62.6	65.3	54.5
50-2/3-G	41.8	42.1	2.2	2.2	84.8	79.5	81.8	68.4
75-2/3-G	55.2	55.6	0.0	0.0	94.9	95.2	93.7	93.6
100-2/3-G	64.6	65.1	2.1	2.0	99.6	100.0	99.6	100.0
25-1/3-L	56.3	56.6	0.0	0.0	74.2	72.2	57.7	23.1
50-1/3-L	49.6	49.8	0.0	0.0	86.1	77.2	81.9	44.9
75-1/3-L	43.9	44.0	0.0	0.0	91.0	83.2	89.3	59.8
100-1/3-L	55.3	55.4	1.2	1.2	99.9	98.7	99.9	85.5
25-2/3-L	34.0	34.1	0.0	0.0	74.5	53.2	70.1	28.8
50-2/3-L	29.0	29.1	1.0	0.7	75.7	61.1	73.3	51.1
75-2/3-L	32.4	32.5	0.3	0.3	92.1	81.2	91.4	68.7
100-2/3-L	35.1	35.2	0.0	0.0	99.9	91.5	99.9	84.6
25-1/3-N	33.1	33.7	1.1	0.9	67.7	68.5	66.6	67.9
50-1/3-N	36.9	37.4	10.9	11.3	81.4	82.2	80.2	81.4
75-1/3-N	42.7	43.4	6.1	4.7	93.9	94.8	92.0	92.9
100-1/3-N	46.0	46.9	11.5	11.6	99.2	100.0	98.9	100.0
25-2/3-N	21.2	21.7	1.0	0.9	68.6	69.8	68.0	69.5
50-2/3-N	27.1	27.6	7.2	7.3	82.2	83.2	77.1	78.1
75-2/3-N	31.6	32.2	6.1	6.2	94.7	95.8	93.4	94.6

Tab. 4.3: Percentage integrality gaps.

N instances, they solve about as quick as the relaxations of the RLT formulations. This is because for the N instances the RLT formulations have fewer variables and constraints than the corresponding mixed 0-1 CQP formulations. For a similar reason, the continuous relaxations of SCF formulations can be solved more quickly than those of the MCF formulations. Interestingly, the continuous relaxations of the mixed 0-1 CQP formulations for the L instances solve faster than those of the G instances. This may be due to the fact that they have less variables in the objective function.

Table 4.3 shows, for the 23 instances that were solved to optimality, and for the same eight formulations, the *percentage integrality gap*, i.e., the difference between the lower bound given by the LP solution and the cost of the true integer optimum,

Instance	SCF-w	SCF-ϵ	MCF-w	MCF-ϵ	RLT-SCF-w	RLT-SCF-ϵ	RLT-MCF-w	RLT-MCF-ϵ
25-1/3-G	195	156	3	0	579	842	56	44
50-1/3-G	47050	50913	9	8	–	–	–	–
75-1/3-G	3652603	–	184	264	–	–	–	–
100-1/3-G	–	–	7	4	–	–	–	–
25-2/3-G	183	52	0	0	110	170	0	0
50-2/3-G	49521	43005	28	32	–	–	–	–
75-2/3-G	1140106	–	25	19	–	–	–	–
100-2/3-G	–	–	142	78	–	–	–	–
25-1/3-L	36	50	0	0	323	201	48	10
50-1/3-L	2264	3545	0	0	–	–	–	–
75-1/3-L	159903	39417	0	0	–	–	–	–
100-1/3-L	–	–	0	57	–	–	–	–
25-2/3-L	54	72	0	0	446	240	105	11
50-2/3-L	66496	82813	0	0	–	–	–	–
75-2/3-L	324141	–	0	0	–	–	–	–
100-2/3-L	–	–	0	9	–	–	–	–
25-1/3-N	263	160	63	40	90	56	8	3
50-1/3-N	349158	143175	12220	–	4151	3295	407	277
75-1/3-N	–	–	24085	–	61206	137587	1002	682
100-1/3-N	–	–	–	–	–	–	10225	16180
25-2/3-N	148	151	61	18	145	44	9	9
50-2/3-N	2859911	1333279	39289	4590	37227	45444	1379	1120
75-2/3-N	–	–	–	–	669891	–	12873	7171

Tab. 4.4: Number of branch-and-bound nodes.

expressed as a percentage of the latter. We see that the gaps are far smaller for MCF-w and MCF-ϵ formulations than for any of the other formulations. The superiority of the MCF formulation over the SCF formulation is explained in [88]. For the 0-1 mixed CQP formulations, the N instances have the smallest gaps and G instances have the biggest. This is probably due to the sparsity of the objective function for the L and N instances. Also, observe that as the proportion of required nodes increases, the gaps decrease for the SCF formulations and for the RLT formulations the LP relaxation gaps increase with n. The reason for these two phenomena is not clear. Finally, we note that gaps may increase if one wanted to consider negative elements in any of the covariance matrices.

Table 4.4 shows, for each of the 23 instances solved, the number of branch-and-bound nodes needed to solve the integer programs to proven optimality. A dash ($-$) indicates that the given formulation could not be solved within the given time limit. We see that, the MCF formulations give the true integer optimal solution using very few branch-and-bound nodes. Particularly for L instances with smaller values of n, they do not use any branch-and-bound. The mixed 0-1 CQP formulations need significantly less branch-and-bound nodes for the L instances than they do for G and N instances. For the RLT formulations, an excessive number of nodes is needed for the G and L instances, even for small values of n but for N instances they need less branch-and-bound nodes than the mixed 0-1 CQP formulations.

It should be noted that, in a few cases, the number of branch-and-bound nodes is zero even though the integrality gap is positive. The explanation for this anomalous behaviour is that, in some cases, CPLEX automatically appended some of its own internal cutting planes to the continuous relaxation. If one were to switch off all internal cutting plane generation in CPLEX, then branching would become necessary whenever there is a positive integrality gap.

Finally, Table 4.5 shows the total time, in seconds, taken by the branch-and-bound procedure. We see that despite having more variables, the MCF formulations perform quite well, enabling one to find a provably optimal solution even for quite large values of n. The mixed 0-1 CQP formulations consume far less time for L instances than they do for G and N instances. This is because less time is spent at each branch-and-bound node due to having less number of variables in the objective function. The RLT formulations, struggle even for small values of n, in the case of G and L instances but they are quicker for N instances. This is due to the fact that they have smaller number of variables for N instances.

All things considered, the MCF formulations outperform for instances with

Instance	SCF-w	SCF-ϵ	MCF-w	MCF-ϵ	RLT-SCF-w	RLT-SCF-ϵ	RLT-MCF-w	RLT-MCF-ϵ
25-1/3-G	0.31	0.30	0.33	0.20	13.67	39.51	12.17	21.50
50-1/3-G	62.20	34.94	1.53	1.64	–	–	–	–
75-1/3-G	3863.30	–	20.58	29.48	–	–	–	–
100-1/3-G	–	–	22.59	10.59	–	–	–	–
25-2/3-G	0.44	0.42	0.34	0.37	87.81	151.66	37.97	54.41
50-2/3-G	24.98	26.39	6.80	7.07	–	–	–	–
75-2/3-G	1161.82	–	12.37	16.08	–	–	–	–
100-2/3-G	–	–	223.41	1713.62	–	–	–	–
25-1/3-L	0.27	0.26	0.13	0.09	7.94	41.28	10.05	13.07
50-1/3-L	1.15	2.48	0.34	0.50	–	–	–	–
75-1/3-L	87.19	29.19	1.17	1.22	–	–	–	–
100-1/3-L	–	–	9.84	738.35	–	–	–	–
25-2/3-L	0.27	0.28	0.25	0.20	32.45	66.72	25.80	7.33
50-2/3-L	20.84	47.99	6.41	10.92	–	–	–	–
75-2/3-L	169.85	–	302.56	26.18	–	–	–	–
100-2/3-L	–	–	4896.34	90.81	–	–	–	–
25-1/3-N	0.28	0.22	0.38	0.36	0.30	0.34	0.36	0.42
50-1/3-N	109.90	70.11	65.61	–	5.15	5.35	5.54	5.97
75-1/3-N	–	–	1020.78	–	104.79	573.38	75.57	66.71
100-1/3-N	–	–	–	–	–	–	1764.29	4633.06
25-2/3-N	0.23	0.31	0.61	0.53	0.33	0.28	0.62	0.56
50-2/3-N	918.86	2078.12	1501.24	163.32	49.97	63.85	46.22	46.66
75-2/3-N	–	–	–	–	1634.69	–	1415.72	749.10

Tab. 4.5: Branch-and-bound time, in seconds.

low rank and general covariance structures. For nearby covariance structures, we recommend using the RLT-MCF formulations. It is not clear why the mixed 0-1 CQP formulations on N instances, do not perform as well as they do on G instances.

4.7 Discussion

In real-life road networks, the costs or travel times for individual roads are stochastic, since they may be affected by unexpected events such as road works or accidents. Moreover, as mentioned in Subsection 4.3.3, there is empirical evidence that the costs or times for pairs of roads can be correlated. Accordingly, we have proposed the Steiner TSP with correlated costs as a new and potentially useful VRP. We considered two ways to reduce this bi-objective problem into a single-objective problem, and then presented eight different integer programming formulations. We also showed how to exploit special structures in the covariance matrix. Our experimental results have shown that instances with up to 100 nodes can be solved to proven optimality with a standard branch-and-bound solver.

A possible extension of this work could be to devise methods for computing the entire efficient frontier for the STSPCC, or at least approximations of it. It would also be interesting to see if larger instances could be tackled by incorporating strong cutting planes into the formulation and solution algorithm. Another possible topic for future research would be to incorporate correlations in costs or travel times into other vehicle routing problems, or indeed into models of traffic and/or transportation in general.

5. PRICING ROUTINES FOR VEHICLE ROUTING WITH TIME WINDOWS ON ROAD NETWORKS

5.1 abstract

Several very effective exact algorithms have been developed for vehicle routing problems with time windows. Unfortunately, most of these algorithms cannot be applied to instances that are defined on road networks, because they implicitly assume that the cheapest path between two customers is equal to the quickest path. Garaix and co-authors proposed to tackle this issue by first storing alternative paths in an auxiliary multi-graph, and then using that multi-graph within a branch-and-price algorithm. We show that, if one works with the original road network rather than the multi-graph, then one can solve the pricing subproblem more quickly, in both theory and practice.

Keywords: Vehicle routing, combinatorial optimization, bi-criteria shortest paths.

5.2 Introduction

Vehicle Routing Problems (VRPs) are a much-studied class of combinatorial optimization problems, and several books have been written about them (e.g., [12, 62, 65, 124]). Many VRPs arising in practical applications involve restrictions on the time at which service begins at the customers. Well-known examples include the *Traveling Salesman Problem with Time Windows* or TSPTW [6], the *Multiple Traveling Salesman Problem with Time Windows* or m-TSPTW [39, 114], and the *Vehicle Routing Problem with Time Windows* or VRPTW [36, 118]. Good surveys on such problems include [27, 38].

As mentioned in the above-mentioned books and surveys, there are several effective exact algorithms available to solve such time-constrained VRPs, some of which are capable of routinely solving instances with up to around 100 customers to proven optimality. There are also several effective heuristics that are able to provide good solutions for even larger instances.

Unfortunately, and surprisingly, most of these algorithms are based on a key assumption that is not guaranteed to hold in the real world. This assumption is that, for all ordered pairs (i, j) of nodes (that represent either depots or customers), one is provided with two numbers: c_{ij}, the cost of traveling from i to j, and t_{ij}, the time taken to travel from i to j. In reality, however, many VRPs are concerned with the routing of vehicles on *road networks*. In a real-life road network, the cheapest path between two points is unlikely to be the same as the quickest path. Therefore, to model and solve time-constrained VRPs on road networks correctly, one should take into account the trade-off between travel costs and travel times.

This issue was explained in detail in a recent paper by Garaix *et al.* [59], who proposed to remedy the situation as follows. First, in a pre-processing stage, they solve a series of *bicriteria shortest path* problems in the road network. This yields, for each pair of customer and/or depot nodes in the road network, a set of paths that completely represent the cost-time trade-off. These paths are stored in an auxiliary *multi-graph*. Then, they use a traditional branch-and-price algorithm, but solve the pricing subproblem on the multi-graph rather than the original road network. Finally, dynamic programming is used to convert the optimal solution into a collection of feasible routes in the road network.

Although the approach of Garaix *et al.* [59] is elegant, it does require the use of specialised techniques for constructing the multi-graph, solving the pricing problem, and converting the solution. Moreover, as we explain in Section 5.4, constructing and storing the multi-graph can take exponential time and space in the worst case. These considerations led us to develop more ‘natural’ algorithms for the pricing subproblem, that work directly on the original road network. It turns out that these natural algorithms, as well as being simpler, are faster in both theory and practice.

The paper is organized as follows. In Section 5.3, we review the relevant literature. In Section 5.4, we present a result about the size of the multi-graph. In Section 5.5, we present our first pricing routine, which is designed for the case of ‘elementary’ routes. We also show that we obtain, as a by-product, an exact algorithm for the ‘road network’ version of the TSPTW. In Section 5.6, we present our second pricing routine, for the case in which ‘non-elementary’ routes are permitted, and show that it is faster (in the worst case) than the one of Garaix *et al.* [59]. In Section 5.7, we show that it is faster also in practical computations. Finally, some concluding remarks appear in Section 5.8.

5.3 Literature Review

We now review the relevant literature. Subsection 5.3.1 deals with the TSPTW, m-TSPTW and VRPTW, Subsection 5.3.2 covers VRPs on road networks, and Subsection 5.3.3 focuses on VRPs with time windows on road networks.

5.3.1 Standard VRPs with time windows

The TSPTW is defined as follows [6]. Let G be a complete directed graph with vertex set $V = \{0, 1, \ldots, n\}$ and arc set A. Vertex 0 represents the depot and the other vertices represent customers. For each $(i, j) \in A$ we are given a cost c_{ij} and a traversing time t_{ij}. Each customer i has a time window $[e_i, \ell_i]$ and a service time s_i. Service at customer i must start no earlier than e_i and no later than ℓ_i. If the vehicle arrives at customer i before e_i, it has to wait. The vehicle departs from the depot at time 0 and must return to the depot by time T. The objective is to find a minimum cost route that services each customer once and satisfies the time window requirements. It is usually assumed that all costs and times are positive integers.

The m-TSPTW is identical to the TSPTW, except that there are several vehicles and each customer must be visited by exactly one vehicle [39, 114]. The VRPTW is similar, except that each customer i has a positive integral demand q_i and the total load of each vehicle must not exceed some positive integral capacity Q [36, 37].

Exact approaches to the TSPTW include, e.g., dynamic programming (DP) [45], hybrid DP / branch-and-bound [10], constraint programming [111] and branch-and-cut [4, 32]. Exact approaches to the multi-vehicle problems include, e.g., Lagrangian relaxation [77], branch-and-cut [91], branch-and-price [17, 37, 39, 51], branch-cut-and-price [35, 74, 80] and, very recently, hybrid DP / dual ascent / branch-and-bound [9].

All of the exact approaches to the multi-vehicle problems are based, either explicitly or implicitly, on *set covering* or *set partitioning* formulations. The set covering formulation takes the form

$$\min \quad \textstyle\sum_{r\in\Omega} c_r \lambda_r \tag{5.1}$$

$$\text{s.t.} \quad \textstyle\sum_{r\in\Omega} a_{ir}\lambda_r \geq 1 \quad (\forall i \in V \setminus \{0\}) \tag{5.2}$$

$$\lambda_r \in \{0,1\} \qquad (\forall r \in \Omega), \tag{5.3}$$

where Ω denotes the set of all feasible routes for a single vehicle, c_r denotes the cost of route r, λ_r is a binary variable taking the value 1 if and only if a vehicle uses route r, and a_{ir} is a binary constant, taking the value 1 if and only if customer i is serviced by route r. The set partitioning formulation is identical, except that the inequalities (5.2) are changed to equations. (If the costs and times obey the triangle inequality, which is usually the case in practice, this change affects neither the optimal solution, nor the lower bound from the LP relaxation.)

Since $|\Omega|$ can be exponentially large, if one wishes to solve the LP relaxation of

(5.1)–(5.3) exactly, it must be solved via *column generation*, i.e, the variant of the simplex method in which columns of negative reduced cost are generated on-the-fly via pricing routines. The pricing subproblem is strongly $\mathcal{NP}$-hard [42], but can be solved in pseudo-polynomial time if one permits *non-elementary* routes, i.e., routes that visit customers more than once [37, 39]. The resulting enlargement of the column set Ω does not change the validity of the set covering / partitioning formulations, but it weakens the lower bound obtained when one solves the LP relaxation. As a compromise, one can forbid some non-elementary routes but not others (e.g., [9, 35, 73]).

5.3.2 Routing on road networks

All of the approaches mentioned in the previous subsection assume that the instance is defined on a complete directed graph. Most VRPs arising in practice, however, take place on road networks. It is usually possible to transform a VRP on a road network into a VRP on a complete graph, via a series of shortest-path computations (e.g., [28, 34, 53]). Nevertheless, it can be preferable to work with the original road network, in an attempt to exploit any properties, such as sparsity or planarity, that it may have (e.g., [28, 53, 88, 89]).

Much of the literature on VRPs on road networks has been concerned with so-called *arc routing* problems, in which the customers are located along the edges or arcs of the network, rather than at nodes. For brevity, we do not review the arc routing literature here, and refer the reader to the book [43]. We mention however that the paper [89], concerned with the so-called *Capacitated Arc Routing Problem* (CARP), can be viewed as a companion paper to the present one. It shows that pricing for the CARP can be performed more quickly if one works on the original road network rather than on a complete graph. The pricing routines presented in [89] are however quite different from the ones presented here. In particular, the routine for elementary routes in [89] is based on integer programming, whereas the one we present in Section 5.5 is based on dynamic programming. Moreover, the routine for non-elementary routes in [89] is slower and more complex than the one we present in Section 5.6, due to the fact that, in the case of the CARP, the vehicle load does not change when an edge is deadheaded.

We now return to node routing. The following 'road network' version of the TSP was defined in [28, 53, 106]. We are given:

- an undirected road network $\tilde{G} = (\tilde{V}, \tilde{E})$,
- a specified depot node, say node 0,
- a set of customer nodes $C \subset \tilde{V} \setminus \{0\}$,
- a cost $\tilde{c}_e$ for each $e \in E$.

The task is to find a minimum-cost tour, starting and ending at the depot, that passes through each customer node at least once. Nodes may be visited more than once, and edges may be traversed more than once, if desired. (Note that nodes in $\tilde{V} \setminus (C \cup \{0\})$ represent road junctions.)

Following [28, 88], we call the above variant of the TSP the *Steiner* TSP. In [26, 53], the Steiner TSP is formulated as an integer program with $\mathcal{O}(|\tilde{E}|)$ variables and an exponential number of constraints, and solved with cutting planes and branch-and-bound. In [88], it is formulated as an integer program with only $\mathcal{O}(|\tilde{E}|)$ variables and constraints, and solved via plain branch-and-bound.

Several generalisations of the Steiner TSP were also presented in [88]. Of relevance to us is the Steiner version of the TSPTW. This is like the Steiner TSP, but each customer $i \in C$ now has a time window $[e_i, \ell_i]$ and a service time s_i, and the vehicle must leave the depot at time 0 and return by time T. One can easily define Steiner versions of the m-TSPTW and VRPTW in a similar way. (In [88], the graph $\tilde{G}$ was assumed to be undirected, but one can also allow it to be directed, or mixed.)

5.3.3 Routing on road networks with time windows

Unfortunately, VRPs on road networks become significantly harder to model and solve when time windows are present. Indeed, in [88], we were able to formulate all problems considered as integer programs with only $\mathcal{O}(|\tilde{E}|)$ variables and constraints, *except* the Steiner TSPTW. The reason for this phenomenon is as follows: for the other problems considered in [88], it is never optimal for a vehicle to traverse an edge more than once in any given direction. When time windows are present, this is no longer true.

As mentioned in the introduction, another key paper in this regard is Garaix *et al.* [59]. They point out that, when time windows are present, one cannot always transform a 'road network' VRP into a standard VRP. This is because in a real-life road network the cheapest path between two points is unlikely to be the same as the quickest path.

To see this, consider the simple road network displayed in Figure 5.1. Suppose the depot is located at node 0, and the customers are located at nodes 2 and 4. Also suppose that we have (undirected) edges rather than (directed) arcs, for simplicity. For each of the seven edges, the corresponding traversing cost and time are displayed in parentheses. Between the depot and node 4, the cheapest path costs 2 and takes 4 time units, and the quickest path costs 4 and takes 2 time units. So there is no unique way to define c_{02} and t_{02}. A similar consideration applies to the paths between the two customers.

Garaix *et al.* proposed to remedy the situation with a three-phase procedure. In the first phase, they solve a series of *bicriteria shortest path* problems in $\tilde{G}$, in order to compute, for each ordered pair of nodes in $\{0\} \cup C$, a complete and non-redundant set of *efficient* (or *non-dominated*, or *pareto-optimal*) paths. (A variety of algorithms exist for generating such complete sets; see [112].) A (loopless and directed) multi-graph is then constructed, which has node set $\{0\} \cup C$ and, for

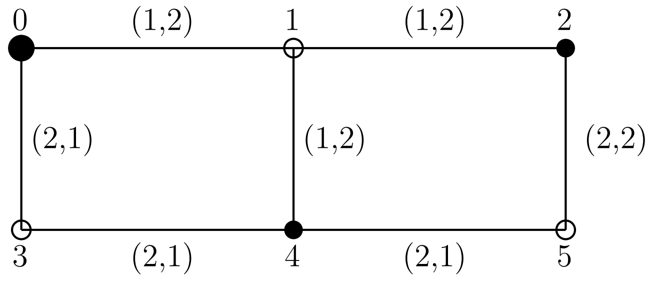


Fig. 5.1: Example of a possible road network $\tilde{G}$.

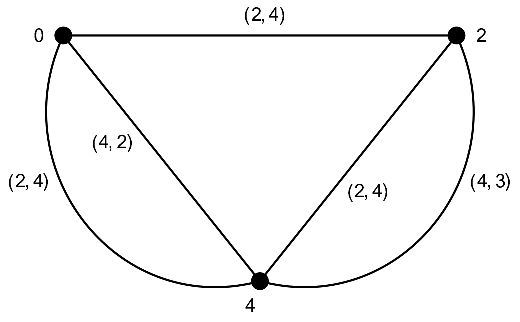


Fig. 5.2: Corresponding multi-graph.

each ordered pair of nodes, a set of parallel arcs, representing the corresponding complete set. (Figure 5.2 is the multi-graph corresponding to the road network in Figure 5.1, but with edges instead of arcs, for clarity.)

In the second phase, Garaix *et al.* formulate the problem as a set covering problem with a variable for each feasible route in the multi-graph, and solve it by branch-and-price. In the third and final phase, the optimal set covering solution is converted into an optimal solution for the original problem, via dynamic programming.

5.4 Drawbacks of the Multi-Graph Approach

Although the branch-and-price approach of Garaix *et al.* [59] performs reasonably well in practice, it seems both simpler and more natural to work with the original road network $\tilde{G}$ rather than the multi-graph. Moreover, the time and space requirements of the multi-graph approach can be very high in the worst case.

Specifically:

- Computing complete sets from a source node to all other nodes takes $\mathcal{O}(|\tilde{A}|T)$ time (see [112]). Therefore, computing complete sets for all ordered pairs (i,j) with i and j in $C \cup \{0\}$ will take $\mathcal{O}(|C|\,|\tilde{A}|\,T)$ time.

- For a given ordered pair (i,j), the cardinality of a complete set can be as large as T [68]. So the number of arcs in the multi-graph can be exponential in the encoding length of T.

- Converting the solution for the transformed problem into a solution for the original problem takes $\mathcal{O}(m\,|\tilde{A}|\,T)$ time, where m is the number of vehicles used.

In fact, the situation is even worse than this, as shown by the following theorem.

Theorem 6. *The multi-graph can contain $\Theta(|C|^2\,T)$ arcs, even if the original graph $\tilde{G}$ contains only $\mathcal{O}(|C|\,\log T)$ arcs.*

Proof. Suppose initially that $|C| = 1$, i.e., there is just a single customer. Let k be any positive integer. Construct a graph $\tilde{G}^k$ as follows. The vertex set is $\{0,\dots,2k\}$, where 0 is the depot and $2k$ is the customer. For $i = 1,\dots,k$, the arc set $\tilde{A}$ contains the arcs $(i-1,i)$, $(i-1,i+k)$ and $(i,i+k)$, together with arcs in the opposite direction. The arcs $(i-1,i)$ have cost $2+2^{i-1}$ and time 1. The arcs $(i-1,i+k)$ have cost 1 and time 1. The arcs $(i,i+k)$ have cost 1 and time 2^{i-1}. For each of these arcs, the arc going in the opposite direction has the same cost and time. (Figure 5.3 shows the graph $\tilde{G}^3$, but with pairs of opposite arcs shown as edges, to aid clarity.)

By construction, for any integer t between 0 and 2^k-1, there is a path from 0 to $2k$ in $\tilde{G}^k$ that has time $k+t$ and cost $2k+2^k-2-t$. Each of these paths is non-dominated. If we set $T = 2^{k+1}$, then each of these paths is also feasible, and therefore corresponds to an arc from 0 to $2k$ in the multi-graph. The multi-graph also contains analogous arcs from $2k$ to 0. Thus, the multi-graph contains $2^{k+1} = T$ arcs, yet $\tilde{G}^k$ contains only $3k = \mathcal{O}(\log T)$ arcs. This proves the result for $|C| = 1$.

To prove the result for $|C| > 1$, create $|C|$ copies of the graph $\tilde{G}^k$, one for each customer, and merge them into a single graph, by identifying the depot nodes. Now, between any pair of customers, for any integer t between 0 and 2^k-2, there is a non-dominated path between the customers that has time $2k+t$. (It suffices to concatenate a non-dominated path from the first customer to the depot, of time $k+\lfloor t/2 \rfloor$, with a non-dominated path from the depot to the other customer, of

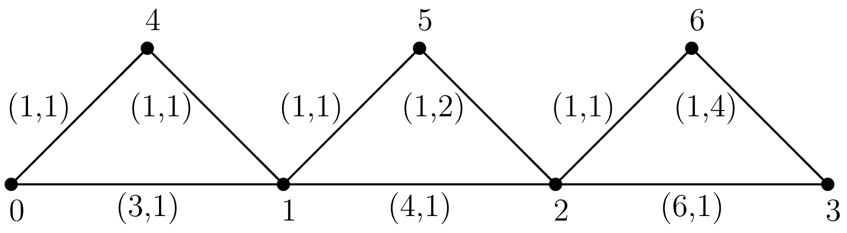


Fig. 5.3: The graph $\tilde{G}^3$, with costs and times.

time $k + \lceil t/2 \rceil$.) Moreover, if we set $T = 2^{k+2}$, then each of these paths is feasible, and therefore corresponds to an arc in the multi-graph. Thus, the multi-graph contains $\Theta(|C|^2 T)$ arcs, yet the original graph contains only $3k|C| = \mathcal{O}(|C| \log T)$ arcs. □

The above considerations suggest that it may be better to work directly with the original road network $\tilde{G}$ rather than with the multi-graph. So, suppose we have an instance of the 'Steiner' m-TSPTW or VRPTW, and consider once again the set covering formulation (5.1)–(5.3). In the Garaix *et al.* approach, Ω is assumed to contain all feasible routes in the multi-graph. We propose instead to assume that Ω contains all feasible routes in $\tilde{G}$. It is important to note that this change in the definition of Ω has no effect on the cost of the optimal solution to the set covering problem, nor on the lower bound from its LP relaxation. Indeed, given any feasible route in the multi-graph, there is a corresponding feasible route in $\tilde{G}$ of no greater cost that services the same set of customers, and vice-versa.

We remark that it is always valid to use set covering rather than set partitioning when working on $\tilde{G}$. Indeed, if an optimal solution to the set covering problem corresponds to a set of routes that collectively service a customer more than once, we can always take one of the routes that services that customer, and modify it so that the vehicle passes through the given customer node without actually servicing the customer. The modified route remains feasible and has the same cost as the original route. Repeating this procedure, if necessary, we obtain an optimal set of routes that collectively service each customer exactly once. (Incidentally, the same argument shows the validity of using set covering in the multi-graph approach.)

As in the case of the standard m-TSPTW and VRPTW [37, 39], one can if desired enlarge the column set Ω so that non-elementary routes are permitted. In the 'Steiner' context, this corresponds to permitting routes in $\tilde{G}$ that *service* some customers more than once. (Recall that even elementary routes in $\tilde{G}$ are permitted to *pass through* nodes more than once, regardless of whether or not they are customer nodes.) In the following two sections, we present pricing routines for the Steiner m-TSPTW and Steiner VRPTW for the cases of elementary and non-elementary routes, respectively.

5.5 Pricing Elementary Routes

In this section, we present an exact pricing routine for the Steiner m-TSPTW, for the case in which only elementary routes are permitted. We also show how to adapt it to the Steiner VRPTW. We will see that these routines are much faster than the analogous routines based on the multi-graph. Throughout, we assume for simplicity of notation that the graph $\tilde{G}$ is directed, with node set $\tilde{V}$ and arc set $\tilde{A}$. The routines can be easily modified for the case in which $\tilde{G}$ is undirected or mixed.

For a given $\Omega' \subset \Omega$, suppose that the LP relaxation of the formulation (5.1)–(5.3) has been solved, but only using columns in Ω'. For each $i \in C$, let $\pi_i \geq 0$ be the dual price of the corresponding constraint (5.2) in the solution to the relaxation. The reduced cost of the variable λ_r for a route $r \in \Omega \setminus \Omega'$ is equal to its true cost c_r, minus $\sum_{i \in C} a_{ir} \pi_i$. We now show that, in the case of the Steiner m-TSPTW, one can find routes of negative reduced cost in $\mathcal{O}\left(2^{|C|}\,|\tilde{A}|\,T\right)$ time.

For all $S \subseteq C$, all $i \in \tilde{V}$ and all $0 \leq t \leq T$, let $f(S, i, t)$ denote the reduced cost of the cheapest feasible path in $\tilde{G}$ (if any) that starts at the depot, services every customer in S exactly once, possibly visiting other nodes in $\tilde{V}$ along the way, and ends at node i at time t. (If no such feasible path exists, we view $f(S, i, t)$

as being infinite.) Also, for any given node $i \in \tilde{V}$, let $n^+(i)$ denote the set of *forward neighbors* of i in $\tilde{G}$, by which we mean the set of nodes j for which the arc (i, j) exists in $\tilde{A}$. One can then compute $f(S, i, t)$ for all feasible triples (S, i, t) as follows:

```
Set $f(\emptyset, 0, 0) = 0$.
For $t = 0, \ldots, T$ do:
    For all $i \in \tilde{V}$ do:
        For all $S \subseteq C$ do:
            For all $j \in n^+(i)$ such that $t + \tilde{t}_{ij} < T$ do:
                If $f(S, i, t) + \tilde{c}_{ij} < f(S, j, t + \tilde{t}_{ij})$,
                    Set $f(S, j, t + \tilde{t}_{ij})$ to $f(S, i, t) + \tilde{c}_{ij}$.
            For all $j \in n^+(i) \cap (C \setminus S)$ such that $t + \tilde{t}_{ij} < e_j$ do:
                If $f(S, i, t) + \tilde{c}_{ij} - \pi_j < f(S \cup \{j\}, j, e_j + s_j)$,
                    Set $f(S \cup \{j\}, j, e_j + s_j)$ to $f(S, i, t) + \tilde{c}_{ij} - \pi_j$.
            For all $j \in n^+(i) \cap (C \setminus S)$ such that $e_j \leq t + \tilde{t}_{ij} \leq \ell_j$ do:
                If $f(S, i, t) + \tilde{c}_{ij} - \pi_j < f(S \cup \{j\}, j, t + \tilde{t}_{ij} + s_j)$,
                    Set $f(S \cup \{j\}, j, t + \tilde{t}_{ij} + s_j)$ to $f(S, i, t) + \tilde{c}_{ij} - \pi_j$.
```

In this DP algorithm, the three inner loops involving j correspond, respectively, to (i) travelling from i to j without servicing j, (ii) travelling from i to j, waiting, and then servicing j, and (iii) travelling from i to j and servicing j immediately. When the DP terminates, any pair (S, t) with $f(S, 0, t) < 0$ corresponds to a column with negative reduced cost. To construct such a column one simply traces the path backward from the end state $(S, 0, t)$ to the initial state $(\emptyset, 0, 0)$.

Note that, for a given t and S, each arc in $\tilde{A}$ is examined no more than three times. Since there are at most $T+1$ choices for t and $2^{|C|}$ choices for S, the running time is $\mathcal{O}\left(2^{|C|}\,|\tilde{A}|\,T\right)$, as claimed. Although this running time is exponential in $|C|$, it compares very favourably with the running time that would be obtained if one used the multi-graph. Indeed, in that case, to compute $f(S,i,t)$ for a given S, i and t, one would have to examine every arc in the multi-graph whose head is i and whose tail is in $S \setminus \{i\}$. From the proof of Theorem 6, this would take $\mathcal{O}(|C|\,T)$ time. Since there are $\mathcal{O}\left(2^{|C|}\right)$ choices for S, $\mathcal{O}(|C|)$ choices for i, and T choices for t, the DP would take $\mathcal{O}\left(2^{|C|}\,|C|^2\,T^2\right)$ time.

As usual with DP algorithms (e.g., [17, 35, 45, 51]), some simple tests can be used to eliminate states from consideration. For example, suppose that we pre-compute, for all $i \in \tilde{V} \setminus \{0\}$, the time taken to travel from node i to the depot, via a quickest path. Then one can eliminate the state (S,i,t) if the time taken to travel from i to the depot exceeds $T-t$. Even with such additional tests, however, the algorithm can be expected to be viable for large values of $|C|$ only if the time windows are very narrow. This is because pricing typically has to be performed many times in a branch-and-price algorithm.

We remark that to adapt the above pricing routine to the Steiner VRPTW it suffices to introduce an additional state variable q, representing the cumulative load of the partial path, and add another 'for' loop in which q ranges from 0 to Q. The running time does however then increase by a factor of Q.

To close this section, we observe that the above pricing routine can easily be converted into an exact algorithm for the Steiner TSPTW. Indeed, if we set π_j to zero for all $j \in C$ and run the routine, then the cost of an optimal Steiner TSPTW solution is given by:

$$\min_{0<t\leq T} \{f(C,0,t)\},$$

and one can construct such an optimal solution by tracing the path backward from the optimal final state to the initial state $(\emptyset, 0, 0)$. The running time remains unchanged at $\mathcal{O}\left(2^{|C|}\,|\tilde{A}|\,T\right)$. We imagine that, with the addition of some standard state-elimination and dominance tests (as in [45]), this could lead to a viable solution algorithm for the Steiner TSPTW, provided the time windows are reasonably narrow. We are not aware of any other exact algorithm for the Steiner TSPTW.

5.6 Pricing Non-Elementary Routes

In this section, we present exact pricing routines for the Steiner m-TSPTW and VRPTW, for the case in which non-elementary routes are permitted. Once again, we will see that the routines are much faster than the analogous routines based on the multi-graph.

The new pricing routine for the Steiner m-TSPTW is similar to the one described in the previous section, but we drop the index S from the state, since we no longer need to keep a record of which customers have already been serviced. Accordingly, for all $i \in \tilde{V}$ and all $0 \leq t \leq T$, let $f(i,t)$ denote the reduced cost of the cheapest feasible path (if any) that starts at the depot, services any customers any number of times, possibly visiting other nodes in $\tilde{V}$ along the way, and ends at node i at time t. One can compute $f(i,t)$ for all i and t as follows:

```
Set $f(0,0) = 0$.
For $t = 0, \dots, T$ do:
    For all $i \in \tilde{V}$ do:
        For all $j \in n^+(i)$ such that $t + \tilde{t}_{ij} < T$ do:
            If $f(i,t) + \tilde{c}_{ij} < f(j, t + \tilde{t}_{ij})$,
                Set $f(j, t + \tilde{t}_{ij})$ to $f(i,t) + \tilde{c}_{ij}$.
        For all $j \in n^+(i) \cap C$ such that $t + \tilde{t}_{ij} < e_j$ do:
            If $f(i,t) + \tilde{c}_{ij} - \pi_j < f(j, e_j + s_j)$,
                Set $f(j, e_j + s_j)$ to $f(i,t) + \tilde{c}_{ij} - \pi_j$.
```

```
For all $j \in n^+(i) \cap C$ such that $e_j \leq t + \tilde{t}_{ij} \leq \ell_j$ do:
    If $f(i,t) + \tilde{c}_{ij} - \pi_j < f(j, t + \tilde{t}_{ij} + s_j)$,
        Set $f(j, t + \tilde{t}_{ij} + s_j)$ to $f(i,t) + \tilde{c}_{ij} - \pi_j$.
```

The explanation of this DP routine is similar to that given in the previous section. When it terminates, any t with $f(0,t) < 0$ corresponds to a column with negative reduced cost.

This DP routine is easily seen to run in $\mathcal{O}(|\tilde{A}|\, T)$ time. As we will show in the next section, this is fast enough to be practically useful. It also compares very favourably, again, with that which would be obtained with the multi-graph approach, which can easily be shown to be $\mathcal{O}\left(|C|^2\, T^2\right)$.

As in the previous section, state-elimination and dominance tests could improve the practical performance of the routine, and one can adapt the routine to the Steiner VRPTW, at the cost of increasing the running time by a factor of Q.

5.7 Computational Experiments

In this section, we report on some computational experiments. Subsection 5.7.1 explains how we created a collection of sparse graphs for use in the experiments. Subsection 5.7.2 is concerned with the construction of the multi-graph and Subsection 5.7.3 is concerned with the pricing routines.

5.7.1 Construction of sparse graphs

To begin with, we constructed ten sparse, undirected graphs, each with its own edge costs. This was done using a similar procedure to the one used in our earlier paper [88], which was specifically designed to produce graphs that resemble real life road networks. In detail, the following was done for $n \in \{50, 100, 150, 200, 250\}$ and then for $n = \{100, 200, 300, 400, 500\}$:

1. Set $\tilde{V} = \{1, \ldots, n\}$ and $\tilde{E} = \emptyset$.

2. Place the n nodes at random in a circle of radius $10\sqrt{n}$.
3. Define the set of *potential edges* $P = \{\{i,j\} : 1 \leq i < j \leq n\}$.
4. For each $\{i,j\} \in P$, let the cost c_{ij} be the Euclidean distance between the end-nodes, rounded up to the nearest integer.
5. Sort the potential edges in non-decreasing order of cost.
6. Examine each edge in P in turn. Insert it into $\tilde{E}$ if both of the following conditions are satisfied:
   (a) It does not cross one of the edges already inserted into $\tilde{E}$.
   (b) It does not form an angle of less than $60°$ with an edge that has already been inserted into $\tilde{E}$ and with which it shares an end-node.
7. If there are no isolated nodes in the resulting graph, output the graph $(\tilde{V}, \tilde{E})$. Otherwise, repeat the procedure.

The only difference between this procedure and the one used in [88] is that, in step 2, the circle is of radius $10\sqrt{n}$, rather than 100. This change was made to ensure that the average edge cost is roughly the same across all instances, which is closer to what happens in real life.

For each of the ten graphs, one node was selected at random to be the depot. For the first five graphs, with $n \in \{50, \ldots, 250\}$, each non-depot node was then given a probability of 2/3 of being required. For the other five graphs, with $n = \{100, \ldots, 500\}$, the probability was set to 1/3.

Figure 5.4 shows the graph that was obtained for $n = 200$ and a probability of 2/3. The required and non-required nodes are represented by small red and green circles, respectively, and the depot is represented by the small black square.

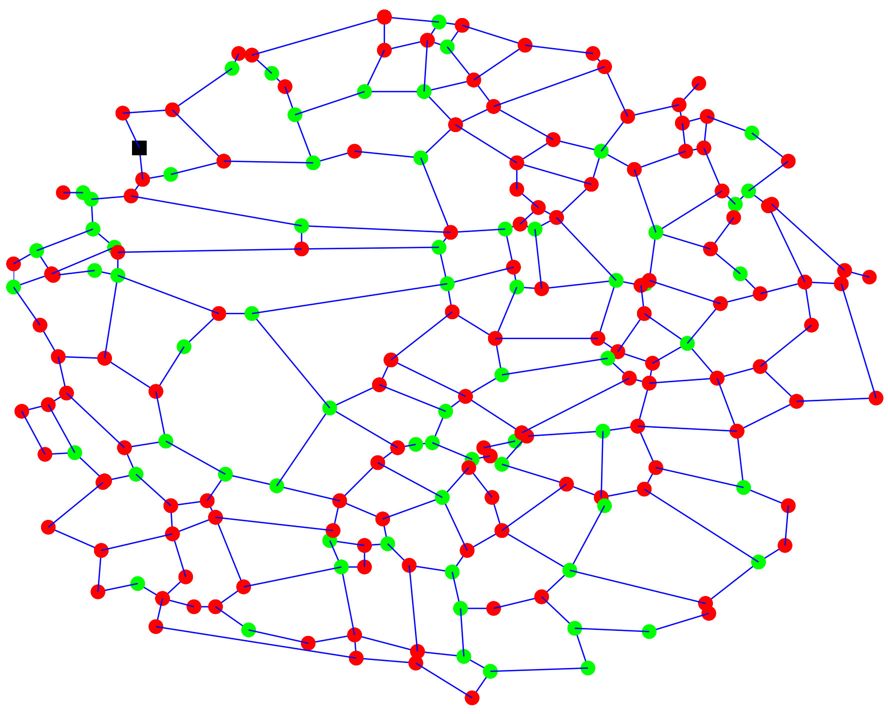


Fig. 5.4: Graph with $n = 200$ and $|C| = 133$.

5.7.2 Experiments with multi-graph construction

In our first set of experiments, we examined the complexity of multi-graph computation in practice. Along the way, we also explored the effect of *correlations* between the costs and travel times. Intuitively, one would expect the number of edges in the multi-graph to increase as the correlation between costs and times decreases. Accordingly, for each of the ten graphs we created three different sets of random travel times, with differing levels of correlation with the costs: strong correlation, weak correlation and no correlation. This was done as follows, for each instance:

- Let $\bar{c}$ be the maximum edge cost.
- For each edge $e \in \tilde{E}$, let r_e be a random number uniformly distributed between 0 and 1. Set:
  - $t_e = \lceil 0.9c_e + 0.1r\bar{c} \rceil$ for strong correlation,

- $t_e = \lceil 0.5c_e + 0.5r\bar{c} \rceil$ for weak correlation,
- $t_e = \lceil r\bar{c} \rceil$ for no correlation.

The formulae used here were intended to make the travel times of the same order of magnitude as the costs.

For each of the resulting 30 cases, we computed the corresponding multi-graph using an algorithm of our own. This consisted of calling a single-source bi-criterion shortest-path subroutine $|C|$ times, with each customer in turn taking the role of the source node. The subroutine was a straightforward dynamic programming algorithm. Although our algorithm is fairly rudimentary, its running time is $\mathcal{O}(|C|\,|\tilde{E}|\,T)$, which matches that of the best algorithms (see Section 5.4).

Table 5.1 shows the results obtained. The first three columns show the number of nodes, edges and customers in each of the ten sparse graphs. The next column shows the degree of correlation (strong, weak, none). The following two columns show the number of edges in the multi-graph, and the time taken to compute the multi-graph using our algorithm, in seconds.

We see that the number of edges in the multi-graph is rather high in all cases. As expected, it increases as the correlation decreases, and increases as $|\tilde{V}|$, $|\tilde{E}|$ and $|C|$ increase. From the way the multi-graph is constructed, one would expect the number of edges in the multi-graph to grow quadratically in $|C|$ and exponentially in $|\tilde{V}|$ and $|\tilde{E}|$. This is consistent with the results in the table.

The running times, on the other hand, are quite reasonable for these instances. This suggests that the time taken to compute the multi-graph is not an issue after all, although it could be, if the travel times were measured to a higher precision.

5.7.3 Comparison of pricing routines

Next, we wished to compare the time taken to perform pricing (with non-elementary routes permitted), when using either our algorithm (described in Section 5.6), or

$\lvert\tilde{V}\rvert$	$\lvert\tilde{E}\rvert$	$\lvert C\rvert$	Corr	$\lvert E\rvert$	Time (s)
50	69	33	S	674	0.5
			W	903	0.5
			N	1324	0.5
100	139	66	S	2850	2.2
			W	5009	2.1
			N	6892	2.2
150	215	100	S	5852	5.1
			W	12765	5.1
			N	18337	5.0
200	291	133	S	11641	9.0
			W	31770	9.0
			N	35617	9.1
250	367	166	S	18454	13.3
			W	37585	13.9
			N	56042	13.4
100	139	33	S	776	1.0
			W	1241	1.0
			N	1646	1.1
200	291	66	S	2857	4.3
			W	6096	4.2
			N	9708	4.3
300	442	100	S	7224	10.0
			W	16030	9.9
			N	23480	9.9
400	597	133	S	12946	18.1
			W	32087	18.9
			N	47359	18.7
500	744	166	S	23650	29.5
			W	60824	29.5
			N	79321	30.3

Tab. 5.1: Results on the construction of the multi-graph.

the corresponding algorithm based on the multi-graph. Since we expected the pricing time to depend on the width of the time windows, for each of the thirty combinations mentioned in the previous subsection, we created two Steiner m-TSPTW instances, with two different widths for the time windows. Each customer node $i \in C$ was assigned an integer service time $s_i \in [1, 2]$ at random. A set of routes was then constructed in a greedy way, so that each customer was serviced by exactly one vehicle. Finally, time windows $[e_i, \ell_i]$ were assigned to each customer such that the given routes were feasible and such that either $e_i + 3 \leq \ell_i \leq e_i + 4$ (for narrow time windows) or $e_i + 10 \leq \ell_i \leq e_i + 15$ (for wide time windows). The time horizon T was set to 3000 in all cases.

Table 5.2 shows the results obtained for the resulting 60 instances. The first four columns have the same meaning as before. The next four columns show the time taken, in seconds, to solve the LP relaxation of the set covering problem under various settings. The subscripts 'n' and 'w' indicate narrow and wide time windows, respectively, and the subscripts 'm' and 'o' indicate the multi-graph approach and our approach, respectively. All times include the time taken to re-optimise the LPs after adding columns, but, for these instances, the bulk of the time was spent pricing. The last two columns present the acceleration in speed gained by using our approach instead of the multi-graph approach, for both narrow and wide windows (i.e., column A_n is computed by dividing column T_{nm} by column T_{no}, and column A_w is computed by dividing column T_{wm} by column T_{wo}).

A first observation is that the pricing times are rather high in all cases. This is because we only added one column after each pricing call, namely, the one with the most negative reduced cost. One could probably reduce the number of pricing iterations, and therefore the pricing time, if one added many columns after each call. A second observation is that the pricing times tend to be higher when time windows are wide. This is probably because the cardinality of the column set Ω is

$\lvert\tilde{V}\rvert$	$\lvert\tilde{E}\rvert$	$\lvert C\rvert$	Corr	T_{nm}	T_{no}	T_{wm}	T_{wo}	A_n	A_w
50	69	33	S	8	2	10	3	4.0	3.3
			W	13	3	10	3	4.3	3.3
			N	32	6	32	7	5.3	4.6
100	139	66	S	103	23	216	29	4.5	7.4
			W	301	49	294	35	6.1	8.4
			N	504	59	518	75	8.5	6.9
150	215	100	S	974	122	760	126	8.0	6.0
			W	1405	139	2344	198	10.1	11.8
			N	7369	219	3713	208	33.6	17.9
200	291	133	S	17377	692	4854	569	25.1	8.5
			W	7777	479	11666	446	16.2	26.2
			N	14151	588	12187	620	24.1	19.7
250	367	166	S	5613	457	4753	300	12.3	15.8
			W	8415	720	10095	411	11.7	24.6
			N	4596	381	14264	538	12.1	26.5
100	139	33	S	17	10	17	9	1.7	1.9
			W	19	8	24	10	2.4	2.4
			N	28	13	37	14	2.2	2.6
200	291	66	S	348	97	398	100	3.6	4.0
			W	1043	223	697	161	4.5	4.3
			N	999	129	796	120	7.7	6.6
300	442	100	S	1008	210	695	365	4.8	1.9
			W	1875	259	1714	248	7.2	6.9
			N	3692	340	2362	272	10.9	8.7
400	597	133	S	1457	300	2462	423	4.9	5.8
			W	2055	194	4171	408	10.6	10.2
			N	11011	380	7428	436	29.0	17.0
500	744	166	S	4554	795	7803	942	5.7	8.3
			W	6643	620	12197	984	10.7	12.4
			N	23220	1775	47820	1551	13.1	30.8

Tab. 5.2: Time taken by pricing routines on Steiner m-TSPTW instances.

larger when the windows are wider.

More importantly, for our purposes, it is clear that our approach is significantly faster than the multi-graph approach. Moreover, the difference seems to grow as the size of the original graph grows. In our view, this demonstrates clearly that working on the original graph is preferable in terms of running time, as well as in terms of ease of implementation.

5.8 Discussion

Despite the vast research effort that has been put into VRPs with time windows, most of the existing exact and heuristic solution algorithms cannot be relied upon to give meaningful solutions when the original instance is defined on a road network. The approach of Garaix *et al.* [59], based on the computation of an auxiliary multi-graph, provides one way around this difficulty. In this paper, we have shown that it is possible instead to work with the original graph throughout, and that this can lead to significant savings in computing time.

We leave to future research the development of a full branch-and-price or branch-cut-and-price algorithm for the Steiner versions of the m-TSPTW or VRPTW. We remark that in order to develop such algorithms one would need to develop specialised branching rules that work on the original graph rather than the multi-graph. For a discussion of this issue in the context of the Capacitated Arc Routing Problem, see Bode & Irnich [16].

6. CONCLUSION

6.1 Summary

The vehicle routing problem and its variants are amongst the most famous combinatorial optimisation problems and have a prominent role in the Operational Research literature. Over the years they have drawn much attention from many researchers in the transportation and logistics sectors due to their complexity and their wide applicability.

The main goal of this thesis is to produce a viable solution algorithm for one or more vehicle routing problems with correlated congestion. At the preliminary stage, alternative integer programming formulations for routing problems on road networks have been proposed and their performances have been examined theoretically and computationally. Then the structure of correlation between travel speeds on roads has been studied using a real traffic data set and different correlation structures have been incorporated in the integer programming formulations that we proposed for the Steiner TSP. The outcome of this research has been four separate research articles that each focusses on modelling aspects or the solution methods of VRPs defined on the original road network graph.

In the first research article presented in chapter 2, we have derived a family of compact formulations for the TSP and some other variants of VRP, that are defined on the original road network graph. The performance of these formulations and strength of their LP relaxation has been investigated both theoretically and experimentally. It has been shown that instances with over 200 nodes can be solved on a standard branch-and-bound solver in under two hours, using the best of our formulations.

In the second article presented in chapter 3, an in-depth study has been carried out into the structure of the correlations between travel speeds on a very congested road network by performing statistical analysis on real traffic data that has been collected in the City of London. It has been shown that for this data set, there is still significant spatial correlations between speeds on roads that are up to twenty links apart and the temporal fluctuations become less correlated as the gap between time periods grows. A principal component analysis of the correlations suggests that although speeds on this congested road network have long-range correlations, the correlation structure itself is not very complex. In fact up to 89% of the variation in the temporal correlation have been explained in 3 dimensions and for spatial correlation 82% of the variation has been explained in 6 dimensions. This could be useful for data compression.

The third article presented in chapter 4, is an extension of our first paper that incorporates our findings in chapter 3 into a modelling framework for VRP. The STSPCC has been defined as a potentially useful variant of VRP that considers the TSP on a real road network where the costs are stochastic random variables with correlation. STSPCC is best thought of as a bi-objective problem, but we have used standard techniques to convert it into a single-objective problem with eight different integer programming formulations presented. It has also been shown how to account for three different correlation structures in each of the formulations. Our experimental results on a number of test instances have shown that instances of up to 100 nodes can be solved on a standard branch-and-bound solver in less than two hours.

Finally, the fourth research article presented in chapter 5 has shown how most of the exact algorithms proposed for the VRPs with time windows, might not be applicable if the original problem is defined on the road network graph. This is due to the underlying assumption of these algorithms that the cheapest path between a pair of customers is the same as the quickest path. This assumption often is not valid on a real road network. Garaix *et al.* [59] have proposed to tackle this issue by first storing alternative paths in an auxiliary multi-graph, and then using that multi-graph within a branch-and-price algorithm. We have shown that it is possible instead to work with the original graph throughout. Considering pricing routines as a key component in an exact branch-and-price solution method, we have proposed some alternative pricing routines for the algorithm that can be used to solve the problem directly on the original graph. It has been shown clearly that using this routine can lead to significant savings in computing time compared to the Garaix *et al.*'s approach.

6.2 Further Research

The transportation problems faced by many companies in real life can generally be stated as transferring goods between a number of customers at different locations by some means of transportation. Typically there are many additional restrictions such as capacity limitations associated with the use of different vehicles or drivers working hours and availability. Furthermore, there may be restrictions on presence of customers and delivery times. These problems are basically contained in CVRP and VRPTW. In current solution methods of these variants, the stochastic nature of the travel times and the correlations between them is often overlooked. A natural direction for future research would be to investigate how to handle this uncertainty and incorporate the correlations in other variants of VRP.

Furthermore, each chapter of this thesis puts forward new ideas and directions for future research. Some of these ideas are as follows:

- Considering the article presented in chapter 2, one could try a *cut-and-branch* approach, in which cutting planes are used to strengthen the initial formulation, before feeding the strengthened formulation into the branch-and-bound solver. This approach may have an affect on the performance of some or all the formulations proposed and is an interesting matter for further investigation.

- In chapter 2, we mentioned that the Steiner TSPTW can be formulated as a mixed $0-1$ linear program with $\mathcal{O}(|C||E|)$ variables and constraints, where C is the set of customers and E is the set of edges. The details are presented in the appendix. However, whether there there exists a formulation with fewer variables and constraints, is still an open question. Also adapting the formulations to other problems, e.g., problems with multiple vehicles could be a potential research area for future.

- Our analysis on the correlation structures between travel speeds on road networks in the second article presented in chapter 3 were based the traffic data for City of London. However, since correlations can depend on a number of factors such as congestion levels, the distance between road links and the road types (i.e., motorways or small city links), results may be different for a less congested area or a road network with more sparsity. A systematic characterization of road networks for this purpose and a study of correlation structures for different types could be another logical step for further research.

- An interesting extension of the third article presented in chapter 4 would be to try to develop methods for computing the entire efficient frontier, or at least approximating it from above and/or below. One could also define and explore different versions of the problem, e.g., two-stage stochastic, robust, chance-constrained or dynamic versions.

- Regarding the pricing routine that was proposed in the fourth article presented in chapter 5 for solving the VRPTW on the original road network, there is a need to develop specialised branching rules, so that the routine can be used effectively within a branch-and-price algorithm.

- Another interesting direction for future research would be to try to develop formulations and bounding techniques for vehicle routing problems where travel times are stochastic, correlated and time-varying (i.e., are random variables with probability distribution functions that vary with time).

Finally, we believe that the work presented in this thesis has has made great contributions towards modelling and solving the vehicle routing problems on real road network and potentially initiates some new ideas for future research.

7. APPENDIX

This appendix provides two sets of formulations for VRPs with time windows that can be used to solve the problem directly on the original road network.

7.1 Time-indexed Formulations

First, we adapt the time-indexed formulation for the TSPTW in [32] to the Steiner version. For each $a \in \tilde{A}$, let $\tilde{c}_a$ be the cost of traversing a and let $\tilde{t}_a$ be the time taken. We define s_i, e_i and ℓ_i as in Subsection 5.3.1, except that they are defined only for $i \in C$. Finally, we define T as in Subsection 5.3.1.

We define three sets of binary variables as follows. For $a \in \tilde{A}$ and $0 \leq t < T$, let x_a^t take the value 1 if and only if the salesman begins to traverse arc a at time t. For $i \in C$ and $e_i \leq t \leq \ell_i$, let y_i^t take the value 1 if and only if service at customer i begins at time t. For $i \in C$ and $0 < t < e_i$, let z_i^t take the value 1 if and only if the salesman arrives at node i at time t, waits until time e_i, and then begins servicing customer i.

We will also need the following additional notation. For any node $i \in \tilde{V}$, we let $\delta^+(i)$ denote the set of arcs leaving i (i.e., whose tail is i), and $\delta^-(i)$ denote the set of arcs arriving at i (i.e., whose head is i). Also, $\bar{C}$ will denote $V \setminus (C \cup \{0\})$.

Then, to solve the Steiner TSPTW, it suffices to minimise

$$\sum_{t=0}^{T} \sum_{a \in \tilde{A}} \tilde{c}_a x_a^t$$

subject to the following constraints:

$$\sum_{t=e_i}^{\ell_i} y_i^t = 1 \qquad (i \in C) \tag{7.1}$$

$$\sum_{a\in\delta^+(0)} x_a^0 = 1 \tag{7.2}$$

$$\sum_{a\in\delta^+(0)} x_a^t \le \sum_{a\in\delta^-(0)} x_a^{t-\tilde{t}_a} \qquad (0 < t < T) \tag{7.3}$$

$$\sum_{a\in\delta^+(i)} x_a^t = \sum_{a\in\delta^-(i)} x_a^{t-\tilde{t}_a} \qquad (i \in \bar{C}, 0 < t < T) \tag{7.4}$$

$$\sum_{a\in\delta^+(i)} x_a^t + z_i^t = \sum_{a\in\delta^-(i)} x_a^{t-\tilde{t}_a} \qquad (i \in C, 0 < t < e_i) \tag{7.5}$$

$$\sum_{a\in\delta^+(i)} x_a^{e_i} + y_i^{e_i} = \sum_{a\in\delta^-(i)} x_a^{e_i-\tilde{t}_a} + \sum_{t=1}^{e_i-1} z_i^t \qquad (i \in C) \tag{7.6}$$

$$\sum_{a\in\delta^+(i)} x_a^t + y_i^t = \sum_{a\in\delta^-(i)} x_a^{t-\tilde{t}_a} + y_i^{t-s_i} \qquad (i \in C, e_i < t < T) \tag{7.7}$$

$$\sum_{t=1}^{e_i-1} z_i^t \le y_i^{e_i} \qquad (i \in C) \tag{7.8}$$

$$\sum_{a\in\delta^+(i)} x_a^{t+s_i} \ge y_i^t \qquad (i \in C, e_i \le t \le \ell_i) \tag{7.9}$$

$$x_a^t \in \{0,1\} \qquad (a \in \tilde{A}, 0 \le t < T) \tag{7.10}$$

$$y_i^t \in \{0,1\} \qquad (i \in C; e_i \le t \le \ell_i) \tag{7.11}$$

$$z_i^t \in \{0,1\} \qquad (i \in C; 0 < t < e_i). \tag{7.12}$$

The equations (7.1) ensure that every customer is serviced within its time window. The constraint (7.2) ensures that the salesman departs from the depot at time 0. The constraints (7.3) permit the salesman to pass through the depot node any number of times, but ensure that the route eventually ends at the depot. The equations (7.4) ensure that, if the salesman arrives at a node that is neither the depot nor a customer, then he departs immediately. The equations (7.5)–(7.7) ensure that, if the salesman arrives at a customer node, then he either waits, or services, or departs immediately. (In constraints (7.7), y variables are to be included on the left-hand side only if they are defined for the given value of t.) The constraints (7.8) ensure that, if the salesman is waiting at a customer node, then he goes on to service that customer at the start of its time window. The constraints (7.9) ensure that, when the salesman has finished servicing a customer, he departs immediately. Finally, constraints (7.10)–(7.12) are the binary conditions.

This time-indexed formulation of the Steiner TSPTW has $\mathcal{O}(|\tilde{A}|\,T)$ variables and constraints. Although its size is pseudo-polynomial (rather than polynomial). When $\tilde{G}$ is sparse and T is fairly small, one can solve small- to medium- instances to proven optimality using a standard branch-and-bound solver.

To adapt the time-indexed formulation to the Steiner m-TSPTW, the most obvious way is to create m copies of each variable, one for each vehicle. Suppose the variables are labelled x^t_{ak}, y^t_{ik} and z^t_{ik}, for $k = 1, \dots, m$. Then, one must make three changes. First, change the objective function to:

$$\sum_{k=1}^{m} \sum_{t=0}^{T} \sum_{a \in \tilde{A}} \tilde{c}_a x^t_{ak}.$$

Second, change the equations (7.1) to:

$$\sum_{k=1}^{m} \sum_{t=e_i}^{\ell_i} y^t_{ik} = 1 \qquad (i \in C).$$

Third, create one copy of the constraints (7.2)–(7.12) for each vehicle. We will call the resulting formulation the *three-index* formulation.

Note that the three-index formulation has $\mathcal{O}(m|\tilde{A}|T)$ variables and constraints. This is so large that the formulation is unlikely to be of practical use.

However, there is a more economical way to adapt the time-indexed formulation of the Steiner TSPTW to the Steiner m-TSPTW: simply change the right-hand side of the constraint (7.2) to m, and permit the x variables to be general integers, rather than binary. This keeps the number of variables and constraints at $\mathcal{O}(|\tilde{A}|T)$. We will call this the *aggregated* formulation of the Steiner m-TSPTW.

These two formulations of the Steiner m-TSPTW can be adapted to the Steiner VRPTW in various ways. For example, to adapt the aggregated formulation, one could:

- Add $2^{|C|} - 1$ constraints, to ensure that, for each non-empty set $S \subseteq C$, at least $\left\lceil \sum_{i \in S} q_i / Q \right\rceil$ vehicles traverse the arcs that connect S to $\tilde{V} \setminus S$.

- Add $\mathcal{O}(|\tilde{A}|\,T)$ variables, representing the total load carried along each arc at each time step, and $\mathcal{O}(|\tilde{A}|\,T)$ constraints, ensuring that each customer receives q_i units and that the vehicle capacity Q is never exceeded at any time.

- Add a 'load' index to the variables, so that, for example, x_a^{tq} represents the number of vehicles that begin to traverse arc a at time t while carrying a load of q. This leads to a huge formulation, with $\mathcal{O}(|\tilde{A}|\,T\,Q)$ variables and constraints.

We omit further details, for the sake of brevity.

7.2 Compact Formulations

The formulation given in the previous section, though smaller than the one of Geraix *et al.* [59], is still of exponential size. In this section, we present formulations of polynomial size.

We begin with the TSPTW. For every $a \in A$ and $k = 0, \dots, n_R$, let the binary variable $\tilde{x}_a^k$ take the value 1 if and only if the salesman traverses arc a after having serviced exactly k customers so far. Also let g_a^k be a non-negative continuous variable representing the total time that has elapsed when the salesman starts to traverse arc a, having exactly serviced k customers, or 0 if no such traversal occurs. Finally, for all $i \in V_R$ and $k = 1, \dots, n_R$, let the binary variable y_i^k take the value 1 if and only if customer i is the kth customer to be serviced.

The objective function is simply:

$$\min \sum_{k=0}^{n_R} \sum_{a \in A} c_a \tilde{x}_a^k.$$

To ensure that each required node is serviced exactly once, we have the following constraints:

$$\begin{aligned} \textstyle\sum_{k=1}^{n_R} y_i^k &= 1 \quad (\forall i \in V_R) \\ \textstyle\sum_{i \in V_R} y_i^k &= 1 \quad (k = 1, \ldots, n_R). \end{aligned}$$

To ensure that the vehicle departs from and returns to the depot a correct number of times, we have:

$$\begin{aligned} \textstyle\sum_{a \in \delta^+(1)} \tilde{x}_a^0 &= 1 \\ \textstyle\sum_{a \in \delta^-(1)} \tilde{x}_a^k &= \textstyle\sum_{a \in \delta^+(1)} \tilde{x}_a^k \quad (\forall k = 1, \ldots, n_R - 1) \\ \textstyle\sum_{a \in \delta^-(1)} \tilde{x}_a^{n_R} &= 1. \end{aligned}$$

Then, to ensure that the vehicle departs from each non-depot node as many times as it arrives, we have:

$$\begin{aligned} \textstyle\sum_{a \in \delta^-(i)} \tilde{x}_a^0 &= y_i^1 + \textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a^0 && (\forall i \in V_R) \\ y_i^k + \textstyle\sum_{a \in \delta^-(i)} \tilde{x}_a^k &= y_i^{k+1} + \textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a^k && (\forall i \in V_R,\ k = 1, \ldots, n_R - 1) \\ y_i^{n_R} + \textstyle\sum_{a \in \delta^-(i)} \tilde{x}_a^{n_R} &= \textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a^{n_R} && (\forall i \in V_R) \\ \textstyle\sum_{a \in \delta^-(i)} \tilde{x}_a^k &= \textstyle\sum_{a \in \delta^+(i)} \tilde{x}_a^k && (\forall i \in V \setminus V_R \cup \{1\},\ k = 0, \ldots, n_R). \end{aligned}$$

Next, to ensure that the g_a^k variables take the value that they should, we add the following constraint for $i \in V_R$ and for $k = 0, \ldots, n_R - 1$:

$$\begin{aligned} \textstyle\sum_{a \in \delta^+(i)} g_a^{k+1} + T(1 - y_i^{k+1}) &\geq \textstyle\sum_{a \in \delta^-(i)} g_a^k + \textstyle\sum_{a \in \delta^-(i)} t_a \tilde{x}_a^k + s_i y_i^{k+1} \\ \textstyle\sum_{a \in \delta^+(i)} g_a^k + T y_i^{k+1} &\geq \textstyle\sum_{a \in \delta^-(i)} g_a^k + \textstyle\sum_{a \in \delta^-(i)} t_a \tilde{x}_a^k. \end{aligned}$$

the following constraints for $i \in V_R$:

$$\sum_{a \in \delta^+(i)} g_a^{n_R} \geq \sum_{a \in \delta^-(i)} g_a^{n_R} + \sum_{a \in \delta^-(i)} t_a \tilde{x}_a^{n_R}.$$

and the following constraint for $i \in V \setminus V_R$ and for $k = 0, \ldots, n_R$:

$$\sum_{a \in \delta^+(i)} g_a^k \geq \sum_{a \in \delta^-(i)} g_a^k + \sum_{a \in \delta^-(i)} t_a \tilde{x}_a^k.$$

Moreover, to ensure that the time windows are obeyed, we add the following constraints:

$$\sum_{a\in\delta^+(i)} g_a^k \geq (a_i + s_i)y_i^k \qquad (\forall i \in V_R, k = 1,\ldots,n_R)$$

$$\sum_{a\in\delta^-(i)} g_a^{k-1} + \sum_{a\in\delta^-(i)} t_a\tilde{x}_a^{k-1} \leq T - (T - b_i)y_i^k \quad (\forall i \in V_R,\, k = 1,\ldots,n_R).$$

Finally, we have the trivial constraints:

$$\tilde{x}_a^k \in \{0,1\} \qquad (\forall a \in A,\, k = 0,\ldots,n_R)$$

$$y_i^k \in \{0,1\} \qquad (\forall i \in V_R,\, k = 1,\ldots,n_R)$$

$$0 \leq g_a^k \leq T\tilde{x}_a^k \quad (\forall a \in A,\, k = 0,\ldots,n_R).$$

This formulation has $\mathcal{O}(|C||E|)$ variables and constraints, and is therefore polynomial in size. It can be adapted to the m-TSPTW by using an additional index. That is, x_{aj}^k takes the value 1 if and only if vehicle j traverses arc a after having serviced k customers, y_{ij}^k takes the value 1 if and only if customer i is the kth customer serviced by vehicle j, and g_{aj}^k is the total time that has elapsed when vehicle j starts to traverse arc a, having serviced exactly k customers, or 0 if no such traversal occurs. The number of variables and constraints would increase to $\mathcal{O}(m|C||E|)$. We omit the details for brevity.

We leave the existence of a significantly smaller compact formulation as an open question.

BIBLIOGRAPHY

[1] W. Adams and H. Sherali. A tight linearisation and an algorithm for zero-one quadratic programming problems. *Management Science*, 32:1274–1290, 1986.

[2] T. Anderson. *An Introduction to Multivariate Statistical Analysis.* Wiley, New York.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. *The Travelling Salesman Problem: A Computational Study.* Princeton NJ:, Princeton University Press, 2006.

[4] N. Ascheuer, M. Fischetti, and M. Grötschel. Solving the asymmetric travelling salesman problem with time windows by branch-and-cut. *Math. Program.*, 90:475–506, 2001.

[5] M. Asif, C. Goh, A. Fathi, M. Xu, M. Dhanya, N. Mitrovic, and P. Jaillet. Spatial and temporal patterns in large-scale traffic speed prediction. IEEE Intelligent Transportation Systems Conference, 2012.

[6] E. Baker. An exact algorithm for the time-constrained travelling salesman problem. *Oper. Res.*, 31:938–945, 1983.

[7] E. Balas. The prize collecting travelling salesman problem. *Networks*, 19:621–636, 1989.

[8] E. Balas. *The Travelling Salesman Problem and its Variations*, chapter The prize collecting travelling salesman problem and its applications. In Gutin and Punnen [67], 2002.

[9] R. Baldacci, A. Mingozzi, and R. Roberti. New route relaxation and pricing strategies for the vehicle routing problem. *Oper. Res.*, 59:1269–1283, 2011.

[10] R. Baldacci, A. Mingozzi, and R. Roberti. New state-space relaxations for solving the travelling salesman problem with time windows. *INFORMS J. Comput.*, 24:356–371, 2012.

[11] M. Balinski and R. Quandt. On an integer program for a delivery problem. *Oper. Res.*, 12:300–304, 1964.

[12] M. Ball, T. Magnanti, C. Monma, and G. Nemhauser. Network routing. *Handbooks in Operations Research and Management Science*, 8, 1995.

[13] E. Beale. On quadratic programming. *Nav. Res. Log. Quart.*, 6:227–243, 1959.

[14] R. Bellman. The theory of dynamic programming. *Bulletin of the American Mathematical Society*, 60:503–516, 1954.

[15] G. Berbeglia, J. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP*, 15:1–31, 2007.

[16] C. Bode and S. Irnich. Cut-first branch-and-price-second for the capacitated arc-routing problem. *Oper. Res.*, 60:1167–1182, 2012.

[17] A. Chabrier. Vehicle routing problem with elementary shortest path based column generation. *Comput. & Oper. Res.*, 33:2972–2990, 2006.

[18] G. Chamberlain and M. Rothschild. Arbitrage, factor structure and mean-variance analysis in large asset markets. *Econometrica*, 51:1305–1324, 1983.

[19] S. Chaudhuri, M. Drton, and T. Richardson. Estimation of a covariance matrix with zeros. *Biometrika*, 94:199–216, 2007.

[20] A. Chen, H. Yang, H. Lo, and W. Tang. Capacity reliability of a road network: an assessment methodology and numerical results. *Transportation Research Part B: Methodological*, 36:225–252, 2002.

[21] B. Chen, W. Lam, A. Sumalee, Q. Li, H. Shao, and Z. Fang. Finding reliable shortest paths in road networks under uncertainty. *Networks and Spatial Economics*, 13:123–148, 2013.

[22] B. Chen, W. Lam, A. Sumalee, and Z. Li. Reliable shortest path finding in stochastic networks with spatial correlated link travel times. *International Journal of Geographical Information Sciences*, 26:365–386, 2012.

[23] T. Cheng, J. Haworth, and J. Wang. Spatio-temporal autocorrelation of road network data. *Journal of Geographical Systems*, 14:389–413, 2012.

[24] A. Claus. A new formulation for the travelling salesman problem. *SIAM J. Alg. Discr. Meth.*, 5:21–25, 1984.

[25] E. Coffman, J. Csirik, G. Galambos, S. Martello, and D. Vigo. Bin packing approximation algorithms: survey and classification. In D. Du, P. Pardalos, and R. Graham, editors, *Handbook of Combinatorial Optimization.* Springer, New York, 2013.

[26] A. Corberán, A. Letchford, and J. Sanchis. A cutting plane algorithm for the general routing problem. *Math. Program.*, 90:291–316, 2001.

[27] J.-F. Cordeau, G. Desaulniers, J. Desrosiers, M. Solomon, and F. Soumis. *The Vehicle Routing Problem*, chapter VRP with time windows. In Toth and Vigo [124], 2002.

[28] G. Cornuéjols, J. Fonlupt, and D. Naddef. The travelling salesman problem on a graph and some related integer polyhedra. *Math. Program.*, 33:1–27, 1985.

[29] G. Dantzig, D. Fulkerson, and S. Johnson. Solution of a large-scale travelling-salesman problem. *Oper. Res.*, 2:363–410, 1954.

[30] G. Dantzig and J. Ramser. The truck dispatching problem. *Management Science.*, 6:80–91, 1959.

[31] G. Dantzig and P. Wolfe. Decomposition principle for linear programs. *Oper. Res.*, 8:101–111, 1960.

[32] S. Dash, O. Günlk, A. Lodi, and A. Tramontani. A time bucket formulation for the traveling salesman problem with time windows. *INFORMS J. Comput.*, 24:132–147, 2012.

[33] D. Davendra. *Travelling Salesman Problem, Theory and Applications.* In-Tech, Croatia, 2010.

[34] M. de Aragão, H. Longo, and E. Uchoa. Solving capacitated arc routing problem using a transformation to the cvrp. *Comput. & Oper. Res.*, 33:1823–1837, 2006.

[35] G. Desaulniers, F. Lessard, and A. Hadjar. Tabu search, partial elementarity, and generalized k-path inequalities for the vehicle with time windows. *Transp. Sci.*, 42:387–404, 2008.

[36] M. Desrochers. Vehicle routing with time windows: optimization and approximation. In B. Golden and A. Assad, editors, *Vehicle Routing: Methods and Studies.* Elsevier, North Holland, 1988.

[37] M. Desrochers, J. Desrosiers, and M. Solomon. A new optimization algorithm for the vehicle routing problem with time windows. *Oper. Res.*, 40:342–354, 1992.

[38] J. Desrosiers, Y. Dumas, M. Solomon, and F. Soumis. Time-constrained routing and scheduling. In M. Ball, T. Magnanti, C. Monma, and G. Nemhauser, editors, *Network routing*. 1995.

[39] J. Desrosiers, F. Soumis, and M. Desrochers. Routing with time windows by column generation. *Networks*, 14:545–565, 1984.

[40] E. Dijkstra. A note on two problems in connection with graphs. *Numerical Mathematics*, 1:269–271, 1959.

[41] W. Dong, M. Li, Q. Vo, and H. Vu. Reliability in stochastic time-dependent traffic networks with correlated link travel times. *International IEEE Conference on Intelligent Transportation Systems*, pages 16–19, 2012.

[42] M. Dror. Note on the complexity of the shortest path models for column generation in vrptw. *Oper. Res.*, 42:977–979, 1994.

[43] M. Dror, editor. *Arc Routing: Theory, Solutions and Applications.* Kluwer Academic Publishers, New York, 2000.

[44] M. Dror, G. Laporte, and P. Trudeau. Vehicle routing with stochastic demands: properties and solutions frameworks. *Transp. Sci.*, 23:166–176, 1989.

[45] Y. Dumas, J. Desrosiers, E. Gélinas, and M. Solomon. An optimal algorithm for the travelling salesman problem with time windows. *Oper. Res.*, 43:367–371, 1995.

[46] M. Ehrgott and X. Gandibleux. Bound sets for biobjective combinatorial optimization problems. *Comput. Oper. Res.*, 34:2674–2694, 2007.

[47] EnviromentalProtectionUK. Car pollution. `http://www.environmental-protection.org.uk/committees/air-quality/air-pollution-and-transport/car-pollution`. Accessed: 2014-06-02.

[48] L. Escudero. An inexact algorithm for the sequential ordering problem. *Eur. J. Oper. Res.*, 37:232–253, 1988.

[49] Y. Fan, R. Kalaba, and J. Moore. Shortest paths in stochastic networks with correlated link costs. *Computers and Mathematics with Applications*, 49:1549–1564, 2005.

[50] D. Feillet, P. Dejax, and M. Gendreau. Travelling salesman problems with profits. *Transp. Sci.*, 39:188–205, 2005.

[51] D. Feillet, P. Dejax, M. Gendreau, and C. Gueguen. An exact algorithm for the elementary shortest path problem with resource constraints: application to some vehicle routing problems. *Networks*, 44:216–229, 2004.

[52] M. Fischetti, J.-J. Salazar-González, and P. Toth. *The Travelling Salesman Problem and its Variations*, chapter The generalized travelling salesman problem and orienteering problems. In Gutin and Punnen [67], 2002.

[53] B. Fleischmann. A cutting plane procedure for the travelling salesman problem on a road network. *Eur. J. Opl Res.*, 21:307–317, 1985.

[54] B. Fleischmann and S. Gnutzmann. Dynamic vehicle routing based on online traffic information. *Transp. Sci.*, 38:420–433, 2004.

[55] L. Ford and D. Fulkerson. Maximal flow through a network. *Canad. J. Math.*, 8:399–404, 1956.

[56] R. Fortet. L'algèbre de boole et ses applications en recherche opérationnelle. *Cahiers Centre Etudes Rech. Oper.*, 4:5–36, 1959.

[57] R. Fukasawa, H. Longo, J. Lysgaard, M. P. de Arago, M. Reis, E. Uchoa, and R. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Math. Program. Series A*, 106:491–511, 2006.

[58] D. Gale. A theorem of flows in networks. *Pacific J. Math.*, 7:1073–1082, 1957.

[59] T. Garaix, C. Artigues, D. Feillet, and D. Josselin. Vehicle routing problems with alternative paths: An application to on-demand transportation. *Eur. J. Oper. Res.*, 204:62–75, 2010.

[60] B. Gavish and S. Graves. The travelling salesman problem and related problems. Working paper, Operations Research Centre, Massachusetts Institute of Technology, 1978.

[61] M. Gendreau, G. Laporte, and R. Seguin. Stochastic vehicle routing. *Eur. J. Oper. Res.*, 88:3–12, 1996.

[62] B. Golden and A. Assad. *Vehicle routing: methods and studies.* North-Holland, Amsterdam, 1988.

[63] B. Golden and A. Assad. *Introduction to linear algebra.* Wellesley–Cambridge Press, Wellesley MA, 2009.

[64] B. Golden, L. Levy, and R. Vohra. The orienteering problem. *Nav. Res. Log.*, 34:307–318, 1987.

[65] B. Golden, S. Raghavan, and E. Wasil. *Series in Operations Research/Computer Science Interfaces*, chapter The vehicle routing problem: latest advances and new challenges. Springer, Berlin, 2008.

[66] L. Gouveia and S. Voss. A classification of formulations of the (time-dependent) travelling salesman problem. *Eur. J. Oper. Res.*, 83:69–82, 1995.

[67] G. Gutin and A. Punnen, editors. *The Travelling Salesman Problem and its Variations.* Kluwer, Dortrecht, 2002.

[68] P. Hansen. Bicriterion path problems. In G. F. . T. Gal, editor, *Multiple Criteria Decision Making: Theory and Applications. Lectures Notes in Economics and Mathematical Systems.* Springer, Heidelberg, 1980.

[69] W. Hardgrave and G. Nemhauser. On the relation between the travelling salesman and longest path problems. *Oper. Res.*, 10:647–657, 1962.

[70] C. Hierholzer. Über die möglichkeit, einen linienzug ohne wierholung und ohne unterbrechung zu umfahren. *Mathematische Annalen*, 6:30–32, 1873.

[71] A. Hoffman. Some recent applications of the theory of linear inequalities to extremal combinatorial analysis. *Proc. Sympos. Appl. Math.*, 10:113–128, 1960.

[72] D. Houck, J. Picard, and R. Vemuganti. The travelling salesman problem as a shortest path problem: Theory and computational experience. *Opsearch*, 17:93–109, 1980.

[73] S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for $k \geq 3$. *INFORMS J. Comput.*, 18:391–406, 2006.

[74] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. Subset-row inequalities applied to the vehicle-routing problem with time windows. *Oper. Res.*, 56:497–511, 2008.

[75] M. Jepsen, B. Petersen, S. Spoorendonk, and D. Pisinger. A branch-and-cut algorithm for the capacitated profitable tour problem. Working paper, Technical University of Denmark, 2012.

[76] H. Jula, M. Dessouky, and P. Ioannu. Truck route planning in non-stationary stochastic networks with time-windows at customer locations. *IEEE Trans. Intell. Transp. Syst.*, 37:51–63, 2006.

[77] B. Kallehauge, J. Larsen, and O. Madsen. Lagrangian duality applied to the vehicle routing problem with time windows. *Comp. & Oper. Res.*, 33:1464–1487, 2006.

[78] E. Kao. A preference order dynamic program for a stochastic traveling salesman problem. *Oper. Res.*, 26:1033–1045, 1978.

[79] A. Kenyon and D. Morton. Stochastic vehicle routing with random travel times. *Transp. Sci.*, 37:69–82, 2003.

[80] N. Kohl, J. Desrosiers, O. Madsen, M. Solomon, and F. Soumis. 2-path cuts for the vehicle routing problem with time windows. *Transp. Sci.*, 33:101–116, 1999.

[81] S. Kumar and R. Panneerselvam. A survey on the vehicle routing problem and its variants. *Intelligent Information Management*, 4:66–74, 2012.

[82] A. Land and A. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28:497–520, 1960.

[83] G. Laporte, F. Louveaux, and H. Mercure. The vehicle routing problem with stochastic travel times. *Transp. Sci.*, 26:161–169, 1992.

[84] G. Laporte and Y. Nobert. A branch and bound algorithm for the capacitated vehicle routing problem. *OR Spektrum*, 5:77–85, 1983.

[85] E. Lawler, J. Lenstra, A. R. Kan, and D. Shmoys. *The Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization.* Wiley, Chichester, 1985.

[86] T. Leipälä. On the solutions of stochastic travelling salesman problems. *Eur. J. Oper. Res.*, 2:291–297, 1977.

[87] A. Letchford, S. Nasiri, and A. Oukil. Pricing routines for vehicle routing with time windows on road networks. *Comput. & Oper. Res.*, 51:331–337, 2014.

[88] A. Letchford, S. Nasiri, and D. Theis. Compact formulations of the steiner travelling salesman problem and related problems. *Eur. J. Oper. Res.*, 228:83–92, 2013.

[89] A. Letchford and A. Oukil. Exploiting sparsity in pricing routines for the capacitated arc routing problem. *Comput. & Oper. Res.*, 36:2320–2327, 2009.

[90] X. Li, P. Tiana, and S. Leung. Vehicle routing problems with time windows and stochastic travel and service times: models and algorithm. *Int. J. Prod. Econ.*, 125:137–145, 2010.

[91] J. Lysgaard. Reachability cuts for the vehicle routing problem with time windows. *Eur. J. Oper. Res.*, 175:210–223, 2006.

[92] J. Lysgaard, A. Letchford, and R. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Math. Program.*, 100:423–445, 2004.

[93] K. Mardia, J. Kent, and J. Bibby. *Multivariate analysis.* Academic Press, 1979.

[94] H. Markowitz. Portfolio selection. *J. Finance*, 7:77–91, 1952.

[95] G. Marsaglia. Choosing a point from the surface of a sphere. *Ann. Math. Stat.*, 43:645–646, 1972.

[96] G. Mavrotas and D. Diakoulaki. A branch and bound algorithm for mixed zero-one multiple objective linear programming. *Eur. J. Oper. Res.*, 107:530–541, 1998.

[97] K. Miettinen. *International Series in Operations Research & Management Science*, chapter Nonlinear Multiobjective Optimization. Springer, New York, 1999.

[98] P. Miliotis, G. Laporte, and Y. Nobert. Computational comparison of two methods for finding the shortest complete cycle or circuit in a graph. *RAIRO*, 15:233–239, 1981.

[99] C. Miller, A. Tucker, and R. Zemlin. Integer programming formulations and travelling salesman problems. *J. Ass. Comp. Mach.*, 7:326–329, 1960.

[100] W. Min and L. Wynter. Real-time road traffic prediction with spatio-temporal correlations. *Transportation Research Part C*, 19:606–616, 2011.

[101] A. Mingozzi, L. Bianco, and S. Ricciardelli. Dynamic programming strategies for the travelling salesman problem with time windows and precedence constraints. *Oper. Res.*, 45:365–377, 1997.

[102] M. Muller. A note on a method for generating points uniformly on n-dimensional spheres. *Comm. Assoc. Comput. Mach.*, 2:19–20, 1959.

[103] D. Naddef. *The Travelling Salesman Problem and its Variations*, chapter Polyhedral theory and branch-and-cut algorithms for the symmetric TSP. In Gutin and Punnen [67], 2002.

[104] S. Nasiri. Correlation between speeds on a congested road network in the city of london. *Submitted for publication*, 2014.

[105] T. Öncan, I. Altinel, and G. Laporte. A comparative analysis of several asymmetric travelling salesman problem formulations. *Comput. & Oper. Res.*, 36:637–654, 2009.

[106] C. Orloff. A fundamental problem in vehicle routing. *Networks*, 4:35–64, 1974.

[107] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric travelling salesman problems. *SIAM Rev.*, 33:60–100, 1991.

[108] M. Padberg and T. Sung. An analytical comparison of different formulations of the travelling salesman problem. *Math. Program.*, 52:315–357, 1991.

[109] C. Papadimitriou. The euclidean travelling salesman problem is np-complete. *Theoretical Computer Science*, 4:237244, 1977.

[110] A. Pepin, G. Desaulniers, A. Hertz, and D. Huisman. A comparison of five heuristics for the multiple depot vehicle scheduling problem. *Journal of Scheduling*, 12:17–30, 2009.

[111] G. Pesant, M. Gendreau, J. Potvin, and J. Rousseau. An exact constraint logic programming algorithm for the travelling salesman problem with time windows. *Transp. Sci.*, 32:12–29, 1999.

[112] A. Raith and M. Ehrgott. A comparison of solution strategies for biobjective shortest path problems. *Comput. & Oper. Res.*, 36:1299–1331, 2009.

[113] S. Samaranayake, S. Blandin, and A. Bayen. A tractable class of algorithms for reliable routing in stochastic networks. *Procedia Social and Behavioural Sciences.*

[114] M. Savelsbergh. Local search in routing problems with time windows. *Ann. Oper. Res.*, 4:285–305, 1985.

[115] N. Secomandi. Analysis of a rollout approach to sequencing problems with stochastic routing applications. *J. Heuristics*, 9:321–352, 2003.

[116] H. Sherali and W. Adams. *A reformulation-linearisation technique for solving discrete and continuous nonconvex problems.* Kluwer, Dordrecht, 1998.

[117] B. Smith, B. Williams, and R. Oswald. Comparison of parametric and non parametric models for traffic flow forecasting. *Transportation Research Part C: Emerging Technologies*, 10:303–321, 2002.

[118] M. Solomon. Algorithms for vehicle routing and scheduling problems with time window constraints. *Oper. Res.*, 35:254–265, 1987.

[119] F. Sourd and O. Spanjaard. A multiobjective branch-and-bound framework: application to the biobjective spanning tree problem. *INFORMS J. Comput.*, 20:472–484, 2008.

[120] F. Stathopoulos and R. Noland. Induced travel and emissions from traffic flow improvements projects. *Transportation Research Record: Journal of the Transportation Research Board*, 1842:57–63, 2003.

[121] S. Teng, H. Ong, and H. Huang. An integer l-shaped algorithm for time-constrained traveling salesman problem with stochastic travel and service times. *Asia-Pacific J. Oper. Res.*, 21:241–257, 2004.

[122] F. Tillman. The multiple terminal delivery problem with probabilistic demands. *Transp. Sci.*, 3:192–204, 1969.

[123] A. Toriello, W. Haskell, and M. Poremba. A dynamic travelling salesman problem with stochastic arc costs. `http://www.optimization-online.org/DB\_FILE/2012/09/3593.pdf`, 2012. Technical report, Optimization online.

[124] P. Toth and D. Vigo, editors. *The Vehicle Routing Problem.* SIAM Publications, Philadelphia, 2002.

[125] E. Ulungu and J. Teghem. The two phases method: An efficient procedure to solve bi-objective combinatorial optimization problems. *Foundations of Computing & Decision Sciences*, 20:149–165, 1995.

[126] S. Vajda. *Mathematical Programming.* Addison-Wesley, London, 1961.

[127] J. Whittaker. *Graphical models in applied multivariate statistics*. John Wiley & Sons, Hoboken, 1990.

[128] P. Wolfe. The simplex method for quadratic programming. *Econometrika*, 27:382–398, 1959.

[129] T. Xing and X. Zhou. Finding the most reliable path with and without link travel time correlation: A lagrangian substitution based approach. *Transportation Research Part B*, 45:1660–1679, 2011.

[130] H. Yin, S. Wong, J. Xu, and C. Wong. Urban traffic flow prediction using a fuzzy-neural approach. *Transportation Research Part C: Emerging Technologies*, 10:85–98, 2002.