Journal of Scheduling (2025) 28:269–287
https://doi.org/10.1007/s10951-024-00828-7

Recoverable robust single machine scheduling with polyhedral uncertainty

Matthew Bold[1] · **Marc Goerigk**[2]

Accepted: 1 October 2024 / Published online: 19 December 2024

Abstract

This paper considers a recoverable robust single-machine scheduling problem under polyhedral uncertainty with the objective of minimising the total flow time. In this setting, a decision-maker must determine a first-stage schedule subject to the uncertain job processing times. Then following the realisation of these processing times, they have the option to swap the positions of up to Δ disjoint pairs of jobs to obtain a second-stage schedule. We first formulate this scheduling problem using a general recoverable robust framework, before we examine the incremental subproblem in further detail. We prove a general result for max-weight matching problems, showing that for edge weights of a specific form, the matching polytope can be fully characterised by polynomially many constraints. We use this result to derive a matching-based compact formulation for the full problem. Further analysis of the incremental problem leads to an additional assignment-based compact formulation. Computational results on budgeted uncertainty sets compare the relative strengths of the three compact models we propose.

Keywords Scheduling · Robust optimization · Recoverable robustness · Polyhedral uncertainty · Budgeted uncertainty

1 Introduction

We consider a scheduling problem where n jobs must be scheduled on a single machine without preemption, such that the total flow time, i.e. the sum of completion times, is minimised. This problem is denoted as $1||\sum C_i$ under the $\alpha|\beta|\gamma$ scheduling problem notation introduced by Graham et al. (1979). In practice, job processing times are often subject to uncertainty, and when this is the case it is important to find robust solutions that account for this uncertainty. In this paper, we propose a recoverable robust approach (Liebchen et al., 2009) to this uncertain single machine scheduling problem. In this recoverable robust setting, we determine a full solution in a first-stage, before an adversarial player chooses a worst-case scenario of processing times from an uncertainty set, and then in response to this, we allow the first-stage solution to be adjusted in a limited way.

The deterministic single machine scheduling problem (SMSP) is one of the simplest and most studied scheduling problems, and can be solved easily in $O(n \log n)$ time by ordering the jobs according to non-decreasing processing times, i.e. by using the shortest processing time (SPT) rule. However, despite the simplicity of the nominal problem, the robust problem has been shown to be NP-hard for even the most basic uncertainty sets (Daniels & Kouvelis, 1995).

In fact, the majority of research to date regarding robust single machine scheduling has been concerned with the presentation of complexity results for a number of different SMSPs. First discussed by Daniels and Kouvelis (1995), Kouvelis and Yu (1997) and Yang and Yu (2002), these papers study the problem with the total flow time objective, and show that it is NP-hard even in the case of two discrete scenarios, for min-max, regret and relative regret robustness. Robust single machine scheduling for discrete uncertain scenarios has been examined extensively. Aloulou and Della Croce (2008) present algorithmic and complexity results for a number of different SMSPs under min-max robustness. Aissi et al. (2011) show that the problem of minimising the number of late jobs in the worst-case scenario, where processing times are known, but due dates are uncertain is NP-hard. Zhao

✉ Matthew Bold
boldmatthew@gmail.com

Marc Goerigk
marc.goerigk@uni-passau.de

[1] STOR-i Centre for Doctoral Training, Lancaster University, Lancaster, UK

[2] Business Decisions and Data Science, University of Passau, Passau, Germany

and Zhao (2010) consider the objective of minimising the weighted sum of completion times in the worst-case scenario, and propose a cutting-plane algorithm to solve the problem. Mastrolilli et al. (2013) study this same problem and show that no polynomial-time approximation scheme exist for the unweighted version. Kasperski and Zieliński (2016) apply the ordered weighted averaging (OWA) criterion, of which classical robustness is a special case, to a number of different SMSPs under discrete uncertainty. The consideration of SMSPs under novel optimality criteria has been continued most recently by Kasperski and Zieliński (2019), where a number of complexity results are presented for the SMSP with the value at risk (VaR) and conditional value at risk (CVaR) criteria.

Robust single machine scheduling in the context of interval uncertainty has also received considerable attention. Daniels and Kouvelis (1995) address interval uncertainty, and describe some dominance relations between the jobs in an optimal schedule based on their processing time intervals. Kasperski (2005) considers an SMSP with precedence constraints, and where the regret of the maximum lateness of a job is to be minimised. A polynomial-time algorithm is presented. Lebedev and Averbakh (2006) show that the SMSP with the total flow time objective is NP-hard in the case of regret robustness. Montemanni (2007) presents a mixed-integer program (MIP) for this same problem, and uses it to solve instances involving up to 45 jobs. Kasperski and Zieliński (2008) also consider this problem, and show that it is 2-approximable when the corresponding deterministic problem is polynomially solvable. Lu et al. (2012) present an SMSP with uncertain job processing and setup times, show this problem is NP-hard, and design a simulated annealing-based algorithm to solve larger instances. Chang et al. (2017) apply distributional robustness to an SMSP, and make use of information about the mean and covariance of the job processing times to minimise the worst-case CVaR. Most recently, Fridman et al. (2020) consider an SMSP with uncertain job processing times and develop polynomial algorithms for solving the min-max regret problem under certain classes of cost functions. For a survey of robust single-machine scheduling in the context of both discrete and interval uncertainty, see Kasperski and Zielinski (2014).

A criticism of classical robustness is that the solutions it provides are overly conservative and hedge against extreme worst-case scenarios that are very unlikely to occur in practice. To reduce the level of conservatism, a restriction to interval uncertainty was introduced by Bertsimas and Sim (2004), known as budgeted uncertainty, in which the number of jobs that can simultaneously achieve their worst-case processing times is restricted. Budgeted uncertainty is a special case of the general compact polyhedral uncertainty that is considered in this paper. Robust single machine scheduling under budgeted uncertainty was first considered by Lu et al. (2014), who present an MIP and heuristic to solve the problem. Following this, Tadayon and Smith (2015) study different versions of the min-max robust SMSP under three different uncertainty sets, including a budgeted uncertainty set. Recently, Bougeret et al. (2019) present complexity results and approximation algorithms for a number of different min-max robust scheduling problems under budgeted uncertainty.

To the best of our knowledge, this paper is the first to solve a single-machine scheduling problem in a recoverable robust setting. However, recoverable robustness has had recent application to a number of closely related matching, assignment and scheduling problems. Fischer et al. (2021) consider a recoverable robust assignment problem, in which two perfect matchings of minimum costs must be chosen, subject to these matchings having at least k edges in common. If the cost of the second matching is evaluated in the worst-case scenario, we arrive in the setting of recoverable robustness with interval uncertainty. Hardness results are presented, and a polynomial-time algorithm is developed for the restricted case in which one cost function is Monge. Recently, Bold and Goerigk (2022) also considered recoverable robust scheduling problems under interval uncertainty, deriving a 2-approximation algorithm for their setting.

Regarding project scheduling, Bendotti et al. (2022) introduce the so-called anchor-robust project scheduling problem in which a baseline schedule is designed under the problem uncertainty, with the objective of maximising the size of the subset of jobs that have their starting times unchanged following the realisation of the activity processing times. This problem is shown to be NP-hard even for budgeted uncertainty. In a series of papers Bruni et al. (2017, 2018); Bold and Goerigk (2021), a two-stage resource-constrained project scheduling problem with budgeted uncertainty is introduced and solved.

The contributions of this paper are as follows. In Sect. 2 we formally define the recoverable robust scheduling problem that we consider in this paper. In Sect. 3 we present a general result that enables the construction of compact formulations for a wide range of recoverable robust problems, and apply this in the context of the scheduling problem at hand. We then analyse the stages of the recoverable robust scheduling problem in detail and show that the incremental problem can be solved using a simple linear programming formulation in Sect. 4. To this end, we prove a general result for max-weight matching problems, arguing that odd-cycle constraints are not required in problems with weights of a specific form. This formulation of the incremental problem then leads to an alternative matching-based compact problem formulation. Additionally, we transfer the matching result to an assignment-based formulation for the incremental problem, which results in a third compact model. These three formulations are theoretically compared in Sect. 5 and it is

shown that no one formulation is dominant. In Sect. 6, computational experiments are presented, showing the benefits of a recourse action, the effects of the uncertainty on the model, and the strength of the assignment-based formulation. Finally, some concluding remarks and potential directions for future research are given in Sect. 7.

2 Problem definition

We consider a single machine scheduling problem with the objective of minimising the sum of completion times. Given a set of jobs $\mathcal{N} = \{1, \ldots, n\}$ with processing times $\boldsymbol{p} = (p_1, \ldots, p_n)$, we aim to find a schedule, i.e. an ordering of the jobs $i \in \mathcal{N}$, that minimises the sum of completion times. This nominal problem is denoted by $1||\sum C_i$ under the $\alpha|\beta|\gamma$ scheduling problem notation introduced by Graham et al. (1979). Recall that this problem is easy to solve; the shortest processing time (SPT) rule of sorting jobs by non-decreasing processing times results in an optimal schedule. This problem can be modelled as the following assignment problem with non-general costs:

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)x_{ij} \tag{1}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} x_{ij} = 1 \qquad \forall j \in \mathcal{N} \tag{2}$$

$$\sum_{j\in\mathcal{N}} x_{ij} = 1 \qquad \forall i \in \mathcal{N} \tag{3}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,\ j \in \mathcal{N}, \tag{4}$$

where $x_{ij} = 1$ if job i is scheduled in position j, and $x_{ij} = 0$ otherwise.

We assume the job processing times $p_i,\ i \in \mathcal{N}$ are uncertain, but are known to lie within a given uncertainty set $\mathcal{U}$. In this paper, we consider a general polyhedral uncertainty set given by

$$\mathcal{U} = \left\{\boldsymbol{p} \in \mathbb{R}^n_+ : \boldsymbol{A}\boldsymbol{p} \leq \boldsymbol{b}\right\},$$

where $\boldsymbol{A} \in \mathbb{R}^{M\times n}$ and $\boldsymbol{b} \in \mathbb{R}^M$, consisting of M linear constraints $a_{m1}p_1 + a_{m2}p_2 + \cdots + a_{mn}p_n \leq b_m$ for $m \in \mathcal{M} = \{1, \ldots, M\}$ on the set of possible processing times $\boldsymbol{p}$. Throughout this paper, we assume $\mathcal{U}$ to be compact. It is also possible to include auxiliary variables in the definition of $\mathcal{U}$; for ease of presentation, such variables have been omitted.

We consider this uncertain single machine scheduling problem in the context of a two-stage decision process, where, having decided on a first-stage schedule $\boldsymbol{x}$ under the problem uncertainty, the decision-maker is given the opportunity to react to the realisation of the uncertain data by choosing up to Δ distinct pairs of jobs and swapping their positions, to obtain a second-stage schedule $\boldsymbol{y}$.

This recoverable robust problem can be written as follows:

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{p}\in\mathcal{U}} \min_{\boldsymbol{y}\in\mathcal{X}(\boldsymbol{x})} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)y_{ij}, \tag{RRS}$$

where $\mathcal{X} = \{\boldsymbol{x} \in \{0,1\}^{n\times n} : (2), (3)\}$ is the set of feasible schedules, and $\mathcal{X}(\boldsymbol{x}) \subseteq \mathcal{X}$ is the set of feasible second-stage assignments given $\boldsymbol{x}$. That is,

$$\mathcal{X}(\boldsymbol{x}) = \{\boldsymbol{y} \in \mathcal{X} : d(\boldsymbol{x}, \boldsymbol{y}) \leq \Delta\},$$

where $d(\boldsymbol{x}, \boldsymbol{y})$ is some measure of the distance between the first and second-stage schedules.

In this paper, we restrict our attention to the case where the recourse action consists of disjoint pairwise swaps to the first-stage positions of the jobs. Such a recourse action is attractive for its explainability and simple implementation. For example, rearranging workplace rotas is made much simpler if staff members are only made to swap with one other staff member since this allows for direct exchanges of briefings, notes, etc. In addition to its conceptual simplicity, the restriction to disjoint pairwise swaps improves the tractability of the problem and leads to the results that we present and analyse in this paper.

Hence, in this case we define $d(\boldsymbol{x}, \boldsymbol{y})$ to be the minimum number of pairwise distinct swaps required to transform $\boldsymbol{x}$ into $\boldsymbol{y}$, if this number exists; otherwise, we set it to ∞. Observe that the value $d(\boldsymbol{x}, \boldsymbol{y})$ can be calculated using the following approach. Let $\mathcal{E}_{\boldsymbol{x}}$ be the edges chosen by $\boldsymbol{x}$ in the corresponding bipartite graph, oriented towards the right, and let $\mathcal{E}_{\boldsymbol{y}}$ be the edges chosen by $\boldsymbol{y}$, oriented towards the left, i.e. $(j, i) \in \mathcal{E}_{\boldsymbol{y}}$ corresponds to assigning job i to position j. If and only if the edges $\mathcal{E}_{\boldsymbol{x}} \cup \mathcal{E}_{\boldsymbol{y}}$ decompose into 2-cycles and 4-cycles, we have $d(\boldsymbol{x}, \boldsymbol{y}) < \infty$, in which case $d(\boldsymbol{x}, \boldsymbol{y})$ is equal to the number of 4-cycles. This is because a 2-cycle corresponds to a job with an unchanged position, whilst a 4-cycle represents a swap of positions of two jobs. An example is given in Fig. 1.

We define the adversarial and incremental problems of (RRS) as follows. Given both a first-stage solution $\boldsymbol{x} \in \mathcal{X}$ and a scenario $\boldsymbol{p} \in \mathcal{U}$, the incremental problem consists of finding the best possible second-stage solution $\boldsymbol{y} \in \mathcal{X}(\boldsymbol{x})$. That is,

$$\text{Inc}(\boldsymbol{x}, \boldsymbol{p}) = \min_{\boldsymbol{y}\in\mathcal{X}(\boldsymbol{x})} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)y_{ij}.$$

The adversarial problem is to find a worst-case scenario $\boldsymbol{p} \in \mathcal{U}$ for a given first-stage schedule $\boldsymbol{x} \in \mathcal{X}$. That is,

$$\begin{aligned}\text{Adv}(\boldsymbol{x}) &= \max_{\boldsymbol{p}\in\mathcal{U}} \min_{\boldsymbol{y}\in\mathcal{X}(\boldsymbol{x})} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)y_{ij} \\ &= \max_{\boldsymbol{p}\in\mathcal{U}} \text{Inc}(\boldsymbol{x}, \boldsymbol{p}).\end{aligned}$$

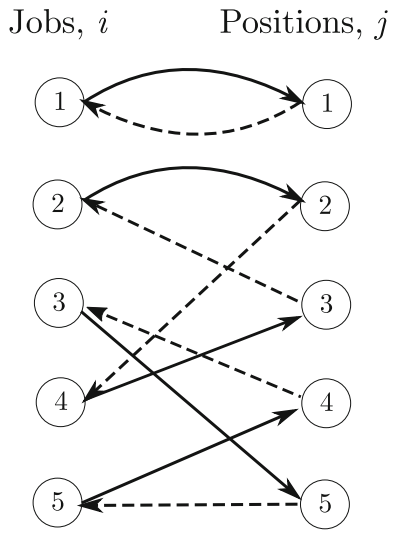

Fig. 1 An example first and second-stage solution. The first-stage assignment is given by the solid arcs oriented towards the right, and corresponds to the schedule (1,2,4,5,3). The second-stage assignment is given by the dashed arcs oriented towards the left, and corresponds to the schedule (1,4,2,3,5). There are two 4-cycles corresponding to the switching of positions of jobs 2 and 4, and 3 and 5. Hence $d(\boldsymbol{x}, \boldsymbol{y}) = 2$

Observe that for the case of general polyhedral uncertainty that we consider here, (RSS) is NP-hard. To see this, suppose that $\Delta = 0$, i.e. there is no recovery option and the second-stage variables are fixed to the corresponding first-stage values. Then the problem reduces to a standard robust single machine scheduling problem of the form

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{p}\in\mathcal{U}} \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)x_{ij}.$$

Now consider a specific polyhedral uncertainty set defined by the linear combination of two discrete points. Since the worst-case scenario must lie at a vertex of the polyhedron, this polyhedral uncertainty set is equivalent to a discrete uncertainty set containing the two extreme points. Since the robust scheduling problem with two scenarios is already NP-hard (see Kouvelis and Yu (1997)), this hardness result also extends to the problem we consider here.

Finally, note that in problem (RRS) we aim to minimise the worst-case costs of the resulting recovery solutions. If the first-stage costs are also relevant, all the results presented in this paper can be adjusted trivially by including these costs in the objective function.

3 A general model for recoverable robustness

In this section we present a general model for recoverable robust optimisation problems, and apply this method to the uncertain single machine scheduling problem (RRS). Our approach is to determine a first-stage solution $\boldsymbol{x} \in \mathcal{X}$ as well as a finite set of candidate recovery solutions $\boldsymbol{y}^1, \ldots, \boldsymbol{y}^K \in \mathcal{X}(\boldsymbol{x})$.

The following result shows that using $K = n+1$ recovery solutions is sufficient to guarantee that this approach provides an exact solution to the problem.

Theorem 1 *Let a recoverable robust problem of the form*

$$\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{\boldsymbol{y}\in\mathcal{X}(\boldsymbol{x})} f(\boldsymbol{y}, \boldsymbol{c})$$

be given, where $\mathcal{X}, \mathcal{X}(\boldsymbol{x}) \subseteq \{0,1\}^n$, $\mathcal{U}$ is a compact convex set, f is linear in $\boldsymbol{y}$, and concave in $\boldsymbol{c}$. Then this problem is equivalent to

$$\min_{\substack{\boldsymbol{x}\in\mathcal{X},\\ \boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(n+1)}\in\mathcal{X}(\boldsymbol{x})}} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{i=1,\ldots,n+1} f(\boldsymbol{y}^{(i)}, \boldsymbol{c}).$$

Proof The idea of the proof is similar to models developed for K-adaptability (see Hanasusanto et al. (2015, Theorem 1) and Buchheim and Kurtz (2017, Corollary 1)). Recall both Carathéodory's theorem and the minimax theorem. Carathéodory's theorem states that any point $x \in \mathbb{R}^n$ lying in $conv(X)$ can be written as a convex combination of $n+1$ points from X. The minimax theorem states that if X and Y are two compact, convex sets, and $f : X \times Y \to \mathbb{R}$ is a continuous compact-concave function (i.e. $f(\cdot, y)$ is concave for fixed values of y and $f(x, \cdot)$ is convex for fixed values of x), then

$$\max_x \min_y f(x,y) = \min_y \max_x f(x,y).$$

We make use of both of these results in the following:

$$\begin{aligned}
&\min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{\boldsymbol{y}\in\mathcal{X}(\boldsymbol{x})} f(\boldsymbol{y}, \boldsymbol{c}) \\
&= \min_{\boldsymbol{x}\in\mathcal{X}} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{\boldsymbol{y}\in conv(\mathcal{X}(\boldsymbol{x}))} f(\boldsymbol{y}, \boldsymbol{c}) \\
&= \min_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}\in conv(\mathcal{X}(\boldsymbol{x}))} \max_{\boldsymbol{c}\in\mathcal{U}} f(\boldsymbol{y}, \boldsymbol{c}) \\
&\quad \text{(by the minimax theorem)} \\
&= \min_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(n+1)}\in\mathcal{X}(\boldsymbol{x})} \min_{\substack{\lambda^1,\ldots,\lambda^{n+1}\geq 0\\ \sum_{i=1}^{n+1}\lambda^i=1}} \max_{\boldsymbol{c}\in\mathcal{U}} f\Big(\sum_{i=1}^{n+1} \lambda^i \boldsymbol{y}^{(i)}, \boldsymbol{c}\Big) \\
&\quad \text{(by Caratheodory's theorem)} \\
&= \min_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(n+1)}\in\mathcal{X}(\boldsymbol{x})} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{\substack{\lambda^1,\ldots,\lambda^{n+1}\geq 0\\ \sum_{i=1}^{n+1}\lambda^i=1}} f\Big(\sum_{i=1}^{n+1} \lambda^i \boldsymbol{y}^{(i)}, \boldsymbol{c}\Big) \\
&\quad \text{(by the minimax theorem)}
\end{aligned}$$

$$= \min_{\boldsymbol{x}\in\mathcal{X}} \min_{\boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(n+1)}\in\mathcal{X}(\boldsymbol{x})} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{\substack{\lambda^1,\ldots,\lambda^{n+1}\geq 0\\ \sum_{i=1}^{n+1}\lambda^i=1}} \sum_{i=1}^{n+1} \lambda^i f(\boldsymbol{y}^{(i)}, \boldsymbol{c})$$

$$= \min_{\substack{\boldsymbol{x}\in\mathcal{X},\\ \boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(n+1)}\in\mathcal{X}(\boldsymbol{x})}} \max_{\boldsymbol{c}\in\mathcal{U}} \min_{i=1,\ldots,n+1} f(\boldsymbol{y}^{(i)}, \boldsymbol{c}).$$

□

This approach can be used to derive a compact formulation to the uncertain single machine scheduling problem (RRS). To this end, we first consider the inner selection problem, given a first-stage solution $\boldsymbol{x}$ and set of recovery solutions $\boldsymbol{y}^1, \ldots, \boldsymbol{y}^K$, and a scenario $\boldsymbol{p}$. This is given by

$$\begin{aligned}
\min \ & \sum_{k\in\mathcal{K}} \left(\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i (n+1-j) y_{ij}^k \right) \lambda_k \\
\text{s.t.} \ & \sum_{k\in\mathcal{K}} \lambda_k = 1 \\
& \lambda_k \geq 0 \qquad\qquad \forall k \in \mathcal{K},
\end{aligned}$$

where $\mathcal{K} = \{1, \ldots, K\}$. The problem of finding a worst-case scenario $\boldsymbol{p} \in \mathcal{U}$ for the choice of first-stage solution $\boldsymbol{x}$ and recovery solutions $\boldsymbol{y}^1, \ldots, \boldsymbol{y}^K$ is therefore:

$$\begin{aligned}
\max \ & t \\
\text{s.t.} \ & t \leq \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i (n+1-j) y_{ij}^k \qquad \forall k \in \mathcal{K} \\
& \sum_{i\in\mathcal{N}} a_{mi} p_i \leq b_m \qquad \forall m \in \mathcal{M} \\
& p_i \geq 0 \qquad \forall i \in \mathcal{N}.
\end{aligned}$$

Here, t is an unbounded variable to represent the minimum of the corresponding right-hand sides for all $k \in \mathcal{K}$ (an epigraph reformulation, which can also be seen as the dual of the inner minimization problem). Dualising this problem then gives the following formulation for (RRS):

$$\min \sum_{m\in\mathcal{M}} b_m q_m \tag{5}$$

$$\text{s.t.} \sum_{k\in\mathcal{K}} \mu_k = 1 \tag{6}$$

$$\sum_{m\in\mathcal{M}} a_{mi} q_m \geq \sum_{k\in\mathcal{K}} \left(\sum_{j\in\mathcal{N}} (n+1-j) y_{ij}^k \right) \mu_k \quad \forall i \in \mathcal{N} \tag{7}$$

$$d(\boldsymbol{x}, \boldsymbol{y}^k) \leq \Delta \quad \forall k \in \mathcal{K} \tag{8}$$

$$\boldsymbol{x} \in \mathcal{X} \tag{9}$$

$$\boldsymbol{y}^k \in \mathcal{X} \quad \forall k \in \mathcal{K} \tag{10}$$

$$\mu_k \geq 0 \quad \forall k \in \mathcal{K} \tag{11}$$

$$q_m \geq 0 \quad \forall m \in \mathcal{M}, \tag{12}$$

where $d(\boldsymbol{x}, \boldsymbol{y}^k)$ is some measure of distance between the first-stage solution and the k-th recovery solution. Note that this model is not restricted to any particular choice of distance measure $d(\boldsymbol{x}, \boldsymbol{y}^k)$.

However, if we opt to calculate the distance between two schedules as the minimum number of disjoint pairwise swaps required to transform one schedule into the other, this can be modelled as follows. Let $z_{ii'}^k$ be a binary variable that is set to 1 when jobs i and i' have swapped positions in recovery solution $\boldsymbol{y}^k$, relative to the first-stage schedule $\boldsymbol{x}$. In this case, we have that

$$y_{ij}^k = \sum_{i'\in\mathcal{N}} z_{ii'}^k x_{i'j}.$$

Hence, $\boldsymbol{y}^k$ can be removed from the model, and replaced by $\boldsymbol{z}^k$ with the inclusion of the following constraints:

$$\sum_{i'\in\mathcal{N}} z_{ii'}^k = 1 \qquad \forall i \in \mathcal{N},\ k \in \mathcal{K} \tag{13}$$

$$\sum_{i\in\mathcal{N}} z_{ii'}^k = 1 \qquad \forall i' \in \mathcal{N},\ k \in \mathcal{K} \tag{14}$$

$$z_{ii'}^k = z_{i'i}^k \qquad \forall i, i' \in \mathcal{N},\ k \in \mathcal{K} \tag{15}$$

$$\sum_{i\in\mathcal{N}} z_{ii}^k \geq n - 2\Delta \qquad \forall k \in \mathcal{K}. \tag{16}$$

Note that although one set of assignment constraints in combination with the symmetry constraints implicitly enforce the other set of assignment constraints (e.g. (13) and (15) implies (14)), we opt to explicitly include the full set of assignment constraints since doing so provides a modest benefit to the computational performance of the model.

To arrive at a mixed-integer linear program, the products $z_{ii'}^k \cdot x_{i'j} \cdot \mu_k$ need to be linearised using standard techniques. The full linearised formulation contains $O(n^3 K)$ constraints and variables and is shown in Appendix B.1.

4 Complexity of subproblems and compact formulations

In this section, we examine the incremental and adversarial problems of (RRS) in more detail and subsequently derive two additional compact formulations.

4.1 Matching-based formulation

We first consider a matching-based formulation for the incremental problem. For the ease of presentation, we assume for

now that $x_{ii} = 1$ for all $i \in \mathcal{N}$, i.e. the first-stage solution is a horizontal matching. Note a change in notation for this section where now the indices i and j are both used to refer to jobs, and ℓ denotes a position in the schedule. Supposing that the positions of jobs i and j are switched in the recovery schedule, the reduction in cost of making this switch is given by

$$p_i(n+1-i) + p_j(n+1-j) - p_i(n+1-j) - p_j(n+1-i) = (p_i - p_j)(j-i).$$

Letting z_{ij} indicate whether or not jobs i and j swap positions in the schedule, the incremental problem can be formulated as:

$$\min \sum_{i\in\mathcal{N}} p_i(n+1-i) - \sum_{e=\{i,j\}\in\mathcal{E}} (p_i - p_j)(j-i)z_e \tag{17}$$

$$\text{s.t.} \sum_{e\in\delta(i)} z_e \le 1 \qquad \forall\, i \in \mathcal{N} \tag{18}$$

$$\sum_{e\in\mathcal{E}} z_e \le \Delta \tag{19}$$

$$z_e \in \{0,1\} \qquad \forall e \in \mathcal{E}, \tag{20}$$

where $\mathcal{E} = \{\{i,j\} : i,j \in \mathcal{N},\ i \ne j\}$ is the set of unique swaps, and $\delta(i)$ is the set of edges incident to vertex i. This is a cardinality-constrained matching problem on a complete graph with one node for each job $i \in \mathcal{N}$.

We examine this matching-based formulation in further detail. First, consider the maximum weight matching problem on a general graph $G = (\mathcal{V}, \mathcal{E})$. This problem can be formulated as the following linear program:

$$\max \sum_{e\in\mathcal{E}} w_e x_e \tag{21}$$

$$\text{s.t.} \sum_{e\in\delta(i)} x_e \le 1 \qquad \forall i \in \mathcal{V} \tag{22}$$

$$\sum_{e\in\mathcal{E}(\mathcal{W})} x_e \le \frac{|\mathcal{W}|-1}{2} \qquad \forall \mathcal{W} \subseteq \mathcal{V},\ |\mathcal{W}| \text{ odd} \tag{23}$$

$$x_e \ge 0 \qquad \forall e \in \mathcal{E}, \tag{24}$$

where $\mathcal{E}(\mathcal{W})$ is the set of edges in the subgraph induced on $\mathcal{W}$. Edmonds (1965) showed that constraints (23), known as odd-cycle constraints or blossom constraints, are required to fully characterise the matching polytope.

In the following theorem, we show that for a matching problem with the same cost structure as (17), odd-cycle constraints are not required.

Theorem 2 *For any $\boldsymbol{a}$, $\boldsymbol{b} \in \mathbb{R}_+^{|\mathcal{V}|}$, the problem*

$$\max \sum_{e=\{i,j\}\in\mathcal{E}} (a_i - a_j)(b_i - b_j)x_e \tag{25}$$

$$\text{s.t.} \sum_{e\in\delta(i)} x_e \le 1 \qquad \forall i \in \mathcal{V} \tag{26}$$

$$x_e \ge 0 \qquad \forall e \in \mathcal{E} \tag{27}$$

has an optimal solution with $x_e \in \{0,1\}$ for all $e \in \mathcal{E}$.

Proof Schrijver (2003, Theorem 30.2, page 522) states that each vertex of the matching polytope described by (26) and (27) is half-integer, i.e. $x_e \in \{0, \frac{1}{2}, 1\}$ for all $e \in \mathcal{E}$ in an optimal solution. Additionally, as observed by Balinski (1965), the vertices of the matching polytope can be partitioned into a matching $\mathcal{P}$, where $x_e = 1$ for each $e \in \mathcal{P}$, and a set of 1/2-fractional cycles of odd length, where $x_e = \frac{1}{2}$ for each e in the odd cycles. Hence, we can restrict our attention only to 1/2-fractional odd cycles, and show that there is an optimal solution where such cycles do not exist.

Suppose we are given an optimal solution containing a 1/2-fractional odd cycle, consisting of edges $\mathcal{C} = \{e_{i_1,i_2}, e_{i_2,i_3}, \ldots, e_{i_{q-1},i_q}, e_{i_q,i_1}\}$, with weights given by $w_{ij} = (a_i - a_j)(b_i - b_j)$. Without loss of generality, we assume an orientation in the cycle, where edges are directed as (i_j, i_{j+1}) for $j = 1, \ldots, q$, where $i_{q+1} = i_1$.

Note that if $w_e \le 0$ for some edge e, it can be removed from $\mathcal{E}$, as such an edge will never be selected in an optimal matching. Hence, we may assume that $w_e > 0$ for all $e \in \mathcal{C}$. Since $w_{ij} = (a_i - a_j)(b_i - b_j) > 0$ for all $e_{ij} \in \mathcal{C}$, $(a_i - a_j)$ and $(b_i - b_j)$ must have the same sign. That is, either $a_i > a_j$ and $b_i > b_j$, in which case we refer to e_{ij} as a *decreasing edge*, or $a_i < a_j$ and $b_i < b_j$, in which case we refer to e_{ij} as an *increasing edge*.

We show that there is an optimal 1/2-fractional cycle that alternates between increasing and decreasing edges. Suppose that there are $p < q$ consecutive decreasing edges in $\mathcal{C}$, $e_{j_1,j_2}, e_{j_2,j_3}, \ldots, e_{j_{p-1},j_p}$, i.e. $a_{j_1} > a_{j_2} > \cdots > a_{j_p}$ and $b_{j_1} > b_{j_2} > \cdots > b_{j_p}$. In this case

$$\begin{aligned}
w_{j_1,j_p} &= (a_{j_1} - a_{j_p})(b_{j_1} - b_{j_p}) \\
&= \Big((a_{j_1} - a_{j_2}) + (a_{j_2} - a_{j_3}) + \cdots + (a_{j_p-1} - a_{j_p})\Big) \\
&\quad \cdot \Big((b_{j_1} - b_{j_2}) + (b_{j_2} - b_{j_3}) + \cdots + (b_{j_{p-1}} - b_{j_p})\Big) \\
&= w_{j_1,j_2} + w_{j_2,j_3} + \cdots + w_{j_{p-1},j_p} \\
&\quad + (a_{j_1} - a_{j_2})\Big((b_{j_2} - b_{j_3}) + \cdots + (b_{j_{p-1}} - b_{j_p})\Big) \\
&\quad + (a_{j_2} - a_{j_3})\Big((b_{j_1} - b_{j_2}) + \cdots + (b_{j_{p-1}} - b_{j_p})\Big) \\
&\quad + \ldots \\
&\quad + (a_{j_{p-1}} - a_{j_p})\Big((b_{j_1} - b_{j_2}) + \cdots + (b_{j_{p-2}} - b_{j_{p-1}})\Big) \\
&> w_{j_1,j_2} + w_{j_2,j_3} + \cdots + w_{j_{p-1},j_p},
\end{aligned}$$

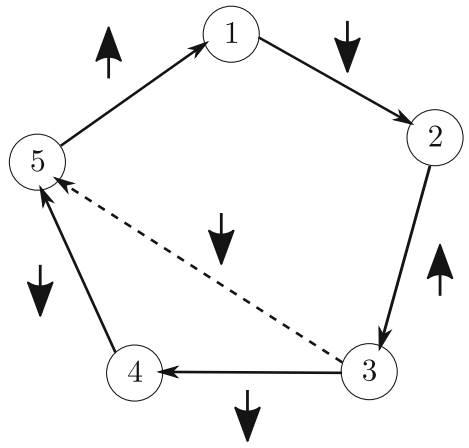

Fig. 2 An example of a 1/2-fractional cycle involving $q = 5$ nodes. Up and down arrows indicate increasing and decreasing edges respectively. It is optimal to replace the two consecutive decreasing edges (3, 4) and (4, 5) with the dashed edge (3, 5), i.e. $w_{35} > w_{34} + w_{45}$

which means that replacing the p consecutive decreasing edges in $\mathcal{C}$ by the edge e_{j_1,j_p} would lead to an even better objective value (see Fig. 2 for an illustration). The same argument can be used to show that there also cannot be p consecutive increasing edges in an optimal 1/2-fractional cycle.

We have therefore constructed an optimal 1/2-fractional cycle that strictly alternates between increasing and decreasing edges. Clearly, this is only possible if q is even. Since a 1/2-fractional even cycle can be written as a convex combination of two feasible matchings, this proves that exists an optimal solution without any 1/2-fractional cycles. □

The following result, presented in Schrijver (2003) (Corollary 18.10a, page 331), states that the integrality of the vertices of the matching polytope is unaffected by the addition of a cardinality constraint.

Theorem 3 *Let $G = (\mathcal{V}, \mathcal{E})$ be an undirected graph and let $k, l \in \mathbb{Z}_+$ with $k \leq l$. Then the convex hull of the incidence vectors of matchings $\mathcal{P}$ satisfying $k \leq |\mathcal{P}| \leq l$ is equal to the set of those vectors $\boldsymbol{x}$ in the matching polytope of G satisfying $k \leq \mathbf{1}^\top \boldsymbol{x} \leq l$.*

This result, in combination with Theorem 2, provides us with the following corollary:

Corollary 4 *For any $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}_+^{|\mathcal{V}|}$, the problem*

$$\max \sum_{e=\{i,j\}\in\mathcal{E}} (a_i - a_j)(b_i - b_j)x_e \tag{28}$$

$$s.t. \sum_{e\in\delta(i)} x_e \leq 1 \qquad \forall i \in \mathcal{V} \tag{29}$$

$$\sum_{e\in\mathcal{E}} x_e \leq \Delta \tag{30}$$

$$x_e \geq 0 \qquad \forall e \in \mathcal{E} \tag{31}$$

has an optimal solution with $x_e \in \{0, 1\}$ for all $e \in \mathcal{E}$.

Hence, given a first-stage solution $\boldsymbol{x}$ and scenario $\boldsymbol{p}$, we can formulate the incremental problem as a linear program with polynomially many constraints. We use this result to derive a compact formulation for the full uncertain single machine scheduling problem (RRS).

We begin by formulating the incremental problem $\mathrm{Inc}(\boldsymbol{x}, \boldsymbol{p})$ according to Corollary 4. Note that we now consider a general first-stage assignment that is not necessarily horizontal, and therefore introduce terms $\sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}$ to track the position in which job i is scheduled in the first-stage schedule. We fix an arbitrary orientation of edges, using $\mathcal{E} = \{(i, j) \in \mathcal{N} \times \mathcal{N} : i < j\}$ in the following.

$$\begin{aligned}
\min_{z} \ & \sum_{i\in\mathcal{N}} p_i \left(n + 1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}\right) \\
& - \sum_{(i,j)\in\mathcal{E}} (p_i - p_j)\left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}\right) z_{ij} \\
\text{s.t.} \ & \sum_{(i,j)\in\mathcal{E}} z_{ij} + \sum_{(j,i)\in\mathcal{E}} z_{ji} \leq 1 && \forall i \in \mathcal{N} \\
& \sum_{(i,j)\in\mathcal{E}} z_{ij} \leq \Delta \\
& z_{ij} \geq 0 && \forall (i, j) \in \mathcal{E}.
\end{aligned}$$

Taking the dual of this, we get the following formulation for the adversarial problem $\mathrm{Adv}(\boldsymbol{x})$:

$$\begin{aligned}
\max_{\boldsymbol{p},\boldsymbol{\alpha},\gamma} \ & \sum_{i\in\mathcal{N}} p_i \left(n + 1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}\right) - \sum_{i\in\mathcal{N}} \alpha_i - \gamma\Delta \\
\text{s.t.} \ & \alpha_i + \alpha_j + \gamma \geq (p_i - p_j)\left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}\right) \\
& \forall (i, j) \in \mathcal{E} \\
& \sum_{i\in\mathcal{N}} a_{mi} p_i \leq b_m && \forall m \in \mathcal{M} \\
& p_i \geq 0 && \forall i \in \mathcal{N} \\
& \alpha_i \geq 0 && \forall i \in \mathcal{N} \\
& \gamma \geq 0.
\end{aligned}$$

Since this is a linear program, we immediately obtain following result:

Corollary 5 *The adversarial problem can be solved in polynomial time.*

Finally, dualising the above adversarial formulation, we get the following compact formulation for problem (RRS):

$$\min_{\boldsymbol{x},z,\boldsymbol{q}} \sum_{m\in\mathcal{M}} b_m q_m \tag{32}$$

$$\text{s.t.} \sum_{m\in\mathcal{M}} a_{mi}q_m + \sum_{(i,j)\in\mathcal{E}} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} \right) z_{ij} - \sum_{(j,i)\in\mathcal{E}} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} \right) z_{ji}$$

$$\geq (n+1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}) \qquad \forall i \in \mathcal{N} \qquad (33)$$

$$\sum_{(i,j)\in\mathcal{E}} z_{ij} + \sum_{(j,i)\in\mathcal{E}} z_{ji} \leq 1 \qquad \forall i \in \mathcal{N} \qquad (34)$$

$$\sum_{(i,j)\in\mathcal{E}} z_{ij} \leq \Delta \qquad (35)$$

$$x \in \mathcal{X} \qquad (36)$$

$$q_m \geq 0 \qquad \forall m \in \mathcal{M} \qquad (37)$$

$$z_{ij} \geq 0 \qquad \forall (i,j) \in \mathcal{E}. \qquad (38)$$

Upon linearising the quadratic $x_{i\ell} \cdot z_{ij}$ and $x_{j\ell} \cdot z_{ij}$ terms, this model becomes a mixed-integer linear program. The fully linearised model contains $O(n^3)$ constraints and variables and is presented in full in Appendix 2.2.

4.2 Assignment-based formulation

We now consider an alternative formulation for the incremental problem. Again, for the purposes of examining the incremental problem, we initially consider the first-stage schedule to be a horizontal assignment, i.e. $x_{ii} = 1$ for all $i \in \mathcal{N}$. By letting variables y_{ij} represent a second-stage assignment (we now return to the convention where the index i is used to denote a job and the index j is used to denote a position in the schedule), we can formulate the incremental problem as follows:

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-j)y_{ij} \qquad (39)$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} y_{ij} = 1 \qquad \forall j \in \mathcal{N} \qquad (40)$$

$$\sum_{j\in\mathcal{N}} y_{ij} = 1 \qquad \forall i \in \mathcal{N} \qquad (41)$$

$$y_{ij} = y_{ji} \qquad \forall i,j \in \mathcal{N} \qquad (42)$$

$$\sum_{i\in\mathcal{N}} y_{ii} \geq n - 2\Delta \qquad (43)$$

$$y_{ij} \in \{0,1\} \qquad \forall i,j \in \mathcal{N}. \qquad (44)$$

Constraints (42) and (43) ensure that the second-stage assignment is a feasible recovery to the first-stage solution, that is, the second-stage assignment is constructed by swapping the first-stage positions of up to Δ disjoint pairs of jobs. Note that this is a level-constrained symmetric perfect matching problem, which can be solved in polynomial time (Thomas, 2015, Theorem 2.28).

We show that problem (39)-(44) can be solved as a linear program as a result of its non-general cost structure. As the proof is technical and based on a reduction to the corresponding maximum weight matching problem, it is omitted here and can be found in Appendix A.

Theorem 6 *For any* $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n_+$, *the problem*

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} a_i b_j y_{ij} \qquad (45)$$

$$\text{s.t. } (40) - (43) \qquad (46)$$

$$y_{ij} \geq 0 \qquad (47)$$

has an optimal solution with $y_{ij} \in \{0,1\}$ *for all* $i, j \in \mathcal{N}$.

We now use this result to find an assignment-based formulation for (RRS). We first write the incremental problem in the form given by (45)-(47). Since we are now considering the case where $\boldsymbol{x}$ is not necessarily a horizontal matching, we rearrange the indices accordingly.

$$\min \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} p_i(n+1-\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell})y_{ij}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} y_{ij} = 1 \qquad \forall j \in \mathcal{N}$$

$$\sum_{j\in\mathcal{N}} y_{ij} = 1 \qquad \forall i \in \mathcal{N}$$

$$y_{ij} = y_{ji} \qquad \forall i,j \in \mathcal{N}$$

$$\sum_{i\in\mathcal{N}} y_{ii} \geq n - 2\Delta$$

$$y_{ij} \geq 0 \qquad \forall i,j \in \mathcal{N}.$$

Taking the dual of this, the adversarial problem can be formulated in the following way:

$$\max_{\boldsymbol{\alpha},\boldsymbol{\beta},\boldsymbol{\gamma},\tau,\boldsymbol{p}} \sum_{i\in\mathcal{N}} (\alpha_i + \beta_i) + (n-2\Delta)\tau$$

$$\text{s.t. } \alpha_j + \beta_i + \gamma_{ij} \leq (n+1-\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell})p_i \qquad \forall i,j \in \mathcal{N} : i < j$$

$$\alpha_j + \beta_i - \gamma_{ji} \leq (n+1-\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell})p_i \qquad \forall i,j \in \mathcal{N} : i > j$$

$$\alpha_i + \beta_i + \tau \leq (n+1-\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell})p_i \qquad \forall i \in \mathcal{N}$$

$$\sum_{i\in\mathcal{N}} a_{mi}p_i \leq b_m \qquad \forall m \in \mathcal{M}$$

$$p_i \geq 0 \qquad \forall i \in \mathcal{N}$$

$$\tau \geq 0.$$

Finally, we dualise this adversarial formulation to derive the following formulation for the recoverable problem:

$$\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{q}} \sum_{m\in\mathcal{M}} b_m q_m \tag{48}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} y_{ij} = 1 \qquad \forall j \in \mathcal{N} \tag{49}$$

$$\sum_{j\in\mathcal{N}} y_{ij} = 1 \qquad \forall i \in \mathcal{N} \tag{50}$$

$$\sum_{i\in\mathcal{N}} y_{ii} \geq n - 2\Delta \tag{51}$$

$$y_{ij} = y_{ji} \qquad \forall i, j \in \mathcal{N} \tag{52}$$

$$\sum_{m\in\mathcal{M}} a_{mi} q_m \geq \sum_{j\in\mathcal{N}} (n + 1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell}) y_{ij} \qquad \forall i \in \mathcal{N} \tag{53}$$

$$\boldsymbol{x} \in \mathcal{X} \tag{54}$$

$$q_m \geq 0 \qquad \forall m \in \mathcal{M} \tag{55}$$

$$y_{ij} \geq 0 \qquad \forall i, j \in \mathcal{N} \tag{56}$$

As before, products $x_{j\ell} \cdot y_{ij}$ can be linearised using standard techniques. The resulting mixed-integer linear program contains $O(n^3)$ constraints and variables and can be found in Appendix 2.3.

5 Comparison of formulations

This section presents a brief investigation into the linear relaxations of the three formulations derived above in order to compare their relative theoretical strengths. We begin by showing that no comparisons can be made between the general formulation and the other two formulations.

In preparation of the proof of this result we introduce budgeted uncertainty as a special case of polyhedral uncertainty. A budgeted uncertainty set can be defined as

$$\mathcal{U}_B = \left\{ \boldsymbol{p} \in \mathbb{R}^n_+ : \sum_{i\in\mathcal{N}} \frac{p_i - \hat{p}_i}{\bar{p}_i} \leq \Gamma,\ p_i \in [\hat{p}_i, \hat{p}_i + \bar{p}_i],\ i \in \mathcal{N} \right\},$$

where $\hat{p}_i$ is the nominal processing time of job i and $\bar{p}_i$ is the worst-case delay to the processing time of job i. Introduced by Bertsimas and Sim (2004), its motivation is to exclude unrealistically pessimistic worst-case scenarios from the uncertainty set and thereby avoid overly conservative and highly-expensive solutions. This is achieved by assuming that at most Γ jobs can simultaneously reach their maximum delays. Note that when $\Gamma = 0$, each job assumes its nominal processing time and the $\mathcal{U}_B$ reduces to a single scenario. Additionally, observe that as $\Gamma \to n$, this budgeted uncertainty set becomes an interval. When $\Gamma = n$ the worst-case scenario is known a priori to be when all jobs achieve their worst-case processing times $\hat{p}_i + \bar{p}_i$. In this case the problem can be solved by simply ordering the jobs accoring to their worst-case processing times, and no recourse action will be required. The proof of the following proposition makes use of an instance involving a bugdeted uncertainty set.

Theorem 7 *The general formulation (61)-(81) is incomparable with both the matching-based formulation (82)-(98) and the assignment-based formulation (99)-(113).*

Proof First consider a problem with two jobs with processing times that lie in the uncertainty set $\mathcal{U} = \{(p_1, p_2) : p_1 \leq 3,\ p_2 \leq 3,\ p_1 + 2p_2 \leq 7\}$. Suppose also that $\Delta = 1$, i.e. one swap can be made to amend the first-stage schedule. In this case, the linear relaxation of the matching-based formulation has an objective value of 7, whilst the linear relaxation of the assignment-based formulation has an objective value of 5.

Now consider an instance involving jobs with $\hat{\boldsymbol{p}} = (10, 8, 9, 4, 1, 5, 7, 1)$ and $\bar{\boldsymbol{p}} = (9, 7, 5, 4, 1, 3, 6, 1)$ lying in the budgeted uncertainty set $\mathcal{U}_B$, and set $\Gamma = 1$ and $\Delta = 1$. The linear relaxation of the matching-based formulation for this instance has an optimal objective value of -9.2 (to 1 decimal place), whilst the linear relaxation of the assignment-based formulation has an objective value of -285.4 (to 1 decimal place).

For both of these instances, the linear relaxation of the general formulation attains an objective value of 0. (In fact, for any polyhedral uncertainty set in which $a_{mi} \geq 0$ and $b_m \geq 0$ for all $i \in \mathcal{N}$, the linear relaxation of the general formulation will be 0, since it is free to set the linearisation variables $h^k_{ii'j} = 0$ for all $i,\ i',\ j \in \mathcal{N},\ k \in \mathcal{K}$ and therefore $q_m = 0$ for all $m \in \mathcal{M}$.)

These examples show that the matching and assignment-based formulations are tighter than the general formulation for some instances, but less tight for other instances. Hence the general formulation is incomparable with the matching and assignment-based formulations. □

It is the case however that the objective value of the linear relaxation of the non-linear matching-based formulation is always greater than or equal to the objective value for the linear relaxation of the non-linear assignment-based formulation. That is, that non-linear matching formulation dominates the non-linear assignment formulation.

Theorem 8 *The non-linear matching-based formulation (32)-(38) dominates the non-linear assignment-based formulation (48)-(56).*

The proof of this statement involves the construction of a transformation ϕ to show that any feasible solution to the matching formulation can be transformed into a feasible solution to the assignment problem. The proof can be found in Appendix A. It does however remain open as to whether this result can be extended to the linearised versions of these formulations given by (82)-(98) and (99)-(113), respectively.

6 Computational experiments

This section presents results from solving the three compact models introduced in this paper. The results of these exact models are compared against each other, as well as against three additional heuristic solution methods. As a particular example of a general polyhedral uncertainty, here we consider budgeted uncertainty as outlined in the previous section. Before introducing the heuristics we propose for solving this problem and examining their performance, we comment on the test instances and computational hardware used for these experiments.

Instances have been generated by randomly sampling both $\hat{p}_i$ and $\bar{p}_i$ from the set $\{1, 2, \ldots, 100\}$. 20 instances of sizes $n \in \{10, 15, 20\}$ have been generated, resulting in a total of 60 deterministic test instances. For each deterministic instance, three uncertain instances have been generated by setting $\Gamma \in \{3, 5, 7\}$, resulting in a total of 180 uncertain instances. These instances, as well as the complete results data, can be found at https://github.com/boldm1/RR-single-machine-scheduling.

All methods have been run on 4 cores of a 2.30GHz Intel Xeon CPU, limited to 16GB RAM. The exact models have been solved using Gurobi 9.0.1, with a time limit of 10 min.

6.1 Heuristics

The three heuristic methods we consider are as follows:

1. *Sorting*. Obtain a schedule by ordering the jobs $i \in \mathcal{N}$ according to non-decreasing $\hat{p}_i + \bar{p}_i$, i.e. a schedule that performs best in the worst-case scenario when $\Gamma = n$, and evaluate by solving Adv($\boldsymbol{x}$).
2. *Max-min*. Solve the max-min problem

$$\max_{\boldsymbol{p} \in \mathcal{U}_B} \min_{\boldsymbol{x} \in \mathcal{X}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} p_i (n + 1 - j) x_{ij}$$

   to obtain a worst-case scenario $\boldsymbol{p} \in \mathcal{U}_B$. Find a schedule $\boldsymbol{x}$ that performs best in this worst-case scenario and evaluate by solving Adv($\boldsymbol{x}$).
3. *Min-max*. Solve the min-max problem

$$\min_{\boldsymbol{x} \in \mathcal{X}} \max_{\boldsymbol{p} \in \mathcal{U}_B} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{N}} p_i (n + 1 - j) x_{ij}$$

   and evaluate by solving Adv($\boldsymbol{x}$).

Note that min-max is equivalent to solving the recoverable robust problem for $\Delta = 0$. Observe that, by construction, the max-min solution will perform worse for the recoverable robust problem with $\Delta = 0$ than the min-max solution. We can thus conjecture that min-max performs also better than max-min for other values of Δ.

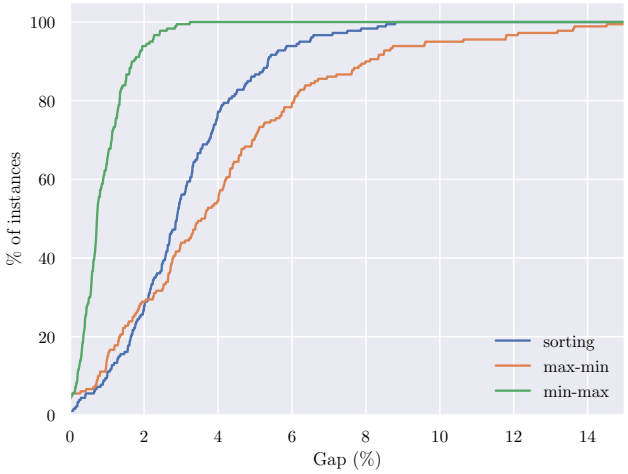

Fig. 3 Cumulative percentage of instances solved to within a given gap of the best known solution

Each heuristic method has been used to find a feasible solution to all 180 uncertain test instances. Figure 3 shows the cumulative percentage of instances solved by each of the heuristics to within a given gap to the best solution found by any method, including the exact models, which have been solved with $\Delta = 2$. It is clear from this plot that min-max is the strongest of the three proposed heuristics, solving all 180 instances to within 3.2% of the best solution. This gap increases to 8.8% for sorting, whilst max-min solves all but one instance to within 15%. The average of these gaps across all instances for min-max, sorting and max-min are 0.9%, 3.1% and 4.1% respectively.

Given its strong performance, we propose using min-max to provide a warm-start solution to the exact models. The benefits of this are assessed in the next section.

6.2 Exact models

We now examine the results of solving the three exact models proposed in this paper and their warm-start variants. The 180 uncertain instances have been solved by each model and its warm-start variant for $\Delta \in \{0, 1, 2, 3\}$. Note that the general model has been implemented with $K = 2$. This has been chosen to make the general model as computationally efficient to solve as possible, whilst actually still providing an advantage over the min-max model, i.e. for $K = 1$ the general model corresponds to the min-max model.

Tables 1 and 2 compare the performance of these exact models for different values of Γ and Δ respectively. For each set of 20 instances with the same combination of instance parameters, Tables 1 and 2 report the following:

- *Time* - Average CPU time (secs) required to solve the instances. Instances that are were solved within the time

Table 1 Comparison of the three exact models proposed in this paper and their warm-start variants, for different values of Γ

			General				General + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	3	2	600.0	100.0	0.2	0	600.0	100.0	0.2	0
10	5	2	600.0	100.0	0.1	0	600.0	100.0	0.1	0
10	7	2	600.0	100.0	0.1	0	600.0	100.0	0	0
15	3	2	600.0	100.0	0.4	0	600.0	100.0	0.3	0
15	5	2	600.0	100.0	0.6	0	600.0	100.0	0.3	0
15	7	2	600.0	100.0	0.4	0	600.0	100.0	0.3	0
20	3	2	600.0	100.0	0.8	0	600.0	100.0	0.3	0
20	5	2	600.0	100.0	0.9	0	600.0	100.0	0.4	0
20	7	2	600.0	100.0	1.2	0	600.0	100.0	0.5	0
						0				0
			Matching				Matching + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	3	2	2.6	0.0	0.0	20	2.6	0.0	0.0	20
10	5	2	4.1	0.0	0.0	20	4.1	0.0	0.0	20
10	7	2	2.1	0.0	0.0	20	3.4	0.0	0.0	20
15	3	2	135.3	0.0	0.0	19	110.0	0.0	0.0	20
15	5	2	202.6	0.0	0.0	17	159.0	0.0	0.0	18
15	7	2	209.4	0.0	0.0	16	198.1	0.0	0.0	16
20	3	2	591.1	0.8	0.0	3	586.4	0.8	0.0	2
20	5	2	599.9	0.5	0.0	1	600.0	0.6	0.0	0
20	7	2	600.0	0.5	0.0	0	600.0	0.4	0.0	0
						116				116
			Assignment				Assignment + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	3	2	8.0	0.0	0.0	20	8.2	0.0	0.0	20
10	5	2	6.8	0.0	0.0	20	5.1	0.0	0.0	20
10	7	2	3.4	0.0	0.0	20	3.0	0.0	0.0	20
15	3	2	50.9	0.0	0.0	20	55.4	0.0	0.0	20
15	5	2	62.6	0.0	0.0	20	59.3	0.0	0.0	20
15	7	2	59.1	0.0	0.0	20	64.3	0.0	0.0	20
20	3	2	344.0	0.0	0.0	20	242.5	0.0	0.0	19
20	5	2	377.7	0.0	0.0	20	292.2	0.0	0.0	19
20	7	2	453.2	0.0	0.0	19	321.3	0.0	0.0	20

limit are counted as having a CPU time of 600 s, i.e. equal to the time limit.

- *LBgap* - Average gap (%) between the best objective bound and the best known feasible solution found by any method.
- *UBgap* - Average gap (%) between the best feasible solution found within the time limit and the best known feasible solution found by any method.
- *#solv* - Number of instances solved to optimality within the time limit.

From Tables 1 and 2, it is clear that the general model is by far the weakest of the three proposed models. Other than for $\Delta = 0$, no instances are solved to optimality. The general model is able to find near-optimal feasible solutions, but fails to begin closing the optimality gap in most instances. The matching-based model improves considerably on the general model, whilst the assignment model is the strongest performing of the three exact models, solving the most number of instances to optimality and having the smallest gaps over those instances that cannot be solved to optimality. The addition of a warm-start solution is clearly beneficial only

Table 2 Comparison of the three exact models proposed in this paper and their warm-start variants, for different values of Δ

			General				General + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	7	0	0.3	0.0	0.0	20	0.3	0.0	0.0	20
10	7	1	600.0	100.0	0.0	0	600.0	100.0	0.0	0
10	7	2	600.0	100.0	0.1	0	600.0	100.0	0.0	0
10	7	3	600.0	100.0	0.0	0	600.0	100.0	0.0	0
15	7	0	3.3	0.0	0.0	20	3.0	0.0	0.0	20
15	7	1	600.0	100.0	0.4	0	600.0	100.0	0.3	0
15	7	2	600.0	100.0	0.4	0	600.0	100.0	0.3	0
15	7	3	600.0	100.0	0.5	0	600.0	100.0	0.3	0
20	7	0	122.5	0.1	0.0	18	133.0	0.1	0.0	18
20	7	1	600.0	100.0	1.1	0	600.0	100.0	0.7	0
20	7	2	600.0	100.0	1.2	0	600.0	100.0	0.5	0
20	7	3	600.0	100.0	1.5	0	600.0	100.0	0.5	0
						58				58
			Matching				Matching + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	7	0	0.0	0.0	0.0	20	0.0	0.0	0.0	20
10	7	1	2.1	0.0	0.0	20	2.2	0.0	0.0	20
10	7	2	2.1	0.0	0.0	20	3.4	0.0	0.0	20
10	7	3	5.3	0.0	0.0	20	5.9	0.0	0.0	20
15	7	0	0.2	0.0	0.0	20	0.2	0.0	0.0	20
15	7	1	383.9	0.1	0.0	11	337.6	0.1	0.0	12
15	7	2	209.4	0.0	0.0	16	198.1	0.0	0.0	16
15	7	3	146.2	0.0	0.0	17	178.2	0.0	0.0	16
20	7	0	1.6	0.0	0.0	20	1.6	0.0	0.0	20
20	7	1	600.0	0.9	0.0	0	600.0	0.9	0.0	0
20	7	2	600.0	0.5	0.0	0	600.0	0.4	0.0	0
20	7	3	518.1	0.2	0.0	8	556.4	0.1	0.0	8
						172				172
			Assignment				Assignment + warm-start			
n	Γ	Δ	*time*	*LBgap*	*UBgap*	*#solv*	*time*	*LBgap*	*UBgap*	*#solv*
10	7	0	0.0	0.0	0.0	20	0.0	0.0	0.0	20
10	7	1	2.8	0.0	0.0	20	2.0	0.0	0.0	20
10	7	2	3.4	0.0	0.0	20	3.0	0.0	0.0	20
10	7	3	5.7	0.0	0.0	20	4.1	0.0	0.0	20
15	7	0	0.2	0.0	0.0	20	0.1	0.0	0.0	20
15	7	1	97.5	0.0	0.0	19	86.8	0.0	0.0	20
15	7	2	59.1	0.0	0.0	20	64.3	0.0	0.0	20
15	7	3	42.4	0.0	0.0	20	50.3	0.0	0.0	20
20	7	0	1.5	0.0	0.0	20	1.3	0.0	0.0	20
20	7	1	583.4	0.3	0.0	2	577.0	0.2	0.0	6
20	7	2	453.2	0.0	0.0	19	321.3	0.0	0.0	20
20	7	3	443.4	0.0	0.0	19	312.0	0.0	0.0	20

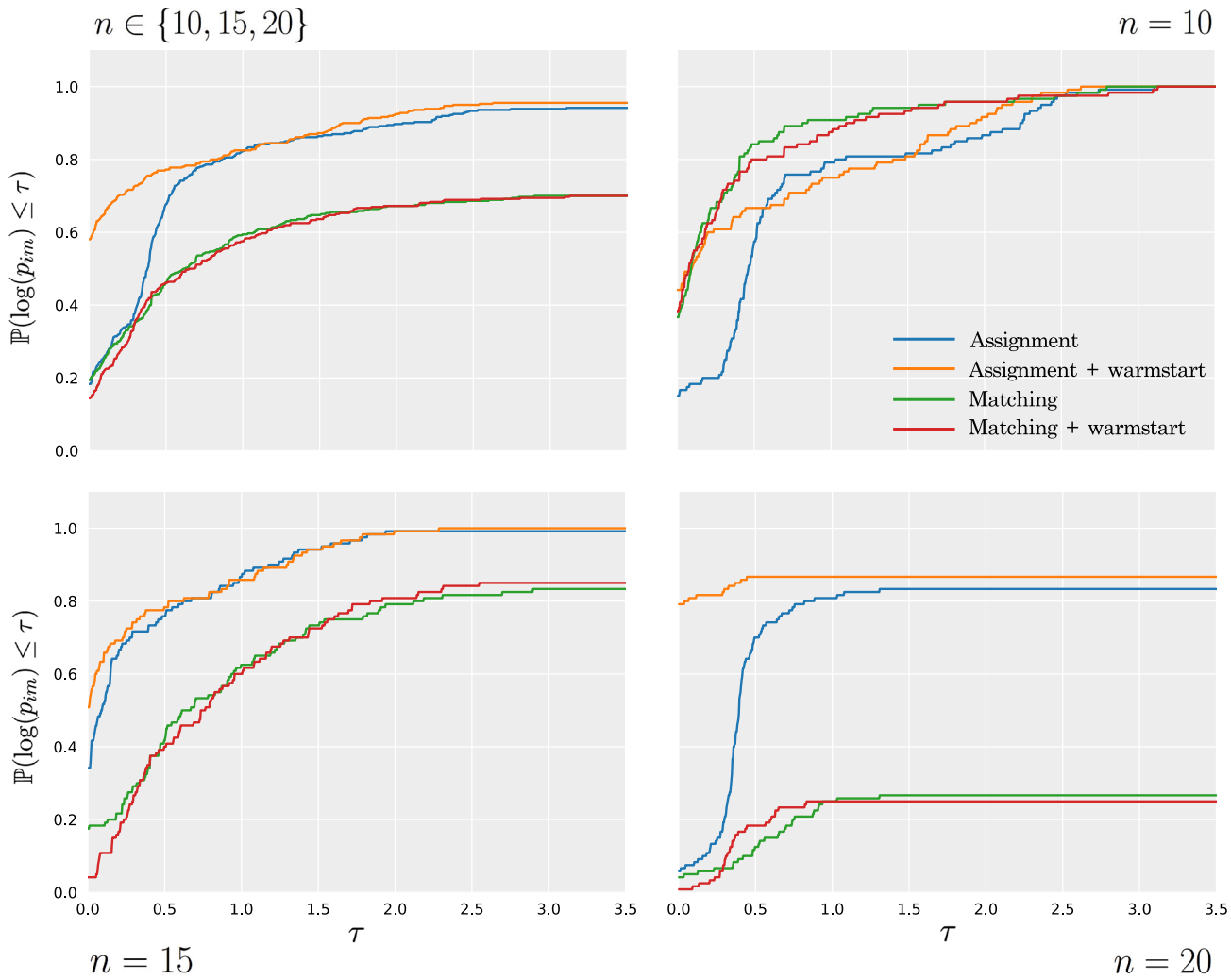

Fig. 4 Performance profiles of relative solution times for different instance sizes

for the assignment-based model, where the addition solves more instances to optimality in less time.

From Table 1 it can be seen that instances tend to become harder to solve as Γ increases from 3 to 7. From Table 2 we observe that, unsurprisingly, instances are easiest to solve to solve when $\Delta = 0$ (this corresponds to solving the min-max model). Interestingly however, when $n = 15$ and $n = 20$, instances are most difficult when $\Delta = 1$, and become easier to solve as the number of recovery swaps allowed, Δ, increases, i.e. the second stage-solution becomes less constrained by the first-stage solution.

Figure 4 shows performance profiles (Dolan and Moré, 2002) of the relative solution times of the matching and assignment-based models and their warm-start variants, for different instance sizes. The general model and its warm-start variant is excluded from these plots given its poor performance. A performance profile is a graphical comparison of the *performance ratios*. The performance ratio of model $m \in \mathcal{M}$ for instance $i \in \mathcal{I}$ is defined as

$$p_{im} = \frac{t_{im}}{\min_{m \in \mathcal{M}} t_{im}},$$

where t_{im} is the time required to solve instance i using model m. If model m fails to find an optimal solution to instance i within the given time limit, then $p_{im} = P$, for some $P > \max_{i,m} t_{im}$ (so long as P is chosen to be greater than the given time limit, the specific value chosen has no effect on the resulting plot). The performance profile of model $m \in \mathcal{M}$ is then defined to be the function

$$\rho_m(\tau) = \frac{|\{p_{im} \leq \tau : i \in \mathcal{I}\}|}{|\mathcal{I}|},$$

that is, the probability that model m is within a factor τ of the best performing model. The performance profiles in Fig. 4 have been plotted on the log-scale for clarity.

The top-left performance profile in Fig. 4 includes data from all instances, whilst the three other performance profiles consider the three sizes of instance separately. We see that for $n = 10$, the matching-based model performs slightly better than the assignment-based model, however the inclusion of a warm-start does not seem to improve the matching model. For $n = 15$ and $n = 20$ however, the assignment model is stronger than the matching model. The benefits of a warm-start solution become most apparent when solving the largest instances, where a warm-start decreases solution times and increases the number of instances solved to optimality of both the matching and assignment model.

6.3 Model parameters

We now examine the impact of the model parameters Γ and Δ on the objective value. For each set of instances, Tables 3 and 4 report the average objective value of the best known feasible solutions found by any method for different values of Γ and Δ respectively, as well as the relative percentage difference in this average from the sets of instances where $\Gamma = 3$ and $\Delta = 0$, respectively.

The results in Table 3 show that, as we would expect, increasing the Γ increases the average objective value in a concave manner. Table 4 shows that the inclusion of a second-stage recourse solution provides an improvement in objective value. However we also see that, for the instances we have solved, increasing Δ beyond $\Delta = 1$ provides little additional benefit. That is, the vast majority of the benefit of allowing a recourse solution can be captured by allowing just a single swap to the first-stage schedule. However, it is important to note the effect of having been limited to instances sizes of 20 and less by the computational intensity of solving the proposed exact models. We expect that for larger instance sizes, a less restricted and more powerful recourse action, i.e. increasing Δ, would become more advantageous. We can see an indication of the effect of the instance sizes in Table 4, with the increase of Δ from 1 to 2 having a bigger impact for the instances with 20, when compared to the instances with 10 jobs. Additionally, for a discrete budgeted uncertainty set where $p_i \in \{\hat{p}_i, \hat{p}_i + \bar{p}_i\}$ for each $i \in \mathcal{N}$, we might expect the benefits of increasing Δ to be more apparent, since in this case the adversary is unable to spread the delay across multiple jobs in an attempt to preempt the recourse response, as is currently the case under the continuous budgeted uncertainty set that we consider. The impact of discrete budgeted uncertainty is an interesting possibility for future research on this problem.

Table 3 The effects of increasing Γ on the average objective value of the best known solution

n	Γ	Δ	*avg. best*	*%diff.*
10	3	2	3946.5	0.0
10	5	2	4578.0	14.1
10	7	2	5053.0	22.1
15	3	2	7164.9	0.0
15	5	2	8177.5	12.7
15	7	2	9002.1	20.7
20	3	2	11814.3	0.0
20	5	2	13317.8	11.5
20	7	2	14582.5	19.3

Table 4 Effect of increasing Δ on the average objective value of the best known solution

n	Γ	Δ	*avg. best*
10	7	0	5079.7
10	7	1	5053.0
10	7	2	5053.0
10	7	3	5053.0
15	7	0	9093.8
15	7	1	9002.2
15	7	2	9002.1
15	7	3	9002.1
20	7	0	14735.6
20	7	1	14583.7
20	7	2	14582.5
20	7	3	14582.5

7 Conclusions

This paper has introduced a recoverable robust model for the single machine scheduling problem with the total flow time criterion. A general result that allows for the construction of compact formulations for a wide range of recoverable robust problems has been presented, and this approach has been applied to the specific scheduling problem we consider. We have analysed the incremental subproblem of the robust scheduling problem in detail in an attempt to develop more tailored and effective compact formulations for this problem. Specifically, we have proved that matching problems with edge weights of the form of (25) have integral solutions, and therefore the inclusion of the odd-cycle constraints of the standard matching polytope is unnecessary. This result allows us to derive a matching-based compact formulation for the full recoverable robust single machine scheduling problem. A symmetric assignment-based formulation has also been presented, and we show how the integral matching result can be transferred to this alternative formulation to enable the derivation of a third compact model for this problem. Computational results show that this assignment-based model is the strongest of the three exact models.

There remain a number of promising directions in which future research on this problem can develop. Firstly, in this work we have considered a limited recourse action of allowing Δ disjoint swaps to be made to the first-stage schedule. Other measures of distance between the first and second-stage solution are certainly possible and worth investigating, especially if the restriction that the swapped pairs be disjoint could be relaxed, and interchanges between the positions of three or more jobs simultaneously can be factored into a recourse action. Another obvious avenue for future research is the analysis of this problem in the context of uncertainty sets different from budgeted uncertainty. Given the vast number of different objective criteria that have been used for single-machine scheduling problems and the unique properties of each, it would be interesting and worthwhile to investigate the application of this recoverable robust model to some of these. As a final suggestion, given the limited size of instance that have been solved by the exact models we propose, an accurate and effective heuristic approach for solving large-scale instances of this problem would certainly be a valuable development.

A Omitted proofs

Proof of Theorem 6 Let an instance I of problem (45)-(47) be given, and define

$$P = \{\boldsymbol{y} \in \mathbb{R}_+^{n\times n} : (40) - (43)\}.$$

To solve I, we construct a graph with a single node for each job $i \in \mathcal{N}$, where an edge between nodes i and j indicates that jobs i and j swap their positions from the first-stage schedule. Since edges correspond to unique swaps, the set of edges in this graph is given by $\mathcal{E} = \{(i, j) : i, j \in \mathcal{N},\ j > i\}$. The weight of an edge (i, j) in this graph is equal to the reduction in objective cost from making the corresponding swap, i.e. $a_ib_i + a_jb_j - a_ib_j - a_jb_i = (a_i - a_j)(b_i - b_j)$. We aim to choose up to Δ edges from this graph to maximise the reduction in objective cost. That is, given I, we construct an instance J of the following cardinality-constrained matching problem:'

$$\min \sum_{i\in\mathcal{N}} a_ib_i - \sum_{(i,j)\in\mathcal{E}} (a_i - a_j)(b_i - b_j)z_{ij} \tag{57}$$

$$\text{s.t.} \sum_{(i,j)\in\mathcal{E}} z_{ij} + \sum_{(j,i)\in\mathcal{E}} z_{ij} \le 1 \qquad \forall i \in \mathcal{N} \tag{58}$$

$$\sum_{(i,j)\in\mathcal{E}} z_{ij} \le \Delta \tag{59}$$

$$z_{ij} \ge 0 \qquad \forall (i, j) \in \mathcal{E}. \tag{60}$$

As stated in Corollary 4, this problem has an integral optimal solution. Hence, letting $P' = \{\boldsymbol{z} \in \mathbb{R}_+^{|\mathcal{E}|} : (58)\text{-}(60)\}$, we construct mappings $\phi : P \to P'$ and $\phi^{-1} : P' \to P$ which preserve objective value and integrality, showing problems I and J are indeed equivalent. To this end, we define $\phi(\boldsymbol{y}) = \boldsymbol{z}$ by $z_{ij} = y_{ij}$ for all $(i, j) \in \mathcal{E}$. Observe that

$$\begin{aligned}
obj_I(\boldsymbol{y}) &= \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}} a_ib_jy_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_iy_{ii} + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j\neq i} a_ib_jy_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_iy_{ii} + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_j + a_jb_i)y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i\left(1 - \sum_{j\in\mathcal{N}:j\neq i} y_{ij}\right) \\
&\quad + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_j + a_jb_i)y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j\neq i} a_ib_iy_{ij} \\
&\quad + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_j + a_jb_i)y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_i + a_jb_j)y_{ij} \\
&\quad + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_j + a_jb_i)y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_i + a_jb_j - a_ib_j - a_jb_i)y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{(i,j)\in\mathcal{E}} (a_i - a_j)(b_i - b_j)z_{ij} \\
&= obj_J(\boldsymbol{z}).
\end{aligned}$$

Conversely, we define $\phi^{-1}(\boldsymbol{z}) = \boldsymbol{y}$ with $y_{ij} = y_{ji} = z_{ij}$ for each $(i, j) \in \mathcal{E}$, and $y_{ii} = (1 - \sum_{j\in\mathcal{N}:j>i} z_{ij})(1 - \sum_{j\in\mathcal{N}:j<i} z_{ji})$ for each $i \in \mathcal{N}$. Then

$$\begin{aligned}
obj_J(\boldsymbol{z}) &= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{(i,j)\in\mathcal{E}} (a_i - a_j)(b_i - b_j)z_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_i + a_jb_j)z_{ij} \\
&\quad + \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{N}:j>i} (a_ib_j + a_jb_i)z_{ij} \\
&= \sum_{i\in\mathcal{N}} a_ib_i - \sum_{i\in\mathcal{N}} a_ib_i \sum_{j\in\mathcal{N}:j>i} z_{ij} \\
&\quad - \sum_{i\in\mathcal{N}} a_ib_i \sum_{j\in\mathcal{N}:j<i} z_{ji}
\end{aligned}$$

$$
\begin{aligned}
&+ \sum_{i\in\mathcal{N}} a_i b_i \underbrace{\sum_{j\in\mathcal{N}:j>i} z_{ij} \sum_{j\in\mathcal{N}:j<i} z_{ji}}_{=0} \\
&+ \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} (a_i b_j + a_j b_i) z_{ij} \\
&= \sum_{i\in\mathcal{N}} a_i b_i \left(1 - \sum_{j\in\mathcal{N}:j>i} z_{ij}\right)\left(1 - \sum_{j\in\mathcal{N}:j<i} z_{ji}\right) \\
&+ \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} (a_i b_j + a_j b_i) z_{ij} \\
&= \sum_{i\in\mathcal{N}} a_i b_i y_{ii} + \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} a_i b_j y_{ij} \\
&+ \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} a_j b_i y_{ji} \\
&= \sum_{i\in\mathcal{N}} a_i b_i y_{ii} \\
&+ \sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j\neq i} a_i b_j y_{ij} \\
&= \sum_{i\in\mathcal{N}} a_i b_j y_{ij} \\
&= obj_I(\boldsymbol{y}).
\end{aligned}
$$

Therefore $\min_{\boldsymbol{y}\in P} obj_I(\boldsymbol{y}) = \min_{\boldsymbol{z}\in P'} obj_J(\boldsymbol{z})$. Since problem (28)-(31) has an optimal integral solution, there must also be an optimal integral solution to I via the mapping ϕ^{-1}, proving the claim. □

Proof of Theorem 8 To prove this, it will suffice to show that $P(F_m) \subsetneq P(F_a)$, where $P(F_m)$ and $P(F_a)$ denote the sets of feasible solutions to the linear relaxations of the non-linear matching-based and assignment-based formulations respectively, i.e. that for each $(\boldsymbol{x}, \boldsymbol{z}, \boldsymbol{q}) \in P(F_m)$, there exists a $\boldsymbol{y}$ such that $(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{q}) \in P(F_a)$. Note that the same vector $\boldsymbol{q}$ is used in both formulations, which results in the objective values being the same. To this end, let $\boldsymbol{z}$ be part of a feasible solution to the matching formulation, and define the transformation $\phi(\boldsymbol{z}) = \boldsymbol{y}$ by $y_{ij} = y_{ji} = z_{ij}$ for $(i, j) \in \mathcal{E} = \{(i, j) \in \mathcal{N} \times \mathcal{N} : i < j\}$ and $y_{ii} = 1 - \sum_{j\in\mathcal{N}:j>i} z_{ij} - \sum_{j\in\mathcal{N}:j<i} z_{ji}$ for $i \in \mathcal{N}$.

Firstly, observe that the assignment constraints (49) and (50) are satisfied by this definition of $\boldsymbol{y}$. For each $j \in \mathcal{N}$ we have that

$$
\begin{aligned}
\sum_{i\in\mathcal{N}} y_{ij} &= \sum_{i\in\mathcal{N}:i<j} y_{ij} + \sum_{i\in\mathcal{N}:i>j} y_{ji} + y_{ii} \\
&= \sum_{i\in\mathcal{N}:i<j} z_{ij} + \sum_{i\in\mathcal{N}:i>j} z_{ji} \\
&\quad + 1 - \sum_{i\in\mathcal{N}:i<j} z_{ij} - \sum_{i\in\mathcal{N}:i>j} z_{ji} = 1.
\end{aligned}
$$

The same can be shown for the 'incoming' assignment constraints for each $i \in \mathcal{N}$.

Now observe that constraint (35) states that

$$
\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} z_{ij} \leq \Delta,
$$

or equivalently

$$
\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}:j>i} y_{ij} \leq \Delta.
$$

Since, $y_{ij} = y_{ji}$ for all $i,\ j \in \mathcal{N}$, we have that

$$
\sum_{i\in\mathcal{N}} \left(\sum_{j\in\mathcal{N}:j>i} y_{ij} + \sum_{j\in\mathcal{N}:j<i} y_{ij} \right) \leq 2\Delta,
$$

and therefore as a consequence of the assignment constraints, which imply

$$
\sum_{i\in\mathcal{N}} \sum_{j\in\mathcal{N}} y_{ij} = \sum_{i\in\mathcal{N}} \left(\sum_{j\in\mathcal{N}:j>i} y_{ij} + \sum_{j\in\mathcal{N}:j<i} y_{ij} + y_{ii} \right) = n,
$$

we have that

$$
\sum_{i\in\mathcal{N}} y_{ii} = n - \sum_{i\in\mathcal{N}} \left(\sum_{j\in\mathcal{N}:j>i} y_{ij} + \sum_{j\in\mathcal{N}:j<i} y_{ij} \right) \geq n - 2\Delta.
$$

This is exactly constraint (51) from the assignment-based formulation.

Finally, consider constraints (33):

$$
\begin{aligned}
&\sum_{m\in\mathcal{M}} a_{mi} q_m + \sum_{j\in\mathcal{N}:j>i} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} \right) z_{ij} \\
&\quad - \sum_{j\in\mathcal{N}:j<i} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} \right. \\
&\quad \left. - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} \right) z_{ji} \geq (n + 1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}) \quad \forall i \in \mathcal{N}.
\end{aligned}
$$

These can be rewritten as

$$
\begin{aligned}
&\sum_{m\in\mathcal{M}} a_{mi} q_m \geq (n + 1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}) \\
&\quad + \sum_{j\in\mathcal{N}:j<i} z_{ji} \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} + \sum_{j\in\mathcal{N}:j>i} z_{ij} \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j<i} z_{ji} \sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j>i} z_{ij} \sum_{\ell\in\mathcal{N}} \ell \cdot x_{j\ell}
\end{aligned}
$$

$$
\begin{aligned}
&= (n+1-\sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell}) + \sum_{j\in\mathcal{N}:j<i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell} \\
&\quad + \sum_{j\in\mathcal{N}:j>i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j<i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j>i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= (n+1-\sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell}) + \sum_{j\in\mathcal{N}:j\neq i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j\neq i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= n+1-\left(1-\sum_{j\in\mathcal{N}:j\neq i} y_{ij}\right) \sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j\neq i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= n+1-y_{ii} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{i\ell} \\
&\quad - \sum_{j\in\mathcal{N}:j\neq i} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= n+1-\sum_{j\in\mathcal{N}} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= (n+1) \sum_{j\in\mathcal{N}} y_{ij} \\
&\quad - \sum_{j\in\mathcal{N}} y_{ij} \sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell} \\
&= \sum_{j\in\mathcal{N}} \big(n+1-\sum_{\ell\in\mathcal{N}} \ell\cdot x_{j\ell}\big) y_{ij},
\end{aligned}
$$

for each $i \in \mathcal{N}$ which are constraints (53) from the assignment formulation.

Hence, every feasible solution to the non-linear matching-based formulation has a corresponding feasible solution for the non-linear assignment-based formulation, i.e. $P(F_m) \subseteq P(F_a)$. The examples used in the proof of Theorem 7 show that there exist instances for which the LP bound of the matching formulation is strictly larger than that of the assignment formulation, demonstrating that $P(F_m) \subset P(F_a)$. □

B Complete formulations

B.1 General model

The complete linear compact formulation for the general recoverable robust model presented in Sect. 3 is as follows:

$$\min \sum_{m\in\mathcal{M}} b_m q_m \tag{61}$$

$$\text{s.t.} \sum_{k\in\mathcal{K}} \mu_k = 1 \tag{62}$$

$$\sum_{m\in\mathcal{M}} a_{mi} q_m \geq \sum_{k\in\mathcal{K}} \left(\sum_{j\in\mathcal{N}} (n+1-j) \sum_{i'\in\mathcal{N}} h^k_{ii'j} \right) \quad \forall i \in \mathcal{N} \tag{63}$$

$$\sum_{i'\in\mathcal{N}} z^k_{ii'} = 1 \qquad \forall i \in \mathcal{N},\ k \in \mathcal{K} \tag{64}$$

$$\sum_{i\in\mathcal{N}} z^k_{ii'} = 1 \qquad \forall i' \in \mathcal{N},\ k \in \mathcal{K} \tag{65}$$

$$z^k_{ii'} = z^k_{i'i} \qquad \forall i, i' \in \mathcal{N},\ k \in \mathcal{K} \tag{66}$$

$$\sum_{i\in\mathcal{N}} z^k_{ii} \geq n - 2\Delta \qquad \forall k \in \mathcal{K} \tag{67}$$

$$\sum_{j\in\mathcal{N}} x_{ij} = 1 \qquad \forall i \in \mathcal{N} \tag{68}$$

$$\sum_{i\in\mathcal{N}} x_{ij} = 1 \qquad \forall j \in \mathcal{N} \tag{69}$$

$$w^k_{ii'j} \leq z^k_{ii'} \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{70}$$

$$w^k_{ii'j} \leq x_{i'j} \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{71}$$

$$w^k_{ii'j} \geq z^k_{ii'} + x_{i'j} - 1 \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{72}$$

$$h^k_{ii'j} \leq w^k_{ii'j} \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{73}$$

$$h^k_{ii'j} \leq \mu_k \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{74}$$

$$h^k_{ii'j} \geq \mu_k + w^k_{ii'j} - 1 \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{75}$$

$$w^k_{ii'j} \in \{0,1\} \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{76}$$

$$h^k_{ii'j} \geq 0 \qquad \forall i, i', j \in \mathcal{N},\ k \in \mathcal{K} \tag{77}$$

$$\mu_k \geq 0 \qquad \forall k \in \mathcal{K} \tag{78}$$

$$q_m \geq 0 \qquad \forall m \in \mathcal{M} \tag{79}$$

$$z^k_{ii'} \in \{0,1\} \qquad \forall i,\ i' \in \mathcal{N},\ k \in \mathcal{K} \tag{80}$$

$$x_{ij} \in \{0,1\} \qquad \forall i,\ j \in \mathcal{N}. \tag{81}$$

B.2 Matching-based model

The fully-linearised formulation of the matching-based model presented in Sect. 4.1 is as follows:

$$\min_{\boldsymbol{x},\boldsymbol{z},\boldsymbol{u},\boldsymbol{v},\boldsymbol{q}} \sum_{m\in\mathcal{M}} b_m q_m \tag{82}$$

$$\text{s.t.} \sum_{(i,j)\in\mathcal{E}} z_{ij} + \sum_{(j,i)\in\mathcal{E}} z_{ji} \leq 1 \qquad \forall i \in \mathcal{N} \tag{83}$$

$$\sum_{(i,j)\in\mathcal{E}} z_{ij} \le \Delta \tag{84}$$

$$\sum_{m\in\mathcal{M}} a_{mi}q_m + \sum_{(i,j)\in\mathcal{E}} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot v_{ij\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot u_{ij\ell} \right) - \sum_{(j,i)\in\mathcal{E}} \left(\sum_{\ell\in\mathcal{N}} \ell \cdot v_{ji\ell} - \sum_{\ell\in\mathcal{N}} \ell \cdot u_{ji\ell} \right) \ge (n+1 - \sum_{\ell\in\mathcal{N}} \ell \cdot x_{i\ell}) \qquad \forall i \in \mathcal{N} \tag{85}$$

$$\sum_{i\in\mathcal{N}} x_{i\ell} = 1 \qquad \forall \ell \in \mathcal{N} \tag{86}$$

$$\sum_{\ell\in\mathcal{N}} x_{i\ell} = 1 \qquad \forall i \in \mathcal{N} \tag{87}$$

$$u_{ij\ell} \le x_{i\ell} \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{88}$$

$$u_{ij\ell} \le z_{ij} \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{89}$$

$$u_{ij\ell} \ge z_{ij} + x_{i\ell} - 1 \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{90}$$

$$v_{ij\ell} \le x_{j\ell} \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{91}$$

$$v_{ij\ell} \le z_{ij} \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{92}$$

$$v_{ij\ell} \ge z_{ij} + x_{j\ell} - 1 \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{93}$$

$$u_{ij\ell} \ge 0 \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{94}$$

$$v_{ij\ell} \ge 0 \qquad \forall (i,j) \in \mathcal{E},\ \ell \in \mathcal{N} \tag{95}$$

$$q_m \ge 0 \qquad \forall m \in \mathcal{M} \tag{96}$$

$$z_{ij} \ge 0 \qquad \forall (i,j) \in \mathcal{E} \tag{97}$$

$$x_{i\ell} \in \{0,1\} \qquad \forall i, \ell \in \mathcal{N}. \tag{98}$$

2.3 Assignment-based model

The fully-linearised formulation of the assignment-based model derived in Sect. 4.2 is as follows:

$$\min_{\boldsymbol{x},\boldsymbol{y},\boldsymbol{w},\boldsymbol{q}} \sum_{m\in\mathcal{M}} b_m q_m \tag{99}$$

$$\text{s.t.} \sum_{i\in\mathcal{N}} y_{ij} = 1 \qquad \forall j \in \mathcal{N} \tag{100}$$

$$\sum_{j\in\mathcal{N}} y_{ij} = 1 \qquad \forall i \in \mathcal{N} \tag{101}$$

$$\sum_{i\in\mathcal{N}} y_{ii} \ge n - 2\Delta \tag{102}$$

$$y_{ij} = y_{ji} \qquad \forall i, j \in \mathcal{N} \tag{103}$$

$$\sum_{m\in\mathcal{M}} a_{mi}q_m \ge \sum_{j\in\mathcal{N}} \left((n+1)y_{ij} - \sum_{\ell\in\mathcal{N}} \ell \cdot w_{ij\ell} \right) \qquad \forall i \in \mathcal{N} \tag{104}$$

$$\sum_{i\in\mathcal{N}} x_{i\ell} = 1 \qquad \forall \ell \in \mathcal{N} \tag{105}$$

$$\sum_{\ell\in\mathcal{N}} x_{i\ell} = 1 \qquad \forall i \in \mathcal{N} \tag{106}$$

$$w_{ij\ell} \le x_{j\ell} \qquad \forall i, j, \ell \in \mathcal{N} \tag{107}$$

$$w_{ij\ell} \le y_{ij} \qquad \forall i, j, \ell \in \mathcal{N} \tag{108}$$

$$w_{ij\ell} \ge x_{j\ell} + y_{ij} - 1 \qquad \forall i, j, \ell \in \mathcal{N} \tag{109}$$

$$w_{ij\ell} \ge 0 \qquad \forall i, j, \ell \in \mathcal{N} \tag{110}$$

$$q_m \ge 0 \qquad \forall m \in \mathcal{M} \tag{111}$$

$$y_{ij} \ge 0 \qquad \forall i, j \in \mathcal{N} \tag{112}$$

$$x_{i\ell} \in \{0,1\} \qquad \forall i, \ell \in \mathcal{N}. \tag{113}$$

Acknowledgements The authors would like to thank the reviewers for the part they played in the improvement of this paper with their constructive and insightful feedback. The authors are also grateful for the support of the EPSRC-funded (EP/L015692/1) STOR-i Centre for Doctoral Training.

References

Aissi, H., Aloulou, M. A., & Kovalyov, M. Y. (2011). Minimizing the number of late jobs on a single machine under due date uncertainty. *Journal of Scheduling, 14*(4), 351–360.

Aloulou, M. A., & Della Croce, F. (2008). Complexity of single machine scheduling problems under scenario-based uncertainty. *Operations Research Letters, 36*(3), 338–342.

Balinski, M. L. (1965). Integer programming: Methods, uses, computations. *Management Science, 12*(3), 253–313.

Bendotti, P., Chrétienne, P., Fouilhoux, P., & Pass-Lanneau, A. (2022). The anchor-robust project scheduling problem. *Operations Research*.

Bertsimas, D., & Sim, M. (2004). The price of robustness. *Operations Research, 52*(1), 35–53.

Bold, M., & Goerigk, M. (2021). A compact reformulation of the two-stage robust resource-constrained project scheduling problem. *Computers & Operations Research, 130*, 105232.

Bold, M., & Goerigk, M. (2022). Investigating the recoverable robust single machine scheduling problem under interval uncertainty. *Discrete Applied Mathematics, 313*, 99–114.

Bougeret, M., Pessoa, A. A., & Poss, M. (2019). Robust scheduling with budgeted uncertainty. *Discrete Applied Mathematics, 261*, 93–107.

Bruni, M. E., Pugliese, L. D. P., Beraldi, P., & Guerriero, F. (2017). An adjustable robust optimization model for the resource-constrained project scheduling problem with uncertain activity durations. *Omega, 71*, 66–84.

Bruni, M. E., Pugliese, L. D. P., Beraldi, P., & Guerriero, F. (2018). A computational study of exact approaches for the adjustable robust resource-constrained project scheduling problem. *Computers & Operations Research, 99*, 178–190.

Buchheim, C., & Kurtz, J. (2017). Min-max-min robust combinatorial optimization. *Mathematical Programming, 163*(1–2), 1–23.

Chang, Z., Song, S., Zhang, Y., Ding, J.-Y., Zhang, R., & Chiong, R. (2017). Distributionally robust single machine scheduling with

risk aversion. *European Journal of Operational Research, 256*(1), 261–274.

Daniels, R. L., & Kouvelis, P. (1995). Robust scheduling to hedge against processing time uncertainty in single-stage production. *Management Science, 41*(2), 363–376.

Dolan, E. D., & Moré, J. J. (2002). Benchmarking optimization software with performance profiles. *Mathematical Programming, 91*(2), 201–213.

Edmonds, J. (1965). Maximum matching and a polyhedron with 0,1-vertices. *Journal of Research of the National Bureau of Standards, 69B*, 125–130.

Fischer, D., Hartmann, T. A., Lendl, S., & Woeginger, G. J. (2021). An investigation of the recoverable robust assignment problem. In *16th International Symposium on Parameterized and Exact Computation (IPEC 2021)*, vol. 214, pp. 19:1–19:14. Schloss Dagstuhl.

Fridman, I., Pesch, E., & Shafransky, Y. (2020). Minimizing maximum cost for a single machine under uncertainty of processing times. *European Journal of Operational Research, 286*(2), 444–457.

Graham, R. L., Lawler, E. L., Lenstra, J. K., & Kan, A. R. (1979). Optimization and approximation in deterministic sequencing and scheduling: a survey. In *Annals of discrete mathematics*, vol. 5, pp. 287–326. Elsevier.

Hanasusanto, G. A., Kuhn, D., & Wiesemann, W. (2015). K-adaptability in two-stage robust binary programming. *Operations Research, 63*(4), 877–891.

Kasperski, A. & Zielinski, P. (2014). Minmax (regret) scheduling problems. *Sequencing and scheduling with inaccurate data*, pp. 159–210.

Kasperski, A. (2005). Minimizing maximal regret in the single machine sequencing problem with maximum lateness criterion. *Operations Research Letters, 33*(4), 431–436.

Kasperski, A., & Zieliński, P. (2008). A 2-approximation algorithm for interval data minmax regret sequencing problems with the total flow time criterion. *Operations Research Letters, 36*(3), 343–344.

Kasperski, A., & Zieliński, P. (2016). Single machine scheduling problems with uncertain parameters and the OWA criterion. *Journal of Scheduling, 19*(2), 177–190.

Kasperski, A., & Zieliński, P. (2019). Risk-averse single machine scheduling: Complexity and approximation. *Journal of Scheduling, 22*(5), 567–580.

Kouvelis, P., & Yu, G. (1997). *Robust discrete optimization and its applications*. Netherlands: Kluwer Academic Publishers Dordrecht.

Lebedev, V., & Averbakh, I. (2006). Complexity of minimizing the total flow time with interval data and minmax regret criterion. *Discrete Applied Mathematics, 154*(15), 2167–2177.

Liebchen, C., Lübbecke, M., Möhring, R., & Stiller, S. (2009). The concept of recoverable robustness, linear programming recovery, and railway applications. In *Robust and online large-scale optimization*, pp. 1–27. Springer.

Lu, C.-C., Lin, S.-W., & Ying, K.-C. (2012). Robust scheduling on a single machine to minimize total flow time. *Computers & Operations Research, 39*(7), 1682–1691.

Lu, C.-C., Ying, K.-C., & Lin, S.-W. (2014). Robust single machine scheduling for minimizing total flow time in the presence of uncertain processing times. *Computers & Industrial Engineering, 74*, 102–110.

Mastrolilli, M., Mutsanas, N., & Svensson, O. (2013). Single machine scheduling with scenarios. *Theoretical Computer Science, 477*, 57–66.

Montemanni, R. (2007). A mixed integer programming formulation for the total flow time single machine robust scheduling problem with interval data. *Journal of Mathematical Modelling and Algorithms, 6*(2), 287–296.

Schrijver, A. (2003). *Combinatorial optimization: Polyhedra and efficiency* (Vol. 24). New York: Springer Science & Business Media.

Tadayon, B., & Smith, J. C. (2015). Algorithms and complexity analysis for robust single-machine scheduling problems. *Journal of Scheduling, 18*(6), 575–592.

Thomas, D. J. (2015). *Matching Problems with Additional Resource Constraints*. PhD thesis, Universität Trier.

Yang, J., & Yu, G. (2002). On the robust single machine scheduling problem. *Journal of Combinatorial Optimization, 6*(1), 17–33.

Zhao, H., Zhao, M., et al. (2010). A family of inequalities valid for the robust single machine scheduling polyhedron. *Computers & Operations Research, 37*(9), 1610–1614.