Minimizing the weighted sum of completion times under processing time uncertainty

Zacharie Alès, Thi Sang Nguyen, Michael Poss

Minimizing the weighted sum of completion times under processing time uncertainty

Zacharie ALES [2]

Laboratoire Informatique d'Avignon
University of Avignon
84911 Avignon, France

Thi Sang NGUYEN [3]

International Francophone Institute
C3- 144, Xuan Thuy, Hanoi, Vietnam

Michael POSS [1]

UMR CNRS 5506 LIRMM, Université de Montpellier, 161 rue Ada, 34392 Montpellier Cedex 5, France

Abstract

We address the robust counterpart of a classical single machine scheduling problem by considering a budgeted uncertainty and an ellipsoidal uncertainty. We prove that the problem is $\mathcal{NP}$-hard for arbitrary ellipsoidal uncertainty sets. Then, a mixed-integer linear programming reformulations and a second order cone programming reformulations are provided. We assess the reformulations on randomly generated instances, comparing them with branch-and-cut algorithms.

Keywords: Integer programming, robust optimization, scheduling.

1 Introduction

Scheduling is a rich topic within combinatorial optimization that has witnessed a large amount of research in the past decades, including applications oriented works, integer programming formulations, polyhedral studies, and approximation algorithms. In this work, we focus in this work on one of the simplest scheduling problem, which can be defined as follows. We are given a set $\mathcal{J} = \{1, \ldots, n\}$ of jobs, each having a processing time p_j and a weight w_j, and we would like to order the jobs so as to minimize the weighted sum of their completion times. Formally, letting $\sigma(i)$ be the position of job i for the permutation σ and $C_j(\sigma) = \sum_{i=1}^{\sigma(j)} p_{\sigma^{-1}(i)}$ be the completion time of job j for permutation σ, we want to solve the optimization problem

$$\min_{\sigma \in P(n)} \sum_{j \in \mathcal{J}} w_j C_j(\sigma), \tag{1}$$

where $P(n)$ represents the set of permutations of $\{1, \ldots, n\}$. It is well-known that Problem (1) can be solved in polynomial time by ordering the jobs according to their non-decreasing value of p_j/w_j, which is known as Smith's rule [12].

In practical scheduling problems, the parameters of the problem are usually subject to variations, and this is particularly true for the vector of processing times p, whose value can be affected by various hazardous events, such as machine breakdowns, working environment changes, worker performance instabilities, to cite a few. We address this issue herein through the lens of min max robust optimization. Specifically, we assume that the uncertainty over p is characterized by a given convex set $U \subset \mathbb{R}^n$, and we study the robust counterpart of (1) that is defined as

$$\min_{\sigma \in P(n)} \max_{p \in U} \sum_{j \in \mathcal{J}} w_j C_j(\sigma, p), \tag{2}$$

where $C_j(\sigma, p) = \sum_{i=1}^{\sigma(j)} p_{\sigma^{-1}(i)}$ denotes the completion time of job j for permutation σ and the vector of processing times taking value p.

For arbitrary uncertainty sets U, it is well known that Problem (2) is $\mathcal{NP}$-hard, see [14], even when $w_j = 1$ for each $j \in \mathcal{J}$ and U is the convex

[1] Email: michael.poss@lirmm.fr
[2] Email: zacharie.ales@univ-avignon.fr
[3] Email: ntsang@ifi.edu.vn

hull of two vectors. This is not a surprising result, as it is well-known that robust combinatorial optimization problems with arbitrary uncertainty sets are, more often than not, harder than their deterministic counterparts [1]. While arbitrary uncertainty sets offer little hope for efficient algorithmic solutions, Bertsimas and Sim [4] have proposed a specific uncertainty set that preserves the complexity of many robust combinatorial optimization problems. In our context, their set can be defined as follows. Given two positive vectors $\overline{p}$ and $\hat{p}$ that respectively represent the nominal value of and the deviation of p, and a positive integer Γ, we consider the set

$$U^{\Gamma} \equiv \left\{ p \in \mathbb{R}^n : p_j = \overline{p}_j + \delta_j \hat{p}_j, j \in \mathcal{J}, \delta \in \{0,1\}^n, j \in \mathcal{J}, \sum_{j \in \mathcal{J}} \delta_j \leq \Gamma \right\}.$$

Following the notations of [6,9], let us denote Problem (2) for set U^{Γ} by $1||U_p^{\Gamma}|\sum_j w_j C_j$. When $w_j = 1$ for each $j \in \mathcal{J}$, which is denoted $1||U_p^{\Gamma}|\sum_j C_j$, Bougeret et al. [6] have proved that the problem can be solved in $O(n^5)$. Tadayon and Smith [13] have proposed a faster algorithm when $\hat{p}_j = \kappa \overline{p}_j$ for some $\kappa > 0$, running in $O(n \log n)$. On the negative side, the problem is $\mathcal{NP}$-hard in the strong sense for arbitrary weights [6].

The hardness result of [6] was the initial motivation for the current work. Specifically, our first contribution is to provide a preliminary mixed-integer linear programming study of $1||U_p^{\Gamma}|\sum_j w_j C_j$. Our second contribution concerns Problem $1||U_p^{\Omega}|\sum_j w_j C_j$, which is defined as Problem (2) when considering the ellipsoidal uncertainty set defined by the positive definite matrix $\Sigma \in \mathbb{R}^{n \times n}$

$$U^{\Omega} \equiv \left\{ p \in \mathbb{R}^n : p_j = \overline{p}_j + \delta_j \hat{p}_j, j \in \mathcal{J}, \|\Sigma^{-\frac{1}{2}} \delta\|_2 \leq \Omega \right\}.$$

We prove that $1||U_p^{\Omega}|\sum_j w_j C_j$ is $\mathcal{NP}$-hard and assess its numerical difficulty through mixed-integer second order cone programming reformulations. The hardness result is provided in Section 2. Section 3 presents our integer programming formulations and Section 4 reports our numerical experiments.

2 Complexity of $1||U_p^{\Omega}|\sum_j w_j C_j$

Suppose that $w_j = 1$ for each $j \in \mathcal{J}$. Given any permutation $\sigma \in P(n)$, notice that

$$\sum_{j \in \mathcal{J}} C_j(\sigma, p) = \sum_{j \in \mathcal{J}} (n + 1 - \sigma(j)) p_j. \quad (3)$$

To obtain a more compact writing for (3), we define in the following $\sigma^*(j) = n+1-\sigma(j)$. We deduce from the definition of U^Ω and a well-known result in convex optimization that $\max_{p\in U^\Omega}\sum_{j\in\mathcal{J}} C_j(\sigma,p) = \sum_{j\in\mathcal{J}}\sigma^*(j)\overline{p}_j + \Omega\sqrt{\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\sigma^*(i)\sigma^*(j)\hat{p}_i\hat{p}_j\Sigma_{ij}}$. We prove below that problem

$$\min_{\sigma\in P(n)}\sum_{j\in\mathcal{J}}\sigma^*(j)\overline{p}_j + \Omega\sqrt{\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\sigma^*(i)\sigma^*(j)\hat{p}_i\hat{p}_j\Sigma_{ij}} \tag{4}$$

is $\mathcal{NP}$-hard for positive semi-definite matrices Σ. Strictly speaking, Problem (4) is more general that Problem $1||U_p^\Omega|\sum_j w_jC_j$ because the former allows Σ to a be a singular matrix, in which case U^Ω is not defined. However, one can show that both problems are equivalent by perturbing any singular matrix by a very small term to make it positive definite, see [7] for details.

Theorem 2.1 *Problem* $1||U_p^\Omega|\sum_j w_jC_j$ *is* $\mathcal{NP}$*-hard.*

Proof. Let p^1 and p^2 be two arbitrary vectors in $\mathbb{R}^n_+$. Yang and Yu [14] proved that the problem

$$\min\nolimits_{\sigma\in P(n)}\max_{p\in\{p^1,p^2\}}\sum_{j\in\mathcal{J}} C_j(\sigma,p) \tag{5}$$

is $\mathcal{NP}$-hard. We prove below that Problem (5) reduces to (4) by appropriate choices of $\overline{p}$, $\hat{p}$, Σ and Ω. Specifically, for each $j\in\mathcal{J}$, we let $\overline{p}_j = \frac{p_j^1+p_j^2}{2}$ and $\hat{p}_j = \frac{p_j^1-p_j^2}{2}$, and define $\Omega = 1$ and $\Sigma_{ij} = 1$ for each $i,j\in\mathcal{J}$. Consider any $\sigma\in P(n)$. The objective function of (4) becomes

$$\begin{aligned}
&\sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1+p_j^2}{2} + \sqrt{\sum_{i\in J}\sum_{j\in\mathcal{J}}\sigma^*(i)\sigma^*(j)\frac{p_i^1-p_i^2}{2}\frac{p_j^1-p_j^2}{2}}\\
&= \sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1+p_j^2}{2} + \sqrt{\left(\sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1-p_j^2}{2}\right)^2}\\
&= \sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1+p_j^2}{2} + \left|\sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1-p_j^2}{2}\right|\\
&= \sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1+p_j^2}{2} + \max\left\{\sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^1-p_j^2}{2}, \sum_{j\in\mathcal{J}}\sigma^*(j)\frac{p_j^2-p_j^1}{2}\right\}\\
&= \max\left\{\sum_{j\in\mathcal{J}}\sigma^*(j)p_j^1, \sum_{j\in\mathcal{J}}\sigma^*(j)p_j^2\right\} = \max_{p\in\{p^1,p^2\}}\sum_{j\in\mathcal{J}} C_j(\sigma,p),
\end{aligned}$$

proving the result. □

3 Integer programming formulations

Next, we introduce two classical mathematical formulations for the problem.

Precedence formulation Let x_{ij} be a binary variable equal to 1 if and only if job i is scheduled prior to job j. We obtain the following formulation:

$$\underset{x\in\{0,1\}^{n^2}}{\text{minimize}} \quad \left(\max_{p\in U}\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}} p_i w_j x_{ij}\right) \tag{6}$$

$$\text{s.t.} \quad x_{ij} + x_{ji} = 1, \quad i,j\in\mathcal{J},\ i\neq j \tag{7}$$

$$x_{ij} + x_{jk} \leq x_{ik} + 1, \quad i,j,k\in\mathcal{J} \tag{8}$$

$$x_{ii} = 1, \quad i\in\mathcal{J}. \tag{9}$$

Using classical techniques, the objective function (6) can be rewritten as:

U^Γ**:** The objective function (6) is replaced by $\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\overline{p}_i w_j x_{ij} + \Gamma z_0 + \sum_{i\in\mathcal{J}} z_i$, where z_0 and z_i are $n+1$ additional non-negative variables that satisfy the additional constraints $z_0 + z_i \geq \hat{p}_i \sum_{j\in\mathcal{J}} w_j x_{ij}$ for each $i\in\mathcal{J}$, leading to a mixed-integer linear programming reformulation.

U^Ω**:** The objective function (6) is replaced by $\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\overline{p}_i w_j x_{ij} + \left\|\tilde{x}^T\Sigma\tilde{x}\right\|_2$ where $\tilde{x}_i = \sum_{j\in\mathcal{J}}\hat{p}_i w_j x_{ij}$, leading to a mixed-integer second order cone programming reformulation.

Whenever $\Gamma = 1$, one readily verifies that there exists an optimal solution with $z_j = 0$ for each $j\in\mathcal{J}$. We obtain the objective function $\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\overline{p}_i w_j x_{ij} + z_0$ combined with the additional constraints $z_0 \geq \hat{p}_i\sum_{j\in\mathcal{J}} w_j x_{ij}$ for each $i\in\mathcal{J}$.

Assignment formulation Let y_{ij} be a binary variable equal to 1 if and only if job i is scheduled in position j. We obtain the following formulation:

$$\underset{y\in\{0,1\}^{n^2}}{\text{minimize}} \quad \left(\max_{p\in U}\sum_{i\in\mathcal{J}}\sum_{j\in\mathcal{J}}\sum_{k\in\mathcal{J}}\sum_{l=1}^{j} p_i w_k y_{ij} y_{kl}\right) \tag{10}$$

$$\text{s.t.} \quad \sum_{i\in\mathcal{J}} y_{ij} = 1, \quad j\in\mathcal{J}$$

$$\sum_{j\in\mathcal{J}} y_{ij} = 1, \quad i\in\mathcal{J}$$

which is a special case of the quadratic assignment problem. We linearize the products between variables y_{ij} and y_{kl} by introducing new real variables z_{ijkl} satisfying $z_{ijkl} \leq y_{ij}$, $z_{ijkl} \leq y_{kl}$, and $z_{ijkl} \geq y_{ij} + y_{kl} - 1$. Then, the two classical reformulations described for the precedence formulation can be used to reformulate the linearized objective function (10).

4 Numerical experiments

We compare the different formulations presented in the previous section for the two uncertainty sets. We solve the formulation using the aforementioned reformulation or using branch-and-cut algorithms, which is further detailed below.

Instances and implementation details The instances were randomly generated as follows. For each $j \in \mathcal{J}$, the processing times $\overline{p}_j$,$\hat{p}_j$, and weight w_j have been uniformly generated in the intervals $[1, 2n], [1, n]$ and $[1, n]$, respectively. The algorithms are coded in Julia language, using the package Jump [8], and solved by Gurobi 6.5. The experiments are carried out on a computer equipped with a CPU at 2.67 GHz and 125 GB of memory. A time limit of 1800 seconds was imposed for each instance and all solution times are reported in seconds.

Value-at-risk The values of Γ and Ω have been computed using the probabilistic bounds provided in [5] and [2], respectively. Specifically, these values ensure that the optimal solution of (2) provides a conservative approximation to the value-at-risk optimization problem $\min_{\sigma \in P(n)} \text{VaR}_\epsilon \left[\max_{p \in U} \sum_{j \in \mathcal{J}} w_j C_j(\sigma, p) \right]$, whenever p is any random vector and

$$\text{VaR}_\epsilon \left[\max_{p \in U} \sum_{j \in \mathcal{J}} w_j C_j(\sigma, p) \right] = \inf \left\{ t : P\big(\max_{p \in U} \sum_{j \in \mathcal{J}} w_j C_j(\sigma, p) \geq t \big) \leq \epsilon \right\},$$

see also [10,11] for more details on the relation between U^Γ and probabilistic constraints [4]. Given a probability $\epsilon \in]0, 1[$, we obtain (see [2]) that $\Omega(\epsilon) = \sqrt{-2 \ln \epsilon}$ while the formula is more complex for Γ, for which we refer to [5].

[4] Notice that the less conservative model proposed in [10,11] cannot be used here to reduce the conservatism of U^Γ because all coefficients of p are non-zero in the objective function.

Branch-and-cut algorithm In addition to the reformulations described in Section 3, we solve the precedence formulation via branch-and-cut algorithms in the line of [3]. Specifically, the algorithm considers a restricted master problem formed by constraints (7)–(9), with the additional constraint $\gamma \geq \sum_{i \in \mathcal{J}} \sum_{j \in \mathcal{J}} p_i w_j x_{ij}$, and minimizes variable γ. Then, at each integer node, we solve the separation problem

$$z = \max_{p \in U} \left(\sum_{i \in \mathcal{J}} \sum_{j \in \mathcal{J}} p_i w_j x^*_{ij} \right), \tag{11}$$

where x^* and γ^* denote the current values of x and γ, respectively. If $z > \gamma^*$, we add the cutting plane $\gamma \geq \sum_{i \in \mathcal{J}} \sum_{j \in \mathcal{J}} p^*_i w_j x_{ij}$ where $p^* \in U$ is an optimal solution of (11). Notice that for U^Γ and U^Ω, the separation problem amounts to sort a vector and to compute an $L2$-norm, respectively (see [3] for details).

Results Table 1 compares the precedence formulation with the assignment formulation using dualized reformulations. The table reports the times spent solving the linear programming relaxation as well as the full problems, and the root gaps. The results shows that the assignment formulation is weaker and takes a longer time to be solved than the precedence formulation. Table 2 compares the classical dualization with the specialized reformulation proposed above when $\Gamma = 1$. The table seems to indicate that both formulations behave similarly with Gurobi. Table 3 compares dualizations with branch-and-cut algorithms for the two uncertainty sets. It appears that the dualization is faster for U^Γ while the branch-and-cut algorithm outperforms the dualization for U^Ω. Table 4 finally compares the solution costs of the two models for the same levels of probability. The results show that the approximation of the value-at-risk provided by U^Ω is cheaper than the one provided by U^Γ.

Acknowledgements

We would like to thank Janis Kurtz for providing us with reference [7].

References

[1] H. Aissi, C. Bazgan, and D. Vanderpooten. Min-max and min-max regret versions of combinatorial optimization problems: A survey. *European Journal of Operational Research*, 197(2):427–438, 2009.

n	Γ	Precedence formulation			Assignment formulation		
		time LP	time IP	Gap	time LP	time IP	Gap
5	4.12	0.005	0.002	0	2.418	0.0056	100
10	5.226	0.014	0.006	0.294	292.193	0.171	100
15	6.11	0.137	0.0219	0.142	1800	1.144	100
20	6.854	0.670	0.075	0.344	1800	4.437	100

Table 1
Comparison of the precedence formulation and the assignment formulation for U^{Γ}

n	Γ	Classical dualization	Specialized reformulation
50	1	10.8840	5.5348
60	1	17.2279	24.9224
70	1	18.3319	16.0779
80	1	14.5652	15.4387
90	1	26.8292	32.3861
100	1	32.8103	26.9855
110	1	75.4261	89.5468
120	1	86.2200	80.9256
130	1	100.4396	100.9902
140	1	108.2194	135.1086
150	1	145.8819	157.2395
200	1	427.3675	489.7448

Table 2
Comparisons of the solution times of the two formulations under the uncertainty set U^{Γ} with $\Gamma = 1$

[2] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Math. Program.*, 88(3):411–424, 2000.

[3] D. Bertsimas, I. Dunning, and M. Lubin. Reformulation versus cutting-planes for robust optimization. *Computational Management Science*, 13(2):195–217, 2016.

[4] D. Bertsimas and M. Sim. Robust discrete optimization and network flows. *Math. Program.*, 98(1-3):49–71, 2003.

[5] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[6] M. Bougeret, A. Pessoa, and M. Poss. Robust scheduling with budgeted uncertainty. Working paper, 2016.

n	ϵ	U^{Γ}		U^{Ω}	
		Dualization	B&C	Dualization	B&C
50	0.01	5.3192	17.7836	1756.1289	14.4455
	0.05	2.6841	10.9062	1600.961	16.725
	0.1	2.8941	12.0554	1333.4534	5.0623
60	0.01	44.5364	81.8779	1800	24.6955
	0.05	31.7562	73.0690	1800	17.7427
	0.1	15.0364	26.9620	1800	16.5177
70	0.01	36.6147	91.0086	1800	48.3793
	0.05	13.1456	42.1255	1800	30.1910
	0.1	3.8862	24.3906	1800	23.7368
80	0.01	47.1291	130.2417	1800	81.7222
	0.05	36.2294	130.1191	1800	52.5677
	0.1	8.8290	21.0278	1800	44.6953
90	0.01	27.2781	181.6671	1800	127.6601
	0.05	27.7035	181.4539	1800	93.1146
	0.1	27.2781	170.6671	1800	73.1298
100	0.01	55.3856	292.3421	1800	232.755
	0.05	21.8976	89.7856	1800	99.1304
	0.1	19.6913	83.9951	1800	82.1937
110	0.01	33.1948	253.8273	1800	107.6149
	0.05	30.1278	186.2178	1800	117.9020
	0.1	26.5111	101.226	1800	147.3104
120	0.01	46.9315	247.8843	1800	226.3862
	0.05	43.7567	199.9976	1800	115.1185
	0.1	44.61242	206.2942	1800	193.9772
130	0.01	81.1503	331.1503	1800	254.1678
	0.05	59.3245	299.3246	1800	483.1201
	0.1	62.1288	202.1288	1800	220.4472
140	0.01	110.1452	523.3927	1800	523.0023
	0.05	103.6665	403.6665	1800	549.0178
	0.1	95.1212	431.2345	1800	289.8857
150	0.01	132.7827	732.617	1800	429.1916
	0.05	112.4356	639.4243	1800	438.6716
	0.1	112.7711	572.771	1800	737.3711
200	0.01	880.156	1800	1800	1219.8972
	0.05	511.3178	1800	1800	1395.2189
	0.1	580.443	1800	1800	1618.8333

Table 3
Solution times for uncertainty sets U^{Γ} and U^{Ω}

n	ϵ	U^Γ	U^Ω
50	0.01	1.236124004e6	1.0973274217e6
	0.05	1.3787137e6	1.2001925699e6
	0.1	1.2636933399e6	1.1033971647e6
100	0.01	1.643406456e7	1.47192456036e7
	0.05	2.010210435e7	1.8023336381e7
	0.1	1.898526685e7	1.70664998303e7
150	0.01	8.0721378208e7	7.18727789599e7
	0.05	8.441931962e7	7.59088424515e7
	0.1	7.31019883599e7	6.54991746364e7
200	0.01	2.8539763498e8	2.4891821698121e8
	0.05	2.31240831376e8	2.077655828647e8
	0.1	2.5827205682e8	2.372011370971e8

Table 4
Solution costs for models U^Γ and U^Ω

[7] C. Buchheim. Robuste optimierung, 2014. http://www.mathematik.tu-dortmund.de/lsv/teaching/robopt/Skript.pdf.

[8] I. Dunning, J. Huchette, and M. Lubin. Jump: A modeling language for mathematical optimization. *CoRR*, abs/1508.01982, 2015.

[9] R. L. Graham, E. L. Lawler, J. K. Lenstra, and A. R. Kan. Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of discrete mathematics*, 5:287–326, 1979.

[10] M. Poss. Robust combinatorial optimization with variable budgeted uncertainty. *4OR*, 11(1):75–92, 2013.

[11] M. Poss. Robust combinatorial optimization with variable cost uncertainty. *European Journal of Operational Research*, 237(3):836–845, 2014.

[12] W. E. Smith. Various optimizers for single-stage production. *Naval Research Logistics Quarterly*, 3(1-2):59–66, 1956.

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

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