Computers & Industrial Engineering 185 (2023) 109610

Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Robust job-sequencing with an uncertain flexible maintenance activity

Paolo Detti [a], Gaia Nicosia [b,*], Andrea Pacifici [c]

[a] Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche, Università di Siena, Via Roma 56, 53100, Italy
[b] Dipartimento di Ingegneria Civile, Informatica e delle Tecnologie Aeronautiche, Università degli Studi "Roma Tre", via della Vasca Navale 79, 00146 Rome, Italy
[c] Dipartimento di Ingegneria Civile e Ingegneria Informatica, Università degli Studi di Roma "Tor Vergata", Via del Politecnico 1, 00133 Rome, Italy

ARTICLE INFO

Keywords:
Single-machine scheduling
Flexible maintenance
Robust optimization
Computational complexity

ABSTRACT

In this study, the problem of scheduling a set of jobs and one uncertain maintenance activity on a single machine, with the objective of minimizing the makespan is addressed. The maintenance activity has a given duration and must be executed within a given time window. Furthermore, duration and time window of the maintenance are uncertain, and can take different values which can be described by different scenarios. The problem is to determine a job sequence which performs well, in terms of makespan, independently on the possible variation of the data concerning the maintenance. A robust scheduling approach is used for the problem, in which four different measures of robustness are considered, namely, maximum absolute regret, maximum relative regret, worst-case scenario, and ordered weighted averaging. Complexity and approximation results are presented. In particular, we show that, for all the four robustness criteria, the problem is strongly NP-hard. A number of special cases are explored, and an exact pseudopolynomial algorithm based on dynamic programming is devised when the number of scenarios is fixed. Two Mixed Integer Programming (MIP) models are also presented for the general problem. Several computational experiments have been conducted to evaluate the efficiency and effectiveness of the MIP models and of the dynamic programming approach.

1. Introduction

In this paper, the problem of scheduling a set of jobs and an uncertain maintenance activity on a single machine with the objective of minimizing the makespan is addressed. The maintenance activity is *flexible*, that is, it must be executed within a given time window. Furthermore, while the processing times of the jobs are deterministic, the maintenance duration and the time window are uncertain, and can take different values which can be described by various scenarios. The problem consists of determining a job sequence which performs "well", in terms of makespan, independently on the possible variation of the data concerning the maintenance.

These types of problems arise in manufacturing plants when a workstation has to process jobs in a given sequence and, concurrently, has to perform a precautionary maintenance task in order to prevent major failures. Maintenance activities are usually carried out by an external party which gives rough estimates of the time required for this task and the time-interval in which it would be executed. In these conditions, duration and starting time of the maintenance are not exactly known *a-priori*, while the job sequence must be decided before the information about the maintenance is disclosed. Hence, the actual schedule will depend on the starting time and length of the maintenance activity, and thus it can be established only once the data become apparent. In such problems, the aim is to find a job sequence that is *robust* against the uncertainties concerning the maintenance task. In general, robust optimization deals with problems in which, due to uncertain input data, a set of scenarios with possibly different optimal solutions exist. Most of the approaches in the literature consist of trying to find a solution satisfying given "robustness criteria", which may be associated to an average or a worst-case solution performance over all scenarios. In this context, the addressed problem can be defined as a robust flexible maintenance single machine scheduling problem with the objective of minimizing the makespan. In this work, different variants of the problems are studied and different robustness criteria are analyzed.

A similar scheduling problem in which uncertainty only affects maintenance duration has been addressed in Detti, Nicosia, Pacifici, and de Lara (2019). In the current work, a more general setting is considered: The maintenance time window in addition to its duration are regarded as varying parameters in the different scenarios. The main contributions of this work are: (*i*) New complexity and approximation results are provided for the general problem under four different robustness criteria, namely, absolute and relative regret, worst case scenario, and ordered weighted averaging. In particular, different from strictly related problems previously addressed in the literature, we are able to prove strong NP-hardness results for all the considered robustness

* Corresponding author.
E-mail addresses: paolo.detti@unisi.it (P. Detti), gaia.nicosia@uniroma3.it (G. Nicosia), andrea.pacifici@uniroma2.it (A. Pacifici).

https://doi.org/10.1016/j.cie.2023.109610
Received 1 May 2023; Received in revised form 9 August 2023; Accepted 11 September 2023
Available online 19 September 2023

criteria. Approximation results are also presented. (*ii*) Different variants and special cases of the problem in which the uncertainty regards one or both the extremes of the time window and/or the duration of the maintenance activity are addressed and analyzed. (*iii*) An exact pseudopolynomial dynamic programming algorithm for the case in which the uncertainty can be described by a fixed number of scenarios is devised, and two Mixed Integer Linear Programming (MIP) models for the general case are proposed. (*iv*) A computational study on randomly generated instances is presented, assessing the performance of the above solution approaches.

The paper is organized as follows. In Section 2, a review of the literature is presented. Some preliminary concepts, definitions, a rigorous statement of the problem together with a summary of the main theoretical contributions are given in Section 3. Section 4 is devoted to present the theoretical results on computational complexity and approximation for all the considered robustness criteria, and to describe an exact dynamic programming algorithm. In Section 5, some special cases, which can be efficiently solved or that can be reduced to problems already addressed in the literature, are investigated. In Section 6 two MIP formulations are presented. The results of the computational experiments on the MIP models, together with the dynamic programming algorithm, are presented in Section 7. Finally, conclusions follow.

2. Literature review

In the literature, scheduling problems with maintenance activities or unavailable periods have been addressed by several authors for the deterministic case, i.e., in which the duration of the maintenance tasks or unavailability periods are known. In Lee (1996) Lee considers several deterministic scheduling problems with unavailability constraints both in the resumable and in the nonresumable cases. (In the resumable case it is possible to interrupt the processing of one job and resume its execution after the completion of the maintenance activity. This is not allowed in the nonresumable case.) In the resumable case many single machine scheduling problems can be solved in polynomial time, while the corresponding nonresumable versions of the same problems often become NP-hard (Lee, 1996; Yang, Maa, Xu, & Yang, 2011). For a survey of scheduling problems with availability constraints see Ma, Chu, and Zuo (2010).

For the makespan objective function, the nonresumable deterministic non-flexible version (in which the maintenance must start at a fixed given time) is proven to be binary NP-hard in Lee (1996). Furthermore, Lee (1996) shows that the Longest Processing Time (LPT) rule has a tight worst-case ratio of 4/3, and He, Ji, and Cheng (2005) present a Fully Polynomial Time Approximation Scheme. Kacem and Kellerer (2016) prove that a simple algorithm that schedules first the longest job has a worst-case ratio of 3/2.

Yang, Hung, Hsu, and Chern (2002) address the problem of scheduling jobs on a single machine with a flexible maintenance activity in the nonresumable case, with the objective of minimizing the makespan. They show that the problem is NP-hard and provide a heuristic algorithm with complexity $O(n \log n)$. The variant of the problem, in which maintenance must be periodically performed is addressed in Chen (2008), Ji, Yong, and Cheng (2007), Lee (1996) and Xu, Yin, and Li (2009). Lee (1996) proves that the problem is strongly NP-hard even in the non-flexible case, and Ji et al. (2007) show that LPT has a tight worst-case ratio of 2. Chen (2008) proposes two mixed integer linear programming models and a heuristic algorithm. Xu et al. (2009) show that the heuristic proposed in Chen (2008) is 2-approximate and that the bound is tight. In Luo, Cheng, and Ji (2015), the authors consider the single machine problem of scheduling jobs and a variable maintenance, which has to start before a given deadline and whose duration is increasing with its starting time. They provide polynomial solution algorithms for a few classical objective functions. A similar problem with different objective is addressed in Ying, Lu, and Chen (2016).

In the literature, robust optimization problems have been addressed in several fields by many authors, in order to take care of incomplete or unreliable data. Mulvey, Vanderbei, and Zenios (1995) present a general framework for addressing conflicting objectives and model robustness, while Kouvelis and Yu (1997) propose different robustness criteria. Robust scheduling problems have been addressed by different works in the literature, too. Daniels and Kouvelis (1995) consider single machine scheduling problems with the objective of minimizing total completion time, in which processing times may vary in a given interval. They introduce different robustness criteria, measuring worst-case absolute or relative deviation from the optimum over all scenarios (i.e., maximum absolute or relative regrets), and establish several properties of robust schedules. Lebedev and Averbakh (2006) show that the problem of finding a schedule minimizing the maximum regret on a single machine, with processing times varying in given intervals, is NP-hard. For the same problem, Kasperski and Zielinski (2014) propose an approximation algorithm, Pereira (2016) presents an exact algorithm for the case with job weights and total weighted completion time objective, and Wang, Cui, Chu, Yu, and Gupta (2020) come up with an approximation algorithm and exact-solution methods based on MIP formulations when the objective is total tardiness.

Robustness concepts have been also adopted in scheduling problems with maintenance activities. For instance in Costa Souza, Ghasemi, Saif, and Gharaei (2022), the authors study how machine unavailability, due to a preventive maintenance with stochastic duration, affects the performance of a job shop. The weighted sum of the expected values of the makespan is used as a robustness criterion. In Golpîra and Tirkolaee (2019), Golpîra and Tirkolaee present a bi-objective model which incorporates the problem of scheduling a maintenance activity with uncertain duration into a robust optimization framework. Different solution techniques aiming at guaranteeing the stability of the output schedule are proposed and evaluated. A recent survey by Shabtay and Gilenson (2023) provides an interesting framework for scheduling problems in which the multi-scenario approach is adopted to cope with uncertain parameters or data.

Finally, in the above mentioned paper (Detti et al., 2019), the authors investigate robust scheduling problems on a single machine in presence of a flexible maintenance activity with uncertain duration: Four robustness criteria are analyzed when the objective is the minimization makespan or total completion time.

3. Introductory concepts, definitions and notation

In the scheduling problem addressed in this paper, we are given a set of n jobs, $J = \{1, 2, \dots, n\}$, and a maintenance activity M, that must be processed on a single machine. The jobs must be processed without interruption, and the machine cannot process any of the jobs during the execution of the maintenance activity. The maintenance M has a duration P and must be performed within a time window $[r, d]$, and the jobs have processing times p_j, $j = 1, \dots, n$. While the jobs processing times p_j are deterministic and known, the maintenance duration P and the interval $[r, d]$ are uncertain quantities, which take their values in a finite and discrete set $\mathcal{S} = \{s_1, s_2, \dots, s_k\}$ of *scenarios*. For each scenario $s \in \mathcal{S}$, we denote by $r(s)$ and $d(s)$ the realizations, i.e., the values, taken by the extremes of the maintenance time window in s, and by $P(s)$ the realization of the maintenance duration. In the remainder of the paper, it is always assumed that $P(s) \le d(s) - r(s)$ for all $s \in \mathcal{S}$.

In what follows, without loss of generality, let us assume that the scenarios are ordered according to non-decreasing values of the latest starting time of the maintenance M. Hence,

$$d(s_1) - P(s_1) \le d(s_1) - P(s_1) \le \cdots \le d(s_k) - P(s_k). \tag{1}$$

The problem consists in determining a job sequence π of the n jobs which performs well, in terms of makespan (i.e., the completion time of the last job in the sequence), independently on the possible variation of the data concerning the maintenance. Namely, we want to find a

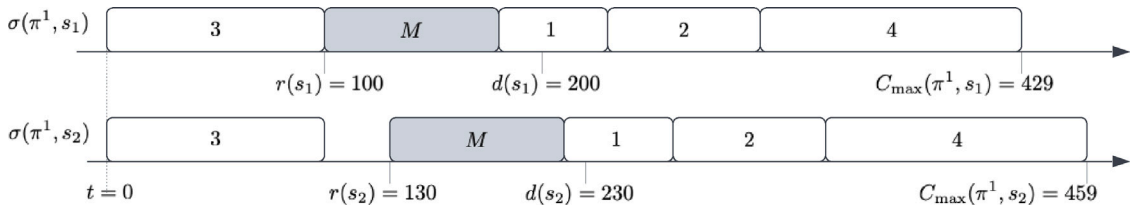

Fig. 1. The realization schedules $\sigma(\pi^1, s_1)$ and $\sigma(\pi^1, s_2)$ of the sequence $\pi^1 = \langle 3, 1, 2, 4\rangle$ under the two scenarios of Example 1.

sequence that is robust against the uncertainties of M described by the scenarios' set S. To this aim, different robustness criteria, which are rigorously defined below, are studied.

Given a job-sequence $\pi = \langle \pi_1, \pi_2, \dots, \dots \pi_n\rangle$ and a scenario $s \in S$ (defining the duration $P(s)$ and the interval $[r(s), d(s)]$ of the maintenance), the makespan of the solution is evaluated in the so-called *realization schedule* $\sigma(\pi, s)$. Here, jobs are processed in the order specified by π and M is executed at the latest available time inside the allotted time window so that unnecessary idle times are avoided (Detti et al., 2019). More specifically, in a realization schedule $\sigma(\pi, s)$, the *crossover* job $\bar{h}$ is the first job in the sequence π that could not be completed before the maintenance period. Since jobs cannot be interrupted,[1] in $\sigma(\pi, s)$, the crossover job is always scheduled after the completion of M and the maintenance activity is scheduled at the earliest possible time between the $(\bar{h}-1)$-th and the $\bar{h}$-th jobs. The resulting sequence is then $\langle \pi_1, \dots \pi_{\bar{h}-1}, M, \pi_{\bar{h}}, \dots \pi_n\rangle$.

For the jobs scheduled before the crossover job, the completion time of the hth job in $\sigma(\pi, s)$ is $C_{\pi_h}(\sigma(\pi, s)) = \sum_{\ell=1}^{h} p_{\pi_\ell}$, for $h = 1, \dots, \bar{h}-1$. For the other jobs, the completion time is the sum of the processing times of preceding jobs, plus the maintenance duration, plus possible unavoidable idle time. Then, for $h = \bar{h}, \dots, n$, $C_{\pi_h}(\sigma(\pi, s)) = \sum_{\ell=1}^{h} p_{\pi_\ell} + P(s) + \max\left\{0, r(s) - \sum_{i=1}^{\bar{h}-1} p_{\pi_i}\right\}$. If, in the latter term, $r(s) - \sum_{i=1}^{\bar{h}-1} p_{\pi_i}$ is positive we have an idle time before the earliest possible start $r(s)$ of the maintenance activity. In the remainder of this paper, the makespan of the realization schedule of a sequence π in scenario $s \in S$ is referred to as $C_{\max}(\pi, s) = \max_{j\in J}\{C_j(\sigma(\pi, s))\}$.

Example 1. Let us consider a problem instance with a set of four jobs $\{1, 2, 3, 4\}$, with processing times $p_1 = 51$, $p_2 = 73$, $p_3 = 100$, and $p_4 = 125$, and two scenarios s_1 and s_2, with $r(s_1) = 100$, $d(s_1) = 200$, $r(s_2) = 130$, $d(s_2) = 230$, $P(s_1) = P(s_2) = 80$. Let us consider the job sequence $\pi^1 = \langle 3, 1, 2, 4\rangle$. In Fig. 1, the two realization schedules $\sigma(\pi^1, s_1)$ and $\sigma(\pi^1, s_2)$ in the two scenarios are reported. Note that, job 1 is the critical job in both scenarios s_1 and s_2. Also observe that, while π^1 is an optimal solution sequence in scenario s_1 (producing no idle time), it is not in s_2, in which the optimal solution is $\pi^2 = \langle 4, 1, 2, 3\rangle$.

Note that, despite the ordering defined by Eq. (1), in general it is not possible to establish domination criteria among the values of the job completion times in different scenarios $s \in S$: For instance, sequence π^1 of Example 1 obtains a worse makespan in the "ampler" scenario s_2 (i.e., such that $d(s_1) - P(s_1) \le d(s_2) - P(s_2)$).

Robustness criteria. As already stated, for a certain scenario s, the performance of a solution π is measured by the makespan of the realization schedule, i.e., the completion time of the last job in the schedule. To assess whether a schedule is satisfactory in all the scenarios, four robustness criteria have been considered (to be minimized). Namely, min–max (MM), maximum absolute regret (ABS) and relative regret (REL), and ordered weighted averaging (OWA), that are defined in the following. Given a job sequence π, let $C_{\max}(\pi, s)$ be the makespan of the realization schedule $\sigma(\pi, s)$, and let $C^*_{\max}(s)$ be the minimum makespan value for scenario $s \in S$. Then:

$$MM(\pi) = \max_{s\in S} C_{\max}(\pi, s) \qquad (min - max); \tag{2}$$

$$ABS(\pi) = \max_{s\in S}\{C_{\max}(\pi, s) - C^*_{\max}(s)\} \qquad (maximum\ absolute\ regret); \tag{3}$$

$$REL(\pi) = \max_{s\in S} \frac{C_{\max}(\pi, s)}{C^*_{\max}(s)} \qquad (maximum\ relative\ regret); \tag{4}$$

$$OWA(\pi) = \sum_{i=1}^{k} \beta_i C_{\max}(\pi, s_{[i]}) \qquad (ordered\ weighted\ averaging); \tag{5}$$

where, in (5), $s_{[i]} \in S$ is the scenario producing the ith largest value of the makespan $C_{\max}$, i.e., $C_{\max}(\pi, s_{[i]}) \ge C_{\max}(\pi, s_{[i+1]})$, $i = 1, \dots, k-1$, and β_i is a given weight assigned to scenario $s_{[i]}$.

The min–max and the maximum absolute and relative regret criteria are widely used criteria in robust optimization (see Aissi, Bazgan, & Vanderpooten, 2009) and have been also applied in the scheduling literature, see e.g., Daniels and Kouvelis (1995). In the MM criterion, one has to find a sequence whose maximum makespan among all scenarios is minimum. ABS and REL provide information on how a given realization schedule is far from the optima of all the scenarios. OWA has been introduced in Yager (1988) and it is a generalization of the min–max criterion: In fact, OWA with $\beta_1 = 1$ and $\beta_i = 0$ for $i = 2, 3, \dots, k$ reduces to MM.

Given a job sequence, i.e., a solution π, and a certain criterion $c \in \{ABS, REL, MM, OWA\}$ (defined in Eqs. (3)–(5)), we indicate by $c(\pi)$ the value of the robustness criterion c associated to solution π. As a consequence, we may rigorously define the addressed robust scheduling problem as follows.

> Robust single-machine Scheduling with Maintenance activity Problem ($RSMP(c)$): *Given* a set J of n jobs with deterministic processing times p_j, $j = 1, \dots, n$, and a set S of discrete scenarios, corresponding to $|S|$ possible realizations of the triple (r, d, P); *find* a sequence π of the jobs *such that* $c(\pi)$ is minimized.

An optimal solution of the above problem $RSMP(c)$ is called a *robust solution*.

Example 2. Let us consider the instance of Example 1. As we already observed, the sequence minimizing the makespan for scenario s_1 is $\pi^1 = \langle 3, 1, 2, 4\rangle$, providing $C^*_{\max}(s_1) = C_{\max}(\pi^1, s_1) = 429$ (with no idle time, see Fig. 1), while the optimal solution for scenario s_2 is $\pi^2 = \langle 4, 1, 2, 3\rangle$, for which $C^*_{\max}(s_2) = C_{\max}(\pi^2, s_2) = 434$. Let us also consider a third solution sequence $\pi^3 = \langle 2, 1, 3, 4\rangle$.

In this case, a robust solution under the min–max criterion MM is not optimal in any of the two scenarios. In fact, we have:

$$C_{\max}(\pi^1, s_1) = 429,\ C_{\max}(\pi^1, s_2) = 459,\ MM(\pi^1) = 459,$$

$$C_{\max}(\pi^2, s_1) = 529,\ C_{\max}(\pi^2, s_2) = 434,\ MM(\pi^2) = 529,$$

[1] This situation is referred to in the literature as "non-resumable" case.

Table 1
Summary of used notation.

n	Number of jobs
p_j	Processing time of job $j = 1, 2, \dots, n$
k	Number of scenarios
S	Set of k scenarios
M	Maintenance activity
$P(s)$	Duration of M in scenario s
$r(s)$	Release date of M in scenario s
$d(s)$	Due date of M in scenario s
$\Delta(s)$	Window slack $= d(s) - P(s) - r(s)$
π	Job sequence
$\sigma(\pi, s)$	Realization schedule of π in s
$C_{\max}(\pi, s)$	Makespan of schedule $\sigma(\pi, s)$
$C^*_{\max}(s)$	Opt. makespan in scenario s
c	Generic robustness criterion
$c = MM$	min–max
$c = ABS$	Maximum absolute regret
$c = REL$	Maximum relative regret
$c = OWA$	Ordered weighted averaging
$c(\pi)$	Criterion objective value of π
$RSMP(c)$	Addressed robust problem

$C_{\max}(\pi^3, s_1) = 456,\ C_{\max}(\pi^3, s_2) = 435,\ MM(\pi^3) = 456.$

Considering that the only subsets of jobs which may be accommodated before M are $\{1\}$, $\{2\}$, $\{3\}$, $\{4\}$, and $\{1, 2\}$, the only sensible sequence beside π^1, π^2, and π^3 is $\pi^0 = \langle 1, 2, 3, 4\rangle$. But $MM(\pi^0) = 478$ and hence π_3 corresponds to a robust solution according the MM criterion.

Regarding the ABS criterion, the robust solution is the optimal solution for scenario s_1, $\pi^1 = \langle 3, 1, 2, 4\rangle$. Indeed, for the three sequences we have:

$$\begin{aligned}
ABS(\pi^1) &= \max\{C_{\max}(\pi^1, s_1) - C^*_{\max}(s_1); C_{\max}(\pi^1, s_2) - C^*_{\max}(s_2)\} = \\
&= \max\{429 - 429; 459 - 434\} = 25 \\
ABS(\pi^2) &= \max\{C_{\max}(\pi^2, s_1) - C^*_{\max}(s_1); C_{\max}(\pi^2, s_2) - C^*_{\max}(s_2)\} = \\
&= \max\{529 - 429; 434 - 434\} = 100 \\
ABS(\pi^3) &= \max\{C_{\max}(\pi^3, s_1) - C^*_{\max}(s_1); C_{\max}(\pi^3, s_2) - C^*_{\max}(s_2)\} = \\
&= \max\{456 - 429; 435 - 434\} = 27.
\end{aligned}$$

Analogously, for the REL criterion we have:

$$\begin{aligned}
REL(\pi^1) &= \max\{C_{\max}(\pi^1, s_1)/C^*_{\max}(s_1); C_{\max}(\pi^1, s_2)/C^*_{\max}(s_2)\} = \\
&= \max\{429/429; 459/434\} = 1.057 \\
REL(\pi^2) &= \max\{C_{\max}(\pi^2, s_1)/C^*_{\max}(s_1); C_{\max}(\pi^2, s_2)/C^*_{\max}(s_2)\} = \\
&= \max\{529/429; 434/434\} = 1.233 \\
REL(\pi^3) &= \max\{C_{\max}(\pi^3, s_1)/C^*_{\max}(s_1); C_{\max}(\pi^3, s_2)/C^*_{\max}(s_2)\} = \\
&= \max\{456/429; 435/434\} = 1.063.
\end{aligned}$$

Hence, in this case π^1 is the robust solution.

Finally, the solution for OWA criterion depends on the weights β_i. As an example, it can be easily verified that, when $\beta_1 = 1$ and $\beta_2 = 10$, we have

$$OWA(\pi^1) = 4749,\ OWA(\pi^2) = 4869,\ OWA(\pi^3) = 4806,$$

implying that π^1 is the robust solution. While, if $\beta_1 = 10$ and $\beta_2 = 1$, we have

$$OWA(\pi^1) = 5019,\ OWA(\pi^2) = 5724,\ OWA(\pi^3) = 4995,$$

and, hence, π^3 is the robust solution for OWA criterion.

For the sake of clarity, in Table 1 some of the notation introduced above is summarized.

In this work, for each of the four robustness measures (2)–(5), we derive several complexity and approximation results. Table 2 reports a summary of the main theoretical contributions presented in the paper.

4. Complexity and approximation

The section is organized as follows. First (Section 4.1) we show how to compute in pseudopolynomial time the optimal solutions of $RSMP(c)$ in each scenario of S, while Section 4.2 is devoted to prove the strong NP-hardness of $RSMP(c)$. On the other hand, in Section 4.3, we show that $RSMP(c)$ can be solved in pseudopolynomial time by dynamic programming when the number of scenarios $|S|$ is fixed. In the last part of this section we briefly discuss about a property called *scenario optimality* (Detti et al., 2019) and present some approximation results. More precisely, we extend a result presented in Detti et al. (2019) to $RSMP(c)$ (Lemma 5), implying that the LPT rule is a $\frac{4}{3}$-approximation algorithm for $RSMP(c)$ when $c \in \{REL, MM, OWA\}$ (Theorem 6).

4.1. Computation of the optimal solutions of all scenarios

Hereafter, we show that the deterministic version of $RSMP(c)$, i.e., with a single scenario, is equivalent to the Subset Sum Problem (SSP) which can be stated as follows: Given a set of non-negative integers $E = \{a_1, a_2, \dots, a_n\}$ and an integer bound B, find a subset $E^* \subseteq E$ whose sum is maximized and does not exceed B.

This equivalence implies that the deterministic version of $RSMP(c)$ is NP-hard in the ordinary sense and can be solved in pseudopolynomial time. Furthermore, we prove that a single run of the standard dynamic programming solution algorithm for SSP can be used to compute the optimal solution values $C^*_{\max}(s)$ of all scenarios $s \in S$, for the general (i.e., multi-scenario) $RSMP(c)$.

Let us consider a deterministic (single-scenario) instance of $RSMP(c)$, and let P and $[r, d]$ be the (deterministic) duration and the time window of the maintenance, respectively. As already observed in Detti et al. (2019), in this case, any solution minimizing the idle time is optimal. Then, a deterministic instance of $RSMP(c)$ can be solved as a SSP by setting $a_j = p_j$ for $j = 1, \dots, n$ and $B = d - P$. If the optimal solution value z_B of this SSP instance has value greater or equal than r, then there is a subset E^* of jobs with total processing time z_B such that $r \le z_B \le d - P$. The job sequence in which the jobs in E^* are scheduled first produces a realization schedules with no idle time. Otherwise, if $z_B < r$, then the optimal realization schedule has an idle time equal to $r - z_B$ and $C^*_{\max} = \sum_j p_j + P + r - z_B$. On the other hand, it is easy to see that any SSP instance can be solved by considering a single-scenario instance of $RSMP(c)$ with n jobs in which $p_j = a_j$ for $j = 1, \dots, n$, $r = B$ and $P = d - r$. The optimal solution of this $RSMP(c)$ instance minimizes the idle time before the maintenance starting at B, providing an optimal solution for the SSP instance.

In our robust optimization problem $RSMP(c)$, with $c \in \{ABS, REL\}$ we are interested in finding the optima $C^*_{\max}(s)$ for all scenarios $s \in S$. Such values can be found by applying a standard dynamic programming algorithm for SSP (Kellerer, Pferschy, & Pisinger, 2003) in which B is set equal to $\max_s\{d(s) - P(s)\} = d(s_k) - P(s_k)$. Since the dynamic programming algorithm finds the optimal solutions for all integers $B' \le B$, it also gives the optimal solution values when $B' = d(s) - P(s)$ for all $s \in S$, corresponding to the optima of all scenarios in $RSMP(c)$. Hence, the optima $C^*_{\max}(s)$ of all scenarios in $RSMP(c)$ can be computed in pseudopolynomial time $O(Bn + |S|)$.

4.2. Complexity results

The following theorems show that $RSMP(c)$ is strongly NP-hard for $c \in \{ABS, REL, MM, OWA\}$ even when the duration $P(s)$ of the maintenance is the same and equal to the time window length in all the scenarios $s \in S$.

Theorem 3. *$RSMP(c)$ with $c \in \{MM, OWA\}$ is strongly NP-hard even when $d(s) - r(s) = P(s) = P$ is constant in all scenarios $s \in S$.*

Table 2
Summary of theoretical results.

$RSMP(c)$ characteristics	Complexity
$c = MM, OWA$, $\|S\|$ arbitrary	Strongly NP-hard (Theorem 3)
$c = ABS, REL$, $\|S\|$ arbitrary	Strongly NP-hard (Theorem 4)
$c = MM, ABS, REL$, $\|S\|$ fixed	Binary NP-hard (see Lee, 1996)
$c = MM, ABS, REL, OWA$, $\|S\|$ fixed	Pseudopolynomially solvable (Section 4.3)
$c = ABS$, $\|S\|$ arbitrary	Not approximable (Section 4.4)
$c = MM, REL, OWA$, $\|S\|$ arbitrary	$\frac{4}{3}$-approximable (Theorem 6)
$c = MM, ABS$, $\|S\|$ arbitrary, r (or d) fixed	Equivalent to both r and d fixed (Theorem 8)
$c = MM, ABS$, $\|S\|$ arbitrary, r (or d) fixed	Pseudopolynomially solvable (Cor. 11)
$c = MM$ $\|S\|$ arbitrary, r (or d) fixed	ϵ-approximable and admits FPTAS (Cor. 10)

Proof. Since OWA generalizes MM, we prove the statement for $c = MM$ only. The proof is by reduction from 3-Partition. Given an instance I_{3P} of 3-Partition, with $3n$ integers, $a_1, \dots, a_{3n}$, such that $\sum_{j=1}^{3n} a_j = nT$ and $T/4 < a_j < T/2$ for $j = 1, \dots, 3n$, we generate an instance I of $RSMP(MM)$ in which, for all $s \in \mathcal{S}$, $P(s) = P$ (i.e., the duration of M does not vary across the scenarios) with $3n$ jobs having processing times $a_1, \dots, a_{3n}$, and with $k = n - 1$ scenarios $\mathcal{S} = \{s_1, s_2, \dots, s_{n-1}\}$. For each scenario $s_i \in \mathcal{S}$, we have $r(s_i) = iT$, $d(s_i) = (i+1)T$ and $P = d(s_i) - r(s_i) = T$. We show that a solution to I_{3P} exists, if and only if, in instance I, there exists a job sequence π such that $MM(\pi) = C_{\max}(\pi, s) = T(n+1)$, for all scenarios s.

In fact, let I_{3P} be a yes-instance of 3-Partition with solution X. We build a job sequence π in which the three jobs corresponding to each triplet of X are consecutive (no matter what the relative order of the triplets is). So doing, it is easy to see that any realization schedule $\sigma(\pi, s_i)$ schedules the jobs corresponding to the first i triplets of X before the maintenance M, for all $i = 1, 2, \dots, k = n$. In fact, the maintenance in scenario s_i starts at time iT and ends at time $(i+1)T$, allowing to schedule i triplets of jobs before it. (Recall that the sum of the processing times of the jobs in each triplet is exactly equal to T.) Then, $C_{\max}(\pi, s) = T(n+1)$ in all scenarios $s \in \mathcal{S}$.

On the other hand, let us assume that I has a robust solution π such that $MM(\pi) = T(n+1)$. Hence, for all $s \in \mathcal{S}$, $C_{\max}(\pi, s) \le T(n+1)$. This in turn implies that the set $J(i)$ of jobs scheduled before M in scenario s_i has total processing time equal to iT. For all $i = 1, 2, \dots, n$, $J(i)$ corresponds to the first i triplets in the solution of I_{3P} which is therefore a yes-instance of 3-Partition. □

Theorem 4. *Problem $RSMP(c)$ with $c \in \{ABS, REL\}$, is strongly NP-hard even when $d(s) - r(s) = P(s) = P$ is constant in all scenarios $s \in \mathcal{S}$.*

Proof. Also in this case, the proof is by reduction from 3-Partition. Given an instance I_{3P} of 3-Partition, with $3n$ integers, $a_1, \dots, a_{3n}$, such that $\sum_{i=j}^{3n} a_j = nT$ and $T/4 < a_j < T/2$ for $j = 1, \dots, 3n$, we build an instance I of $RSMP(c)$ in which, for all $s \in \mathcal{S}$, $P(s) = P$, with $c = ABS$ or $c = REL$, having $3n$ jobs plus one extra job, and $k = n$ scenarios. The $3n$ jobs have processing times $a_1, \dots, a_{3n}$ and the extra job has processing time $0 < b < 1$. Hence, the sum of all job processing times is strictly larger than nT. For each scenario $s_i \in \mathcal{S}$, we let $r(s_i) = iT$, $d(s_i) = (i+1)T$ and $P = d(s_i) - r(s_i) = T$. We prove the thesis, by showing that I_{3P} is a yes-instance if and only if there exists a robust solution sequence π of I such that, for each scenario s, the realization schedule $\sigma(\pi, s)$ has no idle time, i.e., $C_{\max}(\pi, s) = (n+1)T + b$ and hence $ABS(\pi) = 0$ and $REL(\pi) = 1$. Observe that π must necessarily schedule the extra job in the last position. In fact, in the last scenario s_n, at least one job should be processed after the maintenance completing at $(n+1)T$. As all the jobs but the extra job have integer processing times (strictly larger than b), any sequence that schedules last a job different from the extra job has a makespan strictly larger than $(n+1)T + b$.

Assume first that I_{3P} is a yes-instance of 3-Partition with solution X. We build a job sequence π in which the extra job is the last job and the jobs corresponding to each triplet of X are consecutive. It is easy to see that the realization schedule $\sigma(\pi, s)$ has no idle time in all scenarios $s \in \mathcal{S}$ and π has an absolute regret equal to 0 and relative regret equal to 1 in all scenarios and hence π is a robust solution with $ABS(\pi) = 0$ and $REL(\pi) = 1$.

Let us now assume that there exists a robust solution sequence π of I such that, for each scenario s, the realization schedule $\sigma(\pi, s)$ has no idle time, i.e., $C_{\max}(\pi, s) = (n+1)T + b$ and the extra job is scheduled last. Again, this implies that the set $J(i)$ of jobs scheduled before M in scenario s_i, for all $i = 1, 2, \dots, n$, corresponds to the first i triplets in the solution of I_{3P} which is therefore a yes-instance of 3-Partition. This completes the proof.

Note that, if the problem requires integer processing times, in the above reduction, job processing times, $r(s_i)$, $d(s_i)$, and T are all multiplied by an integer factor $m \ge 2$, whereas the extra job processing time b is set equal to one. □

We now briefly discuss about the so-called scenario optimality property. If a robust optimization problem has the scenario optimality property, it means that a robust solution is always optimal in at least one scenario. In Detti et al. (2019), it is proved that the special case of $RSMP(c)$ in which $r(s)$ and $d(s)$ are fixed (and hence only the maintenance duration varies across the scenarios) has the scenario optimality property, for all considered robustness criteria c. However, this is not true for $RSMP(c)$ in general, as multiple parameters vary simultaneously in different scenarios. If scenario optimality held for $RSMP(c)$, then Section 4.1 would imply that $RSMP(c)$ could be solved in pseudopolynomial time by computing the optimal solutions in all scenarios. Nevertheless, Theorems 3 and 4, which state that $RSMP(c)$ is strongly NP-hard, rule out this possibility.

Obviously, Theorems 3 and 4 do not rule out the existence of polynomial or pseudopolynomial algorithms for some specific cases, as shown in the following Section 4.3 and in Section 5.

4.3. A pseudopolynomial algorithm for $RSMP(c)$ with a fixed number of scenarios

In this section, a dynamic programming algorithm (DP) for $RSMP(c)$ is presented. The proposed DP runs in pseudopolynomial time when a fixed number of scenarios k is considered, i.e., when $\mathcal{S} = \{s_1, \dots, s_k\}$ with k fixed.

In the description of the algorithm, as stated in Expression (1), it is important to recall that the scenarios s in $\mathcal{S}$ are sorted in non-decreasing values of $d(s) - P(s)$.

Let $F(j, w_1, w_2, \dots, w_k)$ be a boolean function equal to 1 if there exists a schedule of the jobs $\{1, \dots, j\}$ such that in each scenario s_i the total processing time of the jobs scheduled before the maintenance is *exactly* w_i, for all $s_i \in \mathcal{S}$, and 0 otherwise. Note that, due to Expression (1), we may restrict ourselves to consider w_i values such that $w_1 \le w_2 \le \dots \le w_k$. In fact, for a given schedule, the jobs scheduled before the maintenance M in scenario s_i can be also scheduled before M in scenarios $s_{i+1}, \dots, s_k$. As a consequence, we impose that $F(j, w_1, w_2, \dots, w_k) = 0$ if there exists $i \in \{1, 2, \dots, k-1\}$ such that $w_i > w_{i+1}$. Furthermore, let $F(j, w_1, w_2, \dots, w_k) = 0$ if one of the arguments $w_1, w_2, \dots, w_k$ is negative.

Consider any schedule of the jobs $\{1, \dots, j\}$ such that the total processing time of the jobs scheduled before M in scenario $s_i \in \mathcal{S}$

is exactly equal to w_i, for $i = 1, \dots, k$. Clearly, either there exists a scenario s_ℓ (with $w_\ell \geq p_j$) in which the job j is scheduled before M, or j is scheduled after the maintenance activity in all scenarios. Note that, in the first case, due to (1), job j is scheduled before M in all scenarios $s_\ell, s_{\ell+1}, \dots, s_k$. Hence, the total processing time of the other jobs scheduled before the maintenance in scenarios $s_\ell, \dots, s_k$ is exactly $w_\ell - p_j, w_{\ell+1} - p_j, \dots, w_k - p_j$, respectively. If, on the other hand, there is no scenario in which j can be processed before M, then j will be scheduled after the maintenance in all scenarios.

As an example, let us consider a problem instance with two scenarios s_1 and s_2. Then, the computation of $F(j, w_1, w_2)$ requires to check $F(j-1, w_1, w_2)$, $F(j-1, w_1, w_2 - p_j)$, and $F(j-1, w_1 - p_j, w_2 - p_j)$, corresponding to the cases in which j is scheduled after M, j is scheduled before M in s_2, only, and j is scheduled before M in all the scenarios, respectively. $F(j, w_1, w_2) = 1$ if and only if at least one of the latter three quantities equals to one.

Hence, the computation of the value $F(j, w_1, w_2, \dots, w_k)$ only requires to check if at least one of the $k+1$ entries

$$
\begin{aligned}
&F_0 = F(j-1, w_1, w_2, \dots, w_{k-1}, w_k),\\
&F_1 = F(j-1, w_1, w_2, \dots, w_{k-1}, w_k - p_j),\\
&F_2 = F(j-1, w_1, w_2, \dots, w_{k-1} - p_j, w_k - p_j),\\
&\dots\\
&F_k = F(j-1, w_1 - p_j, w_2 - p_j, \dots, w_{k-1} - p_j, w_k - p_j)
\end{aligned}
$$

takes value 1. In conclusion, the following recursive formula holds

$$F(j, w_1, w_2, \dots, w_k) = \max\left\{F_0, F_1, \dots, F_k\right\}. \tag{6}$$

Recursion (6) must be initialized by setting $F(0, 0, \dots, 0) = 1$.

Note that, if $F(n, w_1, w_2, \dots, w_k) = 1$ for given $w_1, w_2, \dots, w_k$ and a job sequence π, the makespan in the scenario s_i is $C_{\max}(\pi, s_i) = \max\{r(s_i), w_i\} + P(s_i) + (\sum_j p_j) - w_i$, since the maintenance M in scenario s_i cannot start before time $r(s_i)$ or the completion time w_i of all jobs preceding it. Consequently, the total processing time of the job after M is $(\sum_j p_j) - w_i$.

Once all the $F(\cdot)$ values have been computed, in order to obtain the optimal value for $RSMP(c)$, we use the following:

$$\min_{w} \Psi(w) \tag{7}$$

$$\text{s.t.} \quad F(n, w_1, w_2, \dots, w_k) = 1 \tag{8}$$

$$w_i \in \{0, 1, \dots, d(s_i) - P(s_i)\}, \ i = 1, 2, \dots, k \tag{9}$$

in which, depending on the robustness criterion c, we have that $\Psi(w)$ takes one of the expressions of Eqs. (2), (3), (4), and (5), where we plug in $C_{\max}(\pi, s) = \max\{r(s), w_s\} + P(s) + (\sum_j p_j) - w_s$.

For the sake of clarity, consider as an example the robustness criterion $c = MM$, then the optimal solution value of $RSMP(MM)$ is given by (7), in which

$$\Psi(w) = \max_{s_i \in S}\{\max\{r(s_i), w_i\} + P(s_i) + \sum_{j=1}^{n} p_j - w_i\}$$

Observe that each $F(j, w_1, w_2, \dots, w_k)$ can be computed in time $O(k)$ using expression (6). Since there are $O(n(\sum_j p_j)^k)$ such values, then all $F(j, w_1, w_2, \dots, w_k)$ values can be computed in time $O(nk(\sum_j p_j)^k)$. In conclusion, if $c = MM, ABS, REL$, we infer that $RSMP(c)$ can be solved in time $O(nk(\sum_j p_j)^k)$. While, for $c = OWA$, in order to compute the objective function value of Eq. (5), for each choice of $w_1, w_2, \dots, w_k$, the optimal makespan values in the different scenarios are to be sorted. As a consequence the overall computational cost becomes $O(nk^2 \log k(\sum_j p_j)^k)$.

4.4. Approximation

We now report on some approximation results for our problem $RSMP(c)$ with $c \in \{REL, MM, OWA\}$. Note that, in general, if $c = ABS$ and the deterministic (single-scenario) problem is NP-hard, the corresponding robust version is not at all approximable unless $P = NP$ (Detti et al., 2019; Kasperski & Zielinski, 2014).

The following lemma extends an approximation result that was shown in Detti et al. (2019) only for the special case of $RSMP(c)$ in which r and d remain fixed in all scenarios.

Lemma 5. *Let $c \in \{REL, MM, OWA\}$ be a robustness criterion. If algorithm $\mathcal{A}$ is an ε-approximation algorithm for the (deterministic) single-scenario version of $RSMP(c)$ that returns the same sequence $\bar{\pi}$ for all scenarios $s \in S$, then $\mathcal{A}$ is also an ε-approximate algorithm of the robust problem $RSMP(c)$.*

Proof. The proof follows the very same lines of that used in Detti et al. (2019) for the special case with fixed interval $[r, d]$. We report it for completeness.

Let π^* and $c(\pi^*)$ be the robust solution of an instance of $RSMP(c)$ and its value, respectively, according to the considered robustness criterion $c \in \{MM, REL, OWA\}$. Let π be a sequence (returned by the algorithm) and $c(\bar{\pi})$ the corresponding value of the robustness criterion c. In the remainder of the proof, $\hat{s}$ indicates the worst scenario for π under a certain criterion.

Hereafter, we show that, if π is an ε approximate solution in scenario $\hat{s}$, then it is also an approximate solution for the (general) robust problem $RSMP(c)$.

Let us first consider the min–max robustness criterion $c = MM$. In this case, $\pi^* = \arg\min_\pi \left\{\max_{s\in S}\left\{C_{\max}(\pi, s)\right\}\right\}$ and, $c(\pi^*) = \max_{s\in S}\left\{C_{\max}(\pi^*, s)\right\}$. Hence

$$
\begin{aligned}
c(\bar{\pi}) &= \max_{s\in S}\left\{C_{\max}(\bar{\pi}, s)\right\} =\\
&= C_{\max}(\bar{\pi}, \hat{s}) \leq \varepsilon\, C^*_{\max}(\hat{s}) \leq \varepsilon\, C_{\max}(\pi^*, \hat{s}) \leq \varepsilon \max_{s\in S}\left\{C_{\max}(\pi^*, s)\right\}\\
&= \varepsilon\, c(\pi^*).
\end{aligned}
$$

For the relative regret robustness $c = REL$, we have $\pi^* = \arg\min_\pi \max_{s\in S}\left\{\frac{C_{\max}(\pi,s)}{C^*_{\max}(s)}\right\}$ and $c(\pi^*) = \max_{s\in S}\left\{\frac{C_{\max}(\pi^*,s)}{C^*_{\max}(s)}\right\}$. Clearly $c(\pi^*) \geq 1$ and therefore, if $\hat{s} \in S$ is a scenario corresponding to the worst case objective ratio, the following inequalities hold:

$$c(\bar{\pi}) = \max_{s\in S}\left\{\frac{C_{\max}(\bar{\pi}, s)}{C^*_{\max}(s)}\right\} = \frac{C_{\max}(\bar{\pi}, \hat{s})}{C^*_{\max}(\hat{s})} \leq \varepsilon \leq \varepsilon c(\pi^*).$$

Finally, when the robustness criterion is the ordered weighted average $c = OWA$, then $\pi^* = \arg\min_\pi \left\{\sum_{s\in S} \beta_s C_{\max}(\pi, s)\right\}$ and $c(\pi^*) = \sum_{s\in S} \beta_s C_{\max}(\pi^*, s)$. Therefore

$$c(\bar{\pi}) = \sum_{s\in S} \beta_s C_{\max}(\bar{\pi}, s) \leq \sum_{s\in S} \beta_s \varepsilon C^*_{\max}(s) \leq \varepsilon \sum_{s\in S} \beta_s C_{\max}(\pi^*, s) = \varepsilon c(\pi^*).$$

In conclusion, we have shown that for all three robustness criteria $c = MM, REL, OWA$, if $C_{\max}(\bar{\pi}, \hat{s}) \leq \varepsilon C^*_{\max}(\hat{s})$ then $c(\bar{\pi}) \leq \varepsilon c(\pi^*)$ and hence the thesis holds. □

In what follows, approximation results for two simple ordering rules are provided. Let us consider the deterministic (single scenario) version of $RSMP(c)$, that is the problem of seeking a schedule σ minimizing the objective $C_{\max}(\sigma)$ in which the maintenance activity with a given duration has to be scheduled in a certain given interval. Lee in Lee (1996) shows that the Longest Processing Time (LPT) rule provides a 4/3-approximation algorithm for this problem and also proves that the bound is tight. Hence, this result and Lemma 5 directly imply the following statement.

Theorem 6. *The LPT rule is a $\frac{4}{3}$-approximation algorithm for $RSMP(c)$ with $c \in \{REL, MM, OWA\}$.*

In Kacem and Kellerer (2016), a very basic algorithm, called *Algorithm A*, is proposed. Algorithm A simply consists in processing first the longest job with processing time $p_{\max} = \max_j p_j$. Kacem and Kellerer (2016) prove that this algorithm provides a 3/2 approximation for the deterministic version of $RSMP(c)$. Hence, one may want to adopt this linear time algorithm, at the expense of a slightly worse approximation, and thus obtaining a $\frac{3}{2}$-approximation for $RSMP(c)$, with $c \in \{REL, MM, OWA\}$.

5. Special cases

In this section we focus on different subproblems of $RSMP(c)$ obtained when some of maintenance parameters r, d and P do not vary across the scenarios, i.e., they take the same fixed values in all the scenarios. For instance, recall that if r and d are fixed and only the duration of the maintenance varies across the scenarios, then $RSMP(c)$ is equivalent to the problem addressed in Detti et al. (2019). If all the three quantities r, d, and P are fixed, then $RSMP(c)$ is a deterministic single-scenario problem in which we are seeking for a schedule minimizing the makespan, and the maintenance activity with a given duration has to be scheduled in a certain given interval (Lee, 1996). As we observed before, the latter problem is already binary NP-hard. Furthermore, Theorems 3 and 4 prove that $RSMP(c)$ is strongly NP-hard for all the considered robustness criteria, when P is fixed across all scenarios (i.e., only r and d vary).

5.1. Problems with fixed r or d

In this section, we show that when dealing with min–max and absolute regret robustness criteria i.e., $c = MM$ and ABS, the special cases of $RSMP(c)$ in which r or d are fixed are equivalent to the problem in which both r and d are fixed, which is the problem studied in Detti et al. (2019).

In the remainder of this section, we describe the procedure for reducing $RSMP(c)$ with fixed r in all the scenarios, hereafter denoted as $\mathcal{P}$, to the problem in which r and d are both fixed, indicated as $\mathcal{Q}$. In particular if $c = MM$ or $c = ABS$, we show that, given an instance $I_{\mathcal{P}}$ of $\mathcal{P}$, it is possible to build an instance $I_{\mathcal{Q}}$ of $\mathcal{Q}$ such that the robust solution sequence for $I_{\mathcal{Q}}$ is also robust for $I_{\mathcal{P}}$. An analogous procedure can be devised for $RSMP(c)$ with fixed d.

We assume that $\mathcal{S}$, the set of scenarios for problem $\mathcal{P}$, is discrete and contained in the Cartesian product of the two (discrete) sets $\mathcal{S}^P$ and $\mathcal{S}^D$ of realizations of the variable quantities P and d, respectively. Assuming $P_u < P_{u+1}$, $u = 1, \dots, q-1$, and $d_v < d_{v+1}$, $v = 1, \dots, t-1$, we have:

$$\mathcal{S}^P = \{P_1 = P_{\min}, P_2, \dots, P_q = P_{\max}\}, \tag{10}$$

$$\mathcal{S}^d = \{d_1 = d_{\min}, d_2, \dots, d_t = d_{\max}\}, \tag{11}$$

$$\mathcal{S} \subseteq \mathcal{S}^P \times \mathcal{S}^d. \tag{12}$$

Let $I_{\mathcal{P}}$ be any instance of $\mathcal{P}$ with $c = MM$ or $c = ABS$, and a set of scenarios $\mathcal{S}$ as in Eq. (12). Any possible scenario in $\mathcal{S}$ is associated to a pair of possible realizations (values) of the duration P for the maintenance activity M and the due date d, respectively. We denote by (P_u, d_v) the scenario of $\mathcal{S}$ in which the duration of M and the due date d take values P_u and d_v, respectively. (Possibly, some given pairs (P_u, d_v) are not part of $\mathcal{S}$, i.e., $|\mathcal{S}| = k \le qt$.)

Now, starting from $I_{\mathcal{P}}$, we build the following instance $I_{\mathcal{Q}}$ of $\mathcal{Q}$ with the same set of jobs as in instance $I_{\mathcal{P}}$ and a set of scenarios $\tilde{\mathcal{S}} = \{\tilde{P}_{uv},\ u = 1, \dots, q;\ v = 1, \dots, t\ :\ (P_u, d_v) \in \mathcal{S}\}$ in which

$$\tilde{P}_{uv} = P_u + d_{\max} - d_v \forall (P_u, d_v) \in \mathcal{S}. \tag{13}$$

Here $\tilde{P}_{uv}$ indicates the value taken by the duration of maintenance activity in the scenario of $\mathcal{Q}$ corresponding to realizations P_u and d_v for the maintenance activity duration and due date, respectively, in $\mathcal{P}$. The (fixed) extremes of the time window $[\tilde{r}, \tilde{d}]$ of $I_{\mathcal{Q}}$ are set equal to r and, respectively, to the largest realization $d_{\max}$ of d in $I_{\mathcal{P}}$. For example, if scenarios $\tilde{P}_{1t}$ or $\tilde{P}_{q1}$ belong to $\mathcal{S}$, then they are the smallest and largest realizations for $\tilde{P}$ in $I_{\mathcal{Q}}$, corresponding to scenarios $(P_1, d_{\max})$ and $(P_{\max}, d_{\min})$ in $I_{\mathcal{P}}$, respectively.

Lemma 7. *Given a sequence π of the jobs, the idle time lengths in the realization schedules $\sigma(\pi, (P_u, d_v))$ in instance $I_{\mathcal{P}}$ of $RSMP(c)$ with fixed r (problem $\mathcal{P}$) and $\sigma(\pi, \tilde{P}_{uv})$ in $I_{\mathcal{Q}}$ of $RSMP(c)$ with fixed r and d (problem $\mathcal{Q}$) are equal.*

Proof. Clearly, in any realization schedule, the idle time (if any) occurs *before* the execution of the maintenance activity which therefore would start at time r, the left extreme of the time window. In the two realization schedules, the maximum available time ϑ for processing jobs before M is the same in the two corresponding scenarios (P_u, d_u) of $I_{\mathcal{P}}$ and $\tilde{P}_{uv}$ of $I_{\mathcal{Q}}$: In fact, in $I_{\mathcal{Q}}$, we have $\vartheta = \tilde{d} - \tilde{P}_{uv}$. From (13), since $\tilde{d} = d_{\max}$, we have that $\vartheta = d_{\max} - (P_u + d_{\max} - d_v) = d_v - P_u$ which is obviously the ϑ value in scenario (P_u, d_v) of instance $I_{\mathcal{P}}$. Since $\tilde{r} = r$, the idle time produced by a sequence π, would be the same in $I_{\mathcal{P}}$ and $I_{\mathcal{Q}}$. □

Suppose that, in two interrelated scenarios $s = (P_u, d_v) \in \mathcal{S}$ for $I_{\mathcal{P}}$ and $\tilde{s} = \tilde{P}_{uv} = P_u + d_{\max} - d_v \in \tilde{\mathcal{S}}$ for $I_{\mathcal{Q}}$, the maximum available time ϑ for processing jobs before the maintenance is strictly smaller than $\sum_{j \in J} p_j$, i.e., in both instances $I_{\mathcal{P}}$ and $I_{\mathcal{Q}}$, there must be some jobs processed after the maintenance. Lemma 7 implies that the makespan values of the realization schedules of a same sequence π in $I_{\mathcal{P}}$ and $I_{\mathcal{Q}}$ are such that:

$$C_{\max}(\pi, \tilde{s}) = C_{\max}(\pi, s) + d_{\max} - d_v. \tag{14}$$

On the other hand, if $\vartheta \ge \sum_{j \in J} p_j$ then the solution of the two instances becomes trivial and the two makespan values would be equal.

In any case, a sequence π which is a makespan minimizer for scenario s of instance $I_{\mathcal{P}}$, is also optimal in the corresponding scenario $\tilde{s}$ of instance $I_{\mathcal{Q}}$. Furthermore, in these scenarios s and $\tilde{s}$, the absolute regret of any solution sequence π has the same value which implies that, if π minimizes the maximum absolute regret in instance $I_{\mathcal{Q}}$, then it does so in instance $I_{\mathcal{P}}$, as well.

The above discussion implies the equivalence, in terms of optimality, of problems $\mathcal{P}$ and $\mathcal{Q}$. In conclusion the following theorem holds:

Theorem 8. *For $c \in \{ABS, MM\}$, if π is robust for the equivalent instance $I_{\mathcal{Q}}$ of $RSMP(c)$ with fixed r and d (problem $\mathcal{Q}$), then π is robust also for the original instance $I_{\mathcal{P}}$ of $RSMP(c)$ with fixed r (problem $\mathcal{P}$).*

As we already noted, using arguments similar to those employed above, we can prove that, for $c \in \{ABS, MM\}$, given an instance I' of $RSMP(c)$ in which d is fixed, it is possible to define an instance $I'_{\mathcal{Q}}$ of $\mathcal{Q}$ such that a robust solution sequence for the two instances is the same. Moreover, it is important to point out that the result of Theorem 8 does not hold if $c = REL$ or $c = OWA$ or if we are seeking for approximate solutions instead of optimal ones.

In the next Section 5.1.1, we study a special case of $RSMP(c)$ in which we may adapt and use off-the-shelf approximation algorithms.

5.1.1. Approximation results for a special case

In this section, we present approximation results for a special case of $RSMP(c)$ in which the following specific restrictions hold: (i) $c = MM$, (ii) r or d are fixed; (iii) the set of possible scenarios, is *exactly* the Cartesian product:

$$\mathcal{S} = \mathcal{S}^P \times \mathcal{S}^d, \tag{15}$$

where $\mathcal{S}^P$ and $\mathcal{S}^d$ are defined as in Eqs. (10) and (11), i.e., *any pair* (P_u, d_v), $u = 1, \dots, q$, $v = 1, \dots, t$, is a possible scenario for our robust optimization problem. In the remainder of this section, we refer to this special subproblem of $RSMP(c)$ as problem $\mathcal{R}$.

With such a scenario set S, it is easy to see that the maximum makespan of the realization schedules always occurs in scenario $\hat{s} = (P_{\max}, d_{\min})$ in which the maintenance activity has the largest duration P_q and the due date takes its minimum value d_1. In this case, for any solution sequence π, we have $\max_{s\in S}\{C_{\max}(\pi, s)\} = C_{\max}(\pi, \hat{s})$.

An obvious consequence is that, under the min–max robustness criterion $c = MM$, any solution sequence $\bar{\pi}$ within a constant ratio $\varepsilon > 1$ from an optimal solution of scenario $\hat{s}$ is also an ε-approximating solution for $\mathcal{R}$, i.e., if π^* is a robust solution, then

$$MM(\bar{\pi}) = C_{\max}(\bar{\pi}, \hat{s}) \leq \varepsilon\ C^*_{\max}(\hat{s}) = \varepsilon\ MM(\pi^*). \tag{16}$$

The following theorem shows how an approximation algorithm devised for Subset Sum can be exploited to obtain an approximation algorithm for $\mathcal{R}$.

Theorem 9. *Any ρ-approximation algorithm for Subset Sum, can be adapted to obtain an ε-approximate solution algorithm for $RSMP(MM)$ with r fixed (problem $\mathcal{R}$), where $\varepsilon = 1 + \frac{(1-\rho)(d_{\min}-P_{\max})}{\sum_{j\in J} p_j + P_{\max}}$.*

Proof. The proof uses arguments similar to those of Theorem 18 in Detti et al. (2019). We report the details hereafter for the reader's benefit. In its optimization format, Subset Sum is defined as follows: Given a set E of n positive integers $a_1, a_2, \ldots, a_n$ and a bound $B > 0$, define a subset $E^* \subseteq E$ of items whose sum is as large as possible but not greater than B. Suppose there is an algorithm A that, for any instance of Subset Sum, always returns a subset $E^A \subseteq E$ such that

$$\sum_{j\in E^A} a_j \geq \rho \sum_{j\in E^*} a_j \tag{17}$$

for some fixed $\rho \leq 1$. Recall that, for $\mathcal{R}$, (i) the worst scenario $\hat{s}$ corresponds to realizations $P = P_{\max}$ and $d = d_{\min}$ for the duration of the maintenance activity and the due date, respectively; and (ii) a solution sequence minimizing the makespan in scenario $\hat{s}$ is also a robust (optimal) solution of our problem.

Hereafter, given a subset of jobs $J' \subseteq J$, we indicate the total processing time $\sum_{j\in J'} p_j$ of all jobs in J' as $p(J')$. Let π^* be a robust (optimal) solution of a given instance of $\mathcal{R}$ and denote by $C^*_{\max} = C_{\max}(\pi^*, \hat{s})$ the value of its (optimal) makespan. Let $E^* \subseteq J$ be the set of jobs which are scheduled *before* the maintenance activity M in $\sigma(\pi^*, \hat{s})$. Then, it is easy to see that E^* corresponds to an optimal solution set of a Subset Sum instance with $a_j = p_j$ and $B = d_{\min} - P_{\max}$. Moreover, we have $C^*_{\max} = \max\{r, p(E^*)\} + P_{\max} + (p(J) - p(E^*))$.

Now consider a solution π^A obtained by sequencing first the jobs in a set $E^A \subseteq J$ corresponding to the set returned by Algorithm A in the instance where, again, $a_j = p_j$ for all $j \in J$ and $B = d_{\min} - P_{\max}$. Let $C^A_{\max}$ be the makespan $C_{\max}(\pi^A, \hat{s})$ of the realization schedule obtained by π^A in scenario $\hat{s}$. Clearly, $p(E^A) \geq \rho\, p(E^*)$ and $C^A_{\max} = \max\{r, p(E^A)\} + P_{\max} + (p(J) - p(E^A))$. We have three possible cases.

Case 1: $p(E^*) < r$. As $p(E^A) \leq p(E^*)$ then $C^A_{\max} - C^*_{\max} = p(E^*) - p(E^A) \leq (1-\rho)p(E^*)$. As there exists $\theta \leq 1$ such that $p(E^*) \leq \theta\, C^*_{\max}$, by letting $\varepsilon = 1 + \theta - \theta\rho$, we have $C^A_{\max} \leq (1-\rho)p(E^*) + C^*_{\max} \leq \varepsilon\, C^*_{\max}$.

Case 2: $p(E^A) < r \leq p(E^*)$. In this case, we have that $C^A_{\max} - C^*_{\max} = r - p(E^A) \leq p(E^*) - p(E^A)$ and $C^A_{\max} \leq \varepsilon\, C^*_{\max}$ still holds.

Case 3: $r \leq p(E^A) \leq p(E^*)$. No idle time is introduced in the sequence π^A which is therefore an optimal solution.

In conclusion, due to (16), there is an ε-approximation algorithm for $\mathcal{R}$. In order to have a better approximation ratio $\varepsilon = 1 + \theta - \theta\rho$, we would like a large $\rho \leq 1$ and a small $\theta \geq \frac{p(E^*)}{C^*_{\max}}$. For instance, we may always choose $\theta = \frac{d_{\min}-P_{\max}}{P(J)+P_{\max}}$. □

Due to the above theorem and the fact that the Subset Sum problem admits a Fully Polynomial Time Approximation Scheme (FPTAS) (Kellerer, Pferschy, & Pisinger, 2003), the following corollary is immediate.

Corollary 10. *$RSMP(MM)$ with r fixed (problem $\mathcal{R}$) admits a FPTAS.*

We point out that, Corollary 10 is also implied by the results provided in the paper by He et al. (2005) (dealing with the deterministic version of the problem). However our approach has a lower computational complexity $O(n/\epsilon)$ (see Kellerer, Mansini, Pferschy, & Speranza, 2003) which is obtained by running the fastest FPTAS for Subset Sum, as illustrated in the proof of Theorem 9. By Corollary 10, it also follows that $\mathcal{R}$, for $\ell = r, d$ is pseudopolynomially solvable.

As proved in Detti et al. (2019), for any given criterion $c \in \{ABS, REL, OWA\}$, the robust solution for $RSMP(c)$ with fixed r and d (i.e. problem $\mathcal{Q}$), satisfies the scenario optimality, that is it always corresponds to an optimal solution for one of the scenarios. Hence, by Theorem 8, the following result holds:

Corollary 11. *If $c = MM$ or $c = ABS$, the special case of $RSMP(c)$ with fixed r ($\mathcal{P}$ above) is binary NP-hard and can be solved in pseudopolynomial time, even for an arbitrary number of scenarios.*

We conclude by stressing that all the properties presented in this section for $\mathcal{R}$ (namely, approximation results, Theorem 9, Corollary 10, and Corollary 11) can be easily extended to the special case of $RSMP(c)$ with fixed d.

6. Mixed integer linear programming models

In this section, we present two mathematical programming models for $RSMP(c)$. The first one (MIP_1) uses assignment and positional variables. A similar model has been proposed in Detti, Nicosia, Pacifici, and de Lara (2016) where it turned out to the best performing model among a set of different integer programs. The second model (MIP_2) disregards the actual sequencing of the jobs: it only considers whether a job is scheduled before or after the maintenance activity in each scenario.

6.1. MIP_1: Assignment and positional-variable model

Among the classical MIP models for single machine scheduling, and differently from scheduling and transportation problems in which precedence variables with disjunctive constraints and time-indexed variables yield better formulations (see, e.g., Agnetis, Cosmi, Nicosia, & Pacifici, 2023; Benini, Detti, & de Lara, 2022), in our case, a model based on assignment and positional-variables proves to be the best choice. In MIP_1, two types of integer variables, denoted as x and y, and a set of continuous variables denoted as C are used. Positional variables x assign to each job a position in the job sequence. More precisely, we consider binary variables x_{jh}, defined for all $j, h = 1, \ldots, n$, indicating whether job j is the hth job in the solution sequence π (disregarding the position of the maintenance activity in the sequence). We also let $y_h(s)$ be binary variables, defined for $h = 2, \ldots, n$, indicating whether the maintenance activity is scheduled between the $(h-1)$-th and hth job in $\sigma(\pi, s)$. Additionally, $y_1(s)$ and $y_{n+1}(s)$ are binary variables which are equal to 1 if the maintenance activity is scheduled before or after all the n jobs, respectively. Moreover, $C_h(s)$ are variables, defined for all positions $h = 1, \ldots, n$ and all scenarios $s \in S$, representing the completion time of the job in position h in π. Similarly, variable $C_M(s)$, defined for all scenarios $s \in S$, indicates the completion time of the maintenance activity in $\sigma(\pi, s)$.

The model MIP_1 for the robustness criterion $c = MM$ reads as follows:

$$\min \max_{s\in S} \{C_n(s)\} \tag{18}$$

$$\sum_{h=1}^{n} x_{jh} = 1 \qquad j \in J \tag{19}$$

$$\sum_{j\in J} x_{jh} = 1 \qquad h = 1, \ldots, n \tag{20}$$

$$\sum_{h=1}^{n+1} y_h(s) = 1 \qquad s \in S \tag{21}$$

$$C_h(s) \geq C_{h-1}(s) + \sum_{j=1}^{n} p_j x_{jh} \qquad h = 1, \ldots, n,\ s \in S \qquad (22)$$

$$C_h(s) \geq C_M(s) + \sum_{j=1}^{n} p_j x_{jh} - N\left(1 - \sum_{q=1}^{h} y_q(s)\right) \qquad h = 1, \ldots, n,\ s \in S \qquad (23)$$

$$C_M(s) \geq C_h(s) + P(s) - N\left(\sum_{q=1}^{h} y_q(s)\right) \qquad h = 1, \ldots, n,\ s \in S \qquad (24)$$

$$C_M(s) \geq r(s) + P(s) \qquad s \in S \qquad (25)$$

$$C_M(s) \leq d(s) \qquad s \in S \qquad (26)$$

$$x_{jh} \in \{0, 1\} \qquad j \in J,\ h = 1, \ldots, n \qquad (27)$$

$$y_h(s) \in \{0, 1\} \qquad h = 2, \ldots, n,\ s \in S \qquad (28)$$

$$C_h(s) \geq 0 \qquad h = 1, \ldots, n,\ s \in S \qquad (29)$$

$$C_M(s) \geq 0 \qquad s \in S \qquad (30)$$

The expression $\sum_{j=1}^{n} p_j x_{jh}$, in constraints (22) and (23) indicates the processing time of the job in position h of π while, $\sum_{q=1}^{h} y_q(s)$, in constraints (23) and (24) takes value 1 if the maintenance is before the hth job in $\sigma(\pi, s)$.

The first three constraints are standard assignment constraints. Additional constraints define the values $C_h(s)$ of the completion times, in the different scenarios. More precisely, constraints (22) and (23) define a lower bound on the completion times of the jobs (C_0 is set equal to zero). In constraints (23), N is a suitable large constant that can be set equal to the total processing time of the jobs (including the processing time of the maintenance). Constraints (24) define a lower bound on the completion time of the maintenance, while constraints (25) and (26) impose that the maintenance activity is performed within the given window by setting a lower and upper bound on its completion time for each scenario.

The objective function (18) – which can be trivially linearized – models the min–max robustness criterion $c = MM$. For the ABS and REL criteria, it can be modified respectively as:

$$\min \max_{s \in S} \left\{ C_n(s) - C^*_{\max}(s) \right\} \qquad (31)$$

$$\min \max_{s \in S} \left\{ C_n(s) / C^*_{\max}(s) \right\}. \qquad (32)$$

In this case, the values $C^*_{\max}(s)$, $s \in S$, have to be pre-computed through the efficient procedure illustrated at the end of Section 4.1. The number of variables and constraints in the above MIP formulation is $O(n^2 + n|S|)$ and $O(n|S|)$, respectively.

For the OWA criterion, we need a set of additional variables $\chi_i \geq 0$ and $u_i(s) \in \{0, 1\}$, for $s \in S$, $i = 1, \ldots, k$. Binary variable $u_i(s) = 1$ indicates that the makespan $C_n(s)$ of the schedule in scenario s is the ith largest makespan among the $k = |S|$ makespan values, and χ_i would equal such ith value. We may then add the following set of constraints:

$$\sum_{s \in S} u_i(s) = 1 \qquad i = 1, \ldots, k \qquad (33)$$

$$\sum_{i=1}^{k} u_i(s) = 1 \qquad s \in S \qquad (34)$$

$$\chi_i \geq \chi_{i+1} \qquad i = 1, \ldots, k-1 \qquad (35)$$

$$\chi_i \geq C_n(s) - N(1 - u_i(s)) \qquad s \in S,\ i = 1, \ldots, k. \qquad (36)$$

Besides the obvious assignment constraints (33) and (34) that give one of the k possible ranks for the makespan of scenario s, the set of relations (35) and (36) (where, as above, N is a suitably large constant, e.g., $N = d_{\max} + \sum_{j=1}^{n} p_j$) guarantee that χ_i upper bounds the ith largest value of the makespan values. In conclusion, the objective function when the robustness OWA criterion is used, can be expressed as:

$$\min \left\{ \sum_{i=1}^{k} \beta_i \chi_i \right\} \qquad (37)$$

in which β_i is the weight associated to the ith largest makespan scenario (see Eq. (5)). Note that, since the objective pushes the χ_i variables to assume their lowest possible values, together with Eqs. (35), it is ensured that χ_i would equal the ith largest value among the $C_n(s_1), \ldots, C_n(s_k)$.

6.2. MIP_2: Indicator variable model

Hereafter, we present a different mathematical program, denoted as MIP_2, in which we use binary variables $x_j(s)$, defined for all jobs $j \in J$ and all scenarios $s \in S$, equal to 1 if job j is scheduled before M in scenario s and 0 otherwise. Furthermore, let variables $t(s)$ specify the starting time of the maintenance M in scenario s. Clearly, we have that the schedule makespan $C_{\max}$ in scenario s is:

$$t(s) + P(s) + \sum_{j \in J} p_j (1 - x_j(s)).$$

It is important to recall that Eq. (1) holds, i.e., the k scenarios in S are sorted according to non-decreasing values of $d(s) - P(s)$. This ordering implies that there always exists a solution sequence σ such that, if a job j is scheduled before the maintenance M in a scenario s_i, then j is scheduled before M in all scenarios $s_{i'}$ with $i' > i$. MIP_2 when $c = MM$ is presented below.

$$\min \max_{s \in S} \left\{ t(s) + P(s) + \sum_{j \in J} p_j (1 - x_j(s)) \right\} \qquad (38)$$

$$\sum_{j \in J} p_j x_j(s) \leq t(s) \qquad s \in S \qquad (39)$$

$$x_j(s_{i+1}) \geq x_j(s_i) \qquad j \in J,\ i = 1, \ldots, k-1 \qquad (40)$$

$$t(s) \geq r(s) \qquad s \in S \qquad (41)$$

$$t(s) + P(s) \leq d(s) \qquad s \in S \qquad (42)$$

$$x_j(s) \in \{0, 1\} \qquad j \in J,\ s \in S \qquad (43)$$

$$t(s) \geq 0 \qquad s \in S \qquad (44)$$

Constraints (39) state that the starting time of M in scenario s cannot be smaller than the total processing times of jobs assigned before M in scenario s. Constraints (40) imply that if a job j is assigned before M in scenario s_i than j is assigned before M in all scenarios $s_{i'}$ with $i' > i$. Constraints (41) and (42) define lower and upper bounds for the starting time of M in each scenario $s \in S$. The number of variables and constraints in the above MIP formulation is $O(n|S|)$.

The above MIP can be easily adjusted to model different robustness criteria. The expression

$$z(s) = t(s) + P(s) + \sum_{j \in J} p_j (1 - x_j(s)) \qquad (45)$$

measures the makespan in scenario s, so that, the objective functions for the ABS and REL criteria are those indicated, respectively, in Eqs. (31) and (32) in which $C_n(s)$ is replaced by $z(s)$. For the OWA criterion, we need again to add the constraints indicated in Eqs. (33)–(36) (with $z(s)$ in the place of $C_n(s)$) and the objective function is the one in Eq. (37).

7. Computational experiments

In this section we present the results of a computational campaign carried out to assess the effectiveness of the two MIP models and the dynamic program presented in the previous sections. To this aim, different classes of instances have been randomly generated and tested.

Table 3
Experimental results of MIP_1 and MIP_2 on the instances of Set_1 for $c = MM$.

n	MIP_1 time	MIP_2 time	Obj	# opt scen	LB_1	LB_2	LB_1 time	LB_2 time
20	1.49	0.05	642.3	19	554.3	642.3	0.05	0.02
40	5.81	0.06	1188.4	20	1095.6	1188.4	0.15	0.02
60	23.34	0.08	1763.8	19	1675.2	1763.8	0.40	0.02
80	97.81	0.07	2293.7	19	2198.8	2293.7	0.64	0.02
100	162.19	0.08	2794.4	19	2703.1	2794.4	1.01	0.02

Table 4
Experimental results of MIP_1 and MIP_2 on the instances of Set_1 for $c = ABS$.

n	MIP_1 time	MIP_2 time	Obj	# opt scen
20	0.97	0.14	0.10	19
40	4.14	0.09	0.00	20
60	14.26	0.06	0.10	19
80	22.48	0.08	0.05	19
100	72.65	0.14	0.10	19

Table 5
Experimental results of MIP_1 and MIP_2 on the instances of Set_1 for $c = REL$.

n	MIP_1 time	MIP_2 time	Obj	# opt scen
20	0.79	1.02	1.000150	19
40	3.22	0.04	1.000000	20
60	13.17	0.04	1.000057	19
80	22.50	0.07	1.000020	19
100	69.87	0.09	1.000037	19

7.1. MIP performance assessment

In order to evaluate the MIP models presented in Section 6, two sets of instance classes have been generated and tested. In Section 7.1.1, we describe a set of experiments on a first set of instances denoted as Set_1. The results of these tests gave us some insight on the type of solutions we obtain and provided hints on the design of the second set of (harder) instances, denoted as Set_2, which are discussed in Section 7.1.2. The description of the computational experiments on Set_2 is illustrated in Section 7.1.3. All the experiments have been performed using Cplex Optimizer version 12.10, on a 1.19 GHz computer equipped with 8 GB of RAM. A time limit of 30 min has been set in each run. We emphasize that in experiments with absolute and relative robustness criteria, we initially determine the optimal solution values $C^*_{\max}(s)$ for all scenarios using the procedure described in Section 4.1. This process requires a negligible amount of computation time compared to the time taken by the solvers to handle the MIP models. The CPU-times presented in the tables of this section include both the solver processing times and the pre-processing times.

7.1.1. Experiments on instances of Set_1

Set_1 contains 100 randomly generated instances partitioned into 5 classes of 20 instances each, characterized by a different number n of jobs, with $n \in \{20, 40, 60, 80, 100\}$. In each instance, there are $k = 4$ scenarios corresponding to maintenance windows distributed over the time span of the schedule. The integer processing times of the jobs are uniformly distributed in the range $[5, 50]$, and, for each scenario s, the maintenance activity duration $P(s)$ is uniformly drawn in the interval $[50, 100]$, whereas the time window slack $\Delta(s) = d(s) - P(s) - r(s)$ assumes random integer values uniformly distributed in $[0, 3]$.

The computational results on the two MIP models for this set of instances are summarized in Tables 3–5, for $c = MM, ABS, REL$, respectively. In all the tables, the second and third column report the average computation time in seconds over the 20 instances required by the two MIP models, while in the fourth column "Obj" indicates the average objective function value. In the fifth column, the number of times in which the robust solution is also optimal in all four scenarios (out of the 20 instances in each class) is reported, this entry is denoted as "# opt scen". In Table 3, the average solutions of the lower bounds provided by the linear relaxations of the two models, denoted as "LB_1" and "LB_2", are also reported, as well as their computation times in seconds (indicated as "LB_1 time" and "LB_2 time").

A few comments are in order:

- Cplex is extremely fast on MIP_2, requiring at most one second on average on each class, while it spends more time on MIP_1. In fact, when $n = 100$, MIP_1 requires between 70 to 162 s on average, depending on the robustness criterion.
- As columns "# opt scen" of Tables 3–5 show, in most of the cases the robust solution coincides with a sequence which is optimal in all four scenarios in all the classes, for all robustness criteria. Also, in most cases, for all the three robustness criteria, we obtain solutions with zero idle times in all the scenarios, which are, trivially, optimal.
- The solutions seem highly insensitive to the particular robustness criterion adopted. The robust solutions (sequences) obtained when $c = ABS$ or $c = REL$ are almost always the same solutions obtained with $c = MM$. The computation times required to determine robust solutions when $c = ABS$ or $c = REL$ are very similar, while they increase when $c = MM$.
- We also observed that, when $c = MM$, for almost all randomly generated instances, the worst case scenario $\hat{s}$ (for which $C^*_{\max}(\hat{s}) \geq C^*_{\max}(s)$, for all $s \in S$) occurs when the maintenance activity M has the largest duration $P_{\max} = \max_{s \in S}\{P(s)\}$, making the other scenarios irrelevant in terms of selection of a robust solution.
- It is evident from Table 3 that the optimal solution of the linear relaxation of MIP_2, denoted by LB_2, is better than that of MIP_1, and is always equal to the optimal solution value. Furthermore, the computation time of LB_2 is extremely small, 0.02 s on average, while LB_1 requires 0.45 s on average.

All above considerations suggest that the instances in Set_1 are somewhat "manageable" instances. Hence, to better test the performance of the two MIP, we developed more challenging, possibly harder, instances which are described in the next section.

7.1.2. Design of the Set_2 instances

In building the instances of Set_2, we kept in mind the results of the previous section (see the comments above) in order to rule out easy or less significant tests. To this aim, instances in Set_2 have been chosen so that the robust solution does not coincide with the optimal solutions of the scenarios and so that schedules may present some idle time before the maintenance.

The following easy observations are useful to rule out these trivial cases for $RSMP(c)$. First note that, if the job processing times are not larger than the slack $\Delta(s) = d(s) - r(s) - P(s)$ in all the scenarios $s \in S$, then any sequence produces a schedule with no idle times (since a subset of jobs A always exists such that M can be scheduled just after A with no idle time) and hence any sequence is optimal. To avoid this phenomenon we choose very small values for the slacks $\Delta(s)$. Furthermore, when $r(s) = 0$ for all $s \in S$, it easy to see that an optimal schedule exists with no idle time by starting the maintenance at time 0.

Table 6
Classes of instances in Set_2.

Instance Class	# jobs	$[p_{\min}, p_{\max}]$	# scenarios	$P(s)$	window positions	slack
20-U	20	[50,150]	4	50	Spread (U)	[0,1]
20-E	20	[50,150]	4	50	Early (E)	[0,1]
20-M	20	[50,150]	4	50	Median (M)	[0,1]
20-L	20	[50,150]	4	50	Late (L)	[0,1]
30-U	30	[50,150]	4	50	Spread (U)	[0,1]
30-E	30	[50,150]	4	50	Early (E)	[0,1]
30-M	30	[50,150]	4	50	Median (M)	[0,1]
30-L	30	[50,150]	4	50	Late (L)	[0,1]
40-U	40	[50,150]	4	50	Spread (U)	[0,1]
40-E	40	[50,150]	4	50	Early (E)	[0,1]
40-M	40	[50,150]	4	50	Median (M)	[0,1]
40-L	40	[50,150]	4	50	Late (L)	[0,1]
50-U	50	[50,150]	4	50	Spread (U)	[0,1]
50-E	50	[50,150]	4	50	Early (E)	[0,1]
50-M	50	[50,150]	4	50	Median (M)	[0,1]
50-L	50	[50,150]	4	50	Late (L)	[0,1]

Taking all the above considerations into account we randomly generated a set of 320 instances, characterized by the data listed below (and summarized in Table 6).

- The number of jobs n varies in $\{20, 30, 40, 50\}$.
- The number of scenarios is $k = 4$.
- Processing times values are uniformly distributed in interval $[p_{\min}, p_{\max}] = [50, 150]$.
- The maintenance activity duration is fixed to $P(s) = P = 50$ for each scenario $s \in S$.
- In each scenario $s \in S$, the window slack $\Delta(s) = d(s) - P - r(s)$ assumes values equal to 0 or 1, with equal probability.
- The release dates and due dates, for all the k scenarios, are generated by varying the position of the maintenance time window $[r(s), d(s)]$ in the schedule. More precisely, since in general, any realization schedule spans from time 0 to $T \gtrapprox P + \sum_{j \in J} p_j$, we consider the following four modalities (which exclude trivial cases):
  - the k time windows in the different scenarios are spread, i.e., uniformly distributed over the span $[0, T]$ of the schedule (we refer to this choice using the letter U in the class name);
  - the k time windows are chosen close to each other roughly around $\frac{1}{4}T$ ("early time windows", denoted by E);
  - the k time windows are chosen close to each other roughly around $\frac{1}{2}T$ ("median time windows", denoted by M);
  - the k time windows are chosen close to each other roughly around $\frac{3}{4}T$ ("late time windows", denoted by L).

Note that in these instances $P(s)$ is fixed in all scenarios, since – as we observed in Section 7.1.1 – in most cases the worst case scenario occurs in correspondence of the largest duration for the maintenance.

In the following, we denote by n-pos an instance class of Set_2 in which n is the number of jobs of the instances of that class, while pos indicates the maintenance time-window positions, with pos $\in$ {U, E, M, L}. Table 6 summarizes the characteristics of all the instances classes in Set_2.

For each instance class n-pos, 20 instances are randomly generated.

7.1.3. Computational results on instances of Set_2

Tables 7–9 present the results of the computational experiments on the instances in Set_2 for $c = MM, ABS, REL$, respectively. In all three tables, Column 1 contains the instance class, Columns 2 and 3 report the number of instances ("# opt") of each class solved to optimality by MIP_1 and MIP_2, respectively, within the time limit, while Columns 4 and 5 report the average computation times in seconds. Column 6 gives the average solution values ("Obj") of the robust solutions found by the two formulations on the 20 instances of each class.

Table 7, presents additional four columns that report the values of the lower bounds ("LB value") obtained by the linear relaxations of the two MIPs together with their computation time ("LB time").

As shown in Table 7 ("# opt", in Columns 2 and 3), for the MM criterion, MIP_1 is able to solve all the 20 instances of each class within the time limit, while MIP_2 fails to certify the optimality for some instances with $n \in \{30, 40, 50\}$, especially for time window positions E, L and M. A similar behavior can be also observed in Tables 8 and 9 for the ABS and REL criteria. (Actually, MIP_1 is able to optimally solve all the instances but one and three for the ABS and REL criteria, respectively.) However, for all the three robustness criteria, a comparison of the average solutions values shows that MIP_2 finds the optimal solution in all the instances, even if in some cases it is not able to certify the optimality. Regarding the computation times (see Columns 4 and 5 of Tables 7–9), MIP_1 is in general faster than MIP_2, taking from about 1 s (for $n = 20$) to about 184 s for the class 40-U, on average. In fact, Cplex on MIP_2 often reaches the time limit for the instances with $n = 40, 50$.

On the other hand, as shown in the last four Columns of Table 7, the linear relaxation of MIP_2 provides slightly better lower bounds than those provided by MIP_1, in a very short computation time: Fig. 2 reports the trends of the two lower bounds in terms of solution values and computation times.

7.2. Computational experiments on the dynamic programming algorithm

The dynamic programming algorithm presented in Section 4.3 was not able to solve the instances of Set_1 and Set_2 due to the excessive memory requirements, especially when instance numbers (namely, the processing times of the jobs and the duration of the maintenance) are large. In fact, the computational cost of the dynamic program grows exponentially with the number of scenarios k and it is directly proportional to the schedule length. Indeed, the importance of the dynamic programming is mainly theoretic and its performance – being much worse – is not comparable to that of MIP models.

However, in order to give an illustration of the behavior of the dynamic program, we have randomly generated a number of ad-hoc instances characterized by short processing times and duration of the maintenance activity, and larger maintenance slack. More precisely, the new instances are characterized by a number of jobs n varying from 5 to 35, with integer processing times uniformly distributed in interval $[1, 10]$, $c = MM$ robustness criterion, two scenarios, constant maintenance activity duration $P(s) = 10$ and allotted interval $d(s) - r(s) = 15$, respectively for both scenarios. For each value of n, 50 instances have been randomly generated.

Table 10 presents the results of these experiments, depending on the number n of jobs (indicated in the first row of the table). The second row reports the number of optimal solutions found by the algorithm over the 50 instances (and the corresponding success ratio in percent) for each value of n. Observe that, up to $n = 15$, the dynamic programming algorithm always correctly terminates, obviously finding optimal solutions. For larger number of jobs, its performance decreases accordingly: In these cases the algorithm cannot always complete its processing before the maximum amount of available memory is reached. On the other hand, in the solved instances, the computation times remain in any case below one second, for all the 350 considered instances.

8. Conclusions

In this paper, a problem arising in a manufacturing environment concerning the joint scheduling of multiple jobs and a maintenance activity on a single machine has been addressed. The maintenance activity must be processed within a given time window. While the

Table 7
Experimental results of MIP_1 and MIP_2 for $c = MM$ on instance set Set_2.

Instance	# opt		time		Obj	LB value		LB time	
Class	MIP_1	MIP_2	MIP_1	MIP_2	MM	MIP_1	MIP_2	MIP_1	MIP_2
20-U	20	20	2.29	1.12	2004.7	1941.2	1991.2	0.049	0.018
20-E	20	20	1.77	1.09	2122.6	2047.8	2097.8	0.065	0.010
20-M	20	20	2.22	5.75	2086.6	2011.8	2061.8	0.044	0.010
20-L	20	20	2.90	1.31	2099.5	2022.7	2072.7	0.050	0.009
30-U	20	20	4.11	47.75	3084.9	3026.8	3076.8	0.109	0.018
30-E	20	20	3.60	79.05	3093.5	3019.7	3069.7	0.111	0.009
30-M	20	15	3.89	1072.79*	3112.5	3037.1	3087.1	0.144	0.001
30-L	20	20	4.44	124.71	3081.0	3007.0	3057.0	0.153	0.014
40-U	20	3	4.99	1530.05*	4043.1	3957.0	4033.3	0.174	0.013
40-E	20	2	3.50	1620.03*	4071.9	3997.6	4060.7	0.186	0.015
40-M	20	0	5.96	1800.00*	4093.9	4023.3	4073.3	0.196	0.009
40-L	20	1	3.89	1710.03*	4066.0	3993.2	4043.2	0.279	0.025
50-U	20	6	35.40	1260.12*	5093.2	5033.4	5083.4	0.377	0.013
50-E	20	1	4.85	1710.04*	4961.6	4885.9	4935.8	0.251	0.016
50-M	20	1	7.58	1800.00*	5147.5	5075.8	5125.8	0.443	0.009
50-L	20	0	6.44	1620.05*	5158.8	5087.6	5137.6	0.375	0.012

(a) Lower bound values. (b) Computation times.

Fig. 2. Linear relaxation solution values and computation times for MIP_1 and MIP_2.

Table 8
Experimental results of MIP_1 and MIP_2 for $c = ABS$ on the instances of Set_2.

Instance	# opt		time		Obj
Class	MIP_1	MIP_2	MIP_1	MIP_2	ABS
20-U	20	20	0.96	0.85	13.6
20-E	20	20	1.00	0.87	24.8
20-M	20	20	1.26	4.90	24.9
20-L	20	20	1.12	0.76	26.9
30-U	20	20	12.44	41.05	8.1
30-E	20	20	2.07	106.65	23.8
30-M	20	17	2.21	1096.81*	25.4
30-L	20	20	2.44	134.82	24.0
40-U	20	3	5.74	1530.04*	9.9
40-E	20	2	5.36	1620.04*	24.3
40-M	20	3	5.00	1530.30*	20.7
40-L	20	3	4.10	1530.08*	22.8
50-U	19	4	97.64*	1440.13*	9.8
50-E	20	2	4.58	1620.06*	25.8
50-M	20	1	8.13	1710.02*	21.6
50-L	20	2	4.88	1620.07*	21.3

Table 9
Experimental results of MIP_1 and MIP_2 for $c = REL$ on the instances of Set_2.

Instance	# opt		time		Obj
Class	MIP_1	MIP_2	MIP_1	MIP_2	REL
20-U	20	20	9.85	1.16	1,007
20-E	20	20	0.71	0.88	1.012
20-M	20	20	1.36	4.57	1.012
20-L	20	20	1.06	0.81	1.013
30-U	19	20	93.36*	65.62	1.003
30-E	20	20	2.27	103.70	1.008
30-M	20	10	2.87	1371.01*	1.008
30-L	20	20	3.70	79.08	1.008
40-U	18	4	184.88*	1440.11*	1.002
40-E	20	2	3.42	1620.05*	1.006
40-M	20	2	10.07	1620.06*	1.005
40-L	20	1	4.45	1710.03*	1.006
50-U	20	2	8.28	1620.01*	1.002
50-E	20	1	6.90	1710.04*	1.005
50-M	20	2	9.48	1620.07*	1.214
50-L	20	3	6.12	1530.10*	1.004

processing times of the jobs are deterministic, the maintenance time window and duration are uncertain. In this context, the problem of finding job schedules which are robust to any possible change in the maintenance activity characteristics has been addressed. We prove several properties (see Table 2) of robust schedules when minimizing the makespan under four different standard robustness criteria.

Two MIP models and a dynamic programming algorithm are proposed and the results of the computational campaign show that the suggested solution approaches are efficient and effective for instances up to 50 jobs and 4 scenarios. We leave for future research the evaluation of the approaches on instances with a larger number of jobs and scenarios.

Table 10
Number of optimal solutions found by the Dynamic Program.

n	5	10	15	20	25	30	35
# optima	50 (100%)	50 (100%)	50 (100%)	41 (82%)	23 (46%)	23 (46%)	20 (40%)

Other, future research directions may include a theoretical study to characterize the properties of robust schedules in a setting different from the single machine one (for instance, an extension of these results in the context of parallel machines Agnetis, Benini, Detti, Hermans, & Pranzo, 2022; Chen, Huang, Huang, Huang, & Chou, 2021; Yoo & Lee, 2016, or parallel dedicated machines Agnetis, Kellerer, Nicosia, & Pacifici, 2012) and the design of new heuristic algorithms (for instance, math-heuristics exploiting the proposed MIP models). Studying $RSMP(c)$ under a different robustness paradigm, namely that of *recoverable robustness* (Liebchen, Lubbecke, Mohring, & Stiller, 2009; van den Akker, Hoogeveen, & Stoef, 2018) also looks like an interesting research line. Recoverable robustness permits a limited recovery action if a solution is unfeasible after the realization of a specific scenario. In our case, a recovery action can be viewed as a special *re-scheduling* problem (Alfieri, Nicosia, Pacifici, & Pferschy, 2018; Nicosia, Pacifici, Pferschy, Resch, & Righini, 2021). Finally, it would be interesting to investigate the equivalence of the restricted problems in which r or d are fixed (Problem $\mathcal{P}$ in Section 5.1) to the more "constrained" problem in which both r and d are fixed (Problem $\mathcal{Q}$) when the robustness criteria are $c = REL$ or $c = OWA$.

CRediT authorship contribution statement

Paolo Detti: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision. **Gaia Nicosia:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision. **Andrea Pacifici:** Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Writing – original draft, Writing – review & editing, Visualization, Supervision.

Data availability

Data will be made available on request.

References

Agnetis, A., Benini, M., Detti, P., Hermans, B., & Pranzo, M. (2022). Replication and sequencing of unreliable jobs on parallel machines. *Computers & Operations Research*, *139*, Article 105634.

Agnetis, A., Cosmi, M., Nicosia, G., & Pacifici, A. (2023). Two is better than one? Order aggregation in a meal delivery scheduling problem. *Computers & Industrial Engineering*, *183*, Article 109514.

Agnetis, A., Kellerer, H., Nicosia, G., & Pacifici, A. (2012). Parallel dedicated machines scheduling with chain precedence constraints. *European Journal of Operational Research*, *221*, 296–305.

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

Alfieri, A., Nicosia, G., Pacifici, A., & Pferschy, U. (2018). Constrained job rearrangements on a single machine. In P. Daniele, & L. Scrimali (Eds.), *AIRO Springer series*: *vol. 1*, *New trends in emerging complex real life problems*. Cham: Springer.

Benini, M., Detti, P., & de Lara, G. Zabalo Manrique (2022). Mathematical programming formulations and metaheuristics for biological sample transportation problems in healthcare. *Computers & Operations Research*, *146*, Article 105921.

Chen, J. S. (2008). Scheduling of nonresumable jobs and flexible maintenance activities on a single machine to minimize makespan. *European Journal of Operational Research*, *190*(1), 90–102.

Chen, Y. Y., Huang, P. Y., Huang, C. J., Huang, S. Q., & Chou, F. D. (2021). Makespan minimization for scheduling on two identical parallel machines with flexible maintenance and nonresumable jobs. *Journal of Industrial and Production Engineering*, *38*(4), 271–284.

Costa Souza, R. L., Ghasemi, A., Saif, A., & Gharaei, A. (2022). Robust job-shop scheduling under deterministic and stochastic unavailability constraints due to preventive and corrective maintenance. *Computers & Industrial Engineering*, *168*, Article 108130.

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

Detti, P., Nicosia, G., Pacifici, A., & de Lara, G. Zabalo Manrique (2016). Robust single machine scheduling with external-party jobs, proceedings of IFAC conference MIM 2016. *IFAC-PapersOnLine*, *49*(12), 1731–1736.

Detti, P., Nicosia, G., Pacifici, A., & de Lara, G. Zabalo Manrique (2019). Robust single machine scheduling with a flexible maintenance activity. *Computers & Operations Research*, *107*, 19–31.

Golpîra, H., & Tirkolaee, E. B. (2019). Stable maintenance tasks scheduling: A bi-objective robust optimization model. *Computers & Industrial Engineering*, *137*, Article 106007.

He, Y., Ji, M., & Cheng, T. C. E. (2005). Single machine scheduling with a restricted rate-modifying activity. *Naval Research Logistics*, *52*, 361–369.

Ji, M., Yong, H., & Cheng, T. C. E. (2007). Single-machine scheduling with periodic maintenance to minimize makespan. *Computers & Operations Research*, *34*(6), 1764–1770.

Kacem, I., & Kellerer, H. (2016). Semi-online scheduling on a single machine with unexpected breakdown. *Theoretical Computer Science*, *646*, 40–48.

Kasperski, A., & Zielinski, P. (2014). Minmax (regret) scheduling problems. In Sotskov Y., & Werner F. (Eds.), *Sequencing and scheduling with inaccurate data* (pp. 159–210). New York: Nova Science Publishers.

Kellerer, H., Mansini, R., Pferschy, U., & Speranza, M. G. (2003). An efficient fully polynomial approximation scheme for the subset-sum problem. *Journal of Computer and System Sciences*, *66*(2), 349–370.

Kellerer, H., Pferschy, U., & Pisinger, D. (2003). *Knapsack problems*. Springer Verlag.

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

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.

Lee, C. Y. (1996). Machine scheduling with an availability constraint. *Journal of Global Optimization*, *9*, 395–416.

Liebchen, C., Lubbecke, M., Mohring, R., & Stiller, S. (2009). The concept of recoverable robustness, linear programming recovery, and railway applications. In *Lecture notes in computer science*: *vol. 5868*, (pp. 1–27).

Luo, W., Cheng, T. C. E., & Ji, M. (2015). Single-machine scheduling with a variable maintenance activity. *Computers & Industrial Engineering*, *79*, 168–174.

Ma, Y., Chu, C., & Zuo, C. (2010). A survey of scheduling with deterministic machine availability constraints. *Computers & Industrial Engineering*, *58*(2), 199–211.

Mulvey, J. M., Vanderbei, R. J., & Zenios, S. A. (1995). Robust optimization of large-scale systems. *Informs Journal of Computing*, *43*(2), 264–281.

Nicosia, G., Pacifici, A., Pferschy, U., Resch, J., & Righini, G. (2021). Optimally rescheduling jobs with a last-in-first-out buffer. *Journal of Scheduling*, *24*, 663–680.

Pereira, J. (2016). The robust (minmax regret) single machine scheduling with interval processing times and total weighted completion time objective. *Computers & Operations Research*, *66*, 141–152.

Shabtay, D., & Gilenson, M. (2023). A state-of-the-art survey on multi-scenario scheduling. *European Journal of Operational Research*, *310*(1), 3–23.

van den Akker, M., Hoogeveen, H., & Stoef, J. (2018). Combining two-stage stochastic programming and recoverable robustness to minimize the number of late jobs in the case of uncertain processing times. *Journal of Scheduling*, *21*(6), 607–617.

Wang, S., Cui, W., Chu, F., Yu, J., & Gupta, J. N. D. (2020). Robust (min–max regret) single machine scheduling with interval processing times and total tardiness criterion. *Computers & Industrial Engineering*, *149*, Article 106838.

Xu, D., Yin, Y., & Li, H. (2009). A note on scheduling of nonresumable jobs and flexible maintenance activities on a single machine to minimize makespan. *European Journal of Operational Research*, *197*(2), 825–827.

Yager, R. R. (1988). On ordered weighted averaging aggregation operators in multi-criteria decision making. *IEEE Transactions on Systems, Man and Cybernetics*, *18*, 183–190.

Yang, D. L., Hung, C. L., Hsu, C. J., & Chern, M. S. (2002). Minimizing the makespan in a single machine scheduling problem with a flexible maintenance. *Journal of the Chinese Institute of Industrial Engineers*, *19*, 63–66.

Yang, S., Maa, Y., Xu, D., & Yang, J. (2011). Minimizing total completion time on a single machine with a flexible maintenance activity. *Computers & Operations Research*, *38*, 755–770.

Ying, K.-C., Lu, C.-C., & Chen, J.-C. (2016). Exact algorithms for single-machine scheduling problems with a variable maintenance. *Computers & Industrial Engineering*, *98*, 427–433.

Yoo, J., & Lee, I. S. (2016). Parallel machine scheduling with maintenance activities. *Computers & Industrial Engineering*, *101*, 361–371.