

Lawler's minmax cost algorithm: optimality conditions and uncertainty

Nadia Brauner · Gerd Finke · Yakov Shafransky ·
Dzmitry Sledneu

Received: 27 November 2013 / Accepted: 19 December 2014 / Published online: 4 January 2015
© Springer Science+Business Media New York 2014

Abstract The well-known $O(n^2)$ minmax cost algorithm of Lawler (MANAGE SCI 19(5):544–546, 1973) was developed to minimize the maximum cost of jobs processed by a single machine under precedence constraints. We propose two results related to Lawler's algorithm. Lawler's algorithm delivers one specific optimal schedule while there can exist other optimal schedules. We present necessary and sufficient conditions for the optimality of a schedule for the problem with strictly increasing cost functions. The second result concerns the same scheduling problem under uncertainty. The cost function for each job is of a special decomposable form and depends on the job completion time and on an additional numerical parameter, for which only an interval of possible values is known. For this problem we derive an $O(n^2)$ algorithm for constructing a schedule that minimizes the maximum regret criterion. To obtain this schedule, we use Lawler's algorithm as a part of our technique.

Keywords Lawler's minmax cost algorithm · Uncertainty · Maximum regret

N. Brauner · G. Finke (✉)
Université Grenoble Alpes, G-SCOP, F-38000 Grenoble, France
e-mail: gerd.finke@g-scop.inpg.fr

N. Brauner
e-mail: nadia.brauner@imag.fr

N. Brauner · G. Finke
CNRS, G-SCOP, F-38000 Grenoble, France

Y. Shafransky
United Institute of Informatics Problems, NAS of Belarus,
Minsk, Belarus
e-mail: shafr-04@yandex.ru

D. Sledneu
Lund University, Lund, Sweden
e-mail: dzmitry.sledneu@gmail.com

1 Introduction

Lawler (1973) considered the following single-machine scheduling problem. Each of n simultaneously available jobs is to be processed without preemption on a single machine, which can process at most one job at a time. Job j , $j = 1, \dots, n$, has a processing time $p_j \geq 0$. To each of the jobs j is assigned a non-decreasing cost function $\Phi_j(t)$ that specifies the cost incurred if job j is completed at time t . There are precedence constraints represented by an acyclic directed graph $G = (V, E)$ with n vertices, where each vertex $v \in V$ corresponds to one of the jobs. Note that graph G contains all transitive arcs. If job j_1 precedes job j_2 ($j_1 \rightarrow j_2$), i.e., $(j_1, j_2) \in E$, then the processing of job j_2 cannot start before the completion of processing job j_1 . The objective is to find a feasible schedule s that minimizes the maximum cost $\Phi_{\max}(s) = \max_{j=1, \dots, n} \Phi_j(C_j(s))$, where $C_j(s)$ is the completion time of job j under schedule s .

A feasible schedule may be presented by the permutation $\pi = (\pi(1), \dots, \pi(n))$ of jobs. Here $\pi(k)$ is the job in position k of permutation π . The permutation feasibility means that job j_1 is allocated on the left of job j_2 in the permutation if $j_1 \rightarrow j_2$. Denote by $C_j(\pi) = p_{\pi(1)} + \dots + p_{\pi(k)}$ the completion time of job $j = \pi(k)$ allocated in position k of π . A feasible permutation that minimizes $\Phi_{\max}(\pi)$ is called *optimal*. The formulated problem $1|prec|\Phi_{\max}$ is solved by Lawler's minmax cost algorithm that may be described as follows:

Algorithm 1 Lawler (1973)

1. Set $k = n$ and $T = \sum_{j=1}^n p_j$.
2. Let Ω be the set of terminal vertices of graph $G = (V, E)$, i.e., vertices without successors.
3. Find vertex $u \in \Omega$ such that $\Phi_u(T) = \min\{\Phi_v(T) | v \in \Omega\}$. Break ties arbitrarily.

4. Set $\pi(k) = u$, $k = k - 1$, $T = T - p_u$ and modify graph G by setting $V = V \setminus \{u\}$ and removing all arcs adjacent to u .
5. If $k > 0$, then go to Step 2. If $k = 0$, then the optimal permutation is constructed.

The time complexity of the algorithm is $O(n^2)$. It is supposed that graph G is represented by its adjacency matrix, and we assume that each value of $\Phi_j(t)$ can be calculated in $O(1)$ time.

Lawler's algorithm delivers one specific optimal solution while there can exist other optimal solutions. Lin and Wang (2007) derived necessary and sufficient conditions for the optimality of a schedule for a number of scheduling problems without precedence constraints, in particular, for problem $1||L_{\max}$, where L_{\max} is the maximum lateness.

We present necessary and sufficient conditions for the optimality of a schedule for the general problem $1|prec|\Phi_{\max}$ with strictly increasing cost functions. For non-decreasing cost functions these conditions are sufficient.

The traditional scheduling models, in which values of all numerical parameters are given in advance, fail in many practical applications. For a number of the real-world situations, values of some parameters (processing and transportation times, release and due dates, etc.) are not precisely known in advance. Everything that is known is the set of possible values for each of the parameters. These parameters are not controllable, each of them takes any value from the given set, and the choice of the particular value does not depend on the will of the decision maker. If such uncertain parameters are included in the model, then one obtains a model and a corresponding optimization problem under uncertainty.

For a problem under uncertainty, each possible choice of the parameter values generates a scenario. Once a scenario is given, we have to solve a deterministic problem. The main peculiar feature of any problem under uncertainty is that we do not know in advance, which particular scenario will be realized.

One of the most popular approaches of dealing with problems under uncertainty is the worst-case or robust approach, Kouvelis and Yu (1997). Under this approach, the decision maker expects that the real scenario will be the worst possible from the point of view of the problem to be solved, and constructs a solution that is the best one under the expected worst scenario. Intuitively, a robust solution is a solution that remains suitable whatever scenario finally occurs. There are different ways to define the worst scenario. The two main concepts are the worst objective function value and the maximum deviation of the objective function value from the optimum.

To simplify the formal description of the robust solutions, consider as an example a scheduling problem where the aim

is to minimize an objective function. Then in the first concept, the decision maker's aim is to minimize the objective function under maximally unsuitable values of the uncertain parameters. In the second concept, the decision maker tries to find a schedule that minimizes the maximum possible difference between the objective function values for the schedule obtained and the optimal schedule for a real scenario. Respectively, these approaches are called minmax and minmax regret approaches, see Kouvelis and Yu (1997).

To be more formal, denote the set of all possible scenarios by X and the set of all feasible schedules by S . Denote the objective function by $F(x, s)$. Here we point out that the objective function depends on schedule $s \in S$ and on scenario $x \in X$. Then the minmax approach is to find a schedule s that minimizes the function

$$\max\{F(x, s) | x \in X\},$$

whereas the minmax regret approach is to find a schedule s that minimizes the function

$$\max\{F(x, s) - F(x, s^*(x)) | x \in X\},$$

where $s^*(x)$ is an optimal schedule for scenario x .

It appears that Wald (1950) was the first who proposed the minmax approach to deal with uncertainty and Savage (1954) is the author of the minmax regret approach.

In Lawler (1973) the cost functions of jobs are arbitrary non-decreasing functions. In our problem, we assume decomposable non-decreasing functions that will be defined in Sect. 3. A processing time p_j and a cost function parameter λ_j are associated with each job j . We suppose that the additional parameters λ_j are uncertain and the set of possible values of λ_j for each job j is an interval. In our case, the set of all possible scenarios is the Cartesian product of these intervals. We examine the two mentioned approaches: the minmax approach and the minmax regret approach. In both cases we actively use Lawler's algorithm.

The only known result for problems of this type is presented in Kasperski (2005), where the problem $1|prec|L_{\max}$ was considered under uncertainty of processing times and due dates of the jobs. Also in Kasperski (2005) the set of possible values for each of the uncertain parameters is an interval. An $O(n^4)$ algorithm has been developed for constructing a schedule that is optimal with respect to the minmax regret criterion. In our model, we consider a more general objective function that includes L_{\max} as a special case. At the same time, we are dealing with a single uncertain parameter for each job. One of our results is an $O(n^2)$ algorithm for constructing a schedule that is optimal with respect to the minmax regret criterion.

We denote by $A_G(v)$ and $B_G(v)$ the set of all successors of vertex v in graph G and the set of all predecessors of v , respectively.

This paper is organized as follows: In Sect. 2 we consider a deterministic version of the main problem and formulate necessary and sufficient conditions for a feasible permutation to be optimal. In addition, we present some related results that are used in the subsequent sections. In Sect. 3 we deal with the main problem under uncertainty. We present $O(n^2)$ algorithms for constructing schedules that are optimal with respect to the minmax criterion (Sect. 3.1) and to the minmax regret criterion (Sect. 3.2). In the last section, we summarize our results and outline possible directions for further research.

2 Conditions of optimality

For our problem $1|prec|\Phi_{\max}$, we require first that all functions $\Phi_j(t)$ are strictly increasing. Compared to Lin and Wang (2007), where necessary and sufficient conditions of a permutation optimality were derived for problem $1||L_{\max}$, difficulties arise in connection with the precedence constraints and with the presence of zero-length jobs. In Lin and Wang (2007), all processing times are implicitly assumed to be strictly positive.

Let a feasible permutation π be given. We describe a job exchange, as Lawler did in his original paper, but in a slightly modified form. Suppose the job permutation π can be represented schematically as follows:

$$\pi : \quad \boxed{A} \quad \boxed{u} \quad \boxed{B} \quad \boxed{v} \quad \boxed{C}$$

with segments A , B , and C and jobs v and u with $p_u > 0$ and $u \not\rightarrow v$, i.e., $u \notin B_G(v)$. We consider only jobs u that are close to v so that the set $W = B \cap A_G(u)$ of all successor jobs of u in B contains only zero-length jobs, i.e., if $j \in W$ then $p_j = 0$. Set $U = (u, W)$ and move subpermutation U to the right just following v . Then we obtain the feasible permutation π' :

$$\pi' : \quad \boxed{A} \quad \boxed{B'} \quad \boxed{v} \quad \boxed{u} \quad \boxed{W} \quad \boxed{C}$$

where $B' = B \setminus W$. Here it is understood that the inner order of jobs in B' and W is saved from the order in π . The following definition describes the transitions from π to π' that yield some kind of strict improvement, at least locally.

Definition 1 Consider a feasible permutation π and a job v in π with completion time t . We say that job v has a *local improvement* in π if there exists job u , as described above, which satisfies the conditions $\Phi_j(t) < \Phi_v(t)$ for all $j \in \{u\} \cup W$.

If job v has a local improvement in π , then for the permutation π' above, we get the following properties:

1. The positions of jobs $j \in A \cup C$ are unchanged. Hence these jobs keep the same completion times and costs.

2. Let $R = \{u\} \cup \{v\} \cup B$. Then the maximum cost of the jobs in R is strictly reduced, i.e., $\max_{j \in R} \Phi_j(C_j(\pi')) < \max_{j \in R} \Phi_j(C_j(\pi))$. The validity of the property follows from the conditions $p_u > 0$ and $\Phi_j(t) < \Phi_v(t)$ for all $j \in \{u\} \cup W$.
3. We have $\Phi_{\max}(\pi') \leq \Phi_{\max}(\pi)$.

Definition 2 Job v is called *critical* in permutation π , if $\Phi_{\max}(\pi) = \Phi_v(C_v(\pi))$.

Theorem 1 If all functions $\Phi_j(t)$ are strictly increasing, then a feasible permutation π is optimal for problem $1|prec|\Phi_{\max}$ if and only if there exists a critical job in π without any local improvement.

Proof Necessity. Let π be an optimal permutation. We prove the necessity by induction on the number of critical jobs.

Consider a situation with a single critical job v . If job v has a local improvement in π , then in view of the above properties of the local improvement, there exists a feasible permutation π' such that $\Phi_v(C_v(\pi')) < \Phi_v(C_v(\pi))$, and the values of the cost functions of all other jobs are less than $\Phi_v(C_v(\pi))$. So, $\Phi_{\max}(\pi') < \Phi_{\max}(\pi)$ that contradicts the optimality of π .

Suppose that the theorem conditions are necessary for the optimality of any permutation with less than $l \geq 2$ critical jobs.

Consider the case with l critical jobs. If each of the critical jobs has a local improvement, then consider the critical job $v = \pi(k)$ in the left-most position. Acting similarly to the case with a single critical job, we can move job u (see the definition of local improvement) along with its successors in positions j , $j < k$, to the positions to the right of v . As a result, we obtain a feasible permutation π' , in which job v is not a critical job and the general number of critical jobs in π' equals $l - 1$. Besides, $\Phi_{\max}(\pi') \leq \Phi_{\max}(\pi)$, i.e., π' is an optimal permutation. All these $l - 1$ critical jobs are in the same positions as in permutation π . Since for π , each of these $l - 1$ critical jobs has a local improvement, each of them has a local improvement in permutation π' as well. It means that for the optimal permutation π' with $l - 1$ critical jobs, the theorem conditions are violated. That contradicts our induction hypothesis.

Sufficiency Suppose that in a feasible permutation π , there exists a critical job $v = \pi(k)$ without local improvements. We have $\Phi_{\max}(\pi) = \Phi_v(C_v(\pi)) \geq \Phi_j(C_j(\pi))$ for all $j \in \{1, \dots, n\}$. Let π' be an optimal permutation. If $C_v(\pi') \geq C_v(\pi)$, then $\Phi_{\max}(\pi') \geq \Phi_{\max}(\pi)$ and π is an optimal permutation.

Denote $L(v) = \{\pi(l) \notin B_G(v) \mid l < k, p_{\pi(l)} > 0\}$, remembering that job v is in position k in π .

Suppose that for an optimal permutation π' , we have $C_v(\pi') < C_v(\pi)$. Then there exists at least one job $\pi(l) \in$

$L(v)$ on the right of v in π' , otherwise we would have $C_v(\pi') \geq C_v(\pi)$.

Let $\pi(l)$ be the right-most job from $L(v)$ in π' , then $C_{\pi(l)}(\pi') \geq C_v(\pi)$. Since v does not have a local improvement in π , we have $\Phi_{\pi(l)}(C_v(\pi)) \geq \Phi_v(C_v(\pi))$. Therefore, $\Phi_{\pi(l)}(C_{\pi(l)}(\pi')) \geq \Phi_{\pi(l)}(C_v(\pi)) \geq \Phi_v(C_v(\pi))$ and $\Phi_{\max}(\pi') \geq \Phi_{\max}(\pi)$, i.e., π is again an optimal permutation. \square

Corollary 1 *If all $\Phi_j(t)$ are non-decreasing functions, then a feasible permutation π is optimal for problem $1|prec|\Phi_{\max}$ if there exists a critical job in π without any local improvement.*

Definition 3 Given a feasible permutation π and a job v in π with completion time t . Let job u be on the left of v in π , and U is the set of jobs between u and v in π . We say that job v has a *weak improvement* in π if $\Phi_u(t) < \Phi_v(t)$ and none of the jobs in $U \cup \{v\}$ is a successor of u .

Lemma 1 *If all functions $\Phi_j(t)$ are non-decreasing and job v has a weak improvement in a feasible permutation π , then there exists a feasible permutation π' such that $C_v(\pi') \leq C_v(\pi)$, $\Phi_u(C_u(\pi')) < \Phi_v(C_v(\pi))$, and $\Phi_{\max}(\pi') \leq \Phi_{\max}(\pi)$.*

Proof To construct the mentioned permutation π' , it is enough to move job u from its position to position of job v in π , simultaneously shifting all jobs of the set $U \cup \{v\}$ one position to the left. \square

Evidently, if a job has no weak improvement in a feasible permutation π , then it also has no local improvement in π .

Corollary 2 *If all functions $\Phi_j(t)$ are non-decreasing, then a feasible permutation π is optimal for problem $1|prec|\Phi_{\max}$ if there exists a critical job in π without any weak improvement.*

Corollary 3 *If for problem $1|prec|\Phi_{\max}$, permutation π is constructed by Algorithm 1, then none of the jobs has a weak improvement in π .*

Proof Suppose that job $\pi(k)$ has a weak improvement in π . Consider the k th run of Step 3 in Algorithm 1. The definition of weak improvement means that in this run, there is at least one job j in the current set Ω such that $\Phi_j(T) < \Phi_{\pi(k)}(T)$. In accordance with the description of Step 3, job j has to be chosen to be placed in position k in π instead of job $\pi(k)$. \square

3 Decomposable cost functions with uncertainty

Now we formulate our main single-machine problem under uncertainty. We give in detail the differences between our

problem and its deterministic version considered so far with n non-preemptive jobs and precedence constraints presented by graph G . We restrict the cost functions of jobs to be decomposable into a sum of two components. The first component is a monotone function that depends on the job completion time. The second component is an uncertain parameter.

Job j has a processing time $p_j \geq 0$ and an additional real-valued parameter λ_j . Each job j has an associated cost function $\Phi_j(t, \lambda_j) = \varphi(t) + \lambda_j$ that presents a cost to be incurred if the job processing is completed at time t . Here $\varphi(t)$ is a non-decreasing function. The aim is to find a job permutation π that is feasible with respect to the precedence constraints and that minimizes the maximum cost

$$\Phi_{\max}(\pi) = \max\{\Phi_j(C_j(\pi), \lambda_j) | j = 1, \dots, n\}.$$

The best known examples of functions $\Phi_j(t, \lambda_j) = \varphi(t) + \lambda_j$ are

- lateness of job j : $L_j(t, d_j) = t - d_j$, where $d_j \geq 0$ is a due date of job j ; here $\lambda_j = -d_j$, and
- delivery time of job j : $D_j(t, q_j) = t + q_j$, where $q_j \geq 0$ is the transportation time of job j , i.e., after completing the processing, job j is to be delivered to its end user, and the delivering takes q_j time units; here $\lambda_j = q_j$.

We consider the problem above under uncertainty of the additional parameter λ_j : for each job j there are given a lower bound λ_j^- and an upper bound λ_j^+ of possible values of λ_j . The actual value of λ_j is unknown; λ_j may take any value from the interval $[\lambda_j^-, \lambda_j^+]$ of real numbers.

We denote the uncertain version of the problem by

$$1|prec; \lambda_j \in [\lambda_j^-, \lambda_j^+]|\Phi_{\max}.$$

Further, we suppose that the calculation of each value of function $\varphi(t)$ takes a constant time.

Suppose we are given particular values λ_j^-, λ_j^+ , and p_j for each $j \in \{1, \dots, n\}$, a particular function $\varphi(t)$ and a particular graph G presenting the precedence constraints. If we choose the values of the uncertain parameters in any feasible way, i.e., $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ for each $j \in \{1, \dots, n\}$, we get *scenario* I for the problem $1|prec; \lambda_j \in [\lambda_j^-, \lambda_j^+]|\Phi_{\max}$ under uncertainty. We denote by \mathcal{I} the set of all scenarios. Clearly, set \mathcal{I} is infinite if $\lambda_j^- \neq \lambda_j^+$ for at least one $j \in \{1, \dots, n\}$.

Given scenario $I \in \mathcal{I}$, we have a deterministic version of the problem in question, and we can find an optimal permutation using Lawler’s algorithm.

For a given scenario $I \in \mathcal{I}$ and a permutation π , we write $\Phi_{\max}(I, \pi)$ to denote the corresponding value of the objective function, and we write $\Phi_{\max}^*(I)$ to denote the minimum value of the objective function for scenario I . Let $\pi^*(I)$ be an optimal permutation for scenario I .

In the following sections we consider the two robust approaches, described earlier, to solve the problem 1|prec; $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ | Φ_{\max} under uncertainty: minmax and minmax regret criteria.

3.1 Minmax criterion

Since each function $\Phi_j(t, \lambda_j) = \varphi(t) + \lambda_j$ is strictly increasing with the parameter λ_j , for any permutation π , the value of the objective function $\Phi_{\max}(I, \pi)$ gets its maximum if each λ_j takes its maximum value.

Thus, to obtain a permutation that is optimal with respect to the minmax criterion, it is enough to solve the problem of minimizing function $\Phi_{\max}(I^+, \pi)$, where I^+ is the scenario with $\lambda_j = \lambda_j^+$ for all $j = 1, \dots, n$. Hence the permutation may be constructed by Algorithm 1.

The following statement characterizes the quality of solutions obtained according to the minmax criterion.

Theorem 2 *Let π^+ be an optimal permutation for scenario I^+ . Then for any scenario I^0 the bound*

$$\Phi_{\max}(I^0, \pi^+) - \Phi_{\max}(I^0, \pi^*(I^0)) \leq \max\{\lambda_j^+ - \lambda_j^- \mid j = 1, \dots, n\}$$

holds, and this bound is tight.

Proof Let scenario I^0 be defined by the vector $\lambda^0 = (\lambda_1^0, \dots, \lambda_n^0)$. For simplicity, denote permutation $\pi^*(I^0)$ by π^0 . Without loss of generality, suppose that the jobs are numbered in accordance with permutation π^0 . Let job l be a critical job in permutation π^0 for scenario I^+ , i.e., $\Phi_{\max}(I^+, \pi^0) = \varphi(\sum_{j=1}^l p_j) + \lambda_l^+$. Then $\varphi(\sum_{j=1}^l p_j) + \lambda_l^+ \geq \Phi_{\max}(I^+, \pi^+)$.

For I^0 we have $\Phi_{\max}(I^0, \pi^0) \geq \varphi(\sum_{j=1}^l p_j) + \lambda_l^0$. Therefore, $\Phi_{\max}(I^+, \pi^+) - \Phi_{\max}(I^0, \pi^0) \leq \lambda_l^+ - \lambda_l^0$.

Evidently, $\Phi_{\max}(I^+, \pi^+) \geq \Phi_{\max}(I^0, \pi^+)$. Thus, $\delta = \Phi_{\max}(I^0, \pi^+) - \Phi_{\max}(I^0, \pi^0) \leq \Phi_{\max}(I^+, \pi^+) - \Phi_{\max}(I^0, \pi^0) \leq \lambda_l^+ - \lambda_l^0 \leq \max_{1 \leq j \leq n} \{\lambda_j^+ - \lambda_j^-\}$.

Now we construct a simple example that shows the tightness of the bound. Given number $A > 1$, let $\varphi(t) = t$; $n = 2$; $p_1 = A$, $p_2 = 1$, $\lambda_1^- = 1 + \varepsilon$, $\lambda_1^+ = A + 1 + \varepsilon$; $\lambda_2^- = 2$, $\lambda_2^+ = A + 1$, where $\varepsilon > 0$ is an arbitrary small number. It is easy to see that $\pi^+ = (1, 2)$. Suppose that $\lambda_1^0 = 1 + \varepsilon$, $\lambda_2^0 = A + 1$. Then $\pi^0 = (2, 1)$, $\Phi_{\max}(I^0, \pi^+) = 2A + 2$, $\Phi_{\max}(I^0, \pi^0) = A + 2 + \varepsilon$. Thus, $\delta = A - \varepsilon$, and δ tends to A , while ε tends to 0. \square

It is interesting to note that in case of the ‘‘optimistic’’ scenario I^- with $\lambda^- = (\lambda_1^-, \dots, \lambda_n^-)$, we get the same bound.

Note 1 Let π^- be an optimal permutation for scenario I^- . Then for any scenario I^0 , the bound

$$\Phi_{\max}(I^0, \pi^-) - \Phi_{\max}(I^0, \pi^*(I^0)) \leq \max\{\lambda_j^+ - \lambda_j^- \mid j = 1, \dots, n\}$$

holds, and this bound is tight.

The proof of this statement is similar to that of Theorem 2.

3.2 Minmax regret criterion

Let us define the minmax regret criterion for problem 1|prec; $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ | Φ_{\max} .

Definition 4 For problem 1|prec; $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ | Φ_{\max} , a permutation is called *optimal with respect to the maximum regret* if it minimizes the function

$$\Delta(\mathcal{I}, \pi) = \max_{I \in \mathcal{I}} \{\Phi_{\max}(I, \pi) - \Phi_{\max}(I, \pi^*(I))\}. \tag{1}$$

The approach based on minimizing function (1) is known also as the robust deviation decision approach, see Kouvelis and Yu (1997).

Denote by $\lambda' = (\lambda'_1, \dots, \lambda'_n)$ the collection of the parameter values λ that defines scenario $I \in \mathcal{I}$ and rewrite the function $\Phi_{\max}(I, \pi) - \Phi_{\max}(I, \pi^*(I))$ from (1) in the following way:

$$\begin{aligned} & \Phi_{\max}(I, \pi) - \Phi_{\max}(I, \pi^*(I)) \\ &= \max_{j=1, \dots, n} \{\varphi(C_j(\pi)) + \lambda'_j\} - \Phi_{\max}(I, \pi^*(I)) \\ &= \max_{j=1, \dots, n} \{\varphi(C_j(\pi)) - (\Phi_{\max}(I, \pi^*(I)) - \lambda'_j)\}. \end{aligned}$$

Thus,

$$\begin{aligned} \Delta(\mathcal{I}, \pi) &= \max_{I \in \mathcal{I}} \left\{ \max_{j=1, \dots, n} \{\varphi(C_j(\pi)) - (\Phi_{\max}(I, \pi^*(I)) - \lambda'_j)\} \right\} \\ &= \max_{j=1, \dots, n} \left\{ \max_{I \in \mathcal{I}} \{\varphi(C_j(\pi)) - (\Phi_{\max}(I, \pi^*(I)) - \lambda'_j)\} \right\} \\ &= \max_{j=1, \dots, n} \left\{ \varphi(C_j(\pi)) - \min_{I \in \mathcal{I}} \{\Phi_{\max}(I, \pi^*(I)) - \lambda'_j\} \right\}. \end{aligned}$$

Definition 5 For each $h \in \{1, \dots, n\}$ define scenario $I_h \in \mathcal{I}$ by setting $\lambda_h = \lambda_h^+$ and $\lambda_j = \lambda_j^-$ for all $j \neq h$.

Denote by π_h^* an optimal permutation for scenario I_h . The following lemma shows the importance of scenarios I_h .

Lemma 2 *Given scenario $I^0 \in \mathcal{I}$, let π^0 be an optimal permutation for I^0 . Then for any $h \in \{1, \dots, n\}$ the following inequality is valid:*

$$\Phi_{\max}(I_h, \pi_h^*) - \lambda_h^+ \leq \Phi_{\max}(I^0, \pi^0) - \lambda_h^0, \tag{2}$$

where the collection $\lambda^0 = (\lambda_1^0, \dots, \lambda_n^0)$ defines scenario I^0 .

Proof Construct scenario I' replacing λ_j^0 with λ_j^- in scenario I^0 for all $j \neq h$. Evidently, $\Phi_{\max}(I', \pi^0) \leq \Phi_{\max}(I^0, \pi^0)$. Since $\Phi_{\max}(I_h, \pi_h^*) \leq \Phi_{\max}(I_h, \pi^0)$, to prove the validity of (2), it is enough to show that the inequality

$$\Phi_{\max}(I_h, \pi^0) - \lambda_h^+ \leq \Phi_{\max}(I', \pi^0) - \lambda_h^0$$

holds.

If h is not a critical job in π^0 for scenario I_h , then $\Phi_{\max}(I_h, \pi^0) = \Phi_{\max}(I', \pi^0)$ and $\Phi_{\max}(I_h, \pi^0) - \lambda_h^+ \leq \Phi_{\max}(I', \pi^0) - \lambda_h^0$.

If h is a critical job, then $\Phi_{\max}(I_h, \pi^0) - \lambda_h^+ = (\varphi(C_h(\pi^0)) + \lambda_h^+) - \lambda_h^+ = (\varphi(C_h(\pi^0)) + \lambda_h^0) - \lambda_h^0 \leq \Phi_{\max}(I', \pi^0) - \lambda_h^0$. \square

Corollary 4 *The maximum regret is of the form*

$$\Delta(\mathcal{I}, \pi) = \max_{j=1, \dots, n} \{\varphi(C_j(\pi)) + d_j\},$$

where $d_j = -(\Phi_{\max}(I_j, \pi_j^*) - \lambda_j^+)$.

Proof Lemma 2 implies that

$$\begin{aligned} \Delta(\mathcal{I}, \pi) &= \max_{j=1, \dots, n} \left\{ \varphi(C_j(\pi)) \right. \\ &\quad \left. - \min_{I \in \mathcal{I}} \{ \Phi_{\max}(I, \pi^*(I)) - \lambda_j^+ \} \right\} \\ &= \max_{j=1, \dots, n} \{ \varphi(C_j(\pi)) - (\Phi_{\max}(I_j, \pi_j^*) - \lambda_j^+) \}. \end{aligned}$$

\square

Evidently, $\Delta(\mathcal{I}, \pi)$ belongs to the class of maximum cost functions (with decomposable costs). Therefore, Lawler’s algorithm may be used to minimize $\Delta(\mathcal{I}, \pi)$. The following two-step algorithm constructs a permutation that gives the minimum of the maximum regret $\Delta(\mathcal{I}, \pi)$.

Algorithm 2 1. For each scenario $I_j \in \mathcal{I}$, $j = 1, \dots, n$, calculate the value $\Phi_{\max}(I_j, \pi_j^*)$ and set $d_j = -\Phi_{\max}(I_j, \pi_j^*) + \lambda_j^+$.
2. Apply Algorithm 1 to problem 1|prec| Δ_{\max} with cost function $\Delta_j(t) = \varphi(t) + d_j$ to obtain an optimal permutation π^* , and the minimum value $\Delta^* = \Delta_{\max}(\pi^*)$.

The validity of Algorithm 2 is immediate. Using Corollary 4 and Definition 4, π^* is an optimal permutation, and $\Delta^* = \Delta(\mathcal{I}, \pi^*)$ is the minimum value for the minmax regret criterion.

Below (see Algorithm 3) we show that the running time of Step 1 of Algorithm 2 is $O(n^2)$ if we use a special procedure to implement this step. Thus, the running time of Algorithm 2 is $O(n^2)$.

Note that the idea of using scenarios I_h in the minmax regret approach is due to Averbakh (2000). In Averbakh

(2000) the single-machine sequencing problem with the maximum weighted tardiness criterion is considered. However, in this problem, an uncertain weight is given for each job and all the other parameters (i.e., the processing times and the due dates) are precisely known. In such a situation, the problem of minimizing the maximum regret criterion can be solved in time $O(n^3)$. As it is mentioned in Averbakh (2000), the results hold if instead of weighted tardiness $w_j T_j$ we consider any function $w_j f_j$ with uncertain weight w_j , where f_j is non-decreasing in C_j .

Now we describe a procedure that implements Step 1 of Algorithm 2. Given problem 1|prec; $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ | Φ_{\max} , we describe an algorithm for calculating the values $\Phi_{\max}(I_h, \pi_h^*)$ for all n scenarios $I_h \in \mathcal{I}$, $h = 1, \dots, n$, in $O(n^2)$ time.

Algorithm 3 1. Construct scenario $I^- \in \mathcal{I}$ by setting $\lambda_j = \lambda_j^-$ for all $j = 1, \dots, n$. Apply Algorithm 1 to construct an optimal permutation π^- for scenario I^- . Compute the value $\Phi_{\max}(I^-, \pi^-)$.
Renummer the jobs according to permutation π^- (i.e., $\pi^- = (1, \dots, n)$ after the renumbering).
Compute $C_j(\pi^-)$ for all $j = 1, \dots, n$.
Set $h = n$.
2. Construct scenario I_h replacing λ_h^- with λ_h^+ in I^- . Set $v = h - 1$, $\pi_h^* = \pi^-$ and $J_h = (h)$.
3. If $C_h(\pi^-) + \lambda_h^+ \leq \Phi_{\max}(I^-, \pi^-)$, then set $\Phi_{\max}(I_h, \pi_h^*) = \Phi_{\max}(I^-, \pi^-)$ and go to Step 7.
4. If $v = 0$ or $\lambda_h^+ \leq \lambda_v^-$, then go to Step 6, otherwise, go to Step 5.
5. If job v is a predecessor of job h , then set $J_h = (v, J_h)$. Otherwise, modify permutation π_h^* exchanging positions of job v and the subpermutation J_h . Set $v = v - 1$ and go to Step 4.
6. Calculate $\Phi_{\max}(I_h, \pi_h^*)$ for the current permutation π_h^* .
7. Set $h = h - 1$. If $h \geq 1$, then go to Step 2, otherwise, the process of calculating the values $\Phi_{\max}(I_h, \pi_h^*)$ has been completed.

Theorem 3 Algorithm 3 finds permutations π_h^* correctly, $h = 1, \dots, n$, and its running time is $O(n^2)$.

Proof According to Corollary 3, none of the jobs in permutation π^- has a weak improvement. After transforming scenario I^- into I_h , job h is the only job in π^- that may have a weak improvement.

In Step 5 of Algorithm 3, we eliminate weak improvements if they exist. Indeed, if job h has a weak improvement in π^- , then there is job l with $l < h$ such that $\lambda_l^- < \lambda_h^+$ and $A_G(l) \cap \{\pi(l + 1), \dots, \pi(h)\} = \emptyset$. Thus, if $\lambda_h^+ \leq \lambda_v^-$ and $v \notin B_G(h)$ in Step 5, then h has no weak improvements in the current permutation π_h^* . Otherwise, we would have $\lambda_l^- < \lambda_v^-$

and $v \notin A_G(I)$, i.e., job v would have a weak improvement in π^- .

Thus, if h really has a weak improvement in the current permutation π_h^* , then there are only two possibilities for job v . Either $v \rightarrow h$ or $\lambda_v^- < \lambda_h^+$. In the first case, we include v into the current subpermutation J_h ; in the latter case, we interchange the positions of v and J_h and obtain a new feasible permutation π_h^* with the value $\Phi_{\max}(I_h, \pi_h^*)$, which is not greater than the one of the previous permutation π_h^* (see Lemma 1). In the new permutation none of the jobs (excluding possibly job h) has a weak improvement.

If we have a situation with $v = 0$ or $\lambda_h^+ \leq \lambda_v^-$, then in the current permutation π_h^* , job h has no weak improvements. In view of Corollary 2, the final permutation π_h^* is optimal for scenario I_h .

If the inequality $C_h(\pi^-) + \lambda_h^+ \leq \Phi_{\max}(I^-, \pi^-)$ holds in Step 3, then π^- is an optimal permutation for scenario I_h . Indeed, if $\lambda_h^+ > \lambda_h^-$, then the validity of the above inequality means that there exists a critical job $j \neq h$ in π^- and the optimality of π^- follows from Corollary 2.

The running time of Algorithm 3 is $O(n^2)$. In fact, Step 1 takes $O(n^2)$ time. For each h , each run of steps 2–5 and 7 takes $O(1)$ time (for each pair v and h checking the validity or invalidity of the relation $v \rightarrow h$ takes $O(1)$ time if we use the representation of graph G by its adjacency matrix). Each of the steps 2–5 and 7 repeats at most n times. Step 6 takes $O(n)$ time and for each h it runs at most once. Thus, the overall complexity of steps 2–7 is $O(n^2)$. □

There are situations where there exists a permutation that is optimal for all scenarios simultaneously. Therefore, having an instance of a problem under uncertainty, it is reasonable to check the existence of such a permutation.

Definition 6 A permutation π is called *globally optimal* for problem 1|prec; $\lambda_j \in [\lambda_j^-, \lambda_j^+]$ | Φ_{\max} if π is optimal for all scenarios $I \in \mathcal{I}$ simultaneously.

We get immediately two characterizations for the global optimality.

Theorem 4 A globally optimal permutation π is obtained if Algorithm 2 returns the value $\Delta^* = 0$. If $\Delta^* > 0$ then no globally optimal permutation exists. This also means that π is globally optimal if and only if π is optimal for all instances I_h .

Consider a simple example illustrating the notion of global optimality.

Set $\varphi(t) = t$; $n = 3$; $p_1 = 1, p_2 = 3, p_3 = 0$; $\lambda_1^- = \lambda_1^+ = 1, \lambda_2^- = 5, \lambda_2^+ = 7, \lambda_3^- = 4, \lambda_3^+ = 8$. Graph G has a unique arc $(1, 2)$.

For this instance we have only three feasible permutations: $(1, 2, 3)$, $(1, 3, 2)$, and $(3, 1, 2)$. Algorithm 2 gives permutation $\pi = (3, 1, 2)$, and it is a globally optimal permutation

Table 1 Values p_j, λ_j^- and λ_j^+

Jobs	1	2	3	4	5	6	7	8
p_j	1	32	6	20	27	29	29	4
λ_j^-	43	32	47	33	46	54	44	58
λ_j^+	54	41	57	39	56	68	57	70

since $\Delta(I, \pi) = 0$. It is interesting to note that the minmax approach produces the same permutation, whereas the minmin approach (minimizing the objective function under scenario I^-) generates permutation $\pi^- = (1, 2, 3)$, which is not globally optimal (e.g., we have $\Phi_{\max}(I^+, \pi) = 11$ and $\Phi_{\max}(I^+, \pi^-) = 12$). There is one more globally optimal permutation $(1, 3, 2)$, but this permutation cannot be constructed with minmax or minmin approaches.

Consider the following example, which shows that a globally optimal permutation may exist in quite complicated situations.

Set $\varphi(t) = t$; $n = 8$. Values p_j, λ_j^- and λ_j^+ are presented in Table 1.

Graph G is presented by the set of its arcs: $(4, 5), (3, 6), (6, 7), (6, 2), (5, 2), (8, 5)$. Here we do not list the transitive arcs of the graph. We have $\pi = (8, 3, 6, 7, 4, 5, 1, 2)$ as a globally optimal permutation.

4 Conclusion

Our two main results related to the well-known minmax cost algorithm of Lawler (1973) are necessary and sufficient conditions for the permutation optimality and an $O(n^2)$ algorithm for constructing a schedule that minimizes the maximum regret criterion for the problem under interval uncertainty whenever the cost functions are decomposable.

Possible directions for further research include both considering more general cost functions (compared to decomposable functions) and addressing situations with more than one uncertain parameter for each job. Another possibility is to consider the relative regret approach instead of the absolute regret approach discussed in the paper. There are also other criteria that deal with uncertainty such as the OWA operator (Ordered Weighted Averaging aggregation operator). OWA operators were introduced by Yager (1988) to work with multicriteria problems. Recently, OWA operators were used in connection with discrete optimization problems under uncertainty (see Kasperski and Zieliński (2013)) with a discrete scenario set containing a finite number of possible scenarios. In our case of interval uncertainty with the infinite set of scenarios, the direct use of OWA operators is impossible, but some special cases of the general OWA operator may be considered.

One should mention the considerable difference between uncertainty problems with the discrete scenario set and those with interval uncertainty as in our case. The former problems are very often more difficult than the problems with the interval uncertainty. For example, in the recent paper by [Aissi et al. \(2011\)](#), the strong NP-hardness of finding the minmax decision for the single-machine scheduling problem to minimize the number of late unit-time jobs was established under the due date uncertainty and the discrete scenario set where the number of the scenarios is the input parameter. Earlier, the NP-hardness of finding the minmax decision for the same problem was established in [Aloulou and Della Croce \(2008\)](#) under the processing time uncertainty and the discrete scenario set even for the case of two scenarios. Unlike the discrete scenario set situation, finding the minmax decision for this problem is polynomially solvable for the case of interval uncertain due dates and processing times. It is enough to set the minimum values for all due dates, the maximum values for all processing times and to solve the obtained deterministic problem, using for instance the algorithm of [Moore \(1968\)](#).

Acknowledgments Financial support has been provided in part by the joint BRFFR-PICS Project (PICS 5379, BRFFR $\Phi 10\Phi\Pi - 001$). The research of the first two authors have been partially supported by the LabEx PERSYVAL-Lab (ANR-11-LABX-0025). We are grateful to referees for their substantial and helpful comments.

References

- Aissi, H., Aloulou, M. A., & Kovalyov, M. Y. (2011). Minimizing the number of late jobs on a single machine under due date uncertainty. *Journal of Scheduling*, *14*, 351–360.
- Aloulou, M. A., & Della Croce, F. (2008). Complexity of single machine scheduling problems under scenario-based uncertainty. *Operations Research Letters*, *36*, 338–342.
- Averbakh, I. (2000). Minmax regret solutions for minimax optimization problems with uncertainty. *Operations Research Letters*, *27*, 57–65.
- Kasperski, A. (2005). Minimizing maximal regret in the single machine sequencing problem with maximum lateness criterion. *Operations Research Letters*, *33*, 431–436.
- Kasperski, A., & Zieliński, P. (2013). Bottleneck combinatorial optimization problems with uncertain costs and the OWA criterion. *Operations Research Letters*, *41*, 639–643.
- Kouvelis, P., & Yu, G. (1997). *Robust discrete optimization and its applications*. Dordrecht: Kluwer Academic Publishers.
- Lawler, E. L. (1973). Optimal sequencing of a single machine subject to precedence constraints. *Management Science*, *19*(5), 544–546.
- Lin, Y., & Wang, X. (2007). Necessary and sufficient conditions of optimality for some classical scheduling problems. *European Journal of Operational Research*, *176*, 809–818.
- Moore, J. M. (1968). A n job, one machine scheduling algorithm for minimizing the number of late jobs. *Management Science*, *15*, 102–109.
- Savage, L. J. (1954). *The foundations of statistics*. New York: Wiley.
- Wald, A. (1950). *Statistical decision functions*. New York: Wiley.
- Yager, R. R. (1988). On ordered weighted averaging aggregation operators in multicriteria decision making. *IEEE Transaction on Systems, Man, and Cybernetics*, *18*, 183–190.