RAIRO-Oper. Res. 55 (2021) 135–140
https://doi.org/10.1051/ro/2020144

RAIRO Operations Research
www.rairo-ro.org

HIERARCHICAL MINIMIZATION OF TWO MAXIMUM COSTS ON A BOUNDED SERIAL-BATCHING MACHINE

Cheng He*, Hao Lin and Li Li

Abstract. This paper studies a hierarchical optimization problem of scheduling n jobs on a serial-batching machine, in which two objective functions are maximum costs. By a hierarchical optimization problem, we mean the problem of optimizing the secondary criterion under the constraint that the primary criterion is optimized. A serial-batching machine is a machine that can handle up to b jobs in a batch and jobs in a batch start and complete respectively at the same time and the processing time of a batch is equal to the sum of the processing times of jobs in the batch. When a new batch starts, a constant setup time s occurs. We confine ourselves to the bounded model, where $b < n$. We present an $O(n^4)$-time algorithm for this hierarchical optimization problem. For the special case where two objective functions are maximum lateness, we give an $O(n^3 \log n)$-time algorithm.

Mathematics Subject Classification. 90C27, 90B35.

Received August 27, 2018. Accepted December 18, 2020.

1. Introduction

It is well known that the problem $1||f_{\max}$ can be solved by the classical Lawler's algorithm for $1|prec|f_{\max}$ with the precedence constraints ignored (see [1]). For the bicriteria scheduling problem, Hoogeveen [9] showed that the problem of minimizing two maximum cost criteria, that is $1||(f_{\max}, g_{\max})$, is solvable in $O(n^4)$ time. For the scheduling on an unbounded or bounded serial-batching machine, Cheng and Kovalyov [2] studied a serial of constrained optimization problems of minimizing $L_{\max}$, $\sum w_j T_j$, $\sum W_j C_j$ and $\sum W_j U_j$. In our previous work [5, 6, 8] we investigated serial-batching problems $1|s\text{-}batch, b \geq n$ or $b < n|(L_{\max}, C_{\max})$ and $1|s\text{-}batch, b \geq n$ or $b < n|(C_{\max}, \sum C_j)$, respectively. For the problem $1|s\text{-}batch, b \geq n|(f_{\max}, C_{\max})$, He *et al.* [7] present an $O(n^5)$-time algorithm. Recently, Geng *et al.* [3] give an improved $O(n^4)$-time algorithm for the problem $1|s\text{-}batch, b \geq n$ or $b < n|(f_{\max}, C_{\max})$. In the paper we consider the hierarchial scheduling problem, with two maximum costs $f_{\max}$ and $g_{\max}$, on a bounded serial-batching machine. Following the traditional three-field notation scheme of Graham *et al.* [4], this model may be denoted by $1|s\text{-}batch, b < n|Lex(f_{\max}, g_{\max})$, where "*s-batch*" stands for the serial-batching, "$b < n$" means that the batch capacity is bounded, and $Lex(f_{\max}, g_{\max})$ represents minimizing $g_{\max}$ subject to the restriction that $f_{\max}$ is optimized. The motivation of studying sequence batching comes from scheduling manufacturing systems where production items flow between facilities in containers such as boxes, pallets or carts [11]. A set

Keywords. Hierarchical scheduling, serial-batching, maximum cost.

School of Science, Henan University of Technology, Zhengzhou, Henan 450001, P.R. China.
*Corresponding author: hech202@163.com

Article published by EDP Sciences

of items assigned to the same container is considered as a batch. It is often the case that items are processed sequentially and leave the facility together in a batch, thus having equal completion times. The changeover time between different batches represents the setup time. In many practical situations, the capacities of containers are naturally restricted which leads to the bounded model [2]. Two objective functions may refer to different interests of two decision-makers. It is meaningful to study this kind of combination of two aspects. The main result of this paper is to present an $O(n^4)$-time algorithm. Moreover, we give an $O(n^3 \log n)$-time algorithm for the special problem $1|s\text{-}batch, b < n|Lex(L_{\max}, L'_{\max})$.

The rest of the paper is organized as follows. In Section 2 we state some preliminaries. Section 3 is dedicated to the main result, an $O(n^4)$ algorithm for the problem, and an $O(n^3 \log n)$-time algorithm is given for a special case. Section 4 gives a short summary. We shall follow the terminology and notation of Brucker [1].

2. Preliminaries

A serial-batching machine is a machine that can handle up to b jobs in a batch, where b stands for the capacity of a batch. Concerning the capacity b, the bounded model ($b < n$) and the unbounded model ($b \geq n$) are distinguished in the literature. Under the setting of serial-batching, jobs in a batch start and complete respectively at the same time and the processing time of a batch is equal to the sum of the processing times of jobs in the batch. When a new batch starts, a constant setup time s occurs. We only consider the bounded model, *i.e.*, $b < n$.

Suppose that we are given a set of n independent jobs $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$, which are to be scheduled on a serial-batching machine. Job J_j has a processing time p_j and two cost functions $f_j(t)$ and $g_j(t)$ ($j = 1, \ldots, n$). Given a schedule σ, we use $C_j(\sigma)$ to denote the completion time of job J_j in σ. Then $f_j(\sigma) = f_j(C_j(\sigma))$ and $g_j(\sigma) = g_j(C_j(\sigma))$ are defined as two costs of job J_j in σ, and $f_{\max}(\sigma) = \max_{j=1}^{n} f_j(\sigma)$ and $g_{\max}(\sigma) = \max_{j=1}^{n} g_j(\sigma)$ are two maximum costs of σ, respectively. Without loss of generality, we assume that the parameters of jobs and machine are integral. Additionally, we assume that the cost functions $f_j(t)$ and $g_j(t)$ are regular (nondecreasing with respect to the completion time $t = C_j$ of job J_j ($j = 1, \ldots, n$)), and the values of $f_j(t)$ and $g_j(t)$ can be calculated in constant time for every given time $t \geq 0$.

For problems of minimizing a regular objective function without job's release dates, there must be an optimal solution in which the batches are processed contiguously from time zero onwards. Throughout the paper, we restrict our attention to the solutions with this property. Thus, a schedule σ is a sequence of batches $\sigma = (B_1, B_2, \ldots, B_l)$, where each batch B_k ($k = 1, \ldots, l$) is a set of jobs. So the processing time of batch B_k is $p(B_k) = \sum_{J_j \in B_k} p_j$ and its completion time is $C(B_k) = \sum_{q=1}^{k} p(B_q) + ks$. Note that the completion time of job J_j in σ, for each $J_j \in B_k$ and $1 \leq k \leq l$, is $C_j(\sigma) = C(B_k)$.

In this paper, the criteria under consideration are two regular objective functions: maximum costs $f_{\max}$ and $g_{\max}$. Our goal is solving the problem $1|s\text{-}batch, b < n|Lex(f_{\max}, g_{\max})$. Here, the objective $Lex(f_{\max}, g_{\max})$ stands for the hierarchical optimization of minimizing $g_{\max}$ under the constraint that $f_{\max}$ is minimum, namely, the minimization of $g_{\max}$ is taken in the set of all optimal schedules of problem $1|s\text{-}batch, b < n|f_{\max}$. In this paper, we give an $O(n^4)$-time algorithm for the problem. Moreover, we present an $O(n^3 \log n)$-time algorithm for the special problem $1|s\text{-}batch, b < n|Lex(L_{\max}, L'_{\max})$.

3. Polynomial-time algorithm

Suppose that the n jobs in $\mathcal{J}$ have been re-indexed according to the well-known LPT rule so that $p_1 \geq p_2 \geq \cdots \geq p_n$. Then we fix this order throughout the paper. For each subset $Q \subseteq \mathcal{J}$, we use $\boldsymbol{Q}$ to denote the list of the jobs in Q so that the jobs in Q are listed in increasing order of their indices, *i.e.*, for any jobs J_i and J_j in Q, J_i is listed before J_j in $\boldsymbol{Q}$ if and only if $i < j$. Let $|Q|$ be the number of jobs in Q.

Since each schedule has at most n batches, for convenience, we introduce empty batches with processing times 0 and the setup time 0 in schedules such that each schedule has exactly n batches. Thus we always write a schedule in the form of $\sigma = (B_1, B_2, \ldots, B_n)$ throughout the paper. If in σ, the last l batches $B_{n-l+1}, B_{n-l+2}, \ldots, B_n$

are a partition of $\mathcal{J}$ and the first $n-l$ batches are empty batches, where $l \in \{1,2,\ldots,n\}$, then l is called the number of valid batches of schedule σ, denoted by $|\sigma|$, *i.e.*, $|\sigma| = l$.

Note that introduction of the empty batches have no effect on the costs $C_{\max}, f_{\max}$ and $g_{\max}$. It can be observed that the makespan of schedule σ mainly depends on the number of valid batches in schedule σ. He *et al.* [5] and Geng *et al.* [3] gave the following lemmas.

Lemma 3.1 ([5]). *For any schedule σ with $|\sigma| = l$ $(1 \le l \le n)$, $C_{\max}(\sigma^{(l)}) = ls + \sum_{1\le j\le n} p_j$.*

Lemma 3.2 ([3]). *The problem $1|s\text{-}batch, b < n|f$ can be solved in $O(n^4)$ time, where $f \in \{f_{\max}, g_{\max}\}$.*

Let σ^* be an optimal schedule of $1|s\text{-}batch, b < n|f_{\max}$ (if σ is also an optimal schedule, then $C_{\max}(\sigma^*) \le C_{\max}(\sigma)$), and π^* be an optimal schedule of $1|s\text{-}batch, b < n|g_{\max}$. Let $|\sigma^*| := l^*$ $f^* := f_{\max}(\sigma^*)$ and $\overline{g} := g_{\max}(\sigma^*)$ and $\underline{g} := g_{\max}(\pi^*)$. Then problem $1|s\text{-}batch, b < n|Lex(f_{\max}, g_{\max})$ is equivalent to problem $1|s\text{-}batch, b < n, f_{\max} \le f^*|g_{\max}$ and $\underline{g} \le g_{\max} \le \overline{g}$.

Let $C^{(l)}_{\max} := ls + \sum_{1\le j\le n} p_j$. Then we can compute the sums

$$C^{(l)}_{\max} = ls + \sum_{j=1}^{n} p_j \quad (l^* \le l \le n)$$

in a preprocessing step, which takes $O(n)$ time.

Lemma 3.3. *Let σ be an optimal schedule for problem $1|s\text{-}batch, b < n, f_{\max} \le f^*|g_{\max}$ and $|\sigma| = l$. Then $l^* \le l \le n$.*

Proof. First, we have $f_{\max}(\sigma) \le f^* = f_{\max}(\sigma^*)$. So σ is also an optimal schedule of $1|s\text{-}batch, b < n|f_{\max}$ by the optimality of σ^*. If $l < l^*$, then $C_{\max}(\sigma) = C^{(l)}_{\max} < C^{(l^*)}_{\max} = C_{\max}(\sigma^*)$, which contradicts that σ^* is an optimal schedule with the minimum makespan. And each schedule has at most n batches. So $l^* \le l \le n$. □

Assume that there is an *Algorithm FG* (we will state the Algorithm FG later) that may solve the problem $1|s\text{-}batch, b < n, f_{\max} \le f^*, C_{\max} = C^{(l)}_{\max}, g_{\max} \le g|\cdot$. Then by Lemma 3.3, problem $1|s\text{-}batch, b < n, f_{\max} \le f^*|g_{\max}$ $(\underline{g} \le g_{\max} \le \overline{g})$ can be solved by solving a serial of the feasibility problems $1|s\text{-}batch, b < n, f_{\max} \le f^*, C_{\max} = C^{(l)}_{\max}, g_{\max} \le g|\cdot$ for the decreasing g and the increasing l, where $\underline{g} \le g_{\max} \le \overline{g}$ and $l^* \le l \le n$. The iteration Procedure as follows.

Iteration Procedure (FG)

Step 1. Let σ^*, f^* and $\underline{g}$ be defined as above, let $l := l^*$ and $i := 0$ and $\sigma_i := \sigma^*$.

Step 2. If $g_{\max}(\sigma_i) = \underline{g}$, then σ_i is an optimal schedule of $1|s\text{-}batch, b < n, f_{\max} \le f^*|g_{\max}$ and stop. Otherwise, let $g := g_{\max}(\sigma_i) - 1$.

Step 3. Solving the problem $1|s\text{-}batch, b < n, f_{\max} \le f^*, C_{\max} = C^{(l)}_{\max}, g_{\max} \le g|\cdot$ by Algorithm FG. If the problem is infeasible, then if $l = n$, then σ_i is an optimal schedule of $1|s\text{-}batch, b < n, f_{\max} \le f^*|g_{\max}$ and stop; otherwise let $l := l + 1$ and go back to Step 3. If the problem is feasible, then let $i := i + 1$ and σ_i is the schedule obtained by performing Algorithm FG, go back to Step 2.

The following property form a base of Algorithm FG.

Lemma 3.4. *Assume that $1|s\text{-}batch, b < n, f_{\max} \le f^*, C_{\max} = C^{(l)}_{\max}, g_{\max} \le g|\cdot$ is feasible. Then there exists a feasible schedule σ such that the last batch consists of the first $\min\left\{\left|Q\left(C^{(l)}_{\max}\right)\right|, b\right\}$ jobs in $\boldsymbol{Q}\left(C^{(l)}_{\max}\right)$, where $Q\left(C^{(l)}_{\max}\right) = \left\{J_j \in \mathcal{J} : f_j\left(C^{(l)}_{\max}\right) \le f^* \text{ and } g_j\left(C^{(l)}_{\max}\right) \le g\right\}$.*

Proof. Let $\sigma = (B_1, B_2, \ldots, B_n)$ be a feasible schedule. Then $|\sigma| = l$ and $B_n \subseteq Q\left(C_{\max}^{(l)}\right)$ by $f_j\left(C_{\max}^{(l)}\right) \leq f^*$ and $g_j\left(C_{\max}^{(l)}\right) \leq g$ for any $J_j \in B_n$. Suppose that Q_n be the job set that contains the first $\min\left\{\left|Q\left(C_{\max}^{(l)}\right)\right|, b\right\}$ jobs in $\boldsymbol{Q}\left(C_{\max}^{(l)}\right)$. If B_n doesn't meet the property of Lemma 3.4, *i.e.*, $B_n \neq Q_n$, then $|B_n| \leq |Q_n|$ and $\max\{p_j : J_j \in B_n \setminus Q_n\} \leq \min\{p_j : J_j \in Q_n \setminus B_n\}$ by the definition of Q_n (if $B_n \subset Q_n$, then let $\max\{p_j : J_j \in B_n \setminus Q_n\} = 0$). Without loss of generality, assume that $|B_n| = |Q_n|$ (if $|B_n| < |Q_n|$, then transferring the $|Q_n| - |B_n|$ jobs, which belong to $Q_n \setminus B_n$ and have the smallest completion times in σ, to join batch B_n without influence on the feasibility of σ). For any $J_i \in B_n \setminus Q_n$ and $J_j \in Q_n \setminus B_n$ and $J_j \in B_k$, we construct a new schedule σ' by exchanging the positions of jobs J_i and J_j in σ. Then $C_h(\sigma') \leq C_h(\sigma)$ by $p_i \leq p_j$ for $1 \leq h \leq n$ and $h \neq j$, moreover, $f_j(\sigma') = f_j\left(C_{\max}^{(l)}\right) \leq f^*$ and $g_j(\sigma') = g_j\left(C_{\max}^{(l)}\right) \leq g$ by $J_j \in Q_n$. Hence $f_{\max}(\sigma') \leq f^*$ and $g_{\max}(\sigma') \leq g$. Therefore σ' is also a feasible schedule. Dealing with the other jobs similarly if needed, we can eventually obtain a feasible schedule which meet the property of Lemma 3.4. □

By Lemma 3.4, we use a list $\overrightarrow{U}$ with n positions in the order of $p_1 \geq p_2 \geq \cdots \geq p_n$ to store the candidate jobs in a batch temporarily. If a job j is stored in $\overrightarrow{U}$, then j is put into its position of this order (other positions may be left empty). When a batch $\mathcal{B}_k$ is constructed, we take out the first b jobs from $\overrightarrow{U}$ (or take all jobs in $\overrightarrow{U}$ if $|\overrightarrow{U}| \leq b$). Let $|\overrightarrow{U}|$ be the number of nonempty positions in $\overrightarrow{U}$. The problem $1|s\text{-}batch, b < n, f_{\max} \leq f^*, C_{\max} = C_{\max}^{(l)}, g_{\max} \leq g|\cdot$ can be solved by the following algorithm.

Algorithm FG

Step 0. Let $\mathcal{J} = \{J_1, J_2, \ldots, J_n\}$, $t := C_{\max}^{(l)}$ and $k := n$. Let $\overrightarrow{U}$ be defined as above and $|\overrightarrow{U}| = 0$.

Step 1. Let $Q(t) := \{J_j \in \mathcal{J} | f_j(t) \leq f^* \text{ and } g_j(t) \leq g\}$. Update $\overrightarrow{U}$ by inserting the jobs in $Q(t)$ into its position in $\overrightarrow{U}$ and let $\mathcal{J} := \mathcal{J} \setminus Q(t)$. If $|\overrightarrow{U}| = 0$, then the problem is infeasible and stop, else let B_k consists of the first $\min\left\{|\overrightarrow{U}|, b\right\}$ jobs in $|\overrightarrow{U}|$ and update $\overrightarrow{U}$ by deleting the jobs in B_k from $\overrightarrow{U}$ and $k := k - 1$ and $t := t - p(B_k) - s$.

Step 2. If $k = n - l$, then if $t = 0$, then the problem is feasible, return the feasible schedule $\sigma := (B_1, B_2, \cdots, B_n)$ and stop (where $B_1 = \ldots = B_{n-l} = \emptyset$); else $t > 0$, the problem is infeasible and stop. If $k > n - l$, then go back to Step 1.

Theorem 3.5. *The Algorithm FG solves the decision problem* $1|s\text{-}batch, b < n, f_{\max} \leq f^*, C_{\max} = C_{\max}^{(l)}, g_{\max} \leq g|\cdot$ *in* $O(ln + n\log n)$ *time.*

Proof. If the problem is feasible, then the procedure can be carried out until Step 2 and returns answer "yes". Conversely, if the procedure terminates by answer "no", then there is infeasible schedule for the problem. As for the running time, we see that the procedure has $l + 1$ rounds (k changes from n to $n - l$). In each round, we compute at most $2n$ values $f_j(t)$ and $g_j(t)$, and compare them with f^* and g, respectively. So the running time of each round is $O(n)$. And inserting or deleting jobs in Step 1 takes $O(n\log n)$ time. Therefore the overall time bound is $O(ln + n\log n)$. This completes the proof. □

In the following, we assume that σ_i and σ_{i+1} are any two adjacent schedules generated by Iteration Procedure (FG) with values of thresholds g_i and g_{i+1} ($g_i > g_{i+1}$), respectively, and the numbers of the valid batches of σ_i and σ_{i+1} are l_i and l_{i+1}, where $l^* \leq l_i \leq l_{i+1} \leq n$. For convenience, we write $\sigma_i = \left(B_1^{(i)}, B_2^{(i)}, \cdots, B_n^{(i)}\right)$ and $\sigma_{i+1} = \left(B_1^{(i+1)}, B_2^{(i+1)}, \cdots, B_n^{(i+1)}\right)$. We also write $t_k^{(i)} = C\left(B_k^{(i)}\right)$ and $t_k^{(i+1)} = C\left(B_k^{(i+1)}\right)$ for $1 \leq k \leq n$. Write $Q_k^{(i)} = \{J_j \in \mathcal{J} | f_j\left(t_k^{(i)}\right) \leq f^* \text{ and } g_j\left(t_k^{(i)}\right) \leq g_i\}$ and $Q_k^{(i+1)} = \{J_j \in \mathcal{J} | f_j\left(t_k^{(i+1)}\right) \leq f^* \text{ and } g_j\left(t_k^{(i+1)}\right) \leq g_{i+1}\}$ for $1 \leq k \leq n$. Let $P_k^{(i)}$ and $P_k^{(i+1)}$ be the total processing time of the last k batches of σ_i and σ_{i+1},

i.e., $P_k^{(i)} = p\left(B_{n-k+1}^{(i)}\right)+p\left(B_{n-k+2}^{(i)}\right)+\cdots+p\left(B_n^{(i)}\right)$ and $P_k^{(i+1)} = p\left(B_{n-k+1}^{(i+1)}\right)+p\left(B_{n-k+2}^{(i+1)}\right)+\cdots+p\left(B_n^{(i+1)}\right)$. The following property holds for σ_i and σ_{i+1}.

Property 3.6. *$t_k^{(i+1)} \geq t_k^{(i)}$ and $Q_k^{(i+1)} \subseteq Q_k^{(i)}$ and $P_{n-k+1}^{(i)} \geq P_{n-k+1}^{(i+1)}$ for $1 \leq k \leq n$.*

Proof. Since g_i and g_{i+1} are the values of thresholds when Iteration Procedure (FG) generate schedules σ_i and σ_{i+1}, respectively, we have $g_i \geq g_{\max}(\sigma_i) > g_{\max}(\sigma_i) - 1 = g_{i+1}$. We show the results by induction on k backwards. For $k = n$, $t_n^{(i+1)} \geq t_n^{(i)}$ by $l_{i+1} \geq l_i$. If $J_j \in Q_n^{(i+1)}$, then $f_j\left(t_n^{(i)}\right) \leq f_j\left(t_n^{(i+1)}\right) \leq f^*$ and $g_j\left(t_n^{(i)}\right) \leq g_j\left(t_n^{(i+1)}\right) \leq g_{i+1} < g_i$. Therefore $J_j \in Q_n^{(i)}$, *i.e.*, $Q_n^{(i+1)} \subseteq Q_n^{(i)}$. By Algorithm FG, we have $P_1^{(i)} = p\left(B_n^{(i)}\right) \geq p\left(B_n^{(i+1)}\right) = P_1^{(i+1)}$. Moreover, $t_k^{(i)} = 0$ and $B_k^{(i)} = \emptyset\left(\textit{i.e.}, P_{n-k+1}^{(i)} = \sum_{1\leq j\leq n} p_j\right)$ for $1 \leq k \leq n - l_i$. So $t_k^{(i+1)} \geq t_k^{(i)}$ and $Q_k^{(i+1)} \subseteq Q_k^{(i)} = \mathcal{J}$ and $P_{n-k+1}^{(i)} \geq P_{n-k+1}^{(i+1)}$ for $1 \leq k \leq n - l_i$.

Suppose that $t_k^{(i+1)} \geq t_k^{(i)}$ and $Q_k^{(i+1)} \subseteq Q_k^{(i)}$ and $P_{n-k+1}^{(i)} \geq P_{n-k+1}^{(i+1)}$ for $n - l_i + 2 \leq k \leq n - 1$. Then $t_{n-l_i+1}^{(i+1)} = t_n^{(i+1)} - (l_i - 1)\, s - P_{l_i-1}^{(i+1)} \geq t_n^{(i)} - (l_i - 1)\, s - P_{l_i-1}^{(i)} = t_{n-l_i+1}^{(i)}$. Therefore, $Q_{n-l_i+1}^{(i+1)} \subseteq Q_{n-l_i+1}^{(i)}$ and $P_{n-k+1}^{(i)} = P_{l_i}^{(i)} = \sum_{1\leq j\leq n} p_j \geq P_{l_i}^{(i+1)} = P_{n-k+1}^{(i+1)}$ for $k = n - l_i + 1$. This completes the proof. □

Let $\sigma_0, \sigma_1, \ldots, \sigma_k$ be all schedules obtained by Iteration Procedure (FG) and $\Delta(\sigma_i) = |Q_1^{(i)}|+|Q_2^{(i)}|+\cdots+|Q_n^{(i)}|$ for $0 \leq i \leq k$. Then we have the following corollary.

Corollary 3.7. $0 < \Delta(\sigma_k) < \Delta(\sigma_{k-1}) < \cdots < \Delta(\sigma_0) \leq n^2$.

Proof. By Property 3.6 and the definition of $|Q_l^{(i)}|$, we have $n \geq |Q_l^{(i)}| \geq |Q_l^{(i+1)}| \geq 0$ for $0 \leq i \leq k - 1$ and $1 \leq l \leq n$. Therefore $0 < \Delta(\sigma_k) \leq \Delta(\sigma_{k-1}) \leq \cdots \leq \Delta(\sigma_0) \leq n^2$. If $\Delta(\sigma_i) = \Delta(\sigma_{i+1})$ for some $0 \leq i \leq k - 1$, then $|Q_l^{(i)}| = |Q_l^{(i+1)}|$ for $1 \leq l \leq n$. Thus $\sigma_i = \sigma_{i+1}$. This contradicts that $g_{\max}(\sigma_i) > g_{\max}(\sigma_{i+1})$. Hence $\Delta(\sigma_k) < \Delta(\sigma_{k-1}) < \cdots < \Delta(\sigma_0)$, as required. □

Theorem 3.8. *The Iteration Procedure (FG) solves problem $1|s\text{-}batch, b < n|Lex(f_{\max}, g_{\max})$ in $O(n^4)$ time.*

Proof. Iteration Procedure (FG) can solve problem $1|s\text{-}batch, b < n, f_{\max} \leq f^*|g_{\max}$ ($\underline{g} \leq g_{\max} \leq \overline{g}$) by Theorem 3.5 and the implementation of the procedure. Therefore Iteration Procedure (FG) can solve problem $1|s\text{-}batch, b < n|Lex(f_{\max}, g_{\max})$.

As to the running time, the total running time of Step 1 is $O(n^4)$ by Lemma 3.2. Moreover, Step 2 can be performed in constant time. Note that, during the implementation of Step 3, by running Algorithm FG once, we either obtain a schedule, or return an output of infeasibility and let $l := l + 1$ till $l = n$. The latter case occurs $n - l^* + 1$ times. On the other hand, let $\sigma_1, \sigma_2, \ldots, \sigma_k$ be all schedules obtained successively by running Step 3. Then $0 < \Delta(\sigma_k) < \Delta(\sigma_{k-1}) < \cdots < \Delta(\sigma_0) \leq n^2$ by Corollary 3.7. So the number of rounds of Step 3 is at most $n - l^* + 1 + n^2 = O(n^2)$. Since the running time of each round is at most $O(ln + n \log n)$ ($l^* \leq l \leq n$) by Theorem 3.5, the running time of Step 3 is $O(n^4 + n^3 \log n) = O(n^4)$. To summarize, the overall complexity of the procedure is $O(n^4)$. □

If each job J_j has two due dates d_j and d'_j ($1 \leq j \leq n$), then two objective functions of maximum lateness $L_{\max}$ and $L'_{\max}$ are induced, *i.e.*, $L_{\max}(\sigma) = \max_{j=1}^n (C_j(\sigma) - d_j)$ and $L'_{\max}(\sigma) = \max_{j=1}^n \left(C_j(\sigma) - d'_j\right)$ for any given schedule σ. We consider the special problem $1|s\text{-}batch, b < n|Lex\left(L_{\max}, L'_{\max}\right)$ in the following.

First, the problem $1|s\text{-}batch, b < n|L$ can be solved in $O(n^3 \log n)$ time, where $L \in \{L_{\max}, L'_{\max}\}$ [10]. Let $f_{\max} = L_{\max}$ and $g_{\max} = L'_{\max}$ above. Thus $f_{\max} \leq f^*, g_{\max} \leq g \Leftrightarrow C_j \leq d_j + f^*, C_j \leq d'_j + g$ (where f^* denotes the optimal value of $1|s\text{-}batch, b < n|L_{\max}$). Let $\overline{d}_j = \min\{d_j + f^*, d'_j + g\}$. We have $f_{\max} \leq f^*, g_{\max} \leq g \Leftrightarrow C_j \leq \overline{d}_j$.

In order to improve the time complexity of Algorithm FG for the special problem $1|s\text{-}batch, b < n|Lex(L_{\max}, L'_{\max})$, re-indexing the n jobs in $\mathcal{J}$ so that $\overline{d}_1 \geq \overline{d}_2 \geq \cdots \geq \overline{d}_n$. Note that either $Q(t) = \emptyset$

or the subscripts of the jobs in $Q(t)$ are successive in Algorithm FG, *i.e.*, if $Q(t) \neq \emptyset$, then we may suppose that $Q(t) = \{J_i, J_{i+1}, \cdots, J_l\}$, where $t \leq \overline{d}_l$ and $t > \overline{d}_{l+1}$. Furthermore, the current $\mathcal{J} = \{J_{l+1}, J_{l+2}, \cdots, J_n\}$ (after updating $\mathcal{J}$). Thus Algorithm FG can solve problem $1|s\text{-}batch, b < n, L_{\max} \leq f^*, C_{\max} = C^{(l)}_{\max}, L'_{\max} \leq g|\cdot$ in $O(n + n\log n) = O(n\log n)$ time. Hence problem $1|s\text{-}batch, b < n|Lex(L_{\max}, L'_{\max})$ can be solved in $O(n^3 \log n)$ time by Theorem 3.8.

4. Concluding remarks

In the foregoing discussion, we investigate a hierarchical scheduling problem on a bounded serial-batching machine, in which two objective functions are maximum costs or maximum lateness. Moreover, for the simultaneous optimization scheduling problem on a serial-batching machine to minimize maximum cost and makespan, Geng *et al.* [3] presented a polynomial-time algorithm. Our future works would be simultaneous optimization scheduling problems on a serial-batching machine, for example, $1|s\text{-}batch, b \geq n$ or $b < n|(L_{\max}, L'_{\max})$ and $1|s\text{-}batch, b \geq n$ or $b < n|(L_{\max}, \sum C_j)$.

Acknowledgements. This work was supported by by KRPOHNHEI (No. 20A110003) and STAPHNOS (No. 2020-70) and NSFC (No. 12001169) and IFPHNUT (No. 2020ZKCJ08).

References

[1] P. Brucker, Scheduling Algorithms, 3rd edition. Springer, Berlin-Heidelberg (2001).
[2] T.C.E. Cheng and M.Y. Kovalyov, Single machine batch scheduling with sequential job processing. *IIE Trans.* **33** (2001) 413–420.
[3] Z.C. Geng, J.J. Yuan and J.L. Yuan, Scheduling with or without precedence relations on a serial-batch machine to minimize makespan and maximum cost. *Appl. Math. Comput.* **332** (2018) 1–18.
[4] R.L. Graham, E.L. Lawler, J.K. Lenstra and A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: a survey. *Ann. Discrete Math.* **5** (1979) 287–326.
[5] C. He, Y.X. Lin and J.J. Yuan, Bicriteria scheduling of minimizing maximum lateness and makespan on a serial-batching machine. *Found. Comput. Decis. Sci.* **33** (2008) 369–376.
[6] C. He, Y.X. Lin and J.J. Yuan, A DP algorighm for minimizing makespan and total completion time on a series-batching machine. *Inf. Process. Lett.* **109** (2009) 603–607.
[7] C. He, H. Lin, Y.X. Lin and J. Tian, Bicriteria scheduling on a series-batching machine to minimize maximum cost and makespan. *Central Eur. J. Oper. Res.* **21** (2013) 177–186.
[8] C. He, H. Lin and Y.X. Lin, Bounded serial-batching scheduling for minimizing maximum lateness and makespan. *Discrete Optim.* **16** (2015) 70–75.
[9] J.A. Hoogeveen, Single-machine scheduling to minimize a function of two or three maximum cost criteria. *J. Algorithms* **21** (1996) 415–433.
[10] L. Li, *Study on several multicriteria scheduling problems*. Master thesis, Henan University of Technology (2009).
[11] S. Webster and K.R. Baker, (1995) Scheduling groups of jobs on a single machine. *Oper. Res.* **43** (1995) 692–703.