A note on the single machine serial batching scheduling problem to minimize maximum lateness with identical processing times

J.J. Yuan[1,2] , **A.F. Yang**[1,2] **and** **T.C.E. Cheng**[2] *

[1]Department of Mathematics, Zhengzhou University,
Zhengzhou, Henan 450052, People's Republic of China

[2]Department of Management, The Hong Kong Polytechnic University,
Hung Hom, Kowloon, Hong Kong, People's Republic of China

ABSTRACT

We consider the single machine, serial batching scheduling problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$. The complexity of this problem is reported as open in the literature. By reducing this problem to the version without precedence constraints, we show that the problem is polynomially solvable.

Keywords: Scheduling, Precedence constraints, Batches.

1 Introduction and Problem Formulation

Let n jobs $J_1, J_2, ..., J_n$ and a single machine that can handle only one job at a time be given. There are precedence relations $\prec$ between the jobs. Each job J_j has a release date r_j, a processing time p_j and a weight w_j. The jobs are processed in batches. The completion time of all jobs in a batch is defined as the finishing time of the last job in the batch, *i.e.*, the processing time of a batch is equal to the sum of the processing times of all jobs in the batch. If a new batch is formed, a constant setup time s will be incurred. If J_i and J_j are two jobs such that $J_i \prec J_j$, we only require that J_i is processed before J_j, and so it is allowed that J_i and J_j are processed in the same batch. Jobs cannot start

*Corresponding author

before their release dates. We assume that the starting time of the setup of each batch cannot be before the maximum release date of the jobs in the batch. Hence, the starting time of a batch B is at least $\max_{J_j \in B} r_j + s$. Following [3] and [7], this model is called the serial batching scheduling problem and is denoted by

$$1|\textit{prec; s-batch}; r_j|f,$$

where f is the objective function to be minimized.

Coffman *et al.* [5] provide an algorithm to solve the problem $1|\textit{s-batch}|\sum C_j$ in $O(n \log n)$ time. Albers and Brucker [1] provide an $O(n^2)$ time algorithm to solve the problem $1|\textit{prec}; p_j = p; \textit{s-batch}|\sum C_j$, and prove that the problems $1|\textit{s-batch}|\sum w_j C_j$ and $1|\textit{chains}; p_j = 1; \textit{s-batch}|\sum w_j C_j$ are unary NP-hard. Webster and Baker [9] give an algorithm to solve the problem $1|\textit{s-batch}|L_{\max}$ in $O(n^2)$ time. Ng, Cheng and Yuan [8] reduce the problem $1|\textit{prec; s-batch}|L_{\max}$ to the problem $1|\textit{s-batch}|L_{\max}$ and show that the problem $1|\textit{prec; s-batch}|L_{\max}$ can be solved in $O(n^2)$ time by using a revised Webster-Baker algorithm. Recently, Baptiste [2] gives an algorithm to solve the problem $1|p_j = p; \textit{s-batch}; r_j|L_{\max}$ in $O(n^{14} \log n)$ time. Brucker and Knust [4] report that the complexity of the problem $1|\textit{prec}; p_j = p; \textit{s-batch}; r_j|L_{\max}$ is still open.

We show in this paper that, by appropriately modifying the release dates and due dates of the jobs, the problem $1|\textit{prec}; p_j = p; \textit{s-batch}; r_j|L_{\max}$ can be polynomially reduced to the problem $1|p_j = p; \textit{s-batch}; r_j|L_{\max}$ in $O(n^2)$ times. Consequently, the problem $1|\textit{prec}; p_j = p; \textit{s-batch}; r_j|L_{\max}$ is polynomially solvable.

2 Reduction

For the problem $1|\textit{prec}; p_j = p; \textit{s-batch}; r_j|L_{\max}$, a feasible schedule (solution) is a pair (π, BS) such that $\pi = (\pi(1), \pi(2), ..., \pi(n))$ is a sequence of job indices satisfying the precedence constraints, *i.e.*, for any pair of jobs J_i and J_j, if $J_i \prec J_j$, then $\pi^{-1}(i) < \pi^{-1}(j)$, and $BS = \{B_1, B_2, ..., B_k\}$ is a batch sequence that partitions the set of all jobs such that, for each index t with $1 \leq t \leq k$, $\{j : J_{\pi(j)} \in B_t\}$ consists of consecutive integers, *i.e.*, there are two integers x and y with $y = x + |B_t| - 1$ such that $B_t = \{J_{\pi(x)}, J_{\pi(x+1)}, ..., J_{\pi(y)}\}$. The starting time of a batch B under a schedule (π, BS) is denoted by $S_B(\pi, BS)$. The completion time of a job J_j under a schedule (π, BS) is denoted by $C_j(\pi, BS)$ (or simply C_j when there is no ambiguity).

If J_i and J_j are two jobs such that $J_i \prec J_j$, then the starting time of the batch that contains J_j must be at least r_i under any feasible schedule. Modifying the release date of each job J_j by setting

$$r'_j := \max\{r_j, r_i\},$$

we see that the maximum lateness will not change under any feasible schedule. Hence, we can recursively modify the release dates of the jobs such that, for each pair of jobs J_i and J_j, if $J_i \prec J_j$, then $r'_i \leq r'_j$. This procedure can be done in $O(n^2)$ time by the standard

"Algorithm Modify r_j" in the book by Brucker [3]. The modified release dates obtained above can be denoted in the following form:

$$r'_j := \max\{r_i : J_i \prec J_j \text{ or } i = j\}, \quad \text{for } 1 \le j \le n.$$

Again, if J_i and J_j are two jobs such that $J_i \prec J_j$, then the completion time of J_i must be less than or equal to that of J_j under any feasible schedule. Modifying the due date of job J_j by setting

$$d'_i := \min\{d_j, d_i\},$$

we see that the maximum lateness will not change under any feasible schedule. In fact, if (π, BS) is a feasible schedule, then $C_i(\pi, BS) \le C_j(\pi, BS)$, and so

$$\max\{C_i(\pi, BS) - d_i, C_j(\pi, BS) - d_j\} = \max\{C_i(\pi, BS) - d'_i, C_j(\pi, BS) - d_j\},$$

i.e., the maximum lateness of J_i and J_j is not changed after we modify the due date d_i of J_i to d'_i. Hence, we can recursively modify the due dates of the jobs such that, for each pair of jobs J_i and J_j, if $J_i \prec J_j$, then $d'_i \le d'_j$. This procedure can be done in $O(n^2)$ time by the standard "Algorithm Modify d_j" in the book by Brucker [3]. The modified due dates obtained above can be denoted in the following form:

$$d'_i := \min\{d_j : J_i \prec J_j \text{ or } i = j\}, \quad \text{for } 1 \le i \le n.$$

The above discussion leads to the following lemma.

Lemma 2.1 Any optimal schedule of the problem $1|prec; p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$ under the modified release dates and due dates is also an optimal schedule of the problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$ under the original release dates and due dates.

For every job subset $\mathcal{J} \subseteq \{J_1, J_2, ..., J_n\}$, a job $J_i \in \mathcal{J}$ is called a *minimal* job of $\mathcal{J}$ if J_i has no predecessor in $\mathcal{J}$.

The general idea of the procedure constructing a feasible schedule for the problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$ from an optimal solution for the problem $1|p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$ can be described as follows.

Let (π, BS) be an optimal schedule for the problem $1|p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$ under the modified release dates and due dates. If π satisfies the precedence relations $\prec$ between the jobs, then the schedule (π, BS) is also optimal for the version with precedence relations, and thus, we have nothing to do. Otherwise, let $J_{\pi(i)}$ be the first job under π such that $J_{\pi(j)} \prec J_{\pi(i)}$ for some $j > i$. Let $J_{\pi^*(i)}$ be the minimal job of the job subset

$$\{J_{\pi(j)} : j > i \text{ and } J_{\pi(j)} \prec J_{\pi(i)}\}.$$

Since $J_{\pi^*(i)} \prec J_{\pi(i)}$, we must have $r'_{\pi^*(i)} \le r'_{\pi(i)}$ and $d'_{\pi^*(i)} \le d'_{\pi(i)}$. Hence, by exchanging the positions of jobs $J_{\pi(i)}$ and $J_{\pi^*(i)}$ in the schedule (π, BS), the completion times of $J_{\pi(i)}$ and $J_{\pi^*(i)}$ under the new schedule are not greater than the completion times of $J_{\pi^*(i)}$ and

$J_{\pi(i)}$ under (π, BS) , respectively, and so the maximum lateness of the jobs $J_{\pi(i)}$ and $J_{\pi^*(i)}$ is not increased under the new schedule. By noting that the completion time of any job apart from $J_{\pi(i)}$ and $J_{\pi^*(i)}$ is not increased under the new schedule, the resulting new schedule is still optimal for $1|p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$. This procedure is repeated until a schedule (π^*, BS^*) satisfying the precedence relations $\prec$ between the jobs is obtained.

Formally, given any optimal schedule (π, BS) for the problem $1|p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$, we can recursively obtain a feasible schedule (π^*, BS^*) for the original problem in the following way.

(i) If $J_{\pi(1)}$ is a minimal job of $\{J_1, J_2, ..., J_n\}$, then set $J_{\pi^*(1)} = J_{\pi(1)}$, and the schedule is not changed. Otherwise, let $J_{\pi^*(1)}$ be a minimal job of $\mathcal{J}_1 = \{J_j : J_j \prec J_{\pi(1)}\}$. Then we exchange the positions of jobs $J_{\pi^*(1)}$ and $J_{\pi(1)}$ in the job sequence π and batch sequence BS. The present schedule is denoted by (π^1, BS^1).

(ii) Suppose that $(\pi^1, BS^1), (\pi^2, BS^2), ..., (\pi^i, BS^i)$ and also $J_{\pi^*(1)}, J_{\pi^*(2)}, ..., J_{\pi^*(i)}$ $(1 \le i \le n-1)$ have been obtained. If $J_{\pi^i(i+1)}$ is a minimal job of $\{J_{\pi^i(i+1)}, J_{\pi^i(i+2)}, ..., J_{\pi^i(n)}\}$, then set $J_{\pi^*(i+1)} = J_{\pi^i(i+1)}$, and the schedule is not changed. Otherwise, let $J_{\pi^*(i+1)}$ be a minimal job of $\mathcal{J}_{i+1} = \{J_{\pi^i(j)} : j > i + 1 \text{ and } J_{\pi^i(j)} \prec J_{\pi^i(i+1)}\}$. Then we exchange the positions of jobs $J_{\pi^*(i+1)}$ and $J_{\pi^i(i+1)}$ in the job sequence π^i and batch sequence BS^i. The present schedule is denoted by (π^{i+1}, BS^{i+1}).

(iii) Set $(\pi^*, BS^*) = (\pi^n, BS^n)$.

Clearly, the creation of the schedule (π^*, BS^*) takes at most $O(n^2)$ time. By the definition of (π^*, BS^*) and the above discussion, we can easily see the correctness of the following lemma.

Lemma 2.2 (π^*, BS^*) is an optimal schedule for the modified problem

$$1|p_j = p; s\text{-}batch; r'_j; d'_j|L_{\max}$$

under the modified release dates and due dates. Furthermore, if $J_{\pi^*(i)}$ and $J_{\pi^*(j)}$ are two jobs such that $J_{\pi^*(i)} \prec J_{\pi^*(j)}$, then $i < j$

Combining Lemma 2.1 and Lemma 2.2, we deduce that (π^*, BS^*) is an optimal schedule for the original problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$. Hence, the problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$ can be polynomially reduced to the problem $1|p_j = p; s\text{-}batch; r_j|L_{\max}$ in $O(n^2)$ time.

Recall that Baptiste [2] gives an $O(n^{14} \log n)$ algorithm for the problem

$$1|p_j = p; s\text{-}batch; r_j|L_{\max}.$$

This implies that the problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$ can be solved in $O(n^{14} \log n)$ time as well.

3 Conclusions

In this paper, we have studied the single machine, serial batching scheduling problem $1|prec; p_j = p; s\text{-}batch; r_j|L_{\max}$. The complexity of this problem is reported as open in the literature. By reducing this problem to the version without precedence constraints, we show that the problem is solvable in $O(n^{14} \log n)$ time. Since the high order of the polynomial is not practical, any improvement in the complexity is interesting.

Acknowledgements

This research was supported in part by The Hong Kong Polytechnic University under grant number G-T397. The first author was also supported in part by the National Natural Science Foundation of China.

References

[1] S. Albers and P. Brucker, The complexity of one-machine batching problems, *Discrete Applied Mathematics*, **47**(1993), 87-107.

[2] P. Baptiste, Batching identical jobs, *Mathematical Methods of Operations Research*, **52**(2000), 355-367.

[3] P. Brucker, *Scheduling Algorithms*, Springer Verlag, Berlin, 2001.

[4] P. Brucker and S. Knust, Complexity results for scheduling problems, *http://www.mathematik.uni-osnabrueck.de/research/OR/class/*, 2002.

[5] E.G. Coffman, M. Yannakakis, M.J. Magazine and C. Santos, Batch sizing and sequencing on a single machine, *Annals of Operations Research*, **26**(1990), 135-147.

[6] E.L. Lawler, Sequencing jobs to minimize total weighted completion time subject to precedence constraints, *Annals of Discrete Mathematics*, **2**(1978), 75-90.

[7] J.K. Lenstra, A.H.G. Rinnooy Kan and P. Brucker, Complexity of machine scheduling problems, *Annals of Discrete Mathematics,* **1**(1977), 343-362.

[8] C.T. NG, T.C.E. Cheng and J.J. Yuan, A note on the single machine serial batching scheduling problem to minimize maximum lateness with precedence constraints, *Operations Research Letters*, **30**(2002), 66-68.

[9] S.T. Webster and K.R. Baker, Scheduling groups of jobs on a single machine, *Operations Research*, **43**(1995), 692-703.