The Eighth International Symposium on Operations Research and Its Applications (ISORA'09)
Zhangjiajie, China, September 20–22, 2009

Minimizing Total Weighted Completion Time on Uniform Machines with Unbounded Batch*

Cuixia Miao[1, 2,†] Yu-Zhong Zhang[1] Jianfeng Ren[1]

[1]Institute of Operations Research, Qufu Normal University, Rizhao, Shandong, 276826, China
[2]School of Mathematical Sciences, Qufu Normal University, Qufu, Shandong, 273165, China

Abstract In this paper, we consider the scheduling problem of minimizing total weighted completion time on uniform machines with unbounded batch. Each of the machine M_l $(l = 1, \cdots, m)$ has a speed s_l and can process up to $B(\geq n)$ jobs simultaneously as a batch. The processing time of a batch denoted by $P(B)$ is given by the processing time of the longest job in it, and the running time of the batch on machine M_l is $\frac{P(B)}{s_l}$. We present some useful properties of the optimal schedule, and then design an $O(n^{m+2})$ time backward dynamic programming algorithm for the problem.

Keywords Uniform machines with unbounded batch; batch-SPT; dynamic programming algorithm; polynomially solvable

1 Introduction

The batching schedule is motivated by burn-in operations in semiconductor manufacturing. Lee et al. ([1]) provided a background description, Webster and Baker ([2]) presented an overview of algorithms and complexity results for scheduling batch processing machines. This processing system has been extensively studied in the last decade ([3], [9], [12]).

For the batching schedule, there are two distinct models: the *bounded model*, in which the bound B for each batch size is effective, i.e., $B < n$, where n is the number of jobs, the *unbounded model*, in which there is effectively no limit on the size of batch, i.e., $B \geq n$ or $B(\infty)$. Problems of bounded model arise in the manufacture of integrated circuits ([1]), the critical final stage in the production of circuits is the burn-in operation, in which chips are loaded on boards which are then placed in an oven and exposed to high temperatures, each chip has a pre-specified minimum burn-in time and the burn-in oven has a limited capacity; scheduling problems of unbounded model arise for instance in situations where compositions need to be hardened in kilns and the kiln is sufficiently large that it does not restrict batch sizes ([4]). Zhang ([9]) gave a survey on both of the two models.

In this paper, we only address the unbounded model and the objective is to minimizing the total weighted completion time.

*This paper was supported by the National Natural Science Foundation (10671108), the PH.D Funds of Shandong Province (2007BS01014) and the Funds of Qufu Normal University (XJ0714).

†Corresponding author. E-mail address: miaocuixia@126.com

Previous related results: Under the *off-line* setting, for the problem of minimizing total weighted completion time on a single machine with unbounded batch, when all the jobs are released at the same time, Brucker ([4]) presented a dynamic programming algorithm in polynomial time; for general release times, Deng and Zhang ([5]) proved that the problem is NP-hard, and they further showed that several important special cases of the problem can be solved in polynomial time, and for the general case, Li et al. ([6]) gave a polynomial time approximation scheme (PTAS), and they ([7]) also presented a polynomial time approximation scheme (PTAS) for problem of minimizing total weighted completion time on identical parallel unbounded batch machines. Under the *on-line* setting and on a single unbounded batch machine, Chen ([8]) provided an on-line algorithm with $\frac{10}{3}-$competitive. ([4], [10]) give the motive of this paper.

Our contributions: We address the scheduling problem of minimizing total weighted completion time on uniform machines with unbounded batch for the first time. When jobs are released at the same time, we present some useful properties of the optimal schedule, and design an $O(n^{m+2})$ time backward dynamic programming algorithm in this paper.

2 Assumptions, Notations and Preliminaries

We are given a set of n independent jobs $J = \{J_1, \cdots, J_n\}$. Each job J_j $(j = 1, \cdots, n)$ requires processing time p_j which specifies the minimum time needed to process the job, and a weight w_j which is a measure of its importance, all the jobs are released at time zero.

There is a set of m uniform machines $M = \{M_1, \cdots, M_m\}$. Each machine M_l $(l = 1, \cdots, m)$ has a speed s_l, and can process up to $B(\geq n)$ jobs simultaneously as a batch.

The jobs that are processed together form a batch, we also denote it by B, and the processing time of the batch denoted by $p(B)$ is given by the processing time of the longest job in the batch, i.e., $p(B) = max\{p_j | J_j \in B\}$, and the running time of the batch on machine M_l is $\frac{P(B)}{s_l}$. All jobs contained in the same batch start and complete at the same time. Once processing of a batch is initiated, it can not be interrupted and other jobs cannot be introduced into the batch until processing is completed.

The objective is to schedule the jobs on the set of m uniform machines with unbounded batch so as to minimize $\sum w_j C_j$, where C_j denotes the completion time of job J_j. Using the $3-$field notation of Graham et al. ([11]), we denote our problem as: $Q_m|B(\infty)|\sum w_j C_j$, and $Q_m|B(\infty)|\sum C_j$ for $w_j = 1$ $(j = 1, \cdots, n)$.

Definition 1 *We say that a sequence is in batch-SPT if for any two batches B_x and B_y in the sequence, where B_x is processed before B_y, there is no pair of jobs J_i, J_j such that $J_i \in B_y$, $J_j \in B_x$ and $p_i < p_j$.*

3 Dynamic Programming Algorithm for $Q_m|B(\infty)|\sum w_j C_j$

In this section, for the problem $Q_m|B(\infty)|\sum w_j C_j$, we present some properties of the optimal schedule, and design a backward dynamic programming algorithm in polynomial time for it.

Firstly, we assume that the jobs are numbered in SPT order so that $p_1 \leq \cdots \leq p_n$.

3.1 Properties of Optimal Schedule

Theorem 1 *For the problem $Q_m|B(\infty)|\sum w_jC_j$, there exists an optimal schedule satisfying the following properties:*

(i) In each batch of the schedule, the indices of its jobs are consecutive in SPT order, i.e., one batch $B_i = \{J_{n_i}, J_{n_i+1}, \cdots, J_{n_{i+1}-1}\}$ in the schedule.

(ii) On each machine, the batches are sequenced in batch-SPT, i.e., if the schedule on machine M_l $(l = 1,\cdots,m)$ is $\pi^l = \{B^l_1,\cdots,B^l_{n_l}\}$, then $P(B^l_1) < \cdots < P(B^l_{n_l})$, where B^l_j $(j = 1,\cdots,n_l)$ is the j-th batch and n_l is the number of batches assigned to M_l in the schedule.

Proof. We consider any optimal schedule $\pi = \{\pi^1,\cdots,\pi^m\}$ on the number of m uniform machines, where $\pi^l = \{B^l_1,\cdots,B^l_{n_l}\}$ is the subschedule on machine M_l $(l = 1,\cdots,m)$ in π.

(i) Suppose that there are two different batches $B^{l_1}_x$ and $B^{l_2}_y$, and there are three jobs J_j, J_{j+1}, J_{j+2} with $p_j \le p_{j+1} \le p_{j+2}$ and J_j, J_{j+2} belong to $B^{l_1}_x$, J_{j+1} belongs to $B^{l_2}_y$.

We distinguish between two cases:

Case 1. $l_1 = l_2 = l$, that is, $B^{l_1}_x$ and $B^{l_2}_y$ are on the same machine.

Case 1.1. $x < y$

We can get a new subschedule $\pi'^l = \{B^l_1,\cdots,B^l_x \cup \{J_{j+1}\},\cdots,B^l_y \setminus \{J_{j+1}\},\cdots,B^l_{n_l}\}$ on M_l by moving job J_{j+1} from B^l_y to B^l_x, and we get a new schedule $\pi^* = \{\pi^1, \cdots, \pi'^l, \cdots, \pi^m\}$. Since $P_j \le P_{j+1} \le P_{j+2}$, we have that $p(B^l_x \cup \{J_{j+1}\}) = p(B^l_x)$, $p(B^l_y \setminus \{J_{j+1}\}) \le p(B^l_y)$.

Accordingly, the completion time of job J_{j+1} decreases from $C(B^l_y)$ to $C(B^l_x)$, while the completion times of the other jobs do not increase, i.e., $C_{j+1}(\pi^*) < C_{j+1}(\pi)$, and $C_i(\pi^*) \le C_i(\pi)$, for $i = 1,\cdots,n$, $i \ne j+1$, obviously, $w_{j+1}C_{j+1}(\pi^*) < w_{j+1}C_{j+1}(\pi)$, and $w_iC_i(\pi^*) \le w_iC_i(\pi)$, for $i = 1,\cdots,n$, $i \ne j+1$. This contradicts the optimal schedule π.

Case 1.2. $x > y$

We can get a new schedule $\pi'^l = \{B^l_1,\cdots,B^l_y \cup \{J_j\},\cdots,B^l_x \setminus \{J_j\},\cdots,B^l_{n_l}\}$ on M_l by moving job J_j from B^l_x to B^l_y, and we get a new schedule $\pi^* = \{\pi^1,\cdots,\pi'^l,\cdots,\pi^m\}$. Since $p_j \le p_{j+1} \le p_{j+2}$, we have that $p(B^l_y \cup \{J_j\}) = p(B^l_y)$, $p(B^l_x \setminus \{J_j\}) = p(B^l_x)$.

Consequently, $C_j(\pi^*) < C_j(\pi)$, and $C_i(\pi^*) = C_i(\pi)$, for $i = 1,\cdots,n$, $i \ne j$. A contradiction.

Case 2. $l_1 \ne l_2$, that is, $B^{l_1}_x$ and $B^{l_2}_y$ are on different machines, w.l.o.g let $1 \le l_1 < l_2 \le m$.

We distinguish three cases:

Case 2.1. $C(B^{l_1}_x) < C(B^{l_2}_y)$

We get $\pi'^{l_1} = \{B^{l_1}_1,\cdots,B^{l_1}_x \cup \{J_{j+1}\},\cdots,B^{l_1}_{n_{l_1}}\}$, $\pi'^{l_2} = \{B^{l_2}_1,\cdots,B^{l_2}_y \setminus \{J_{j+1}\},\cdots,B^{l_2}_{n_{l_2}}\}$ by moving job J_{j+1} from $B^{l_2}_y$ to $B^{l_1}_x$, and we get a new schedule

$$\pi^* = \{\pi^1,\cdots,\pi'^{l_1},\cdots,\pi'^{l_2},\cdots,\pi^m\}.$$

Since $p_j \le p_{j+1} \le p_{j+2}$, we have that $p(B^{l_1}_x \cup \{J_{j+1}\}) = p(B^{l_1}_x)$, $p(B^{l_2}_y \setminus \{J_{j+1}\}) \le p(B^{l_2}_y)$.

Consequently, the completion time of job J_{j+1} decrease from $C(B_y^{l_2})$ to $C(B_x^{l_1})$, while the completion times of other jobs do not increase, i.e., $C_{j+1}(\pi^*) < C_{j+1}(\pi)$, and $C_i(\pi^*) \leq C_i(\pi)$, for $i = 1, \cdots, n$, $i \neq j+1$. A contradiction.

Case 2.2. $C(B_x^{l_1}) > C(B_y^{l_2})$

We get $\pi'^{l_1} = \{B_1^{l_1}, \cdots, B_x^{l_1} \backslash \{J_j\}, \cdots, B_{n_{l_1}}^{l_1}\}$, $\pi'^{l_2} = \{B_1^{l_2}, \cdots, B_y^{l_2} \cup \{J_j\}, \cdots, B_{n_{l_2}}^{l_2}\}$

by moving job J_j from $B_x^{l_1}$ to $B_y^{l_2}$, and we get a new schedule

$$\pi^* = \{\pi^1, \cdots, \pi'^{l_1}, \cdots, \pi'^{l_2}, \cdots, \pi^m\}.$$

Since $p_j \leq p_{j+1} \leq p_{j+2}$, we have that $p(B_x^{l_1} \backslash \{J_j\}) = p(B_x^{l_1})$, $p(B_y^{l_2} \cup \{J_j\}) = p(B_y^{l_2})$.

Consequently, $C_j(\pi^*) < C_j(\pi)$, and $C_i(\pi^*) = C_i(\pi)$, for $i = 1, \cdots, n$, $i \neq j$. A contradiction.

Case 2.3. $C(B_x^{l_1}) = C(B_y^{l_2})$

We get $\pi'^{l_1} = \{B_1^{l_1}, \cdots, B_x^{l_1} \backslash \{J_j\}, \cdots, B_{n_{l_1}}^{l_1}\}$, $\pi'^{l_2} = \{B_1^{l_2}, \cdots, B_y^{l_2} \cup \{J_j\}, \cdots, B_{n_{l_2}}^{l_2}\}$

by moving job J_j from $B_x^{l_1}$ to $B_y^{l_2}$, and we get a new schedule

$$\pi^* = \{\pi^1, \cdots, \pi'^{l_1}, \cdots, \pi'^{l_2}, \cdots, \pi^m\}.$$

Since $p_j \leq p_{j+1} \leq p_{j+2}$, we have that $p(B_x^{l_1} \backslash \{J_j\}) = p(B_x^{l_1})$, $p(B_y^{l_2} \cup \{J_j\}) = p(B_y^{l_2})$.

Consequently, the completion times of all jobs are unchanged, then the objective value is unchanged.

A finite number of repetitions of this procedure yields an optimal schedule of the required form.

Now, the conclusion of (i) holds.

(ii) For the subschedule $\pi^l = \{B_1^l, \cdots, B_x^l, \cdots, B_y^l, \cdots, B_{n_l}^l\}$ on machine M_l ($l = 1, \cdots, m$) in π, to prove $P(B_1^l) < \cdots < P(B_x^l) < \cdots < P(B_y^l) < \cdots < P(B_{n_l}^l)$, assume the opposite, w.l.o.g suppose that $P(B_x^l) \geq P(B_y^l)$, from (i), we know that all the processing time of jobs in batch B_x^l are not smaller than those of jobs in B_y^l. If we move all the jobs in batch B_y^l to batch B_x^l, then the objective function value is strictly decreased, which contradicting the optimal schedule π. So, the conclusion of (ii) holds.

This completes the proof of Theorem 1.

3.2 Algorithm and Example

Based on Theorem 1, we present an $O(n^{m+2})$ time backward dynamic programming algorithm in this subsection.

Let $F_j(|J_1^j|, \ldots, |J_m^j|)$ be the minimum total weighted completion time in SPT order schedule on the number of m uniform machines with unbounded batch, and each machine M_l ($l = 1, \cdots, m$) contains the job subset J_l^j among $\{J_j, J_{j+1}, \cdots, J_n\}$. Where $\bigcup_{l=1}^m J_l^j = \{J_j, J_{j+1}, \cdots, J_n\}$, and $\sum_{l=1}^m |J_l^j| = n - j + 1$. Where $|J_l^j|$ denotes the cardinality of set J_l^j, i.e., $|J_l^j|$ is the total number of jobs on machine M_l among $\{J_j, J_{j+1}, \cdots, J_n\}$. Here, we must use the cardinality $|J_l^j|$ to denote the number of jobs instead of n_l^j, which is a natural number, because, when we calculate the objective in the iteration, we would consider the weights of jobs in set J_l^j.

Processing the first batch in the schedule starts at time zero on machine M_l $(l = 1, \cdots, m)$. Whenever a new batch is added to the beginning of this schedule and assigned to M_l, there is a corresponding delay to the processing of all those batches on that machine. Suppose that a batch $\{J_j, J_{j+1}, \cdots, J_{k-1}\}$, which has processing time p_{k-1}, is inserted at the start of the schedule and assigned to M_l.

For jobs $\{J_k, \cdots, J_n\}$, the total weighted completion time of some of them which scheduled on M_l increases by

$$\frac{p_{k-1}\sum_{J_i \in J_l^k} w_i}{s_l},$$

where J_l^k is the job set of the schedule among $\{J_k, \cdots, J_n\}$ on M_l, while the total weighted completion time of $\{J_j, J_{j+1}, \cdots, J_{k-1}\}$ is $\frac{p_{k-1}\sum_{i=j}^{k-1} w_i}{s_l}$.

As $J_l^k \bigcup \{J_j, \cdots, J_{k-1}\} = J_l^j$, thus, the overall increase in the total weighted completion time is

$$\frac{p_{k-1}\sum_{J_i \in J_l^k} w_i}{s_l} + \frac{p_{k-1}\sum_{i=j}^{k-1} w_i}{s_l} = \frac{p_{k-1}\sum_{J_i \in J_l^j} w_i}{s_l}.$$

A formal description of backward Dynamic Programming Algorithm is given bellow.

ALGORITHM DP(Dynamic Programming)

Step 1. Re-index all jobs according to the SPT order so that $p_1 \leq \cdots \leq p_n$.

Step 2. Let $F_{n+1}(|\phi|, \ldots, |\phi|) = 0$. If $(j; |J_1^j|, \ldots, |J_m^j|) \neq (n+1; |\phi|, \ldots, |\phi|)$, then $F_j(|J_1^j|, \ldots, |J_m^j|) = \infty$. Let $j = n$.

Step 3. For each tuple $(|J_1^j|, \ldots, |J_m^j|)$ such that $|J_l^j| \in \{0, 1, \cdots, n-j+1\}$, $l = 1, \cdots, m$, and $\sum_{l=1}^m |J_l^j| = n - j + 1$, $\bigcup_{l=1}^m J_l^j = \{J_j, \cdots, J_n\}$. Compute the following

$$F_j(|J_1^j|, \ldots, |J_m^j|) = min_{j<k\leq n+1}\{F_k(|J_1^j|, \ldots, (|J_l^j| - |\{J_j, J_{j+1}, \cdots, J_{k-1}\}|), \ldots, |J_m^j|)$$
$$+ \frac{p_{k-1}\sum_{J_i \in J_l^j} w_i}{s_l} : 1 \leq l \leq m\}.$$

If $j = 1$, go to **step 4**, otherwise , let $j = j - 1$, repeat **step 3**.

Step 4. Define

$$F^* = min\{F_1(|J_1^1|, \ldots, |J_m^1|) : |J_l^1| \in \{0, 1, \cdots, n\}, l = 1, \cdots, m, and \sum_{l=1}^m |J_l^1| = n, \bigcup_{l=1}^m J_l^1 = J\},$$

which is the optimal value , then to find the optimal schedule by backtracking.

Remarks:

1. The time complexity of **ALGORITHM DP** is $O(n^{m+2})$, and our problem is polynomially solvable if the number of machines m is constant.

2. In step 3, if

$$F_j(|J_1^j|, \ldots, |J_m^j|) = F_k(|J_1^j|, \ldots, (|J_l^j| - |\{J_j, J_{j+1}, \cdots, J_{k-1}\}|), \ldots, |J_m^j|) + \frac{p_{k-1}\sum_{J_i \in J_l^j} w_i}{s_l},$$

then batch $\{J_j, J_{j+1}, \cdots, J_{k-1}\}$ is inserted from the start on machine M_l.

Example: Consider the two-machine problem with the following dates.

$J = \{J_1, J_2, J_3\}$ with processing times $p_1 = 2, p_2 = 4, p_3 = 6$ and weights $w_1 = 3, w_2 = 1, w_3 = 5$.

$M = \{M_1, M_2\}$, and the speeds are $s_1 = 1$ and $s_2 = 2$.

The order of jobs is already in SPT order. If we use **ALGORITHM DP**, we can get the schedule: J_1 is assigned to M_1, J_2 and J_3 are in one batch assigned to M_2, and the objective function value is $F^* = \sum w_j C_j = 24$, which is the optimal value.

4 The Special Case of $w_j = 1$ for $j = 1, ..., n$

In this section, we discuss the special case of $w_j = 1$ for $j = 1, \cdots, n$, i.e., the problem $Q_m|B(\infty)|\sum C_j$. Theorem 1 is valid for the special case.

In section 3, if we replace the cardinality $|J_l^j|$ by a natural number n_l^j for $l = 1, \cdots, m$ and $j = 1, \cdots, n$, we can get the similar dynamic programming algorithm for the problem $Q_m|B(\infty)|\sum C_j$. We only recount it in sketch:

Let

$$F_j(|J_1^j|, \ldots, |J_m^j|) = F_j(n_1^j, \ldots, n_m^j).$$

Suppose that a batch $\{J_j, J_{j+1}, \cdots, J_{k-1}\}$ which has processing time p_{k-1} is inserted at the start of the schedule and assigned to M_l. The overall increase in the total completion time is

$$\frac{(k - j + n_l^k) p_{k-1}}{s_l} = \frac{n_l^j p_{k-1}}{s_l}.$$

The iterative formula is

$$F_j(n_1^j, \ldots, n_m^j) = min_{j<k\leq n+1}\{F_k(n_1^j, \ldots, n_l^j - (k-j), \ldots, n_m^j) + \frac{n_l^j p_{k-1}}{s_l} : 1 \leq l \leq m\}.$$

Example: Consider the two-machine problem with the following dates.

$J = \{J_1, J_2, J_3, J_4\}$ with processing times $p_1 = 1, p_2 = 2, p_3 = 4, p_4 = 6$.

$M = \{M_1, M_2\}$, and the speeds are $s_1 = 1$ and $s_2 = 2$.

The schedule is that J_2 is assigned to M_1, J_1 as a batch, is assigned to M_2 from the start, J_2 and J_3 are in one batch assigned to M_2 after J_1. The optimal objective function value is $F^* = min\{F_1(4,0), F_1(0,4), F_1(2,2), F_1(1,3), F_1(3,1)\} = min\{20, 10, 10, 9.5, 13\} = 9.5$.

5 Conclusion

In this paper, we present a backward dynamic programming algorithm for $Q_m|B(\infty)|\sum w_j C_j$, and it is polynomially solvable if the number of machine m is constant. An interesting problem for further research is $Q_m|r_j, B(\infty)|\sum w_j C_j$.

References

[1] C.-Y. Lee, R. Uzsoy and L.A. Martin-Vega, Efficient algorithms for scheduling semiconductor burn-in operations, *Operations Research,* **40,** 1992, 764-775.

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

[3] C.N. Potts and M.Y. Kovalyov, Scheduling with batching: A review, *European Journal of Operational Research,* **120,** 2000, 228-249.

[4] P. Brucker, A. Gladky, H. Hoogeveen, M.Y. Kovalyov, C.N. Potts, T. Tautenhahn, and S. van de Velde, Scheduling a batching machine, *J Sched,* **1,** 1998, 31-54.

[5] Xiaotie Deng and Yuzhong Zhang, Minimizing mean response time in batch processing system, *Lecture Notes in Computer Science,* **1627,** 1999, 231-240.

[6] Shuguang Li, Guojun Li and Hao Zhao, A linear time approximation scheme for minimizing total weighted completion time of unbounded batch scheduling, *Operations Research Transactions,* **8(4),** 2004, 27-32.

[7] Shuguang Li, Guojun Li and Xiuhong Wang, Minimizing total weighted completion time on parallel unbounded batch machines, *Journal of Software,* 2006, **17,** 2063-2068.

[8] Bo Chen, Xiaotie Deng and Wenan Zang, On-line scheduling a batch processing system to minimize total weighted job completion time, *Journal of Combinatorial Optimization,* **8,** 2004, 85-95.

[9] Yuzhong Zhang and Zhigang Cao, Parallel batch scheduling: A Survey, *Advances of mathematics,* **37,** 2008, 392-408.

[10] T.C.E. Cheng, Z.-L. Chen, M.Y. Kovalyov and B.M.T. Lin, Parallel-machine batching and scheduling to minimize total completion time, *IIE Transactions,* **28,** 1996, 953-956.

[11] R.L. Graham, Lawler, J. K. Lenstra and A.H.G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey, *Ann Disc Math,* **5,** 1979, 287-326.

[12] Yuzhong Zhang, Chunsong Bai and Shouyang Wang, Duplicating and its applications in batch scheduling, *Lecture Notes in Operations Research,* **5,** 2005, 108-117.