On Scheduling an Unbounded Batch Machine

Zhaohui Liu[1,3], Jinjiang Yuan[2,3] and T.C. Edwin Cheng[3,*]

[1]Department of Mathematics, East China University of Science and Technology

Shanghai 200237, People's Republic of China

[2]Department of Mathematics, Zhengzhou University

Zhengzhou, Henan 450052, People's Republic of China

[3]Department of Management, The Hong Kong Polytechnic University

Kowloon, Hong Kong SAR, People's Republic of China

August 27, 2002

Abstract

A batch machine is a machine that can process up to c jobs simultaneously as a batch, and the processing time of the batch is equal to the longest processing time of the jobs assigned to it. In this paper, we deal with the complexity of scheduling an unbounded batch machine, i.e., $c = +\infty$. We prove that minimizing total tardiness is binary NP-hard, which has been an open problem in the literature. Also, we establish the pseudopolynomial solvability of the unbounded batch machine scheduling problem with job release dates and any regular objective. This is distinct from the bounded batch machine and the classical single machine scheduling problems, most of which with different release dates are unary NP-hard. Combined with the existing results, this paper provides a nearly complete mapping of the complexity of scheduling an unbounded batch machine.

Keywords: Scheduling; Batch Processing; Complexity

*Corresponding author.

1 Introduction

A batch machine or batch processing machine is a machine that can process several jobs simultaneously as a batch, and the processing time of the batch is equal to the longest processing time of the jobs assigned to it. The research on batch machine scheduling is motivated by burn-in operations in semiconductor manufacturing (Lee et al. [8]). Potts and Kovalyov [11] review the existing results.

The problems that we study in this paper can be formulated as the following model. There are n independent jobs $J_1, J_2, \ldots, J_n$ to be scheduled on a batch machine that can process up to c jobs simultaneously, where c is called the capacity of the batch machine. Each job J_j $(1 \leq j \leq n)$ is associated with a processing time p_j and a release date r_j, before which the job cannot be scheduled. The scheduling objective is to minimize a regular minsum function $\sum f_j = \sum_{j=1}^{n} f_j(C_j)$ or a regular minmax function $f_{\max} = \max_{j=1}^{n} f_j(C_j)$, where f_j is a nondecreasing function of the completion time C_j of job J_j. Among the popular regular objectives are $C_{\max}, L_{\max}, \sum C_j, \sum w_j C_j, \sum U_j, \sum w_j U_j, \sum T_j$ and $\sum w_j T_j$. Specifically, we focus on the total tardiness $\sum T_j = \sum_{j=1}^{n} \max\{0, C_j - d_j\}$, where d_j is given as the due date of job J_j and $\max\{0, C_j - d_j\}$ is the tardiness of job J_j under a schedule. See Lawler et al. [6] for definitions of other objectives. As in Liu and Yu [10], the batch machine with capacity c is denoted by $B(c)$. In this paper, we restrict ourselves to the unbounded case, i.e., $B(\infty)$. Using the three-field notation, we denote the problems under consideration by $B(\infty)|\, r_j \,|\sum f_j$, $B(\infty)|\, r_j \,|f_{\max}$, and so on.

Cheng et al. [4] prove that $B(\infty)|\, r_j \,|L_{\max}$ is NP-hard. In addition, they establish the polynomial solvability of a wide variety of special cases of $B(\infty)|\, r_j \,|f_{\max}$ (including $r_j = 0$). It is shown in Brucker et al. [2] that $B(\infty)||\sum U_j$ is polynomially solvable, $B(\infty)||\sum w_j U_j$ and $B(\infty)||\sum w_j T_j$ are NP-hard, and $B(\infty)||\sum f_j$ is pseudopolynomially solvable. But it is open whether $B(\infty)||\sum T_j$ is polynomially solvable or binary NP-hard. Concerning $B(\infty)|\, r_j \,|\sum w_j C_j$, Deng and Zhang [5] establish its NP-hardness and present polynomial algorithms for several special cases.

As to the bounded case, $B(c)||C_{\max}$ is solved by a simple method due to Bartholdi (Lee and Uzsoy [7]). Brucker et al. [2] prove that $B(2)||L_{\max}$ (and hence $B(2)|\, r_j \,|C_{\max}$) is unary NP-hard. Baptiste [1] presents polynomial dynamic programming algorithms for problems $B(c)|\, r_j\,, p_j = p \,|\, F$ with $F \in \{\sum w_j C_j, \sum w_j U_j, \sum T_j\}$. Li and Lee [9] solve $B(c)|\, r_j \,|\sum U_j$ under some agreeability assumption on job processing times, release dates and due dates. However, the complexity of $B(c)||\sum C_j$ and $B(c)||\sum w_j C_j$ remains open, but $B(c)||\sum C_j$ can be solved in $O(n^{c(c-1)})$ time ([2]).

This paper is organized as follows. In Section 2, we prove the binary NP-hardness of $B(\infty)||\sum T_j$. This answers the open question posed in [2] and Brucker and Knust [3].

In Section 3, we show the pseudopolynomial solvability of the problems $B(\infty)|\, r_j \,|\sum f_j$ and $B(\infty)|\, r_j \,|f_{\max}$. Finally, In Section 4, we present a summary of the complexity status of various unbounded batch machine scheduling problems.

2 NP-hardness of total tardiness problem

In this section, we establish the binary NP-hardness of the problem $B(\infty)||\sum T_j$ by a reduction from the binary NP-complete PARTITION problem.

PARTITION Given t positive integers $a_1, a_2, \ldots, a_t$ with $\sum_{i=1}^{t} a_i = 2B$, decide if there exists a partition of the index set $I = \{1, 2, \ldots, t\}$ into two disjoint subsets I_1 and I_2 such that $\sum_{i\in I_1} a_i = \sum_{i\in I_2} a_i = B$.

Given an instance $\mathcal{P}$ of PARTITION, we first define $3t+1$ integers:

$$\begin{aligned}
M_t &= \sum_{i=1}^{t}(t-i)a_i + 8B \\
M_k &= 2\sum_{i=k+1}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + 8B \quad (k = t-1, t-2, \ldots, 1) \\
L_1 &= 7\sum_{i=1}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + 4B \\
L_k &= 2\sum_{i=1}^{k-1} L_i + 7\sum_{i=1}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + 4B \quad (k = 2, 3, \ldots, 2t+1)\,.
\end{aligned}$$

Obviously, the integers are such that

$$2B \ll M_t \ll M_{t-1} \ll \cdots \ll M_1 \ll L_1 \ll L_2 \ll \cdots \ll L_{2t+1}\,.$$

Now we define an instance $\mathcal{Q}$ of $B(\infty)||\sum T_j$ as follows.

$\mathcal{Q}$ consists of $10t+3$ jobs that are classified into $2t+1$ types. Each type $2k-1$ $(1 \le k \le t)$ contains five jobs: $J^1_{2k-1}, J^2_{2k-1}, J^3_{2k-1}$ and two additional copies of J^1_{2k-1}. Their processing times and due dates are given by

$$\begin{aligned}
p^1_{2k-1} &= L_{2k-1} \\
p^2_{2k-1} &= L_{2k-1} + M_k \\
p^3_{2k-1} &= L_{2k-1} + 2M_k \\
d^1_{2k-1} &= 2\sum_{i=1}^{2k-2} L_i + 5\sum_{i=1}^{k-1} M_i + L_{2k-1} + M_k + 2B \\
d^2_{2k-1} &= 2\sum_{i=1}^{2k-1} L_i + 5\sum_{i=1}^{k-1} M_i \\
d^3_{2k-1} &= 2\sum_{i=1}^{2k-1} L_i + 5\sum_{i=1}^{t} M_i + 2B\,.
\end{aligned}$$

Each type $2k$ ($1 \le k \le t$) also contains five jobs: $J_{2k}^1, J_{2k}^2, J_{2k}^3$ and two additional copies of J_{2k}^1. Their processing times and due dates are given by

$$\begin{aligned}
p_{2k}^1 &= L_{2k} \\
p_{2k}^2 &= L_{2k} + M_k + a_k \\
p_{2k}^3 &= L_{2k} + 2M_k \\
d_{2k}^1 &= 2\sum_{i=1}^{2k-1} L_i + 5\sum_{i=1}^{k-1} M_i + L_{2k} + 3M_k + 2B \\
d_{2k}^2 &= 2\sum_{i=1}^{2k} L_i + 5\sum_{i=1}^{k-1} M_i + 3M_k - (t-k+1)a_k \\
d_{2k}^3 &= 2\sum_{i=1}^{2k} L_i + 5\sum_{i=1}^{t} M_i + 2B\,.
\end{aligned}$$

Type $2t+1$ contains three copies of job J_{2t+1}^1 with

$$\begin{aligned}
p_{2t+1}^1 &= L_{2t+1} \\
d_{2t+1}^1 &= L_{2t+1} + 2\sum_{i=1}^{2t} L_i + 5\sum_{i=1}^{t} M_i + B\,.
\end{aligned}$$

Set the threshold value

$$T^* = 2\sum_{i=1}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + B\,.$$

We are asked to answer whether there exists a schedule σ for instance $\mathcal{Q}$ such that $T(\sigma) \le T^*$, where $T(\sigma)$ denotes the total tardiness of σ.

Clearly, the construction of $\mathcal{Q}$ takes a polynomial time under the binary coding. In the remainder of this section, we will show that $\mathcal{Q}$ has a schedule σ such that $T(\sigma) \le T^*$ if and only if the PARTITION instance $\mathcal{P}$ has a solution $\{I_1, I_2\}$ such that $\sum_{i\in I_1} a_i = \sum_{i\in I_2} a_i = B$. Note that if putting the jobs in instance $\mathcal{Q}$ according to the shortest processing time (SPT) rule, we obtain the sequence:

$$(J_1^1, J_1^2, J_1^3, J_2^1, J_2^2, J_2^3, \ldots, J_{2t}^1, J_{2t}^2, J_{2t}^3, J_{2t+1}^1)\,.$$

Suppose that $\mathcal{Q}$ has a schedule $\sigma = (\mathcal{B}_1, \mathcal{B}_2, \ldots, \mathcal{B}_m)$ such that $T(\sigma) \le T^*$, where each $\mathcal{B}_i$ is a batch. Let $p(\mathcal{B}_i)$ denote the processing time of batch $\mathcal{B}_i$. It is reasonable to require that the processing time of each job in $\mathcal{B}_{i+1}$ is larger than $p(\mathcal{B}_i)$; otherwise, shifting the jobs in $\mathcal{B}_{i+1}$ with processing times no larger than $p(\mathcal{B}_i)$ to $\mathcal{B}_i$ does not increase $T(\sigma)$. Then, σ has the properties:

(i) the jobs in each $\mathcal{B}_i$ come from a contiguous segment of the SPT sequence, and all $\mathcal{B}_i$s are arranged in order of increasing $p(\mathcal{B}_i)$;

(ii) for each k ($1 \leq k \leq 2t+1$), all J_k^1 s are processed in a batch.

Lemma 1 will give more explanations about the structure of σ.

Lemma 1 *σ has the following further properties:*

(iii) each batch contains only jobs of one type;

(iv) for each k ($1 \leq k \leq t$), the jobs of types $2k-1$ and $2k$ are divided into four batches: $\{J_{2k-1}^1, J_{2k-1}^2\}$, $\{J_{2k-1}^3\}$, $\{J_{2k}^1\}$, $\{J_{2k}^2, J_{2k}^3\}$ (Pattern 2112) with total processing time $2(L_{2k-1}+L_{2k})+5M_k$; or $\{J_{2k-1}^1\}$, $\{J_{2k-1}^2, J_{2k-1}^3\}$, $\{J_{2k}^1, J_{2k}^2\}$, $\{J_{2k}^3\}$ (Pattern 1221) with total processing time $2(L_{2k-1}+L_{2k})+5M_k+a_k$.

Proof If property (iii) does not hold, there must exist some k ($1 \leq k \leq 2t$) such that J_k^3 and J_{k+1}^1 are processed in a batch. But

$$\begin{aligned} p_{k+1}^1 - d_k^3 &= L_{k+1} - 2\sum_{i=1}^{k} L_i - 5\sum_{i=1}^{t} M_i - 2B \\ &= 2\sum_{i=1}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + 2B > T^*, \end{aligned}$$

which implies the tardiness of J_k^3 is larger than T^*, a contradiction to $T(\sigma) \leq T^*$.

We prove property (iv) by induction. If the jobs of type 1 are processed in a batch, then the total tardiness of three J_1^1 s is equal to

$$3(p_1^3 - d_1^1) = 3(M_1 - 2B) = 2M_1 + 2\sum_{i=2}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + 2B > T^*.$$

On the other hand, if J_1^1, J_1^2 and J_1^3 are processed in three batches, then the tardiness of J_1^3 is

$$\sum_{j=1}^{3} p_1^j - d_1^3 > L_1 - 5\sum_{i=1}^{t} M_i - 2B > T^*.$$

Thus, J_1^1, J_1^2 and J_1^3 must be processed in two batches: $\{J_1^1, J_1^2\}$, $\{J_1^3\}$; or $\{J_1^1\}$, $\{J_1^2, J_1^3\}$. Further, noticing that the two batches of type 1 require at least $2L_1 + 2M_1$ units of processing time, we can similarly prove that the jobs of type 2 are processed in two batches: $\{J_2^1, J_2^2\}$, $\{J_2^3\}$; or $\{J_2^1\}$, $\{J_2^2, J_2^3\}$.

If batches $\{J_1^1, J_1^2\}$ and $\{J_2^1, J_2^2\}$ both exist, then the total tardiness of three J_2^1 s is

$$3(p_1^2 + p_1^3 + p_2^2 - d_2^1) > 3(M_1 - 2B) > T^*.$$

If both $\{J_1^2, J_1^3\}$ and $\{J_2^2, J_2^3\}$ exist, the total tardiness of J_1^2 and J_2^2 is

$$2(p_1^1 + p_1^3) + p_2^1 + p_2^3 - d_1^2 - d_2^2 > 3M_1 > T^*.$$

So the four batches of types 1 and 2 must be: $\{J_1^1, J_1^2\}$, $\{J_1^3\}$, $\{J_2^1\}$, $\{J_2^2, J_2^3\}$; or $\{J_1^1\}$, $\{J_1^2, J_1^3\}$, $\{J_2^1, J_2^2\}$, $\{J_2^3\}$.

Suppose that the conclusion in property (iv) is true for each $i = 1, 2, \ldots, k-1$. Then, the start time of the first batch of types $2i-1$ and $2i$ is not less than $2\sum_{j=1}^{2i-2} L_j + 5\sum_{j=1}^{i-1} M_j$. If the four batches of types $2i-1$ and $2i$ are of Pattern 2112, then the tardiness of J_{2i}^2 is at least $2M_i$; if they are of Pattern 1221, then the tardiness of J_{2i-1}^2 is at least $2M_i$. Therefore, the jobs of types $1, 2, \ldots, 2k-2$ have total tardiness of at least $2\sum_{i=1}^{k-1} M_i$, which implies the jobs of types $2k-1, 2k, \ldots, 2t+1$ have total tardiness of at most $2\sum_{i=k}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + B$. Noticing that the start time of the first batch of types $2k-1$ and $2k$ will not be less than $2\sum_{i=1}^{2k-2} L_i + 5\sum_{i=1}^{k-1} M_i$, we can prove by an analysis similar to that for types 1 and 2 that if property (iv) does not hold for k, the jobs of types $2k-1$ and $2k$ will have total tardiness larger than $2\sum_{i=k}^{t} M_i + \sum_{i=1}^{t}(t-i)a_i + B$, which leads to a contradiction. □

Let I_1 be the set of indices k ($1 \le k \le t$) such that the four batches of types $2k-1$ and $2k$ are of Pattern 2112. Let $I_2 = I \setminus I_1$, where $I = \{1, 2, \ldots, t\}$. A schedule with properties (i)-(iv) must contain $4t+1$ batches in the following form:

$$(\mathcal{B}_{4k-3}, \mathcal{B}_{4k-2}, \mathcal{B}_{4k-1}, \mathcal{B}_{4k}) = \begin{cases} \left(\{J_{2k-1}^1, J_{2k-1}^2\}, \{J_{2k-1}^3\}, \{J_{2k}^1\}, \{J_{2k}^2, J_{2k}^3\}\right), & k \in I_1 \\ \left(\{J_{2k-1}^1\}, \{J_{2k-1}^2, J_{2k-1}^3\}, \{J_{2k}^1, J_{2k}^2\}, \{J_{2k}^3\}\right), & k \in I_2 \end{cases}$$

$$\mathcal{B}_{4t+1} = \{J_{2t+1}^1\}.$$

The tardiness of each job of types $1, 2, \ldots, 2t$ in the schedule is given by Lemma 2.

Lemma 2 *For each $k \in I_1$, J_{2k}^2 is the only tardy job in $\mathcal{B}_{4k-3}$, $\mathcal{B}_{4k-2}$, $\mathcal{B}_{4k-1}$ and $\mathcal{B}_{4k}$, and its tardiness is*

$$2M_k + (t-k+1)a_k + \sum\{a_i |\, i < k\,, i \in I_2\}.$$

For each $k \in I_2$, J_{2k-1}^2 is the only tardy job in $\mathcal{B}_{4k-3}$, $\mathcal{B}_{4k-2}$, $\mathcal{B}_{4k-1}$ and $\mathcal{B}_{4k}$, and its tardiness is

$$2M_k + \sum\{a_i |\, i < k\,, i \in I_2\}.$$

Proof By property (iv), the total processing time of batches $\mathcal{B}_{4k-3}$, $\mathcal{B}_{4k-2}$, $\mathcal{B}_{4k-1}$ and $\mathcal{B}_{4k}$ is $2(L_{2k-1} + L_{2k}) + 5M_k$ if $k \in I_1$, or $2(L_{2k-1} + L_{2k}) + 5M_k + a_k$ if $k \in I_2$. Then the start time of batch $\mathcal{B}_{4k-3}$ is equal to

$$2\sum_{i=1}^{2k-2} L_i + 5\sum_{i=1}^{k-1} M_i + \sum\{a_i |\, i < k\,, i \in I_2\}.$$

By computation, it is easy to verify that if $k \in I_1$, $\mathcal{B}_{4k-3}$, $\mathcal{B}_{4k-2}$ and $\mathcal{B}_{4k-1}$ contain no tardy jobs; if $k \in I_2$, $\mathcal{B}_{4k-3}$, $\mathcal{B}_{4k-1}$ and $\mathcal{B}_{4k}$ contain no tardy jobs. For $k \in I_1$, the completion time of $\mathcal{B}_{4k} = \{J^2_{2k}, J^3_{2k}\}$ is

$$2\sum_{i=1}^{2k} L_i + 5\sum_{i=1}^{k} M_i + \sum\{a_i |\, i < k\,, i \in I_2\}\,.$$

Thus, J^3_{2k} is on-time and J^2_{2k} has tardiness of $2M_k + (t-k+1)a_k + \sum\{a_i |\, i < k\,, i \in I_2\}$. For $k \in I_2$, the completion time of $\mathcal{B}_{4k-2} = \{J^2_{2k-1}, J^3_{2k-1}\}$ is

$$2\sum_{i=1}^{2k-1} L_i + 5\sum_{i=1}^{k-1} M_i + 2M_k + \sum\{a_i |\, i < k\,, i \in I_2\}\,.$$

Thus, J^3_{2k-1} is on-time and J^2_{2k-1} has tardiness of $2M_k + \sum\{a_i |\, i < k\,, i \in I_2\}$. □

Lemma 3 *Let σ be a schedule with properties (i)-(iv). Then, $T(\sigma) \le T^*$ if and only if $\sum_{k\in I_1} a_k = \sum_{k\in I_2} a_k = B$.*

Proof By Lemma 2, we have

$$\begin{aligned} T(\sigma) &= \sum_{k=1}^{t} \Big(2M_k + \sum\{a_i |\, i < k\,, i \in I_2\}\Big) \\ &\quad + \sum_{k\in I_1} (t-k+1)a_k + 3\max\left\{0,\, \sum_{k\in I_2} a_k - B\right\}, \end{aligned}$$

where the third term is the total tardiness of three J^1_{2t+1} s. Since

$$\sum_{k=1}^{t} \sum\{a_i |\, i < k\,, i \in I_2\} = \sum_{i\in I_2} (t-i)a_i\,,$$

it holds that

$$T(\sigma) = 2\sum_{k=1}^{t} M_k + \sum_{k=1}^{t} (t-k)a_k + \sum_{k\in I_1} a_k + 3\max\left\{0,\, \sum_{k\in I_2} a_k - B\right\}.$$

Then, $T(\sigma) \le T^*$ if and only if $\sum_{k\in I_1} a_k \le B$ and $\sum_{k\in I_1} a_k + 3\sum_{k\in I_2} a_k \le 4B$. Noticing that $\sum_{k\in I_1} a_k + \sum_{k\in I_2} a_k = 2B$, we have completed the proof. □

We now prove the main result of this section.

Theorem 1 *$B(\infty)||\sum T_j$ is binary NP-hard.*

Proof If $\mathcal{Q}$ has a schedule σ such that $T(\sigma) \le T^*$, then we may require or prove that σ possesses properties (i)-(iv). It follows from Lemma 3 that the PARTITION instance $\mathcal{P}$ has a solution $\{I_1, I_2\}$. Conversely, if the PARTITION instance $\mathcal{P}$ has a solution $\{I_1, I_2\}$, we simply construct a schedule with properties (i)-(iv). It again follows from Lemma 3 that the constructed schedule has total tardiness no larger than T^*. □

3 Pseudopolynomial solvability for problems with job release dates and regular objectives

In this section, we develop a pseudopolynomial time algorithm for the general problems $B(\infty)|\, r_j \,|\sum f_j$ and $B(\infty)|\, r_j \,|f_{\max}$. The algorithm is based on the following observation: there exists an optimal schedule in which if the longest job is started at time t, then all the jobs released at or before t should be started at or before t and all the jobs released after t should be started after t.

Let α and γ be the job index sequences such that $r_{\alpha(1)} \leq r_{\alpha(2)} \leq \cdots \leq r_{\alpha(n)}$ and $p_{\gamma(1)} \leq p_{\gamma(2)} \leq \cdots \leq p_{\gamma(n)}$, respectively. Let $\alpha(i,j) = \{\alpha(i), \alpha(i+1), \cdots, \alpha(j)\}$ and $\gamma(i,j) = \{\gamma(i), \gamma(i+1), \cdots, \gamma(j)\}$. Let $J(i_1, i_2; k)$ denote the subset of jobs with indices in $\alpha(i_1, i_2) \cap \gamma(1, k)$. Note that the number of such subsets is $O(n^3)$. The main idea of our algorithm is to schedule the jobs among $J(i_1, i_2; k_1) \cup \{J_{\gamma(k_2)}\}$ $(k_1 < k_2)$ into a given interval such that $J_{\gamma(k_2)}$ is completed at the end of the interval and the objective value of the subschedule is minimized.

To simplify the exposition, we introduce an auxiliary job J_{n+1} with $r_{n+1} = r_{\alpha(n)} + \sum_{j=1}^{n} p_j$, $p_{n+1} = p_{\gamma(n)}$ and $f_{n+1}(t) \equiv 0$. It is easy to see that J_{n+1} should be scheduled at the end of an optimal schedule.

3.1 Problem $B(\infty)|\, r_j \,|\sum f_j$

Let $F(i_1, i_2; k_1, k_2; x, y)$ $(k_1 < k_2)$ denote the minimum objective value when scheduling the jobs among $J(i_1, i_2; k_1) \cup \{J_{\gamma(k_2)}\}$ into the interval $[x, y]$, subject to the constraint that $J_{\gamma(k_2)}$ is completed at time y. If $J(i_1, i_2; k_1) = \emptyset$, then

$$F(i_1, i_2; k_1, k_2; x, y) = \begin{cases} f_{\gamma(k_2)}(y), & \text{if } \max\{r_{\gamma(k_2)},\, x\} \leq y - p_{\gamma(k_2)} \\ +\infty, & \text{otherwise.} \end{cases}$$

Generally, $F(i_1, i_2; k_1, k_2; x, y)$ can be computed recursively.

(i) If $\gamma(k_1) \notin \alpha(i_1, i_2)$, then $J(i_1, i_2; k_1) = J(i_1, i_2; k_1 - 1)$ and we have

$$F(i_1, i_2; k_1, k_2; x, y) = F(i_1, i_2; k_1 - 1, k_2; x, y)\,.$$

(ii) If $\gamma(k_1) \in \alpha(i_1, i_2)$ and $r_{\gamma(k_1)} > y - p_{\gamma(k_2)}$, then $J_{\gamma(k_1)}$ cannot be scheduled in $[x, y]$, and hence $F(i_1, i_2; k_1, k_2; x, y) = +\infty$.

(iii) If $\gamma(k_1) \in \alpha(i_1, i_2)$ and $r_{\gamma(k_1)} \leq y - p_{\gamma(k_2)}$, we have

$$F(i_1, i_2; k_1, k_2; x, y) = \min \left\{ \begin{array}{c} F(i_1, i_2; k_1 - 1, k_2; x, y) + f_{\gamma(k_1)}(y) \\ \min\limits_{\max\{r_{\gamma(k_1)}, x\} \leq t \leq y - p_{\gamma(k_1)} - p_{\gamma(k_2)}} H(t) \end{array} \right\},$$

where the first term is taken if $J_{\gamma(k_1)}$ is processed in the batch including $J_{\gamma(k_2)}$, and $H(t) = H_1(t) + H_2(t)$ in the second term is taken if $J_{\gamma(k_1)}$ is started at time t. We also note that the first term will not be taken when $k_2 = n + 1$, i.e., J_{n+1} will occupy the last batch alone.

$H_1(t)$ is the contribution to $H(t)$ of jobs processed in $[x, t + p_{\gamma(k_1)}]$. It is reasonable to assume that none of the jobs with release dates no more than t in $J(i_1, i_2; k_1 - 1)$ is scheduled after the batch including $J_{\gamma(k_1)}$ since they have processing times no more than $p_{\gamma(k_1)}$. Let i'_2 $(i_1 \le i'_2 \le i_2)$ be the maximum index satisfying $r_{\alpha(i'_2)} \le t$. Then,

$$H_1(t) = F(i_1, i'_2; k_1 - 1, k_1; x, t + p_{\gamma(k_1)}) .$$

$H_2(t)$ is the contribution to $H(t)$ of jobs processed in $[t + p_{\gamma(k_1)}, y]$. It obviously holds that

$$H_2(t) = F(i'_2 + 1, i_2; k_1 - 1, k_2; t + p_{\gamma(k_1)}, y) .$$

By computing $F(1, n; n, n + 1; r_{\alpha(1)}, r_{n+1} + p_{n+1})$ recursively, we can obtain the optimal objective value. An optimal schedule can be found by backtracking.

Now we analyse the complexity of the recursion. The size of the domain of function $F(i_1, i_2; k_1, k_2; x, y)$ is $O(n^4P^2)$, where $P = r_{\alpha(n)} + \sum_{i=1}^{n} p_i$. To obtain the vaule of each $F(i_1, i_2; k_1, k_2; x, y)$, we need at most $O(P)$ time (see cases (i)-(iii)). Thus, the complexity of the recursion is at most $O(n^4P^3)$, which is pseudopolynomial.

3.2 Problem $B(\infty)|\, r_j \,|f_{\max}$

For the problem $B(\infty)|\, r_j \,|f_{\max}$, our analysis is similar to that for the problem $B(\infty)|\, r_j \,|\sum f_j$ except that in case (iii),

$$F(i_1, i_2; k_1, k_2; x, y) = \min \left\{ \begin{array}{l} \max\left\{F(i_1, i_2; k_1 - 1, k_2; x, y),\ f_{\gamma(k_1)}(y)\right\} \\ \min\limits_{\max\{r_{\gamma(k_1)}, x\} \le t \le y - p_{\gamma(k_1)} - p_{\gamma(k_2)}} H(t) \end{array} \right\},$$

where $H(t) = \max\{H_1(t), H_2(t)\}$.

4 Complexity status of unbounded batch machine problems

In this paper, we have addressed the complexity of scheduling an unbounded batch machine. Our results show that all problems with regular objectives are pseudopolynomially solvable even if the jobs have different release dates. This is distinct from the bounded batch machine and the classical single machine scheduling problems, most of which with different release dates are unary NP-hard.

Finally, we present a summary of the complexity status of various unbounded batch machine scheduling problems. Following Brucker and Knust [3], we use the terminology: maximal polynomially solvable, maximal pseudopolynomially solvable and minimal NP-hard.

- maximal polynomially solvable:

  $B(\infty)||\sum U_j$ (Brucker et al. [2])

  $B(\infty)|r_j|f_{\max}$ with a fixed number of r_j or p_j (Cheng et al. [4])

  $B(\infty)|r_j|\sum w_jC_j$ with a fixed number of r_j or p_j (Deng and Zhang [5])

- maximal pseudopolynomially solvable:

  $B(\infty)|r_j|f_{\max}$ (this paper)

  $B(\infty)|r_j|\sum f_j$ (this paper)

- minimal NP-hard:

  $B(\infty)||\sum T_j$ (this paper)

  $B(\infty)||\sum w_jU_j$ (Brucker et al. [2])

  $B(\infty)|r_j|L_{\max}$ (Cheng et al. [4])

  $B(\infty)|r_j|\sum w_jC_j$ (Deng and Zhang [5])

- Open:

  $B(\infty)|r_j|\sum C_j$

Acknowledgment

This research is supported in part by The Hong Kong Polytechnic University under grant number G-YW59. The first author is also supported by the National Natural Science Foundation of China under grant number 10101007.

References

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

[2] P. Brucker, A. Gladky, H. Hoogeveen, M.Y. Kovalyov, C.N. Potts, T. Tautenhahn, S.L. van de Velde, Scheduling a batching machine, Journal of Scheduling 1 (1998) 31–54.

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

[4] T.C.E. Cheng, Z. Liu, W. Yu, Scheduling jobs with release dates and deadlines on a batch processing machine, IIE Transactions 33 (2001) 685–690.

[5] X. Deng, Y. Zhang, Minimizing mean response time in batch processing system, Lecture Notes in Computer Science 1627 (1999) 231–240.

[6] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling: algorithms and complexity, in Graves, S.C., Rinnooy Kan, A.H.G. and Zipkin, P.H. (eds.), Handbooks in Operations Research and Management Science, Volume 4, Logistics of Production and Inventory, North Holland, Amsterdam, 1993, 445–522.

[7] C.-Y. Lee, R. Uzsoy, Minimizing makespan on a single batch processing machine with dynamic job arrivals, International Journal of Production Research 37 (1999) 219–236.

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

[9] C.-L. Li, C.-Y. Lee, Scheduling with agreeable release times and due dates on a batch processing machine, European Journal of Operational Research 96 (1997) 564–569.

[10] Z. Liu, W. Yu, Scheduling one batch processor subject to job release dates, Discrete Applied Mathematics 105 (2000) 129–136.

[11] C.N. Potts, M.Y. Kovalyov, Scheduling with batching: a review, European Journal of Operational Research 120 (2000) 228–249.