Multi-agent scheduling on a single machine with max-form criteria

T.C.E. Cheng[1,*], **C.T. Ng**[1] *and* **J.J. Yuan**[2]

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

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

Abstract: We consider multi-agent scheduling on a single machine, where the objective functions of the agents are of the max-form. For the feasibility model, we show that the problem can be solved in polynomial time even when the jobs are subject to precedence restrictions. For the minimality model, we show that the problem is strongly NP-hard in general, but can be solved in pseudo-polynomial time when the number of agents is a given constant. We then identify some special cases of the minimality model that can be solved in polynomial time.

Keywords: Scheduling, Multi-agent, Fixed jobs.

1 Introduction and Problem Formulation

The following single-machine multi-agent scheduling problem was introduced by Agnetis et al. (2004) and Baker and Smith (2003). There are several agents, each with a set of jobs. The agents have to schedule their jobs on a common processing resource, i.e., a single machine, and each agent wishes to minimize an objective function that depends on the completion times of his own set of jobs. The problem is either to find a schedule that minimizes a combination of the agents' objective functions or to find a schedule that satisfies each agent's requirement for his own objective function.

Scheduling is in fact concerned with the allocation of limited resources over time. Scheduling problems involving multiple customers (agents) competing for a common processing resource arise naturally in many settings. For example, in industrial management, the multi-agent scheduling problem is formulated as a sequencing game, where the objective is to devise some mechanisms to encourage the agents to cooperate with a view

*Corresponding author

to minimizing the overall cost (see, for example, Curiel et al. (1989), and Hamers et al. (1995)). In project scheduling, the problem is concerned with negotiation to resolve conflicts whenever the agents find their own schedules unacceptable (Kim et al. (1999)). In telecommunication services, the problem is to do with satisfying the service requirements of individual agents, who compete for the use of a commercial satellite to transfer voice, image and text files for their clients (Schultz et al. (2002)). Following Agnetis et al. (2004) and Baker and Smith (2003), and intending to generalize their results on two-agent scheduling, we focus this study on deriving feasible or optimal solutions for different scenarios of competition among the agents, and examining the computational complexity issues of some intractable cases of the single-machine multi-agent scheduling problem.

Now we define the multi-agent scheduling in terms of common scheduling terminology. We have m families of jobs $\mathcal{J}^{(1)}, \mathcal{J}^{(2)}, ..., \mathcal{J}^{(m)}$, where, for each i with $1 \le i \le m$, $\mathcal{J}^{(i)} = \{J_1^{(i)}, J_2^{(i)}, ..., J_{n_i}^{(i)}\}$. The jobs in $\mathcal{J}^{(i)}$ are called the i-th agent's jobs. Each job $J_j^{(i)}$ has a positive integral processing time $p_j^{(i)}$, a positive integral due date $d_j^{(i)}$, and a positive weight $w_j^{(i)}$. All the jobs have zero release dates. The jobs will be processed on a single machine starting at time zero without overlapping and idle time between them. A schedule is a sequence of the jobs that specifies the processing order of the jobs on the machine. Under a schedule σ, the completion time of job $J_j^{(i)}$ is denoted by $C_j^{(i)}(\sigma)$; job $J_j^{(i)}$ is called tardy if $C_j^{(i)}(\sigma) > d_j^{(i)}$; $U_j^{(i)}(\sigma) = 1$ if $J_j^{(i)}$ is tardy and zero otherwise; the lateness of $J_j^{(i)}$ is defined as $L_j^{(i)}(\sigma) = C_j^{(i)}(\sigma) - d_j^{(i)}$; the tardiness of $J_j^{(i)}$ is defined as $T_j^{(i)}(\sigma) = \max\{0, L_j^{(i)}(\sigma)\}$; the maximum lateness of the i-th agent is defined as $L_{\max}^{(i)}(\sigma) = \max_{1 \le j \le n_i} L_j^{(i)}(\sigma)$; and the maximum tardiness of the i-th agent is defined as $T_{\max}^{(i)}(\sigma) = \max_{1 \le j \le n_i} T_j^{(i)}(\sigma)$. For each job $J_j^{(i)}$, let $f_j^{(i)}(\cdot)$ be a nondecreasing function of the completion time of job $J_j^{(i)}$ (such an objective function is called *regular* in the scheduling literature). We assume in this paper that $f_j^{(i)}(\cdot)$ can be evaluated in constant time. The i-th agent's objective function $F^{(i)}(\sigma)$ takes either one of the following two forms:

$$\text{max-form} \quad F^{(i)}(\sigma) = \max_{1 \le j \le n_i} f_j^{(i)}(C_j^{(i)}(\sigma)),$$

$$\text{sum-form} \quad F^{(i)}(\sigma) = \sum_{1 \le j \le n_i} f_j^{(i)}(C_j^{(i)}(\sigma)).$$

Throughout this paper, each agent's objective function is assumed to be of the max-form.

The scheduling problem studied in this paper includes the following two models:

• **Feasibility model:** $1|prec|F^{(i)} \le Q_i, 1 \le i \le m$. In this model the goal is to find a feasible schedule σ that satisfies $F^{(i)}(\sigma) \le Q_i$, $1 \le i \le m$.

• **Minimality model:** $1|prec|\sum_{1 \le i \le m} F^{(i)}$. In this model the goal is to find a schedule σ that minimizes $\sum_{1 \le i \le m} F^{(i)}(\sigma)$.

In the above scheduling models, *prec* denotes that the jobs are subject to precedence restrictions. When the jobs are independent, i.e., there are no precedence relations among the jobs, we leave the second field of the model description blank.

The feasibility model of multi-agent scheduling was first studied by Agnetis et al. (2004), and the minimality model of multi-agent scheduling was first studied by Baker and Smith (2003). This paper seeks to present general methods to treat the single-machine multi-agent scheduling problem. The results presented in this paper embrace some of the results on single-machine two-agent scheduling reported in the literature (Agnetis et al. (2004), Baker and Smith (2003), and Yuan et al. (2005)).

The work of this paper is related to scheduling with fixed jobs. This scheduling problem was first introduced by Scharbrodt et al. (1999). In this model we have two types of jobs: free jobs and fixed jobs. The fixed jobs are already fixed in the schedule. The intervals occupied by the fixed jobs can be referred to as unavailability intervals of the machine. The remaining free jobs, which consist of the jobs of all the agents, are to be assigned to the remaining time slots of the machine in such a way that they do not overlap with one another nor with the fixed jobs. In our discussion we allow preemption of the free jobs.

The adding of fixed jobs in the scheduling model is motivated by the need to perform maintenance on the machine in real-life settings. Due to the need for maintenance, the machine is unavailable during certain periods of time over the scheduling horizon. Usually machine maintenance operations are planned in advance and so the unavailable periods on the machine are pre-specified (i.e., fixed). Kovalyov et al. (2007) provided a detailed survey on fixed interval scheduling. In deriving the results in this paper, we use a technique to transform a free job into a fixed job. Hence, we use the terminology "fixed jobs" as in Scharbrodt et al. (1999).

The feasibility model of single-machine multi-agent preemptive scheduling with fixed jobs is denoted by $1|\mathit{prec; fix; pmtn}|F^{(i)} \leq Q_i, 1 \leq i \leq m$, and the minimality version is denoted by $1|\mathit{prec; fix; pmtn}|\sum_{1\leq i\leq m} F^{(i)}$. Here *fix* denotes the constraint due to the presence of the fixed jobs, and *pmtn* means that the free jobs are preemptively scheduled. Throughout this paper, n^* and n are used to denote the number of fixed jobs and the number of free jobs, respectively. Then we have $n = n_1 + n_2 + \cdots + n_m$.

Since the objective functions are assumed to be regular, we need only to consider regular schedules, where a schedule is called regular if under the schedule there is no idle time on the machine before the completion time of the last free job.

The idea in this paper comes partially from Agnetis et al. (2004), Baker and Smith (2003), and Yuan et al. (2005). This paper is organized as follows. In Section 2 we show that the feasibility model can be solved in polynomial time. In Section 3 we show that the minimality model is strongly NP-hard. In Section 4 we show that, when the number of agents is a constant, the minimality model can be solved in pseudo-polynomial time. In Section 5 we give some polynomial-time solvable sub-cases of the minimality model.

2 Polynomial-time algorithm for the feasibility model

Consider the feasibility problem $1|prec; fix; pmtn|F^{(i)} \le Q_i, 1 \le i \le m$, where each $F^{(i)}$ is of the max-form $F^{(i)}(\pi) = \max_{1\le j\le n_i} f_j^{(i)}(\pi)$. We present an $O(n^* + n^2)$ time algorithm. When the free jobs are independent, the running time of the algorithm is $O(n^* + n\log n)$. The idea of our discussion comes from Lawler's (1973) algorithm for the single machine scheduling $1|prec|f_{\max}$. The same procedure was also used in Agnetis et al. (2004).

The first technical issue to deal with is that, if a block of free jobs of total length l is to be processed preemptively under a regular schedule on the machine with fixed jobs, how do we determine the completion time $\Delta(l)$ of the last free job in the block?

Let the n^* intervals occupied by the fixed jobs be $[s_1, t_1), [s_2, t_2), ..., [s_{n^*}, t_{n^*})$, where $0 \le s_1 < t_1 \le s_2 < t_2 \le \cdots \le s_{n^*} < t_{n^*}$. Note that the sorting of these intervals needs $O(n^* \log n^*)$ time. We further define $t_0 = 0$ for convenience. We first calculate all $FB_j = \sum_{1\le i\le j}(t_i - s_i)$ in $O(n^*)$ time, where FB_j is the total length of the first j intervals occupied by the fixed jobs. Clearly, $\Delta(l) > t_j$ if and only if $l + FB_j > t_j$. Hence, we have

Lemma 2.1 The value $\Delta(l)$ can be calculated in $O(n^*)$ time by the formula $\Delta(l) = l + FB_{j^*}$, where j^* is the maximum index j such that $l + FB_j > t_j$. □

For each job $J_j^{(i)}$, we define its deadline as

$$D_j^{(i)} = \max\{C : C \text{ is a positive integer and } f_j^{(i)}(C) \le Q_i\}.$$

In the case that $D_j^{(i)} > \Delta(P)$, we re-define $D_j^{(i)} = P$, where P is the sum of the length of all the free jobs. As stated in Agnetis et al. (2004), if the inverse function of $f_j^{(i)}$ is available, the deadline $D_j^{(i)}$ can be computed in constant time; otherwise this requires $O(\log P_j^{(i)})$ time. We assume the former case for every job.

Now we denote the set of all the free jobs by $\mathcal{J} = \{J_1, J_2, ..., J_n\}$. That is, $\mathcal{J}$ is the union of the free jobs from each agent. The processing time and deadline of J_j are denoted by p_j and D_j, respectively. We write $J_i \prec J_j$ if job J_i must be processed before job J_j.

If J_i and J_j are two jobs such that $J_i \prec J_j$, then the completion time C_i of J_i must be less than or equal to $C_j - p_j$ under any feasible schedule. Modifying the deadline of job J_i by setting $D_i := \min\{D_i, D_j - p_j\}$, we lose nothing, but we obtain a beneficial relation $D_i < D_j$. Hence, we can recursively modify the deadlines of the jobs such that, for each pair of jobs J_i and J_j, if $J_i \prec J_j$, then $D_i < D_j$. This procedure can be performed in $O(n^2)$ time using the standard "Algorithm Modify d_j" given by Brucker (2001).

Clearly, the feasibility problem has a solution if and only if the regular schedule obtained by the EDD rule on the modified deadline meets every job's deadline. Suppose that the deadlines have been modified. Then we re-label the free jobs in the EDD order, i.e., $D_1 \le D_2 \le \cdots \le D_n$, which will take $O(n \log n)$ time. Let π be the regular preemptive schedule obtained by the free job sequence $(J_1, J_2, ..., J_n)$. Then the feasibility problem has a solution if and only if $\Delta(p_1 + p_2 + \cdots + p_j) \le D_j$ for $1 \le j \le n$.

Note that all the values $P(j) = p_1 + p_2 + \cdots + p_j$, $1 \le j \le n$, can be calculated in $O(n)$ time recursively, and all the values $\Delta(P(j))$, $1 \le j \le n$, can be calculated in $O(n^* + n)$ time recursively by the following dynamic programming recursion: Let $k_0 = 0$ and, for j from 1 to n, let $k_j \ge k_{j-1}$ be the maximum value such that $P(j) + FB_{k_j} > t_{k_j}$, and then set $\Delta(P(j)) = P(j) + FB_{k_j}$. From this, we conclude the following result.

Theorem 2.2 The feasibility problem $1|\mathit{prec; fix; pmtn}|F^{(i)} \le Q_i, 1 \le i \le m$ can be solved in $O(n^* \log n^* + n^2)$ time.

When the set of fixed jobs is empty, the problem under study is equivalent to $1| \prec |F^{(i)} \le Q_i, 1 \le i \le m$. Hence, we have

Corollary 2.3 $1|\mathit{prec}|F^{(i)} \le Q_i, 1 \le i \le m$ can be solved in $O(n^2)$ time.

When the free jobs are independent, the procedure for modifying the deadlines can be omitted. Hence, we have

Theorem 2.4 $1|\mathit{fix; pmtn}|F^{(i)} \le Q_i, 1 \le i \le m$ can be solved in $O(n^* \log n^* + n \log n)$ time.

Corollary 2.5 $1||F^{(i)} \le Q_i, 1 \le i \le m$ can be solved in $O(n \log n)$ time.

3 NP-hardness for the minimality model

Theorem 3.1 Both $1||\sum_{1\le i\le m} L^{(i)}_{\max}$ and $1||\sum_{1\le i\le m} T^{(i)}_{\max}$ are binary NP-hard.

Proof Recall that, by Du and Leung (1990), the feasibility scheduling problem $1||\sum T_j \le Y$ is binary NP-complete.

Let us be given an instance I of $1||\sum T_j \le Y$: $(p_1, p_2, ..., p_n; d_1, d_2, ..., d_n; Y)$, where p_i and d_i are the processing time and due date of the i-th job J_i, $1 \le i \le n$, and $Y \ge 0$ is the threshold value of the total tardiness. The decision asks whether there is a schedule σ for the n jobs such that $\sum_{1\le i\le n} T_i(\sigma) \le Y$.

Write $P = \sum_{1\le i\le n} p_i$. By the proof of the NP-completeness of $1||\sum T_j \le Y$ in Du and Leung (1990), we can further assume that $0 \le d_i \le P$ for each i, and $Y \le nP$. We construct an instance I^* of the feasibility scheduling problem $1||\sum_{1\le i\le m} L^{(i)}_{\max} \le Q$ as follows.

- n agents with each agent having exactly two jobs, i.e., $\mathcal{J}^{(i)} = \{J^{(i)}_1, J^{(i)}_2\}$, $1 \le i \le n$.
- Processing times of the jobs are defined by $p^{(i)}_1 = p_i$ and $p^{(i)}_2 = X$, $1 \le i \le n$, where $X = nP + Y + 1$. The jobs $J^{(i)}_2$, $1 \le i \le n$, are called large jobs, and the jobs $J^{(i)}_1$, $1 \le i \le n$, are called small jobs.
- Due dates of the jobs are defined by $d^{(i)}_1 = d_i$ and $d^{(i)}_2 = P + iX$, $1 \le i \le n$.
- Threshold value is defined by $Q = Y$, and the decision asks whether there is a schedule π such that $\sum_{1\le i\le m} L^{(i)}_{\max}(\pi) \le Q$.

The above construction can be done in polynomial time. We show in the following that I is feasible if and only if I^* is feasible.

If there is a feasible schedule π for I^*, we claim that: $C_1^{(i)}(\pi) \leq P$, and $C_2^{(i)}(\pi) = d_2^{(i)} = P + iX, \; 1 \leq i \leq n$.

We notice that $L_{\max}^{(i)}(\pi) \geq -P$ for each i, since $d_i \leq P$. If $C_1^{(x)}(\pi) > P$ for a certain x, then there must be at least one large job completed before $J_1^{(x)}$, and thus $C_1^{(x)}(\pi) \geq X$. This means that $L_{\max}^{(x)}(\pi) \geq X - P$. Consequently, $\sum_{1\leq i\leq m} L_{\max}^{(i)}(\pi) \geq X - nP = Y + 1 > Q$, contradicting the assumption that π is feasible. So, under π, any small job is processed before any large job. This also means that the completion times of the large jobs are $P + iX$, $1 \leq i \leq n$. If possible, let y, $1 \leq y \leq n$, be the minimum such that $C_2^{(y)}(\pi) > d_2^{(y)} = P + yX$. Then, $L_{\max}^{(y)}(\pi) \geq X$. Again, this implies $\sum_{1\leq i\leq m} L_{\max}^{(i)}(\pi) > Q$, contradicting the assumption that π is feasible. We conclude that the claim holds.

The above claim means that under the feasible schedule π each large job completes at its due dates, and $L_{\max}^{(i)}(\pi) = T_{\max}^{(i)}(\pi) = T_1^{(i)}$, $1 \leq i \leq n$. Since the small jobs in I^* correspond one-to-one to the jobs in I with the same processing times and due dates, we deduce that I is feasible if and only if I^* is feasible. We conclude that $1||\sum_{1\leq i\leq m} L_{\max}^{(i)} \leq Q$ is NP-complete.

The above proof is also valid for $1||\sum_{1\leq i\leq m} T_{\max}^{(i)} \leq Q$. Hence, the result follows. □

The classical scheduling problem $1||\sum w_j T_j$ is strongly NP-hard. One of the well-known proofs for the strongly NP-hardness of this problem was given by Lawler (1977) by using a reduction from the strongly NP-complete 3-Partition problem to the decision version of the scheduling problem. In the constructed instance of $1||\sum w_j T_j$ in Lawler (1977), the jobs are of two families $\mathcal{J}^{(1)}$ and $\mathcal{J}^{(2)}$: each job $J_j \in \mathcal{J}^{(1)}$ has a due date $d_j = 0$ and so $T_j = C_j$ in any schedule; and each job $J_j \in \mathcal{J}^{(2)}$ has a positive due date d_j and a sufficiently large weight w_j so that $T_j = 0$ in any feasible schedule. This just implies that the feasibility problem $1||\sum w_j^{(1)} C_j^{(1)} \leq Y : L_{\max}^{(2)} \leq 0$ is strongly NP-complete. Note that Agnetis et al. (2004) have proved the ordinary NP-completeness of the same problem.

Theorem 3.2 $1||\sum_{1\leq i\leq m} \max_{1\leq j\leq n_i} w_j^{(i)} C_j^{(i)}$ is strongly NP-hard.

Proof Let us be given an instance I of $1||\sum w_j^{(1)} C_j^{(1)} \leq Y : L_{\max}^{(2)} \leq 0$:

$$(p_1, , ..., p_n; w_1, ..., w_k; d_{k+1}, ..., d_n; Y),$$

where $J_1, ..., J_k$ are the first agent's jobs and $J_{k+1}, ..., J_n$ are the second agent's jobs; $p_j \geq 1$ is the integral processing time of job J_j, $1 \leq j \leq n$; $w_j \geq 1$ is the weight of job J_j, $1 \leq j \leq k$; $d_j \geq 1$ is the integral due date of job J_j, $k+1 \leq j \leq n$; and $Y > 0$ is the threshold value of the total weighted completion time of the first agent's jobs. The decision asks whether there is a schedule σ on the machine such that $L_{\max}^{(2)}(\sigma) \leq 0$ and $\sum_{1\leq i\leq k} w_i C_i(\sigma) \leq Y$.

Since the completion time of each job cannot exceed $P = \sum_{1\leq j\leq n} p_j$, we can assume that $Y \leq WP$, where $W = \sum_{1\leq j\leq k} w_j$. It is also reasonable to suppose that $d_j \leq P$ for every J_j with $j > k$. We construct an instance I^* of the feasibility scheduling problem $1||\sum_{1\le i\le m}\max_{1\le j\le n_i} w_j^{(i)}C_j^{(i)} \le Q$ as follows.

• $k+1$ agents and $n+1$ jobs $J'_1, J'_2, ..., J'_{n+1}$ with J'_j, $1 \le j \le n$, corresponding to job J_j in I. For $1 \le i \le k$, the i-th agent has exactly one job J'_i. The job set of the $(k+1)$-th agent is defined by $\{J'_{k+1}, J'_{k+2}, ..., J'_{n+1}\}$.

• Processing times of the jobs are defined by $p'_j = p_j$ for $1 \le j \le n$ and $p'_{n+1} = M - P$, where $M = d_{\max}(Y+W) + Y + P$ with $d_{\max} = \max_{k+1\le j\le n} d_j$.

• Weights of the jobs are defined by $w'_i = 2w_i$ for $1 \le i \le k$, $w'_j = M/d_j$ for $k+1 \le j \le n$, and $w'_{n+1} = 1$.

• Threshold value is defined by $Q = 2Y + M$, and the decision asks whether there is a schedule π such that $\sum_{1\le i\le m}\max_{1\le j\le n_i} w_j^{(i)}C_j^{(i)} \le Q$.

The above construction can be done in polynomial time. We show in the following that I is feasible if and only if I^* is feasible.

Suppose I has a feasible schedule π. Then we obtain a corresponding sequence π^* of the jobs in I apart from J'_{n+1}. We insert job J'_{n+1} in the last position of π^* to obtain a schedule σ. It can be seen that $C'_j(\sigma) \le C_j(\pi)$ for $1 \le j \le n$, and $\max_{k+1\le j\le n+1} w'_jC'_j(\sigma) = M$. It follows that the objective value of the instance I^* is no more than $2Y + M = Q$.

Conversely, suppose that π is a feasible schedule of I^*. From the construction of I^*, we see that $\sum_{1\le j\le n+1} p'_j = P + M - P = M$, $2M > Q$ and $w'_j(d_j + 1) > 2Y + M = Q$ for $k+1 \le j \le n$. If, for some j with $1 \le j \le k$, J'_j is the last job in π, then $Q \ge w'_jC'_j(\pi) \ge 2M > Q$, a contradiction. If, for some j with $k+1 \le j \le n$, $C'_j(\pi) \ge d_j+1$, then $Q \ge w'_jC'_j(\pi) \ge w'_j(d_j + 1) > Q$, a contradiction again. It follows that the last job in π must be J'_{n+1}, and $C'_j(\pi) \le d_j$ for each j with $k+1 \le j \le n$. Hence, we must have $\max\{w'_jC'_j(\pi) : 1 \le j \le n+1\} = M$. Since the cost of π is no more than $Q = 2Y + M$, we deduce that $\sum_{1\le j\le k} w'_jC'_j(\pi) \le 2Y$.

Since J_j and J'_j have the same processing time for $1 \le j \le n$, π, restricted on the jobs J'_j, $1 \le j \le n$, can also be seen as a schedule for the instance I. Now, $w'_j = 2w_j$ for each j and $\sum_{1\le j\le k} w'_jC'_j(\pi) \le 2Y$ implies that $\sum_{1\le j\le k} w_jC_j(\pi) \le Y$. Consequently, π is a feasible schedule for I. The result follows. □

It is still open whether the problem $1||\sum_{1\le i\le m} L_{\max}^{(i)}$ is strongly NP-hard.

4 Pseudo-polynomial-time algorithms for given m

Suppose that $F^{(i)}$ is an integral regular function for each agent i. When m, the number of agents, is a given constant, even the general problem $1|prec; fix; pmtn|\sum_{1\le i\le m} F^{(i)}$ can be theoretically solved in pseudo-polynomial time provided that, for each agent i, the upper bound $UB(i)$ and the lower bound $LB(i)$ of $F^{(i)}$, with $LB(i) \le F^{(i)}(\pi) \le UB(i)$ for any regular schedule π, can be determined in pseudo-polynomial time, say $\mathcal{P}(i)$.

By the discussion of Section 2, for every set $\{Q_1, Q_2, ..., Q_m\}$ with $LB(i) \leq Q_i \leq UB(i)$, $1 \leq i \leq m$, the feasibility problem $1|prec; fix; pmtn|F^{(i)} \leq Q_i, 1 \leq i \leq m$ can be solved in $O(n^* \log n^* + n^2)$ time. By enumerating all the possibilities of $Q_1, Q_2, ..., Q_m$ and choosing the best one, we eventually solve the problem. Let $B(i) = UB(i) - LB(i) + 1$, $1 \leq i \leq m$. Then each Q_i has at most $B(i)$ choices. Hence, we have

Theorem 4.1 $1|prec; fix; pmtn|\sum_{1\leq i\leq m} F^{(i)}$ can be solved in $O(\sum_{1\leq i\leq m} \mathcal{P}(i) + (n^* \log n^* + n^2)\ \Pi_{1\leq i\leq m} B(i))$ time.

Similarly, we have

Theorem 4.2 $1|prec|\sum_{1\leq i\leq m} F^{(i)}$ can be solved in $O(\sum_{1\leq i\leq m} \mathcal{P}(i) + n^2\ \Pi_{1\leq i\leq m} B(i))$ time.

Theorem 4.3 $1|fix; pmtn|\sum_{1\leq i\leq m} F^{(i)}$ can be solved in $O(\sum_{1\leq i\leq m} \mathcal{P}(i) + (n^* \log n^* + n \log n)\ \Pi_{1\leq i\leq m} B(i))$ time.

Theorem 4.4 $1||\sum_{1\leq i\leq m} F^{(i)}$ can be solved in $O(\sum_{1\leq i\leq m} \mathcal{P}(i) + (n \log n)\ \Pi_{1\leq i\leq m} B(i))$ time.

When each $F^{(i)}$ is in a form familiar to us, the complexity can be reduced to a normal size. For example, if $F^{(i)} = \max_j w_j^{(i)} T_j^{(i)}$, then we can choose $UB(i) = \max_{1\leq j\leq n_i} w_j^{(i)} \times P$ and $LB(i) = 0$; in this case, $\mathcal{P}(i)$ can be omitted from the complexity.

5 Polynomial-time algorithms for special models

Let K be a subset of the free jobs, and let π be a schedule of $1|fix; pmtn|\sum_{1\leq i\leq m} F^{(i)}$. We use S_K and C_K to denote the minimum starting time, and the maximum completion time of the jobs in K under π, respectively. We say the jobs in K are consecutively processed in π if, under the schedule π, the machine has no idle time in $[S_K, C_K)$, and no free jobs other than that in K are processed in $[S_K, C_K)$.

The following observation, which holds for two-agent scheduling (see Baker and Smith (2003), and Yuan et al. (2005)), still holds for multi-agent scheduling.

Observation 5.1 For the problem $1|fix; pmtn|\sum_{1\leq i\leq m} F^{(i)}$, there is an optimal schedule σ such that,

(1) if there is a certain i such that $F^{(i)} = \max_{1\leq j\leq n_i} w^{(i)} C_j^{(i)}$, then the jobs in $\mathcal{J}^{(i)}$ are consecutively processed;

(2) if there is a certain i such that $F^{(i)} = \max_{1\leq j\leq n_i} w_j^{(i)} C_j^{(i)}$, then the jobs in $\mathcal{J}^{(i)}$ are processed in the maximum weight (MAX-W) order; and furthermore, the jobs with the same weights are consecutively processed; and

(3) if there is a certain i such that $F^{(i)} = w^{(i)} L_{\max}^{(i)}$, then the jobs in $\mathcal{J}^{(i)}$ are processed in the EDD order; and furthermore, the jobs with the same due dates are consecutively processed.

Proof By the job-shifting argument. □

Theorem 5.2 $1||\sum_{1\le i\le m} w^{(i)}C^{(i)}_{\max}$ can be solved in $O(n + m\log m)$ time.

Proof By Observation 5.1(1), we first combine the jobs in each $\mathcal{J}^{(i)}$ into a large job J_i with processing time $p_i = \sum_{1\le j\le n_i} p^{(i)}_j$ and weight $w^{(i)}$. As a result, the problem is transformed into the standard single-machine scheduling problem $1||\sum w^{(i)}C_i$, which can be solved by the SWPT rule in $O(m\log m)$ time. The result follows. □

In the following, we consider that, for each agent i, either $F^{(i)} = \max_{1\le j\le n_i} w^{(i)}_j C^{(i)}_j$ or $F^{(i)} = w^{(i)}L^{(i)}_{\max}$. By Observation 5.1, the i-th agent's jobs with the same weights ($F^{(i)} = \max_{1\le j\le n_i} w^{(i)}_j C^{(i)}_j$) or the same due dates ($F^{(i)} = w^{(i)}L^{(i)}_{\max}$) can be combined into a large job. Hence, in the following, we suppose that the i-th agent's jobs have distinct weights or due dates, according to $F^{(i)} = \max_{1\le j\le n_i} w^{(i)}_j C^{(i)}_j$ or $F^{(i)} = w^{(i)}L^{(i)}_{\max}$.

Furthermore, we suppose that, when $F^{(i)} = \max_{1\le j\le n_i} w^{(i)}_j C^{(i)}_j$, the jobs in $\mathcal{J}^{(i)} = \{J_1(i), J_2(i), ..., J_{n_i}(i)\}$ are indexed in the MAX-W order $w^{(i)}_1 < w^{(i)}_2 < \cdots < w^{(i)}_{n_i}$, and when $F^{(i)} = w^{(i)}L^{(i)}_{\max}$, the jobs in $\mathcal{J}^{(i)} = \{J^{(i)}_1, J^{(i)}_2, ..., J^{(i)}_{n_i}\}$ are indexed in the EDD order $d^{(i)}_1 < d^{(i)}_2 < \cdots < d^{(i)}_{n_i}$. By Observation 5.1 again, there is an optimal schedule such that, for each agent i, its jobs are processed in the order $(J^{(i)}_1, J^{(i)}_2, ..., J^{(i)}_{n_i})$. It follows that in an optimal schedule, the completion time of any job is of the form $\Delta(P(x_1, x_2, ..., x_m))$ for some $x_1, x_2, ..., x_m$ with $0 \le x_i \le n_i$, $1 \le i \le m$, where $P(x_1, x_2, ..., x_m) = \sum_{1\le i\le m}\sum_{1\le j\le x_i} p^{(i)}_j$ and $\Delta(\cdot)$ is the same as that in Section 2. Consequently, in an optimal schedule, the possible value of $F^{(i)}$ belongs to the set $R(i) = \{f^{(i)}_j(\Delta(P(x_1, x_2, ..., x_m)) : 0 \le x_i \le n_i\}$. Note that $R(i) = O(n_1 n_2 \cdots n_m)$ for each i.

By the discussion of Section 2, for every sequence $Q_1, Q_2, ..., Q_m$ with $Q_i \in R(i)$, $1 \le i \le m$, the feasibility problem $1|FB; pmtn|F^{(i)} \le Q_i, 1 \le i \le m$ can be solved in $O(n^*\log n^* + n\log n)$ time. By enumerating all possibilities of $Q_1, Q_2, ..., Q_m$ and choosing the best one, we eventually solve the problem. Since the sequence $Q_1, Q_2, ..., Q_m$ has at most $O((n_1 n_2 \cdots n_m)^m)$ choices and all the values $\Delta(P(x_1, x_2, ..., x_m))$ can be calculated in $O(n^* n_1 n_2 \cdots n_m)$ time, the total complexity is $O((n^*\log n^* + n\log n)(n_1 n_2 \cdots n_m)^m)$, which is polynomial when m is fixed.

Theorem 5.3 The problem $1|\mathit{fix};\ pmtn|\sum_{1\le i\le m} F^{(i)}$ can be solved in $O((n^*\log n^* + n\log n)(n_1 n_2 \cdots n_m)^m)$ time.

Similarly, we have

Theorem 5.4 $1||\sum_{1\le i\le m} F^{(i)}$ can be solved in $O((\log n)(n_1 n_2 \cdots n_m)^m)$ time.

Acknowledgements

We are grateful for the helpful comments of two anonymous referees on an earlier version of this paper. This research was supported in part by The Hong Kong Polytechnic University under grant number S818 and The National Natural Science Foundation of China under grant number 10671183.

References

Agnetis, A., P.B. Mirchandani, D. Pacciarelli, and A. Pacifici, "Scheduling problems with two competing agents", *Operations Research*, **52**, 229-242 (2004).

Baker, K.R., and J.C. Smith, "A multiple-criterion model for machine scheduling", *Journal of Scheduling*, **6**, 7-16 (2003).

Brucker, P., *Scheduling Algorithms*, Springer Verlag, Berlin, 2001.

Curiel, I., G. Pederzoli, and S. Tijs, "Sequencing games", *European Journal of Operational Research*, **40**, 344-351 (1989).

Du, J., and J.Y.-T. Leung, "Minimizing total tardiness on one machine is NP-hard", *Mathematics of Operations Research*, **15**, 483-495 (1990).

Garey, M.R., and D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*, Freeman, San Francisco, CA, 1979.

Hamers, H., P. Borm, and S. Tijs, "On games corresponding to sequencing situations with ready times", *Mathematical Programming*, **70**, 1-13 (1995).

Kim, K., B.C. Paulson, C.J. Petrie, and V.R. Lesser, "Compensatory negotiation for agent-based project schedule coordination", CIFE working paper #55, Stanford University, Stanford, CA, 1999

Kovalyov, M.Y., C.T. Ng, and T.C.E. Cheng, "Fixed interval scheduling: models, applications, computational complexity and algorithms", *European Journal of Operational Research*, **178**, 331-342 (2007).

Lawler, E.L., "Optimal sequencing of a single machine subject to precedence constraints", *Management Science*, **19**, 544-546 (1973).

Lawler, E.L., "A pseudopolynomial algorithm for sequencing jobs to minimize total tardiness", *Annals of Discrete Mathematics*, **1**, 331-342 (1977).

Scharbrodt, M., A. Steger, and H. Weisser, "Approximability of scheduling with fixed jobs", *Journal of Scheduling*, **2**, 267-284 (1999).

Schultz, D., S.-H. Oh, C.F. Grecas, M. Albani, J. Sanchez, C. Arbib, V. Arvia, M. Servilio, F. Del Sorbo, A. Giralda, and G. Lombardi, "A QoS concept for packet oriented S-UMTS services", *Proceedings of the 1st Mobile Summit* 2002, Thessaloniki, Greece.

Yuan, J.J., W.P. Shang, and Q. Feng, "A note on the scheduling with two families of jobs", *Journal of Scheduling*, **8**, 537-542 (2005).