2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)
Banff Center, Banff, Canada, October 5-8, 2017

Scheduling Jobs with Multitasking and Asymmetric Switching Costs

Kevin I.-J. Ho
Department of Comp. Sci. & Comm. Eng.
Providence University, Shalu
Taichung, Taiwan.
ho@pu.edu.tw

John Sum
Institute of Technology Management
National Chung Hsing University
Taichung 40227, Taiwan.
pfsum@nchu.edu.tw

***Abstract*—In this paper, we investigate the job scheduling problems with human multitasking and asymmetric switching costs. It is shown that the makespan problem is binary NP-hard. Both the total completion time and the due date assignment problems are unary NP-hard. We then consider a special case in which the cost of switching from job A (interrupted job) to job B (interrupting job) is in a form of $\kappa_1 f_A^p + \kappa_2 f_B^w$, where κ_1 and κ_2 are constants and f_A^p and f_B^w are costs depending on job A and B respectively. With this special form of switching cost, we show that the makespan, the total completion time and the due date assignment problems can be formulated as linear assignment problems and thus be solved in polynomial time. Even if stress effect is introduced, these three scheduling problems with the special form of asymmetric switching costs are polynomial time solvable.**

***Index Terms*—Asymmetric Switching Costs, Human Multitasking, Linear Assignment, NP-hardness, Pseudo Polynomial-time Algorithm**

I. Introduction

Integrating human factors in scheduling problems has been studied for many decades [1]. Two notable factors are the learning effect [2]–[4] and aging effect [5]. Multitasking behavior has not been incorporated even though this behavior can be found in many industries. Until recently, [6] have introduced the ideas of switching cost and interruption-time functions in scheduling problems to model multitasking. Switching cost specifies the amount of time lag in a job switching. In their model, this value is a function of the cardinality of the incomplete job set. Interruption-time function determines the amount of a incomplete job that a worker has to process when the job interrupts. In [6], Hall *et. al.* showed the motivation of multitasking and the feasibility of the model based on empirical literature and real-world examples. Interested readers may refer the paper. They found that the total weighted completion time (TWCT) and the maximum lateness (ML) problems can be solved by polynomial-time algorithms. With the assumption that all the late jobs must be fully processed, the total number of late jobs (TNLJ) problem is binary NP-hard while the total weighted number of late jobs (TWNLJ) problem is unary NP-hard[1]. If the amount of interruption-time of a incomplete job is proportional to its remaining processing time, they showed that the TNLJ problem can be solved by a polynomial-time algorithm similar to Moore's algorithm and the TWNLJ problem can be solved by a pseudo polynomial-time algorithm.

Following the work done by [6], [8] showed that the TWCT and ML problems can be solved by polynomial-time algorithms even if the switching costs are job dependent and symmetric. For the TNLJ and TWNLJ problems, [8] introduced a new setting in which the late jobs are rejected and not scheduled so that they will not interrupt the on-time jobs[2]. we also showed that the complexities of these problems are respectively binary and unary NP-hard.

In the model proposed in [6], the switching cost is defined as a function of the number of unfinished tasks. This definition implies that the switching cost is independent of task characteristics and task sequence. Some psychological studies [9], [10] showed that switching costs are affected by different aspects of the tasks, including complexity, familiarity, or sequencing. To extend from our previous works, the scheduling problems with multitasking and asymmetric switching costs are investigated in this paper. To clarify the differences among the definitions of the switching cost in [6], [8] and the one defined in this paper, their definitions are depicted in Table I. In the table, the notation $J_A \leftarrow J_B$ means that J_B is a incomplete job interrupting J_A while J_A is processing. In the next section, the background on scheduling problems with multitasking will be reviewed. In the presence of multitasking and asymmetric switching costs,

TABLE I
Definitions of switching cost.

Reference	$J_A \leftarrow J_B$	$J_B \leftarrow J_A$	Definition
[6]	f_{AB}	f_{BA}	$f_{AB} = f_{BA} = 1$
[8]	f_{AB}	f_{BA}	$f_{AB} = f_{BA}$
This paper	f_{AB}	f_{BA}	$f_{AB} \neq f_{BA}$
This paper	f_{AB}	–	$f_{AB} = \kappa_1 f_A^p + \kappa_2 f_B^w$
This paper	–	f_{BA}	$f_{BA} = \kappa_1 f_B^p + \kappa_2 f_A^w$

f_A^p is the cost that J_A is interrupted by other jobs.
f_A^w is the cost that J_A interrupts other jobs.

[1]In Pinedo's book [7], binary and unary NP-hard are named as ordinary and strongly NP-hard respectively.

[2]In the original setting in [6], all the late jobs are scheduled and interrupt the on-time jobs. Clearly, the setting will introduce a controversial issue. As a late job cannot be completed by its due date, why should a worker handle any part of it while the worker is processing the on-time jobs. If late jobs are rejected and reassigned/outsourced, the worker could possible complete more jobs on-time.

978-1-5386-1645-1/17/$31.00 ©2017 IEEE

i.e. $f_{AB} \neq f_{BA}$, we then show in Section 3 that the makespan problem is binary NP-hard, the total completion time and the due date assignment problems are unary NP-hard. In Section 4, we investigate a special case that $f_{AB} = \kappa_1 f_A + \kappa_2 f_B$, where κ_1 and κ_2 are constants. The factors f_A and f_B are costs depending on job A and job B. A discussion on the stress effect (rate-modifying activity) on this special asymmetric switching costs is presented in Section 5. Finally, we conclude the paper in Section 6.

II. Background

Given a set of n jobs $J = \{J_1, \cdots, J_n\}$ with known processing times $p_1, \cdots, p_n$, weighting factors $w_1, \cdots, w_n$ and a criteria, the purpose of classical single machine scheduling is to find a sequence of jobs $S = \{J_{\pi_1}, \cdots, J_{\pi_n}\}$ such that the criteria is optimized. Then, the jobs are processed one by one in accordance with the sequence. The completion time of J_{π_k} is thus $\sum_{j=1}^{k} p_{\pi_j}$.

Following [6], in the presence of multitasking, it is assumed that $J_{\pi_{k+1}}$, $J_{\pi_{k+2}}$, $\cdots$, J_{π_n} will interrupt, not preempt, J_{π_k}. In other words, part of each incomplete job has to be processed when a scheduled job is executed. Let $p'_{\pi_i}(k)$ be the remaining processing time of J_{π_i} after it has interrupted k times. $g_{\pi_i}(p'_{\pi}(k-1))$ is the amount of time that J_{π_i} has to be processed in the k^{th} interruption. To simplify the notation, we use $g_{\pi_i k}$ to denote $g_{\pi_i}(p'_{\pi}(k-1))$. Then, we have

$$p'_{\pi_i}(k) = \begin{cases} p_{\pi_i} & \text{if } k = 0, \\ p'_{\pi_i}(k-1) - g_{\pi_i k} & \text{if } 1 \leq k \leq i-1, \\ 0 & \text{if } k \geq i \end{cases} \quad (1)$$

for $i, k = 1, 2, \cdots, n$. Let c_{π_k} be the completion time of J_{π_k}. Then, by setting $c_{\pi_0} = 0$, the completion time of J_{π_k} can be given by

$$c_{\pi_k} = c_{\pi_{k-1}} + p'_{\pi_k}(k-1) + \sum_{j=k+1}^{n} \left(g_{\pi_j k} + f_{\pi_k \pi_j}\right) \quad (2)$$

for $k = 1, \cdots, n-1$ and

$$c_{\pi_n} = c_{\pi_{n-1}} + p'_{\pi_n}(n-1). \quad (3)$$

Equivalently,

$$c_{\pi_k} = \sum_{i=1}^{k} p_{\pi_i} + \sum_{i=1}^{k} \sum_{j=i+1}^{n} \left(g_{\pi_j i} + f_{\pi_i \pi_j}\right) \quad (4)$$

for $k = 1, \cdots, n-1$, and

$$c_{\pi_n} = \sum_{i=1}^{n} p_{\pi_i} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(g_{\pi_j i} + f_{\pi_i \pi_j}\right). \quad (5)$$

To ensure that $p_{\pi_i}(k)$ in (1) is non-negative, we further assume that $\sum_{k=1}^{n-1} g_{\pi_i k} \leq p_{\pi_i}$.

Following [6], we use 'mt' in the β field to denote 'multitasking'. In the following, C_{max} denotes the makespan, i.e. $C_{\max} = c_{\pi_n}$. E_i and L_i denotes the earliness and lateness of job J_i, i.e. $E_i = \max\{0, c_i - d_i\}$ and $L_i = c_i - d_i$. U_i is used as the indicator that job J_i is late, i.e. $U_i = 1$ if J_i is late and zero otherwise. The problems to be investigated are defined as follows.

Problem 1 (Makespan): Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, w_i, $g_i(\cdot)$ and $f_{ij} \neq f_{ji}$ for all $i, j = 1, 2, \cdots, n$, find a feasible schedule $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$ such that C_{max} is minimum among all feasible schedules.

Problem 2 (Total Completion Time): Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$ and $f_{ij} \neq f_{ji}$ for all $i, j = 1, \cdots, n$, find a schedule $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$ such that $\sum_i c_{\pi_i}$ is minimum among all feasible schedules.

Problem 3 (Due Date Assignment): Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$ and $f_{ij} \neq f_{ji}$ for all $i, j = 1, \cdots, n$, find a schedule $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$ and a common due date d such that $\sum_i \xi_1 d + \xi_2 E_i + \xi_3 T_i + \xi_4 c_i$ is minimum among all feasible schedules.

III. NP-Hard Problems

First, we consider the problem $1|mt, f_{ij}|C_{max}$. For a schedule $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$, the makespan is given by

$$C_{max}(S) = \sum_{i=1}^{n} p_{\pi_i} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} f_{\pi_i \pi_j}. \quad (6)$$

As $\sum_{i=1}^{n} p_{\pi_i}$ is a constant for any schedule, the problem $1|mt, f_{ij}|C_{max}$ is equivalent to finding a schedule S such that $\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} f_{\pi_i \pi_j}$ is minimum. Its NP-hardness is stated in the following theorem.

Theorem 1: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$, f_{ij} for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij}|C_{max}$ is binary NP-hard.

Proof. The proof is accomplished by reducing the feedback arc set problem to the makespan problem. The feedback arc set problem is defined as below [11, p.192 Problem GT8].

Problem 4 (Feedback Arc Set (FAS)): Given a directed graph $G = (V, A)$ and a positive integer $K \leq |A|$, find a subset $A' \subseteq A$ with $|A'| \leq K$ such that A' contains at least one arc from every directed cycle in G.

Let $G = (V, A)$ be a arbitrary directed graph, where $V = \{V_1, V_2, \cdots, V_n\}$ is a vertex set with n vertices and an arc $(V_i, V_j) \in A$ denotes the arc from V_i to V_j. Given any instance of FAS, G and K, an instance of $1|mt, f_{ij}|C_{max}$ in decision version, J and y, can be constructed as follows. $|J| = n$ and $y = n^2 + K$. For each job $J_i \in J$ which corresponds to $V_i \in V$, its processing time, interruption-time function and switching costs are defined as follows :

$$p_i = n, \quad (7)$$

$$g_i = 1, \quad (8)$$

$$f_{ij} = \begin{cases} 1 & \text{if } (V_j, V_i) \in A, \\ 0 & \text{otherwise.} \end{cases} \quad (9)$$

Note that the value of $f_{ij} = 1$ if $(V_j, V_i) \in A$ and 0 otherwise. It implies that all the non-zero switching costs incur only when J_j is scheduled after J_i.

($\Rightarrow$) Let $A' \subseteq A$ be an arc set such that $G' = (V, A \setminus A')$ is a directed acyclic graph and $|A'| \leq K$. A schedule S of

J can be constructed by sequencing all the jobs according to the topological ordering of the corresponding vertices in G'. Without counting the switching costs introducing by the arcs in A', the makespan of S is n^2. Let $(V_{i_1}, V_{i_2}, \cdots, V_{i_r})$ be a directed path in G' and (V_{i_r}, V_{i_1}) be an arc in A'. It is obvious that scheduling $J_{i_1}, J_{i_2}, \cdots, J_{i_r}$ sequentially will introduce 0 switching cost. Since $(V_{i_r}, V_{i_1}) \in A'$, 1 time unit of switching cost will be added. As a result, the number of arcs in A' is equivalent to the total switching costs incurring in S. Therefore,

$$\begin{aligned} C_{max}(S) &= \sum_{i=1}^{n} p_i + \sum_{(V_j,V_i)\in A'} f_{ij} \\ &\leq n^2 + K \\ &= y. \end{aligned}$$

($\Leftarrow$) Let $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$ be a schedule of the constructed instance such that $C_{max}(S) \leq n^2 + K$. The arc set A' can be built as follows.

$$A' = \{(V_{\pi_i}, V_{\pi_j}) | J_{\pi_i} \text{ is scheduled after } J_{\pi_j} \text{ in } S\}$$

We now show that at least one arc in each directed cycle in G is in A'. Assume that there is a directed cycle $C = (V_{i_1}, V_{i_2}, \cdots, V_{i_r}, V_{i_1})$ in G such that none of the arcs in C is in A'. Since S is a single machine schedule, the jobs corresponding to the vertices in C are scheduled as a linear sequence. Therefore, there must exist a job pair $(J_{i_k}, J_{i_{k+1}})$ such that $(V_{i_k}, V_{i_{k+1}}) \in A$ and J_{i_k} is scheduled after $J_{i_{k+1}}$. From the construction of A', $(V_{i_k}, V_{i_{k+1}})$ must be in A', contradicting to the assumption. Since only the job pair corresponding to an arc in A' introduces 1 time unit of switching cost, the size of A' is equal to the total switching cost of S. The total processing time of all the jobs is n^2 and $C_{max}(S) \leq n^2 + K$, implying that the total switching costs incurred in S is not greater than K. Hence, $|A'| \leq K$.

As it has been proved by [12] that the feedback arc set problem is NP-Complete, $1|mt, f_{ij}|C_{max}$ is binary NP-hard. **Q.E.D.**

The following theorem shows that the total completion time problem is unary NP-hard.

Theorem 2: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$ and f_{ij} for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij}|\sum_i c_i$ is unary NP-hard.

Proof. The proof is accomplished by reducing the problem $1|prec|\sum_i c_i$ to the problem $1|mt, f_{ij}|\sum_i c_i$. Let (J, G, K) be an arbitrary instance of the problem $1|prec|\sum_i c_i$. $J = \{J_1, J_2, \cdots, J_n\}$ denotes a set of n job where p_i $(1 \leq i \leq n)$ is the processing time of J_i. All the jobs are available at time 0. The precedence constraints among jobs are represented by the directed acyclic graph $G = (J, A)$ in which, an arc $(J_i, J_j) \in A$ corresponds to the constraint that J_j cannot start executing before J_i completes. The decision version of $1|prec|\sum_i c_i$ is to ask if there is a schedule S such that the job sequence fulfills all the precedence constraints and $\sum_{J_i \in J} c_i(S) \leq K$.

Let (J^*, y) be the instance of the problem $1|mt, f_{ij}|\sum_i c_i$ constructed based on (J, G, K). The characteristics of each job J_i^*, $1 \leq i \leq n$, in J^* is shown as below. For all $i, j = 1, 2, \cdots, n$,

$$p_i^* = p_i \quad (10)$$

$$g_{ik}^* = 0 \text{ for } k = 1, \cdots, n-1 \quad (11)$$

$$f_{ij}^* = \begin{cases} K+1 & \text{if } (J_j, J_i) \in A, \\ 0 & \text{otherwise.} \end{cases} \quad (12)$$

The threshold y is set to K.

($\Rightarrow$) Let $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$ be a schedule fulfilling the precedence constraints and $\sum_{J_i \in J} c_i(S) \leq K$. We can build the schedule S^* for (J^*, y) by sequencing jobs in J^* as S. As S fulfills the precedence constraints, for all $(J_i, J_j) \in A$, J_i must be completed before J_j. It implies that J_i^* must also be completed before J_j^* starts. By the definitions of the switching costs, $f_{ij}^* = K + 1$ only incurs when J_j^* is completed before J_i^*. Therefore, no switching cost incurs in the schedule S^*. The completion time of $J_{\pi_i}^*$ must be equal to that of J_{π_i}. As a result, $\sum_{J_i^* \in J^*} c_i(S^*) = \sum_{J_i \in J} c_i(S)$ and thus $\sum_{J_i^* \in J^*} c_i(S^*) \leq y$.

($\Leftarrow$) Let $S^* = \{J_{\pi_1}^*, J_{\pi_2}^*, \cdots, J_{\pi_n}^*\}$ be a schedule for (J^*, y) and $\sum_{J_i^* \in J^*} c_i(S^*) \leq y$. A schedule S for (J, G, K) can be formed by sequencing jobs in J as S^*. From the definitions of switching cost functions, it is easy to see that no switching cost is incurred in S^*. Therefore, S satisfies all the precedence constraints. Moreover, the completion time of J_{π_i} in S must be equal to that of $J_{\pi_i}^*$ in S^*. As a result, we can conclude that $\sum_{J_i \in J} c_i(S) \leq K$.

As $1|prec|\sum_i c_i$ has been proved to be unary NP-hard [13], [14], $1|mt, f_{ij}|\sum_i c_i$ is unary NP-hard. **Q.E.D.**

With reference to [15], we define the criteria to be minimized as $\sum_i (\xi_1 d + \xi_2 E_i + \xi_3 L_i + \xi_4 c_i)$.

Theorem 3: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$ and f_{ij} for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij}|\sum_i \xi_1 d + \xi_2 E_i + \xi_3 L_i + \xi_4 c_i$ is unary NP-hard.

Proof. Clearly, the total completion time problem is a special case of the common due date problem if $\xi_1 = \xi_2 = \xi_3 = 0$, it can be implied by Theorem 2 that the due date assignment problem with mulitasking and asymmetric switching costs is unary NP-hard. **Q.E.D.**

The implications of Theorem 1 and Theorem 2 are, with asymmetric switching costs, (1) the maximum lateness problem is binary NP-hard, (2) the total weighted completion time and the total weighted number of late jobs problems are both unary NP-hard.

IV. Polynomial Time Solvable Problems

In this section, we will show that some problems can be solved in polynomial time if the switching cost is defined as follows :

$$f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w, \quad (13)$$

where f_i^p is the cost due to the primary job J_i, f_j^w is the cost due to the incomplete job J_j. In (13), κ_1 and κ_2 are constants and $\kappa_1 \neq \kappa_2$.

A. Makespan

The following theorem states that makespan problems can be solved in polynomial time.

Theorem 4: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$, f_j^w, f_i^p for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w|C_{max}$ can be solved in $\mathcal{O}(n^3)$ time.

Proof. For $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$,

$$\begin{aligned} C_{max} &= \sum_{i=1}^{n} p_{\pi_i} + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(\kappa_1 f_{\pi_i}^p + \kappa_2 f_{\pi_j}^w\right), \\ &= P + \kappa_1 \sum_{i=1}^{n-1} (n-i) f_{\pi_i}^p \\ &\quad + \kappa_2 \sum_{j=2}^{n} (j-1) f_{\pi_j}^w, \end{aligned} \tag{14}$$

where $P = \sum_i p_i$.

Let $x_{ik} \in \{0, 1\}$ be the decision variable if J_i is assigned as the J_{π_k}.

$$x_{ik} = \begin{cases} 1 & \text{if } J_i \text{ is assigned as } J_{\pi_k}, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Then, we replace $f_{\pi_k}^p$ and $f_{\pi_k}^w$ in (14) by

$$f_{\pi_k}^p = \sum_{i'=1}^{n} f_{i'}^p x_{i'k}, \quad f_{\pi_k}^w = \sum_{i'=1}^{n} f_{i'}^w x_{i'k} \tag{16}$$

for all $k = 1, \cdots, n$. Thus, C_{max} can be expressed as a linear combination of x_{ik} for $i, k = 1, \cdots, n$. The problem $1|mt, f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w|C_{max}$ can be formulated as a linear assignment problem that minimizes

$$\begin{aligned} C_{max} &= P + \kappa_1 \sum_{i=1}^{n-1} (n-i) \sum_{i'=1}^{n} f_{i'}^p x_{i'i} \\ &\quad + \kappa_2 \sum_{j=2}^{n} (j-1) \sum_{i'=1}^{n} f_{i'}^w x_{i'j} \end{aligned} \tag{17}$$

subject to

$$\sum_{i=1}^{n} x_{ik} = 1, \text{ for all } k = 1, 2, \cdots, n. \tag{18}$$

$$\sum_{k=1}^{n} x_{ik} = 1, \text{ for all } i = 1, 2, \cdots, n. \tag{19}$$

This problem can be solved in $\mathcal{O}(n^3)$ time [16]. The proof is completed. **Q.E.D.**

It is clear that if κ_1 in (13) is zero, an optimal schedule can be obtained by sorting the $f_{(\cdot)}^w$ values in descending order, i.e. $f_{\pi_1}^w \geq \cdots \geq f_{\pi_n}^w$. On the other hand, if $\kappa_2 = 0$, an optimal schedule can be obtained by sorting the $f_{(\cdot)}^p$ values in ascending order, i.e. $f_{\pi_1}^p \leq \cdots \leq f_{\pi_n}^p$. The complexity of this special case is $\mathcal{O}(n \log(n))$.

B. Total Completion Time

For the total completion time problem with the switching cost defined as (13), it can be formulated as a linear assignment problem as well and solved in $\mathcal{O}(n^3)$ time.

Theorem 5: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$, f_j^w, f_i^p for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w| \sum_i c_i$ can be solved in $\mathcal{O}(n^3)$ time.

Proof. Since $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$, from (2), we can get that

$$\begin{aligned} \sum_i c_i &= \sum_{k=1}^{n} (n-k+1) \left\{ p_{\pi_k} - \sum_{l=0}^{k-1} g_{\pi_k l} \right\} \\ &\quad + \sum_{k=1}^{n-1} (n-k+1) \sum_{j=k+1}^{n} \left(g_{\pi_j k} + \kappa_1 f_{\pi_i}^p + \kappa_2 f_{\pi_j}^w\right). \end{aligned} \tag{20}$$

where $g_{i0} = 0$ for all $i = 1, \cdots, 0$.

Again, we define $x_{ik} \in \{0, 1\}$ as in (15). Replace both p_{π_k} and $g_{\pi_k l}$ in (20) by

$$p_{\pi_k} = \sum_{i'=1}^{n} p_{i'} x_{i'k}, \quad g_{\pi_k l} = \sum_{i'=1}^{n} g_{i'l} x_{i'k} \tag{21}$$

and both $f_{\pi_k}^p$ and $f_{\pi_k}^w$ by (16) for all $k = 1, \cdots, n$ and $l = 1, \cdots, n-1$. Thus, $\sum_i c_i$ can be expressed as a linear combination of x_{ik} for $i, k = 1, \cdots, n$. The problem $1|mt, f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w| \sum_i c_i$ can be formulated as a linear assignment problem that minimizes $\sum_i c_i$ subject to (18) and (19). It can be solved in $\mathcal{O}(n^3)$ time [16]. The proof is completed. **Q.E.D.**

C. Due Date Assignment

To show that the due date assignment problem is polynomial time solvable if $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$, we need the following lemma by [15, Lemma 2].

Lemma 1: If $\xi_3 > \xi_1$, for an optimal schedule S, there exists an optimal due date d equal to c_{π_K}, i.e. $d = c_{\pi_K}$, where K is the smallest integral value greater than or equal to $n(\xi_3 - \xi_1)/(\xi_2 + \xi_3)$.

Proof. Refer to P.393-394,397 in [15]. **Q.E.D.**

For the sake of presentation, we denote $F(S, d)$ as the criteria to be minimized, i.e. $F(S, d) = \sum_i \xi_1 d + \xi_2 E_i + \xi_3 L_i + \xi_4 c_i$. By Lemma 1, we can get that

$$F(S, d) = \sum_i \left(\xi_1 c_{\pi_K} + \xi_2 E_i + \xi_3 L_i + \xi_4 c_i\right).$$

Theorem 6: Given a set of jobs $J = \{J_1, J_2, \cdots, J_n\}$ with known p_i, $g_i(\cdot)$, f_j^w, f_i^p for all $i, j = 1, \cdots, n$, the problem $1|mt, f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w|F(S, d)$ can be solved in $\mathcal{O}(n^3)$ time.

Proof. For a schedule $S = \{J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n}\}$, we get by Lemma 1 that

$$\begin{aligned} F(S,d) &= \sum_{i=1}^{K} (\xi_1 c_{\pi_K} + \xi_2 (c_{\pi_K} - c_{\pi_i}) + \xi_4 c_{\pi_i}) \\ &\quad + \sum_{i=K+1}^{n} (\xi_1 c_{\pi_K} + \xi_3 (c_{\pi_i} - c_{\pi_K}) + \xi_4 c_{\pi_i}) \\ &= (n(\xi_1 - \xi_3) + K(\xi_2 + \xi_3))\, c_{\pi_K} \\ &\quad + (\xi_4 - \xi_2) \sum_{i=1}^{K} c_{\pi_i} + (\xi_4 + \xi_3) \sum_{i=K+1}^{n} c_{\pi_i} \\ &= \sum_{i=1}^{n} \eta_i c_{\pi_i}, \end{aligned} \tag{22}$$

where

$$\eta_1 = \cdots = \eta_{K-1} = \xi_4 - \xi_2,$$

$$\eta_K = n(\xi_1 - \xi_3) + K(\xi_2 + \xi_3) + (\xi_4 - \xi_2)$$

and

$$\eta_{K+1} = \cdots = \eta_n = \xi_4 + \xi_3.$$

Recall, from Lemma 1, that $K = \lceil n(\xi_3 - \xi_1)/(\xi_2 + \xi_3) \rceil$. Rewriting (22), we can get that

$$\begin{aligned} F(S,d) &= \sum_{i=1}^{n-1} \left(\sum_{j=i}^{n} \eta_j \right) \left\{ \left(p_{\pi_i} - \sum_{l=0}^{i-1} g_{\pi_i l} \right) \right. \\ &\quad \left. + \sum_{j=i+1}^{n} \left(g_{\pi_j i} + \kappa_1 f_{\pi_i}^{p} + \kappa_2 f_{\pi_j}^{w} \right) \right\} \\ &\quad + \eta_n \left(p_{\pi_n} - \sum_{l=0}^{n-1} g_{\pi_n l} \right). \\ &= \sum_{i=1}^{n-1} \left(\sum_{j=i}^{n} \eta_j \right) \left\{ \left(p_{\pi_i} - \sum_{l=0}^{i-1} g_{\pi_i l} \right) \right. \\ &\quad \left. + (n-i)\kappa_1 f_{\pi_i}^{p} + \sum_{j=i+1}^{n} \left(g_{\pi_j i} + \kappa_2 f_{\pi_j}^{w} \right) \right\} \\ &\quad + \eta_n \left(p_{\pi_n} - \sum_{l=0}^{n-1} g_{\pi_n l} \right). \end{aligned} \tag{23}$$

Introducing the binary variables x_{ik} as in (15) and replacing $f_{\pi_k}^{p}$, $f_{\pi_k}^{w}$, p_{π_k} and $g_{\pi_k l}$ in (23) by (16) and (21), the due date assignment problem can thus be formulated as a linear assignment problem which can be solved in $\mathcal{O}(n^3)$ time. The proof is completed. **Q.E.D.**

If $g_{ik} = Dp_i'(k-1)$, $f_i^p = D'p_i$ and $f_i^w = D''p_i$, where $D, D', D'' \in [0,1)$, it can be shown that $\sum_i c_i$ (20) and $F(S,d)$ (23) can be reduced to a form of $\gamma_1 p_{\pi_1} + \gamma_2 p_{\pi_2} + \cdots + \gamma_n p_{\pi_n}$. Let $\gamma_{s_1} \le \gamma_{s_2} \le \cdots \le \gamma_{s_n}$ and $p_{s_1'} \ge p_{s_2'} \ge \cdots \ge p_{s_n'}'$. Then we can assign $J_{s_k'}$ as $J_{\pi_{s_k}}$. So that the total completion time and due date assignment problems can be solved in $\mathcal{O}(n \log(n))$ time.

V. With Stress Effect

In accordance with Yerkes-Dodson Law [17], the efficiency of a worker could be alleviated if a proper stress is incurred. Too much or too less stress, the efficiency declines. A common source of stress is clearly due to the number of incomplete jobs. Thus, the scheduling problems investigated could be formulated as scheduling problems with rate-modifying activity [18]–[20]. The completion times are given by

$$c_{\pi_1} = \beta_1 \left(p_{\pi_1} + \sum_{j=2}^{n} g_{\pi_j 1} \right) + \sum_{j=2}^{n} f_{\pi_1 \pi_j}, \tag{24}$$

$$\begin{aligned} c_{\pi_k} &= c_{\pi_{k-1}} + \beta_k \left(p_{\pi_k} - \sum_{l=1}^{k-1} g_{\pi_k l} + \sum_{j=k+1}^{n} g_{\pi_j k} \right) \\ &\quad + \sum_{j=k+1}^{n} f_{\pi_k \pi_j}, \end{aligned} \tag{25}$$

for $k = 2, \cdots, n-1$ and

$$c_{\pi_n} = c_{\pi_{n-1}} + \beta_n \left(p_{\pi_n} - \sum_{l=1}^{n-1} g_{\pi_n l} \right). \tag{26}$$

Here, $\beta_1 > \cdots > \beta_{i^*}$ and $\beta_{i^*} < \cdots < \beta_n$, which is in a form of U-shape. Obviously, the scheduling problems must be at least binary NP-hard. However, for the special case that $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$, the makespan, the total completion time and due date assignment problems can be solved in $\mathcal{O}(n^3)$. It is because these problems can be formulated as linear assignment problems. Due to page limit, the details will be presented elsewhere.

VI. Conclusions

For the scheduling problems with multitasking and asymmetric switching costs, we have shown in this paper the makespan problem is binary NP-hard, the total completion time and the due date assignment problems are unary NP-hard. These results indicate that the scheduling jobs in the presence of multitasking and asymmetric switching costs is normally intractable. With the special form of switch cost in which $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$ for all $i, j = 1, \cdots, n$, it has been shown that the makespan, the total completion time and the due date assignment problems can be formulated as linear assignment problems and thus solved in $\mathcal{O}(n^3)$ time. If either κ_1 or κ_2 is zero, the makespan problem can be solved in $\mathcal{O}(n \log(n))$ time. If $g_{ik} = Dp_i'(k-1)$, $f_i^p = D'p_i$ and $f_i^w = D''p_i$, where $D, D', D'' \in [0,1)$, the total completion time and due date assignment problems can be solved in $\mathcal{O}(n \log(n))$ time. Even with stress effect (i.e. rate-modifying activity), the makespan, the total completion time and the due date assignment problems can also be formulated by formulating the corresponding linear assignment problems if $f_{ij} = \kappa_1 f_i^p + \kappa_2 f_j^w$ and thus solved $\mathcal{O}(n^3)$ time. These results supplement to our previous works on the scheduling problems with multitasking and symmetric switching costs [8].

Acknowledgement

The work presented in this paper is supported in part by Taiwan Ministry of Science and Technology (MOST) under the grants 103-2410-H-005-036, 104-2410-H-005-026 and 105-2410-H-126-003.

REFERENCES

[1] E. J. Lodree, C. D. Geiger, and X. Jiang, "Taxonomy for integrating scheduling theory and human factors: Review and research opportunities," *International Journal of Industrial Ergonomics*, vol. 39, no. 1, pp. 39–51, 2009.

[2] K. I. Ho, J. Y. Leung, and W. Wei, "Complexity of scheduling tasks with time-dependent execution times," *Information Processing Letters*, vol. 48, no. 6, pp. 315–320, 1993.

[3] D. Biskup, "Single-machine scheduling with learning considerations," *European Journal of Operational Research*, vol. 115, no. 1, pp. 173–178, 1999.

[4] ——, "A state-of-the-art review on scheduling with learning effects," *European Journal of Operational Research*, vol. 188, no. 2, pp. 315–329, 2008.

[5] C.-l. Zhao and H.-Y. Tang, "Single machine scheduling with general job-dependent aging effect and maintenance activities to minimize makespan," *Applied Mathematical Modelling*, vol. 34, no. 3, pp. 837–841, 2010.

[6] N. G. Hall, J. Y.-T. Leung, and C.-L. Li, "The effects of multitasking on operations scheduling," *Production and Operations Management*, vol. 24, no. 8, pp. 1248–1265, 2015.

[7] M. Pinedo, *Scheduling: Theory, Algorithms, and Systems*, 3rd ed. Springer, 2015.

[8] J. Sum and K. I.-J. Ho, "Operations scheduling in the presence of multitasking and symmetric switching cost," in *Proceedings of International Conference on Management Science and Decision Making*, Taipei, Taiwan, 2015.

[9] "Multitasking: Switching costs," http://www.apa.org/research/action/multitask.aspxt, 2006.

[10] J. S. Rubinstein, D. E. Meyer, and J. E. Evans, "Executive control of cognitive processes in task switching," *Journal of Experimental Psychology: Human Perception and Performance*, pp. 763–797, 2001.

[11] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-completeness*. New York: WH Freeman and Company, 1979.

[12] R. M. Karp, "Reducibility among combinatorial problems," in *Complexity of Computer Computations*. Springer, 1972, pp. 85–103.

[13] E. L. Lawler, "Sequencing jobs to minimize total weighted completion time subject to precedence constraints," *Annals of Discrete Mathematics*, vol. 2, pp. 75–90, 1978.

[14] J. K. Lenstra and A. Rinnooy Kan, "Complexity of scheduling under precedence constraints," *Operations Research*, vol. 26, no. 1, pp. 22–35, 1978.

[15] S. Panwalkar, M. Smith, and A. Seidmann, "Common due date assignment to minimize total penalty for the one machine scheduling problem," *Operations Research*, vol. 30, no. 2, pp. 391–399, 1982.

[16] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*. Courier Corporation, 1982.

[17] R. M. Yerkes and J. D. Dodson, "The relation of strength of stimulus to rapidity of habit-formation," *Journal of comparative neurology and psychology*, vol. 18, no. 5, pp. 459–482, 1908.

[18] C.-Y. Lee and V. J. Leon, "Machine scheduling with a rate-modifying activity," *European Journal of Operational Research*, vol. 128, no. 1, pp. 119–128, 2001.

[19] Z. Zhu, F. Zheng, and C. Chu, "Multitasking scheduling problems with a rate-modifying activity," *International Journal of Production Research*, vol. 55, no. 1, pp. 296–312, 2017.

[20] Z. Zhu, J. Li, and C. Chu, "A note on multitasking scheduling problems with deterioration effect," *Mathematical Problems in Engineering*, 2006.