Operations Scheduling in the Presence of Multitasking and Symmetric Switching Cost

John Sum[1], Kevin I.-J. Ho[2]

[1] Institute of Technology Management, National Chung Hsing University, Taichung 402, Taiwan. (pfsum@nchu.edu.tw)

[2] Department of Computer Science and Communication Engineering, Providence University, Shalu, Taichung, Taiwan. (ho@pu.edu.tw)

ABSTRACT

Hall, Leung & Li have recently proposed a new model for studying operations scheduling in the presence of multitasking – Once a primary job is being processed, the worker gets interruptions from the waiting jobs and needs to switch from the primary job to process for each of the interrupting jobs part of them. Afterward, the worker switches back to process the rest of the primary job. In this paper, we follow their model by introducing two new conditions: (i) the switching costs are job-dependent and symmetric, and (ii) the late jobs are not allowed to interrupt. Complexities of six scheduling problems are investigated. They include the makespan, the total weighted completion time (TWCT), the maximum weighted tardiness (MWT) and the maximum weighted lateness (MWL), the total number of late jobs (TNLJ) and the total weighted number of late jobs (TWNLJ) problems. We show that the makespan, TWCT, MWT and MWL problems are polynomial-time solvable. Under mild conditions on the switching cost function and the interruption function, the total completion time problem can be solved by the shortest processing time first rule. For our late job problems, we show that the TNLJ problem is NP-hard and the TWNLJ is strongly NP-hard. For certain special cases, these problems are polynomial time solvable. Findings in the areas of psychology and management have indicated that multitasking could hamper the mental health and the productivity of a worker. If multitasking is unavoidable, the algorithms presented in this paper could help the human worker to schedule their jobs so as to minimize the effect.

Keywords: Algorithm, Complexity Analysis, Multitasking, Scheduling.

1. INTRODUCTION

Given a set of jobs, suppose the jobs have been schedule. Without multitasking, a worker will process the jobs in sequential manner. A waiting job will not be processed unless all the jobs scheduled before it have complete. IN reality, we worker usually will receive request from the client of a waiting job to handle in advance part of the job. The worker has then to stop the current job and switch to the waiting job. While multitasking behavior can be found in many industries, it has not been studied under the context of scheduling. Recently, Hall, Leung and Li have introduced the ideas of *switching cost* and *interruption function* in the scheduling problems in the presence of multitasking. Switching cost determines the amount of time a worker required in job switching. Interruption function determines the amount of the job a worker has to process for the waiting job during an interruption. For constant switching cost, they have found that the total weight completion time and the maximum lateness problem can be solved by polynomial time algorithms. The total (weighted) number of late job problem is (strongly) NP hard. Their late job problems are defined in a way the same as in the classical scheduling theory – All the jobs have to complete.

In this paper, we extend the analysis by introducing two new conditions.

- *The switching costs are not constant. They are job-dependent and symmetric. That is, the cost of switching from job A to job B is equal to the cost of switching from job B to job A.*
- *For the (weighted) number of late job problem, the late jobs are discarded and they do not have to process.*

As the late jobs are discarded and will not interrupt the on-time jobs, the (weighted) number on on-time jobs will definitely be increased. With the above conditions, six scheduling problems corresponding to six different criteria will be investigated in this paper. They include the makespan, the total weighted completion time (TWCT), the maximum weighted tardiness (MWT), the maximum weighted lateness (MWL), the total number of late jobs (TNLJ) and the total weighted number of late jobs (TWNLJ) problems. For the late job problems, while we do not specify, the late jobs are discarded.

While multitasking is a new topic in the area of scheduling, it has been researched in the areas of psychology, management, information systems, and behavior economics for several decades. Thus, in Section 2, some of the key research results in these areas will be reviewed and their limitations in solving job scheduling problems are highlighted. Formulations of the scheduling problems in the presence of multitasking will be presented in Section 3. In particular, the model of multitasking, the ideas of interruption functions, symmetric switching costs and models of the late jobs problems are elucidated. Complexities of six scheduling problems in the presence of multitasking are analyzed in Section 4. For some problems, polynomial time algorithms are developed. The conclusion is presented in Section 5.

2. RESEARCH WORKS IN MULTITASKING

In our daily lives, multitasking can be found in many places. On the road, some people drive a car and at the same time talk over the cell phone. At school, some students open multiple browsers at the same time and search information concurrently for different assignments (Spink, Ozmutlu & Ozmutlu 2002, Ozmutlu, Ozmutlu & Spink 2003, Spink 2004, Spink, Park, Jansen & Pedersen 2006). In a recruiting firm, recruiters need to handle candidate search for one project and at the same time answer enquiries (in the form of emails or phone calls) from their clients for another project (Aral, Brynjolfsson & Van Alstyne 2007). In an Italian court, judges are assigned new cases in every morning, and at the same time they need to process cases which are already in the pipe (Coviello, Ichino & Persico 2010). Pilots have to read multiple sources of data over the panel and at the same time listen over his headset the enquiries and instructions from the air traffic control tower to ensure that the plane can safely take off and land (Loukopoulos, Dismukes & Barshi 2009). Air traffic controllers have to gather multiple sources of information, including radar signals, weather reports and the information from the pilots, and co-ordinate the taking off and landing of the planes (Loukopoulos, Dismukes & Barshi 2009). In a busy hospital emergency department, each physician needs to handle multiple patients concurrently in a shift (KC 2013).

2.1 Laboratory & Field Studies

Researchers have found that there are many reasons why people multitask. The very first reason is due to the job nature, like plane control and air traffic control. The second reason is due to interruptions, from clients, colleagues, friends and family members. These interruptions are usually not predictable. Some of them are unavoidable and need immediate attention. It is especially true for senior managers (Mintzberg 1973, Jett & George 2003). The third reason is due to personality trait. People with high levels of impulsivity and sensation seeking often engage in multitasking (Sanbonmatsu, Strayer, Medeiros-Ward & Watson 2013). As multitasking behavior can be found in almost everywhere, one important question is aroused. *Is multitasking good or bad?* In this regard, researchers from management, psychology and information systems have long been studying the effect of multitasking on the productivity of human worker in office works, on the mental health of a person and the design of computer-human interface.

While some studies (Lee & Taatgen 2002, Taatgen & Lee 2003) have found that multitasking can help to improve the skills of a worker in handling multiple tasks, the actual benefit of multitasking has yet to be discovered. Some scholars suggested that multitasking is not recommended (Rosen 2008). Multitasking incurs a cost of job switching (Rogers & Monsell 1995, Rubinstein, & Evan 2001, Monsell 2003) which includes response delay (Meuter & Allport 1999) and resumption lag (Trafton, Altmann, Brock & Mintz 2003). (A good summary regarding switching costs in multitasking can be found in (APA 2006).) The delay and lag are job dependent (Gillie & Broadbent 1989). In a field study, it is found that the average resumption lag could be as long as 25 minutes (Mark, Gonzalez & Harris 2005, p.326). Rehearsal could reduce the response delay or resumption lag. But still, they can hardly be eliminated (Gillie & Broadbent 1989, Trafton, Altmann, Brock & Mintz 2003).

Multitasking could cause problem in learning (Foerde, Knowlton & Poldrack 2006) and lead to attention deficit trait (ADT) (Hallowell 2005). A person with ADT has a normal brain but his behavior is similar to a person with attention deficit disorder, problem in paying attention. In some laboratory experiments (Watson & Strayer 2010, Sanbonmatsu, Strayer, Medeiros-Ward & Watson 2013), it is found that the performance of a subject in multitasking environment drops as compared with working in sequential processing environment. Only very few subjects are able to perform equally well in both multitasking and sequential processing environments (Watson & Strayer 2010). Nevertheless, subjects who perceived they are good at multitasking usually underperform in both environments, as compared with the subjects who perceived they are not good at multitasking (Sanbonmatsu, Strayer, Medeiros-Ward & Watson 2013). *These results indicate that multitasking should be avoided.*

By analyzing the historical data of an executive recruitment firm, Aral, Brynjolfsson & Van Alstyne (2007) found that increasing multitasking could increase productivity. Beyond an optimum amount, increasing multitasking would reduce the project completion rates. Similar findings have also been observed in the judges in an Italian court (Coviello, Ichino & Persico 2010) and the physicians in emergence department (KC 2013). *These studies indicate that increasing number of tasks would increase productivity (like job completion rate) if a worker is under-loaded. However, productivity and even the job quality could be declined if a worker is over-loaded.*

2.2. Behavioral Economics Studies

Allocating effort to handle multiple jobs has long been a research problem in behavioral economics. The problem is solved by two different approaches. The first approach is to model the problem as a Markov decision problem. A worker has to work on multiple jobs in infinite horizon. In each time slot, the worker can only devote his effort on one job. If a job is allocated with effort, its quality level will be either incremented or staying the same. If a job has not been allocated with effort, its quality level will be either decremented or staying the same. The problem is to determine in each time slot which job the effort should be allocated so that all the quality levels can be optimized. Radner & Rothschild (1975), amongst the first research group, worked on this problem. Instead of determining the optimal effort allocation rule, they analyzed how the quality level changes under different rules of effort allocation (see (Radner & Rothschild 1975, p.360)).

The second approach of multitasking studies in behavioral economics is from a seminal work by Holmstrom & Milgram (1991). The effort allocation problem is modeled as a principal-agent problem (Itoh 1994, Prasad 2009). A worker (i.e. an agent) has to work on multiple jobs in a period of time and the worker has his own cost function to measure the effort he puts. The payment to the worker is determined by his output quality. The higher the quality, the more payment will be got. The utility of the worker is thus determined by the payment he gets minus the cost on his effort. The manager (i.e. the principal) has his profit function which is determined by the output of the worker. As a manager, he has to design the effort the worker will put on the jobs. So that both the profit is maximized as well as the expected utility of the worker is optimized (see (Holmstrom & Milgram 1991, Section 2)). Once the effort has been determined, the worker will work on the jobs in accordance with the determined effort.

While some other models were proposed and analyzed after the works by (Radner & Rothschild 1975) and (Holmstrom & Milgram 1991), none of them considered switching cost in their models. To handle multiple jobs, Rothschild (1974) considered that the incremental change of a job's quality level will be higher if a job gets more consecutive steps of effort. In other words, the incremental change of a job's quality will be higher if it has fewer interruptions. Seshadri & Shapira (2001) considered the situation that a worker has to handle one long-term primary job and many short-term secondary jobs. Similar to the problem setting in (Radner & Rothschild 1975), the worker in each time slot can only devote his effort on one job. If a job is allocated with effort, its quality level will be either incremented or staying the same. If a job has not been allocated with effort, its quality level will be either decremented or staying the same. But more than that, Seshadri & Shapira (2001) assumed that the primary job will get an immediate decrement of the quality level if it is interrupted by a short-term secondary job. Even in some recent papers like (Corts 2007, Prasad 2009, Matsushima, Miyazaki & Yagi 2010) which follow the principle-agent approach, switching cost has neither been considered.

To attain a theoretical basis for their findings from the Italian courts, Coviello, Ichino & Persico (2014) developed a stochastic model for single-worker environment. In their model, it is assumed that (i) all the jobs are of the same type and same size, (ii) the worker could randomly pick any number of jobs in the queue, and (iii) in any moment of time the worker allocates effort equally to all the incomplete jobs. Then, they (Coviello, Ichino & Persico 2014, Proposition 1) show that the job completion rate increases as the pick-up rate increases until the pick-up rate has reached a certain value. After that value, the productivity decreases as the pick-up rate increases. While Coviello, Ichino & Persico (2014) did introduce switching cost in their model, they assumed it zero in their analysis. Clearly, if the switching cost is non-zero, their theory will collapse.

Therefore, (i) allocation of effort to handle multiple jobs with inclusion of switching cost, (ii) modeling the effect of interruption and (iii) scheduling jobs in the presence of multitasking have yet to be investigated in behavioral economics.

3. SCHEDULING WITH MULTITASKING

In classical scheduling theory (Pinedo 1995), human multitasking behavior has also not been considered. Until recently, Hall, Leung & Li (2014) proposed a new scheduling model, in which interruption function and switching cost are introduced to capture the multitasking behavior.

3.1 Hall-Leung-Li Model

Given a set of jobs, a job J_i is characterized by a 5-tuple (p_i, d_i, w_i, $g_i(.)$, $f_{ij}(.)$), where p_i, d_i, and w_i, denote the processing time, due date and weight of J_i as in classical models. The function $g_i(.)$ is the interruption function and the function $f_{ij}(.)$ is the cost of switching from J_i to J_j. A schedule is denoted by $S = (J_{\pi_1}, J_{\pi_2}, \cdots, J_{\pi_n})$. By convention, J_{π_i} is scheduled before $J_{\pi_{i+1}}$. A job J_{π_i} is called primary in a time period when it is scheduled to be processed in that period. When J_{π_i} becomes a primary job, all the jobs scheduled before have complete. The execution of a primary job J_{π_i} will be interrupted by job J_{π_j} $g_{\pi_j}(i)$ units of time. To represent the remaining processing time of J_{π_i} after each interruption, we use $p_{\pi_i}(k)$ to denote the remaining processing time of J_{π_i} after k interruptions. Thus, we can have that $p_{\pi_i}(0) = p_{\pi_i}$,

$$p_{\pi_i}(k) = p_{\pi_i}(k-1) - g_{\pi_i}(k) \qquad (1)$$

for $1 \le k \le i-1$, and $p_{\pi_i}(k) = 0$, for $i \le k \le n$. We assume that $\sum_{k=1}^{n-1} g_i(k) < p_i$. Thus, $p_{\pi_i}(k)$ in (1) are positive for $1 \le k \le i$.

The interruption function $g_i(.)$ can be a constant function, a function of the remaining processing time $p_i(k)$ or a function k. For illustration, here are a few examples for $g_i(.)$. Let $n = 5$, $p_i = 10$ and $0 \le D < 1$.

$$\begin{aligned}
&Example\ 1: g_i(1) = g_i(2) = g_i(3) = g_i(4) = 1.\\
&Example\ 2: g_i(p_i(k)) = D p_i(k), \quad for \quad 1 \le k \le i-1.\\
&Example\ 3: g_i(1) = g_i(2) = 2, \quad g_i(3) = 3, \quad g_i(4) = 1.
\end{aligned}$$

For constant switching cost, Hall, Leung & Li (2014) investigate four different scheduling criteria, including (i) total weighted completion time (TWCT), (ii) maximum lateness (ML), (iii) total number of late jobs (TNLJ) and (iv) total weighted number of late jobs (TNWLJ), under the following conditions.

- *All the switching costs are constant.*
- *All jobs have to be processed.*
- *While a primary job is being processed, all waiting jobs will interrupt.*

In this regard, the completion time of J_{π_i} is given by

$$c_{\pi_i} = \sum_{j=1}^{i}\left(p_{\pi_j} + \sum_{j'=j+1}^{n} f_{\pi_j \pi_{j'}}\right) + \sum_{j=i+1}^{n}\left(\sum_{k=1}^{i} g_{\pi_j}(k)\right) \qquad (2)$$

for $i=1,2,\ldots,n-1$, and

$$c_{\pi_n} = p_{\pi_n} + \sum_{j=1}^{n-1}\left(p_{\pi_j} + \sum_{j'=j+1}^{n} f_{\pi_j \pi_{j'}}\right). \qquad (3)$$

Here f_{ij} is constant. They showed that the TWCT problem and ML problem are polynomial-time solvable. The TNLJ problem is NP-hard. The TNWLJ problem is strongly NP-hard. If the interruption function is defined as that $g_i(p'_i) = Dp'_i$, for $0 < D < 1$, the TNLJ problem can be solved by an $O(n\log(n))$ algorithm and the TNWLJ problem can be solved by a dynamic programming algorithm.

3.2 Our Model

In our model, the switching costs are job-dependent and symmetric, i.e. $f_{ij} = f_{ji}$. It is in line with the findings in (Gillie & Broadbent 1989, Trafton, Altmann, Brock & Mintz 2003). For the late job problems, the late jobs are discarded.

For the makespan and the total weighted completion time problems, the criteria are defined as follows:

$$\text{Makespan} = \max_i \{c_i\}. \qquad (4)$$

$$\text{TWCT} = \sum_{i=1}^{n} w_i c_i. \qquad (5)$$

The maximum weighted tardiness and maximum weighted lateness are defined as follows:

$$\text{MWT} = \max_i\{\max\{0, w_i(c_i - d_i)\}\}, \qquad (6)$$

$$\text{MWL} = \max_i\{c_i - d_i\}. \qquad (7)$$

As the late jobs will be discarded, our definitions of the late job problems are slightly different from the convention late job problems. Instead of minimizing the total (weighted) number of late jobs, we define an optimal schedule as the one that maximizes the total (weighted) number of on-time jobs. Without introducing new notations, we use the convention notation U_i to indicate if J_i is late. The criteria are given by

$$\text{TNLJ} = \sum_{i=1}^{n} U_i \qquad (8)$$

$$\text{TWNLJ} = \sum_{i=1}^{n} w_i U_i. \qquad (9)$$

3.3 Scheduling Problems

Follow Hall, Leung & Li (2014), we use ‘mt’ in the β field to denote ‘multitasking’. For our late job problems, we add ‘DLJ’ in the β field to synonymize discarding late jobs. The problems to be investigated are stated as following.

Problem: *Given a set of jobs $J = \{J_1, J_2, \ldots, J_n\}$ with known p_i, d_i, w_i, $g_i(p'_i)$, f_{ij}, for all $i,j = 1,2,\ldots,n$, find a* *feasible schedule $S=(J_{\pi_1},J_{\pi_2},\cdots,J_{\pi_n})$ such that the criterion z(S) is minimum.*

Here *z(S)* corresponds to any one of the criteria as defined in (4)-(9). Using the Graham notation, the problems are denoted in the following.

(P1) Makespan: $1|mt, f_{ij}=f_{ji}|\max_i\{c_i\}$.

(P2) TWCT: $1|mt, f_{ij}=f_{ji}|\sum_{i=1}^{n} w_i c_i$.

(P3) MWT: $1|mt, f_{ij}=f_{ji}|\max_i\{\max\{0,(c_i-d_i)\}\}$.

(P4) MWL: $1|mt, f_{ij}=f_{ji}|\max_i\{c_i-d_i\}$.

(P5) TNLJ: $1|mt, f_{ij}=f_{ji}, DLJ|\sum_{i=1}^{n} U_i$.

(P6) TWNLJ: $1|mt, f_{ij}=f_{ji}, DLJ|\sum_{i=1}^{n} w_i U_i$.

Without special notice, we use c_i to denote the completion time of the J_i in the optimal schedule. Unless we want to specify the completion times of the J_i in two different schedules *S* and *S'*, we use $c_i(S)$ and $c_i(S')$ respectively.

Remark on discarding late jobs: To explain the reason why we propose to discard late jobs, let us have a simple example. Suppose we have four jobs. Their processing times are 2, 3, 4 and 10. Their due dates are 4, 6, 9 and 15. Assume that there is no switching cost. Every time a waiting job interrupts a primary job, the amount of interruption time is 1. If all the jobs have to be processed, the optimal schedules are {2, 3, 1, 4}, {2, 4, 1, 3} or {3, 4, 1, 2}. In either case, only the first two jobs in the schedule are on-time. If we can discard late job, the optimal schedule is {1, 2, 3} and job 4 is discarded. The number of on-time jobs is 3. Clearly, more jobs can be scheduled on-time if the late jobs can be discarded.

4. MAIN RESULTS

Even with our new conditions on the switching cost and the way of handling late jobs, it can be shown that the first four problems P1 to P4 are easy problems. While the last two problems P5 and P6 are hard problems.

Due to page limit, we only outline the proofs. Readers are referred to (Sum, Leung & Ho 2014) for detail steps.

4.1 Makespan

For the makespan problem, one can readily show that the makespan is simply the completion time of the last job in the schedule. By (3), we can get that

$$c_{\pi_n}=\sum_{j=1}^{n} p_{\pi_j}+\sum_{j=1}^{n-1}\left(\sum_{j'=j+1}^{n} f_{\pi_j\pi_{j'}}\right). \qquad (10)$$

Obviously, it is a constant value. As $f_{ij} = f_{ji}$, interchanging the sequences of any two jobs does not change the value of makespan. We can state without proof the following theorem.

Theorem 1: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i, w_i, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all i,j = 1,2,…,n, the makespan is given by (10), a constant irrespective to the actual schedule.*

By definition, $f_{ii} = 0$, the makespan (10) can be written as follows:

$$c_{\pi_n}=\sum_{j=1}^{n} p_{\pi_j}+\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} f_{ij}.$$

Clearly, it is not depended on the actual schedule. If there is no switching cost, the makespan equals to the total processing time.

4.2 Total Weighted Completion Time

Recall from (1) that $g_i(k)$ is the amount of job J_i to be processed during its k^{th} interruption and $p_i(k)$ is the remaining processing time after the k^{th} interruption. The total weighted completion time problem can be solved by the following algorithm.

```
Algorithm TWCT
Step 1: Initialize $S' = J, S = \{\}$.
Step 2: For $l$ from $n$ to 2,
    $J_{\pi_l} = \arg\min_{J_i \in S'} \left\{ \frac{w_i}{p_i(l-1) + \sum_{J_j \in S} \left(g_j(l) + f_{ij}\right)} \right\}$,
    $S' = S' \setminus \{J_{\pi_l}\}$,  $S = \{J_{\pi_l}\} || S$.
Step 3: $S = S' || S$.
```

In Step 2-3, the operator $A||B$ means concatenating set *A* in front of set *B*. Note that the for-loop in Step 2 is in backward manner. It can be shown that the above algorithm can find an optimal schedule in $O(n^2)$ running time.

Theorem 2: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i, w_i, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all i,j = 1,2,…,n, the total weighted completion time problem can be solved in $O(n^2)$ running time.*

Outline of the Proof: The proof is accomplished by contradiction. The selection criterion in Step 2 implies that the schedule obtained by the Algorithm TWCT satisfies the following inequality for any pair of neighbor jobs. That is to say, for $l = 2,3,\cdots,n$,

$$\frac{w_{\pi_{l-1}}}{p_{\pi_{l-1}}(l-1)+\sum_{J_j\in S}(g_j(l)+f_{\pi_{l-1}j})} \geq \frac{w_{\pi_l}}{p_{\pi_l}(l-1)+\sum_{J_j\in S}(g_j(l)+f_{\pi_l j})}.$$

Suppose an optimal schedule, say S', in which one pair of neighbor jobs $J_{\pi'_{l-1}}$ and $J_{\pi'_l}$ do not fulfill this inequality. We can show that the total weighted completion time of S' can be reduced by interchanging jobs $J_{\pi'_{l-1}}$ and $J_{\pi'_l}$. So, it contradicts to the hypothesis that S' is an optimal.

Q.E.D.

Suppose $w_1=\cdots=w_n=1$, $f_{ij}(p_i,p_j)=\kappa(p_i,p_j)$ is an increasing function of p_i and p_j, such that

$$\frac{\partial}{\partial x}\kappa(x,y)\geq 0,\quad \kappa(x,y)=\kappa(y,x),\quad \kappa(0,0)=0. \quad (11)$$

Here are a few examples for the switching function.

$$0,\quad \xi,\quad \xi(x^a+y^a),\quad \xi(xy)^a,\quad \xi(log(xy)),$$

where a is an positive integer. The interruption function $g_i(p)=\nu(p)$, such that

$$\frac{\partial}{\partial x}\nu(x)\geq 0,\quad \nu(x)\geq 0,\quad \nu(0)=0 \qquad (12)$$

for $x\geq 0$. Here are a few examples for the interruption function.

$$0,\quad Dx,\quad Dx^a,\quad Dx^{1/a}\quad Dlog(1+x),$$

where a is a positive integer. Under such setting, the problem can be solved by the shortest processing time first (SPTF) rule. The optimal schedule is identical to the one without multitasking.

Theorem 3: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i, w_i=1, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all i,j = 1,2,…,n, and the switching functions and the interruption functions fulfill (11) and (12), the total completion time problem can be solved by SPTF rule in O(n log(n)) running time.*

Outline of the Proof: If all the weights are equal, the selection criterion in Step 2 can be rewritten as follows:

$$J_{\pi_l}=\underset{J_i\in S'}{arg\,max}\left\{p_i(l-1)+\sum_{J_j\in S}\left(g_j(l)+f_{ij}\right)\right\}.$$

Condition (12) implies that the remaining processing times $p_i(k)\geq p_j(k)$ for $k=1,2,\cdots,n-1$, if $p_i\geq p_j$. Condition (11) implies that $\kappa(p_i,p_\pi)\geq\kappa(p_j,p_\pi)$ if $p_i\geq p_j$. As a result, the job to be assigned as J_{π_l} must be the one with the largest processing time in the unscheduled job set. The optimal schedule for this special case is thus the same as the one obtained by SPTF rule.

Q.E.D.

Theorem 3 is an important theorem. For other scheduling problems, the optimal schedule obtained in the presence of multitasking, say S^{MUL}, is usually different from that obtained in the absence of multitasking, say S^{NM}. Thus, comparing the performances between S^{MUL}, and S^{NM} has to rely on computer simulations (Hall, Leung, Li 2014). Only for the total completion time problem with the conditions (11) and (12), $S^{MUL}=S^{NM}$. Then the effect of multitasking on the total completion time can then be analyzed mathematically.

4.3 Maximum Weighted Tardiness

To solve the MWT problem, the idea can be outline as following. Recall that the tardiness function for the job J_i is defined as follows:

$$T_i(x)=max\{0,x-d_i\}. \qquad (13)$$

In accordance with Theorem 1, we can get the completion time of the last job by (10). Let the value be c_{max}. Job J_{π_n} is thus assigned as the last job if the following condition is satisfied.

$$J_{\pi_n}=\underset{J_i\in J}{arg\,min}\{w_iT_i(c_{max})\}.$$

Once the last job has been determined, the completion time of the second last job can be calculated by

$$c=c_{max}-p_{\pi_n}(n-1).$$

Similarly, job $J_{\pi_{n-1}}$ is thus assigned as the last job if the following condition is satisfied.

$$J_{\pi_{n-1}}=\underset{J_i\in S'}{arg\,min}\{w_iT_i(c)\},$$

where $S'=S\setminus\{J_{\pi_n}\}$. So, the maximum weighted tardiness problem can be solved by the following algorithm.

```
Algorithm MWT
Step 1: Initialize  $S'=J, S=\{\}, c=c_{max}$.
Step 2: For $l$ from $n$ to 2,
        $J_{\pi_l}=\underset{J_i\in S'}{arg\,min}\{w_iT_i(c)\}$,
        $c=c-p_{\pi_l}(l-1)-\sum_{J_j\in S}(g_j(l)+f_{\pi_l j})$,
        $S'=S'\setminus\{J_{\pi_l}\},\quad S=\{J_{\pi_l}\}||S$.
Step 3: $S=S'||S$.
```

It can be shown that the above algorithm can find an optimal schedule in *O(n²)* running time. So, we can state without proof the following theorem.

Theorem 4: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i, w_i, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all i,j = 1,2,…,n, the* *maximum weighted tardiness problem can be solved in $O(n^2)$ running time.*

Two points should be noted from the above results. First, Algorithm MWT is applied to any arbitrary definitions of $g_i(k)$ as long as it has fulfilled the mild conditions mentioned in Section 3.1. Second, if all the weights are of equal value, say $w_1 = w_2 = \cdots = w_n = 1$, the job with the minimum tardiness $T_i(c_{max})$ is the one with the latest due date. Once the last job has been determined, the completion time of the second last job, say *c*, can readily be computed. Then, the unscheduled job with the minimum tardiness must be the one with the second latest due date. Repeating the steps for the completion of the third last job and so on, the optimal schedule for this special case is essentially the same as the schedule obtained by earliest due date (EDD) rule.

Theorem 5: *Given a set of jobs $J = \{J_1, J_2,...,J_n\}$ with known p_i, d_i, $w_i=1$, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all $i,j = 1,2,\ldots,n$, the maximum weighted tardiness problem can be solved by EDD rule in $O(n \log(n))$ running time.*

4.4 Maximum Weighted Lateness

To solve the MWL problem, the idea is the same as in Algorithm MWT. Replacing the tardiness function $T_i(x)$ in (13) and in Step 2 by $L_i(x)$, where

$$L_i(x) = max\{\, x - d_i \,\}. \qquad (14)$$

Accordingly, the maximum weighted lateness problem can be solved by the following algorithm.

```
Algorithm MWL
Step 1: Initialize  $S' = J, S = \{\}, c = c_{max}$.
Step 2: For $l$ from $n$ to 2,
        $J_{\pi_l} = \arg\min_{J_i \in S'}\{ w_i L_i(c)\}$,
        $c = c - p_{\pi_l}(l-1) - \sum_{J_j \in S} (g_j(l) + f_{\pi_l j})$,
        $S' = S' \backslash \{J_{\pi_l}\}$,  $S = \{J_{\pi_l}\} || S$.
Step 3: $S = S' || S$.
```

Similarly, we state without proof the following theorem.

Theorem 6: *Given a set of jobs $J = \{J_1, J_2,...,J_n\}$ with known p_i, d_i, w_i, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all $i,j = 1,2,\ldots,n$, the maximum weighted lateness problem can be solved in $O(n^2)$ running time.*

Similar to the MWT problem, we can apply EDD rule to obtain the optimal schedule for the MWL problem if all the weights are equal.

Theorem 7: *Given a set of jobs $J = \{J_1, J_2,...,J_n\}$ with known p_i, d_i, $w_i=1$, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all $i,j = 1,2,\ldots,n$, the maximum weighted lateness problem can be solved by EDD rule in $O(n \log(n))$ running time.*

4.5 Total Number of Late Jobs

Recall that the late jobs are discarded in our late jobs problems. For the TNLJ problem, it can be shown that the problem is NP-hard.

Theorem 8: *Given a set of jobs $J = \{J_1, J_2,...,J_n\}$ with known p_i, d_i, $g_i(p'_i)$, $f_{ij}=f_{ji}$, for all $i,j = 1,2,\ldots,n$, the total number late jobs problem with late jobs being discard is NP-hard.*

Outline of Proof: The proof is by reduction from the Partition problem. Furthermore, we consider a special case that $f_{ij} = 0$. If this special case is NP-hard, the general case must be NP-hard.

Given a set *A* of *2q* integers, the problem asks if there exists a subset A' such that $|A'| = q$ and $\sum_{i\in A'} a_i = \sum_{i\in A\backslash A'} a_i$.

The corresponding instance of our late job problem can be constructed as that. Let $n = 3q$ and $B = (\sum_{i\in A} a_i)/2$. The processing times, due dates and interruption functions of the jobs are given by

$$p_i = \begin{cases} a_i + B & for\ i = 1,2,\cdots,2q, \\ 1 & for\ i = 2q+1, 2q+2,\cdots,3q. \end{cases}$$

$$d_i = \begin{cases} q + (q+1)B & for\ i = 1,2,\cdots,2q, \\ q + \dfrac{(q-1)B}{N} & for\ i = 2q+1,\cdots,3q. \end{cases}$$

$$g_i(k) = \begin{cases} \dfrac{B - a_i}{qN} & for\ i = 1,2,\cdots,2q, \\ 0 & for\ i = 2q+1,\cdots,3q, \end{cases}$$

for $k = 1,2,\cdots,3q-1$. Here N is an arbitrary large positive integer to ensure that $p_i(k) > 0$.

If part: Jobs $J_{2q+1},\cdots,J_{3q}$ are the enforcement jobs. As their due dates are earlier than the other jobs, they must be scheduled first. If there is a solution for the partition problem, it can be easily shown that there exists a subset A', in which the corresponding jobs are scheduled after the enforcement jobs. The completion times of the last enforcement job and the last job in the schedule will meet their due dates.

Only If part: On the other hand, if there is an optimal schedule for the late job problem, there exists a subset

J' of jobs such that the completion times of the last enforcement job and the last job in the schedule must fulfill the following inequalities.

$$q + q\sum_{i\in J'} \frac{B-a_i}{qN} \leq q + (q-1)\frac{B}{N},$$
$$q + \sum_{i\in J'}(a_i + B) \leq q + (q+1)B.$$

The first inequality implies that $\sum_{i\in J'} a_i \geq B.$ The second inequality implies that $\sum_{i\in J'} a_i \leq B.$ Hence, $\sum_{i\in J'} a_i = B.$ So that, we can assign $A' = J'$. Clearly, all the sum of the elements in A' must be equal to B.

As partition is NP-hard (Garey & Johnson 1979), the problem $1|mt, f_{ij} = f_{ji}, DLJ|\sum_i U_i$ must be NP-hard.

Q.E.D.

It should be noted that Hall, Leung & Li (2014) have showed that the problem

$$1|mt, p_i, d_i, g_i(p'_i) = Dp'_i, f_{ij} = \xi|\sum_i U_i$$

can be solved by an $O(n\log(n))$ algorithm. However, we have found that their algorithm is not applicable to our late job problem even if there is no switching cost. The complexity of the following problem is still an open problem.

$$1|mt, p_i, d_i, g_i(p'_i) = Dp'_i, f_{ij} = 0, DLJ|\sum_i U_i.$$

For $g_i(p'_i) = Dp'_i$, all the jobs have the same due date d, and there is no switching cost, the problem can be solved by the following polynomial time algorithm. Here we use 'SC1' referring to 'special case 1'.

Algorithm TNLJ-SC1

```
Step 1: Sort the jobs by their processing times s.t.
        $p_{\pi_i} \leq p_{\pi_{i+1}}, i = 1, \cdots, n-1.$
        Set $S = \{\}, c_{max} = 0, l = 0.$
Step 2: While $(c_{max} \leq d)$,
            $l = l + 1,$
            $c_{max} = c_{max} + p_{\pi_l},$
            $S = S || \{J_{\pi_l}\}.$
Step 3: If $(c_{max} > d)$, $S = S \setminus \{J_{\pi_l}\}.$
```

Theorem 9: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i=d, $g_i(p'_i)$=Dp'_i, f_{ij}=f_{ji}, for all i,j = 1,2,…,n, the total number late jobs problem with late jobs being discard is polynomial time solvable.*

We have also identified two special cases which are solvable in $O(n^2 \log(n))$ runtime.

$$1|mt, p_i = p, d_i, g_i(p'_i) = Dp'_i, f_{ij} = 0, DLJ|\sum_i U_i.$$
$$1|mt, p_i > n, d_i, g_i = 1, f_{ij} = 0, DLJ|\sum_i U_i.$$

For the first case, all jobs have the same processing time, their due dates are different and there is no switching cost. For the second case, the jobs have different processing times and different due dates. The interruption functions of all the jobs are the same constant, say $g_i(k) = 1$ for $i = 1, \cdots, n$ and there is no switching cost.

4.6 Total Weighted Number of Late Jobs

For the TWNLJ problem, it can be shown that the problem is strongly NP-hard.

Theorem 10: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, d_i, w_i, $g_i(p'_i)$, f_{ij}=f_{ji}, for all i,j = 1,2,…,n, the total weighted number of late jobs problem with late jobs being discard is strongly NP-hard.*

Outline of Proof: The proof is by reduction from the Exact Cover by 3-Sets (X3C) problems. As the construction of the instance is cumbersome, it is advised that interested readers refer to our unpublished manuscript (Sum, Leung & Ho 2014).

Q.E.D.

For the special case that the due dates of the jobs are the same, the interruption function is proportional to the remaining process time and there is no switching cost, $1|mt, p_i, d_i = d, g_i(p'_i) = Dp'_i, f_{ij} = 0, DLJ|\sum_i w_i U_i$, it can readily be shown that it is equivalent to the knapsack problem. So, we can state without proof the following algorithm for this special case of total weighted number of late jobs.

Theorem 11: *Given a set of jobs J = {J_1, J_2,...,J_n} with known p_i, w_i, d_i=d, $g_i(p'_i)$=Dp'_i, f_{ij}=f_{ji}, for all i,j = 1,2,…,n, the total number late jobs problem with late jobs being discard is NP-hard.*

Any pseudo-polynomial time algorithm applied to solve the knapsack problem can clearly be applied to solve this special case.

5. CONCLUSIONS

In this paper, we have reviewed some key research results in the areas of psychology, management, information systems and behavioral economics regarding the issue of multitasking. The laboratory studies have revealed that our brain structures are not designed for handling multitasking. Thus, multitasking could cause mental problem to human being. Besides, people peceived themselves good at multitasking always underperform as compared with people who engage in sequential processing. Field studies on daily works of the executives and the office workers have revealed that job inerruptions introduce costs on jobs switching and this cost could hamper the productivity of a worker. Some field studies have found that increasing the number of concurrent tasks could increase the productivity of a worker. Beyond a threshold, the productivity drops as the number of concurrent tasks increases. If this phenomena is a manifest of result of underloaded and overloaded, the benefit of multitasking is still unclear. However, in terms of mental health and productivity, multitasking should be avoided. In behavioral economics, reserachers have studied the problem of multitasking (in the topic of effort allocation) for half a century, switching cost has not been considered in their analysis and scheduling of jobs has not been investigated.

Scheduling with multitasking is a relatively new topic in the area of scheduling theory. Hall, Leung & Li (2014) introduced the ideas of interruption function and switching cost to some classical single machine scheduling problems. Extend from their formulations, we introduce in this paper two new conditions. First, the switching costs are job-dependent and symmetric. Second, for the late job problems, we consider that the late jobs are not allowed to interrupt. The scheduling problems of six different criteria are then investigated, including the makespan problem, the TWCT problem, the MWT problem, the MWL problem, the TNLJ problem and the TWNLJ problem.

We show that the makespan can be evaluated in advance if both the processing times of the jobs and their switching costs are given. The TWCT, MWT and MWL problems are polynomial-time solvable. If the interruption function and the switching cost function fulfill the conditions as stated in (11) and (12), the total completion problem (TCT) can be solved in $O(n\log(n))$ runtime. Moreover, the optimal schedule of the TCT problem in the presence of multitasking is equivalent to the optimal schedule of the TCT problem in the absence of multitasking. For the unweighted cases of MWT and MWL, i.e. the maximum tardiness problem and the maximum lateness problem, they can be solved in $O(n\log(n))$ runtime. For our late jobs problems, we have showed that The TNLJ problem is NP-hard and the TWNLJ is strongly NP-hard. Under the special case that $g_i(p'_i) = Dp'_i$, all the jobs have the same due dates and there is no switching cost, the TNLJ problem can be solved in $O(n\log(n))$runtime but the TWNLJ is NP-hard.

Nevertheless, it is still unknown on the complexity of our TNLJ problem under the special case that $g_i(p'_i) = Dp'_i$, the jobs have the different processing times and due dates, and there is no switching cost. We leave this problem open for future research.

The results presented in this paper can help the human workers to schedule their jobs so as to minimize the effect of multitasking.

Acknowledgement

The works presented in this paper is support in part by the MOST Taiwan research grant, number MOST 103-2410-H-005 -036.

REFERENCES

[1] American Psychological Association, “Multitasking: Switching costs”. 2006. Available online: www.apa.org/research/action/multitask.aspx.

[2] Aral, S., E. Brynjolfsson, and M. Van Alstyne. “Information, technology, and information worker productivity.” *Information Systems Research* Vol.23(3)-part-2, pp.849-867, 2012.

[3] Brucker, P., Scheduling Algorithms, 5th Edition, Springer-Verlag Berlin Heidelberg, 2007.

[4] Corts, K.S., “Teams versus individual accountability: Solving multitask problems through job design”, *RAND Journal of Economics*,Vol.38(2), 467-479, 2007.

[5] Coviello, D., A. Ichino, and N. Persico. “Don’t spread yourself too thin: the impact of task juggling on workers speed of job completion”, National Bureau of Economic Research (NBER) Working paper No.16502, 2010.

[6] Coviello, D., A. Ichino, N. Persico. “Time allocation and task juggling”, *The American Economic Review*, 104(2), 609-623, 2014.

[7] Deshmukh, S. D., D.S. Chikte, “Stochastic evolution and control of an economic activity”, *Journal of Economic Theory*, 15(1), 112-122, 1977.

[8] Du, J., J.Y.T. Leung, “Minimizing total tardiness in one machine is NP-hard”, *Mathematics of Operations Research*, Vol.15, 483-495, 1990.

[9] Foerde, K., B.J. Knowlton, R.A. Poldrack, “Modulation of competing memory systems by distraction”, *Proceedings of the National Academy of Sciences*, 103(31), 11778-11783, 2006.

[10] Garey, M.R., D.S. Johnson. Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman, New York. 1979.

[11] Gillie, T., D. Broadbent,“What makes interruptions disruptive? A study of length, similarity, and complexity”, *Psychological Research*, Vol.50(4), 243-250, 1989.

[12] Hall, N.G., J.Y.T. Leung, C.L. Li, “The effects of multitasking on operations scheduling”, In submission, 2014.

[13] Hallowell, E.M., “Overloaded circuits: Why smart people underperform”, *Harvard Business Review*, Vol.83(1), 55-62, 2005.

[14] Holmstrom, B. & P. Milgrom, "Multitask principal-agent analyses: Incentive contracts, asset ownership, and job design", *Journal of Law, Economics & Organization*, Vol.7, 24-52, 1991.

[15] Itoh, H., "Job design, delegation and cooperation: A principal-agent analysis", *European Economic Review*, 38(3), 691-700, 1994.

[16] Jackson, J.R., "Scheduling a production line to minimize maximum tardiness", Research Report No. 43, Management Science Research Report, UCLA, 1995.

[17] Jett, Q.R., J.M. George, "Work interrupted: A closer look at the role of interruptions in organizational life", *Academy of Management Review*, Vol.28, 494-507, 2003.

[18] Karp, K. M., "Reducibility among combinatorial problems", In R.E. Miller, J.W. Thatcher, eds. Complexity of Computer Computations. Plenum, New York, 85-103, 1972.

[19] KC, Diwas Singh. "Does multitasking improve performance? Evidence from the emergency department". *Manufacturing & Service Operations Management*, Vol.16(2), pp.168-183, 2013.

[20] Lageweg, B.J., J.K. Lenstra, E.L. Lawler, A.H.G. Rinnooy Kan, "Computer-aided complexity classification of combinatorial problems", *Communications of ACM*, Vol.25, 817-822, 1982.

[21] Lawler, E.T., "Optimal sequencing of a single machine subject to precedence constraints", *Management Science*, Vol.19, 544-546, 1973.

[22] Lawler, E.L., "A 'pseudopolynomial' time algorithm for sequencing jobs to minimize total tardiness", *Annals of Discrete Mathematics*, Vol.1, 331-342, 1977.

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

[24] Lee, F.J., N.A. Taatgen NA., "Multi-tasking as skill acquisition", *Proceedings of the Twenty-Fourth Annual Conference of the Cognitive Science Society*, pp. 572-577, 2002.

[25] Lenstra, J.K., A.H.G. Rinnooy Kan, P. Bruker "Complexity of machine scheduling problems", *Annals of Discrete Mathematics*, Vol.1, 343-362, 1977.

[26] Lenstra, J.K., A.H.G. Rinnooy Kan, "Complexity of scheduling under precedence constraints", *Operations Research*, Vol.26(1), 22-35, 1978.

[27] Loukopoulos, L.D., R.K. Dismukes, I. Barshi, The Multitasking Myth: Handling complexity in real world operations, Ashgate Publishing Limited, England, 2009.

[28] Mark, G., V.M. Gonzalez, J. Harris, "No task left behind?: examining the nature of fragmented work", *Proceeedings of ACM SGICHI 2005 Conference on Human Factors in Computing Systems*, pp.321-330, 2005.

[29] Matsushima, H., K. Miyazaki, N. Yagi, "Role of linking mechanisms in multitask agency with hidden information", *Journal of Economic Theory*, Vol.145(6), 2241-2259, 2010.

[30] McNaughton, R., "Scheduling with deadlines and loss functions", *Management Science*, Vol.6, pp.1-12, 1959.

[31] Meuter, R.F., A. Allport, "Bilingual language switching in naming: Asymmetrical costs of language selection", *Journal of Memory and Language*, Vol.40(1), 25-40, 1999.

[32] Mintzberg, H., The Nature of Managerial Work, Harper & Row, New York, 1973.

[33] Monsell, S., "Task switching", *Trends in Cognitive Sciences*, Vol.7(3), 134-140, 2003.

[34] Moore J.M., "An n job, one machine sequencing algorithm for minimizing the number of late jobs", *Management Science*, Vol.14, pp.102-109, 1968.

[35] Ozmutlu, S., H.C. Ozmutlu, A. Spink, "Multitasking web searching: implications for design", *Proceedings of ASIST03: Annual meeting of the American Society for Information Science and Technology*, 2003.

[36] Pinedo, M., Scheduling Theory, Algorithms, and Systems, Prentice Hall, Englewood Cliffs, NJ, 1995.

[37] Prasad, S., "Task assignments and incentives: generalists versus specialists", *RAND Journal of Economics*, Vol.40(2), pp.380-403, 2009.

[38] Radner, R., M. Rothschild, "On the allocation of effort", *Journal of Economic Theory*, Vol.10(3), pp.358-376, 1975.

[39] Rogers, R.D., S. Monsell, "Costs of a predictible switch between simple cognitive tasks", *Journal of Experimental Psychology: General*, Vol.124(2), 207, 1995.

[40] Rosen, C., "The myth of multitasking", The New Atlantis, 20(Spring), pp.105-110, 2008.

[41] Rothschild, M., "Further notes on the allocation of effort", Princeton University, Econometric Research Program Report No.171, 1974.

[42] Rubinstein, J.S., D.E. Meyer, J.E. Evans, "Executive control of cognitive processes in task switching", *Journal of Experimental Psychology: Human Perception and Performance*, Vol.27(4), pp.763, 2001.

[43] Salvucci, D.D., N.A. Taatgen, J.P. Borst, "Toward a unified theory of the multitasking continuum: From concurrent performance to task switching,

interruption, and resumption", *Proceeedings of ACM SGICHI 2009 Conference on Human Factors in Computing Systems*, pp.1819-1828, 2009.

[44] Sanbonmatsu,D.M.,D.L.Strayer, N.Medeiros-Ward, J.M. Watson, "Who multi-tasks and why? Multi-tasking ability, perceived multi-tasking ability, impulsivity, and sensation seeking", *PloS One*, Vol.8(1), e54402, 2013.

[45] Seshadri,S., Z.Shapira, "Managerial allocation of time and effort: The effects of interruptions", *Management Science*, Vol.47, 647-662, 2001.

[46] Spink,A., H.C.Ozmutlu, S.Ozmutlu, "Multitasking information seeking and searching processes", *Journal of the American Society for Information Science and Technology*, Vol.53(8), 639-652, 2002.

[47] Spink,A., "Multitasking information behavior and information task switching: An exploratory study", *Journal of Documentation*, 60(3), 336-345, 2004.

[48] Spink, A., M. Park, B.J. Jansen, J. Pedersen, "Multitasking during web search sessions", *Information Processing and Management*, Vol.42(1), pp.264-275, 2006.

[49] Sum, J., C.S. Leung, K.I-.J. Ho, "Intractability of the operations scheduling in the presence of multitasking", unpublished manuscript, 2014.

[50] Taatgen, N.A., F.J. Lee, "Production compilation: A simple mechanism to model complex skill acquisition", *Human Factors: The Journal of the Human Factors and Ergonomics Society*, Vol.45(1), pp.61-67, 2003.

[51] Trafton, J.G., E.M. Altmann, D.P. Brock, F.E. Mintz, "Preparing to resume an interrupted task: Effects of prospective goal encoding and retrospective rehearsal", *International Journal of Human-Computer Studies*, Vol.58, pp.583-603, 2003.

[52] Watson, J.M., D.L. Strayer, "Supertaskers: Profiles in extraordinary multitasking ability", *Psychonomic Bulletin & Review*, Vol.17(4), pp.479-485, 2010.