OPERATIONS RESEARCH
DOI 10.1287/opre.1090.0744ec

e-companion

ONLY AVAILABLE IN ELECTRONIC FORM

Electronic Companion—"Competitive Two-Agent Scheduling and Its Applications" by Joseph Y.-T. Leung, Michael Pinedo, and Guohua Wan, *Operations Research*, DOI 10.1287/opre.1090.0744.

Competitive Two-agent Scheduling and its Applications

Joseph Y-T. Leung

Department of Computer Science

New Jersey Institute of Technology, Newark, NJ 07102, leung@oak.njit.edu

Michael Pinedo

Stern School of Business, New York University

44 West Fourth Street, New York, NY 10012, mpinedo@stern.nyu.edu

Guohua Wan

Antai College of Economics and Management

Shanghai Jiao Tong University, Shanghai 200052, China, ghwan@sjtu.edu.cn

APPENDIX: Proof of Theorem 1

We will prove Theorem 1 through a reduction from the Even-Odd Partition problem. The Even-Odd Partition problem has been shown to be binary NP-complete by Garey, Tarjan and Wilfong (1988); see also Garey and Johnson (1979).

Even-Odd Partition: Given $2n$ positive integers $a_1 < a_2 < \cdots < a_{2n}$, where $A = \frac{1}{2}\sum_{j=1}^{2n} a_j$, is there a partition of these integers into two sets A_1 and A_2, such that $\sum_{a_j \in A_1} a_j = \sum_{a_j \in A_2} a_j = A$, where A_1 and A_2 each contains exactly one element from $\{a_{2i-1},\ a_{2i}\},\ i = 1, 2, \ldots, n$?

Let $\sigma_i = a_{2i} - a_{2i-1}, i = 1, \ldots, n$. Notice that since each pair of integers, a_{2i-1} and a_{2i}, must

be put into two different sets, we can add a constant c_i to each pair without changing the problem instance. By carefully choosing c_i, we may assume that the given instance of Even-Odd Partition satisfies the following properties:

Property 1: $a_1 > (2n+2)\max(\sigma_1, \ldots, \sigma_n)$.

Property 2: $a_{2i-1} > \sum_{j=1}^{2i-2} a_j$.

Moreover, we may assume:

Property 3: a_i/j is an integer for each $1 \le i \le 2n$ and $1 \le j \le n$.

If this is not true, we can multiply each a_i by $n!$ without changing the problem instance. Note that although the numbers a_i may become exponential, the size of the binary input remains polynomial.

Given an instance of the Even-Odd Partition problem, we create an instance I of our scheduling problem as follows: There are $2n$ P-jobs, each of which corresponds to an integer in the Even-Odd Partition instance, and a large R-job for Agent B. There are n Q-jobs for Agent A. The processing times and due dates of these jobs are shown in Table 5, where

Table 1: Job data in instance I

Job	Processing Time	Due Date
P_{2i-1}	a_{2i-1} $(= p_{2i-1})$	$\sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i-1} x_k + p_{2i-1}$
P_{2i}	$a_{2i} + (l_i - 1)\sigma_i$ $(= p_{2i})$	$\sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i} x_k + p_{2i}$
R	L	$\sum_{i=1}^{n} x_i + [A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i] + L$
Q_i	x_i	

- L is an integer larger than $2A$.

• $x_1 = 1,\ x_i = \frac{n-i+1}{n-i+2}a_{2i-3}$ for $i = 2, \ldots, n-1$, and $x_n = \frac{1}{2}a_{2n-3} + a_{2n-5}$. Note that $x_1 < x_2 < \cdots < x_n$ and they are all integers.

• $l_i\sigma_i = \frac{1}{n-i+1}a_{2i-1}$ for $i = 1, \ldots, n$. By Property 3, $l_i\sigma_i$ is an integer.

Let the threshold for the total completion time of Agent A be TC, where

$$TC = \sum_{i=1}^{n}(n-i)[a_{2i} + (l_i - 1)\sigma_i] + \sum_{i=1}^{n}(n-i+1)x_i + \frac{1}{2}\sum_{i=1}^{n} l_i\sigma_i = A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i,$$

and let the threshold for the number of tardy jobs of Agent B be n. Clearly, this transformation can be done in polynomial time.

Note that based on the data given in Table 5, we have

• processing times $p_1 < p_2 < \cdots < p_{2n-1} < p_{2n}$, and

• due dates $d_{P_1} < d_{P_2} < \cdots < d_{P_{2n-1}} < d_{P_{2n}} < d_R$. Figure 2 shows the due date pattern of the jobs.

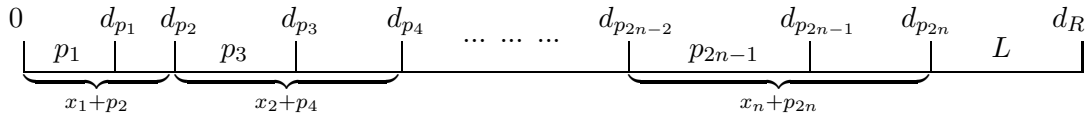

Figure 1: The due dates of jobs in instance I

In an instance I of the scheduling problem $1 \mid\mid \sum C_j^a : \sum U_j^b$, a schedule is said to be *feasible* if the number of tardy jobs of agent B is less than or equal to n. The decision problem asks: is there a feasible schedule with total completion time of the jobs of Agent A less than or equal to TC? That is, is there a solution to the problem $1 \mid\mid \sum C_j^a \leq TC : \sum U_j^b \leq n$?

First of all, we have:

Lemma 1: *Given an instance of the Even-Odd Partition problem, if there is a solution to this instance, then there exists a solution to the corresponding instance of the problem* $1 \mid\mid \sum C_j^a \leq TC :$

$\sum U_j^b \leq n$.

Proof: If there is a solution to an instance of the Even-Odd Partition problem, i.e., there is a partition A_1 and A_2 such that $\sum_{a_j \in A_1} a_j = \sum_{a_j \in A_2} a_j = A$, where A_1 and A_2 each contains exactly one element from $\{a_{2i-1},\ a_{2i}\},\ i = 1, 2, \ldots, n$, then we can schedule the jobs of the corresponding instance I as follows.

First, the R-job is scheduled to complete exactly at its due date. Second, the n P-jobs corresponding to A_1 are scheduled before the R-job in EDD order. Third, the n Q-jobs are scheduled together with the n P-jobs corresponding to A_1 as follows: if the P-job corresponds to a_{2i}, then job Q_i is scheduled just before P_{2i} as Q_iP_{2i}; if the P-job corresponds to a_{2i-1}, then job Q_i is scheduled just after P_{2i-1} as $P_{2i-1}Q_i$. Finally, the n P-jobs corresponding to A_2 are scheduled after job R in any order. It is easy to check that this schedule is feasible (the n P-jobs corresponding to A_1 and the R-job are non-tardy) and the total completion time (of the n Q-jobs) of Agent A is exactly TC. This means that there is a solution to the instance I of the scheduling problem. □

In the following, we show that if there exists a solution to an instance of the problem $1 \mid\mid \sum C_j^a \leq TC : \sum U_j^b \leq n$, then there exists a solution to the corresponding instance of the Even-Odd Partition problem. In fact, we will show that for any feasible schedule, the minimum value of $\sum C_j^a$ is exactly TC. In other words, any schedule with $\sum C_j^a < TC$ must not be a feasible schedule. Since the threshold for $\sum C_j^a$ in the problem $1 \mid\mid \sum C_j^a \leq TC : \sum U_j^b \leq n$ is TC, a solution to the problem must imply that $\sum C_j^a = TC$. Moreover, to obtain a schedule with $\sum C_j^a = TC$, there must be a solution to the corresponding instance of the Even-Odd problem.

The basic idea of the proof is to define a P-job for each integer a_j, and a large R-job whose due date is the largest among all the jobs, for Agent B. (Here, the P-job corresponding to a_{2j} is called

an *even* P-job, and the P-job corresponding to a_{2j-1} is called an *odd* P-job.) For Agent A, we define n Q-jobs each of which can and has to be scheduled in between an adjacent pair of P-jobs in SPT order. By properly choosing the processing times and due dates of the jobs, we can show that for any feasible schedule of instance I the following four assertions hold:

(a) Exactly one job from each pair $\{P_{2i-1}, P_{2i}\}$ must be tardy.

(b) The R-job must be on time and scheduled after all the other on-time P-jobs.

(c) The Q-jobs must be scheduled in SPT order before the R-job.

(d) The total processing time of the on-time P-jobs cannot exceed $A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i$.

It can be shown that for every i, there are two ways to schedule P_{2i-1}, P_{2i} and Q_i: either P_{2i-1} followed by Q_i (which implies that P_{2i} will be tardy), or Q_i followed by P_{2i} (which implies that P_{2i-1} will be tardy). In order to minimize the total completion time of the Q-jobs (which are the jobs of Agent A), we need to have more even P-jobs on time. It can be shown that every time we interchange a pair of even and odd P-jobs by making the even P-job tardy and the odd P-job on-time, the total completion time of Agent A will be increased by a quantity equal to the difference between the processing times of the two P-jobs, which is exactly the quantity reduced in the total processing time of the on-time P-jobs. Thus, a feasible schedule with $\sum C_j^a \leq TC$ is obtained when the total processing time of the on-time P-jobs is exactly $A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i$. But this occurs only when there is a solution for the instance of the Even-Odd Partition problem. Notice that the first two terms in the formula of TC represent the total completion time when all the even P-jobs are on time (which does not yield a feasible schedule since the R-job will be tardy). The last term is the minimum increase in the total completion time when the total processing time of the on-time P-jobs is reduced to $A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i$ (which yields a feasible schedule since the R-job will then be on time).

Before we proceed to prove the assertions made above, we first prove the following inequalities in order to facilitate the presentation of the proofs.

Lemma 2: (2.1) $\sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i} x_k < p_{2i-1}$ *for* $i = 1, \ldots, n-1$.

(2.2) $\sum_{k=1}^{n-1} p_{2k} + \sum_{k=1}^{n} x_k < p_{2n-5} + p_{2n-1}$.

Proof: (2.1) For $i = 1$, the left hand side is $x_1 = 1$ and the right hand side is $p_1 = a_1$. Clearly, the left hand side is smaller than the right hand side. We now consider $i > 1$.

Recall that

$$l_k \sigma_k = \frac{1}{n-k+1} a_{2k-1}, \qquad k = 1, \ldots, n.$$

Furthermore, $x_1 = 1$ and

$$x_k = \frac{n-k+1}{n-k+2} a_{2k-3}, \qquad k = 2, \ldots, n-1.$$

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

$$\begin{aligned}
\sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i} x_k &= \sum_{k=1}^{i-1} [a_{2k} + (l_k - 1)\sigma_k] + \sum_{k=1}^{i} x_k \\
&= \sum_{k=1}^{i-1} (a_{2k} + l_k \sigma_k - \sigma_k) + x_1 + \sum_{k=1}^{i-1} x_{k+1} \\
&= \sum_{k=1}^{i-1} (a_{2k} + \frac{a_{2k-1}}{n-k+1} - \sigma_k) + x_1 + \sum_{k=1}^{i-1} \frac{n-k}{n-k+1} a_{2k-1} \\
&= \sum_{k=1}^{i-1} a_{2k} + x_1 - \sum_{k=1}^{i-1} \sigma_k + \sum_{k=1}^{i-1} a_{2k-1} \ \leq \ \sum_{k=1}^{i-1} a_{2k} + \sum_{k=1}^{i-1} a_{2k-1} \\
&< \ a_{2i-1} \ = \ p_{2i-1}. \text{ (By Property 2)}
\end{aligned}$$

(2.2) Recall that $x_n = \frac{1}{2} a_{2n-3} + a_{2n-5}$. Thus,

$$\sum_{k=1}^{n-1} p_{2k} + \sum_{k=1}^{n} x_k = \sum_{k=1}^{n-1} [a_{2k} + (l_k - 1)\sigma_k] + \sum_{k=1}^{n} x_k$$

$$= \sum_{k=1}^{n-1}(a_{2k} + \frac{1}{n-k+1}a_{2k-1} - \sigma_k) + x_1 + \sum_{k=1}^{n-1}\frac{n-k}{n-k+1}a_{2k-1} + a_{2n-5}$$

(since $x_n = \frac{1}{2}a_{2n-3} + a_{2n-5}$)

$$\leq \sum_{k=1}^{n-1}(a_{2k} + a_{2k-1}) + a_{2n-5} < a_{2n-1} + a_{2n-5} = p_{2n-5} + p_{2n-1}$$

(by Property 2). □

Lemma 3:

$$\sum_{k=1}^{i-1} p_{2k} < \sum_{k=1}^{i-1} p_{2k-1} + x_i, \qquad i = 2, 3, \ldots, n.$$

Proof: For $i = 2$, the left hand side is

$$p_2 = a_2 + (l_1 - 1)\sigma_1 = a_1 + \sigma_1 + (l_1 - 1)\sigma_1 = a_1 + l_1\sigma_1 = a_1 + \frac{1}{n}a_1$$

and the right hand side is

$$p_1 + x_2 = a_1 + \frac{n-1}{n}a_1.$$

Clearly, the left hand side is smaller than the right hand side.

For $i = 3, \ldots, n-1$, we have

$$x_i = \frac{n-i+1}{n-i+2}a_{2i-3} = \frac{n-i}{n-i+2}a_{2i-3} + \frac{1}{n-i+2}a_{2i-3}$$
$$> \frac{n-i}{n-i+2}\sum_{j=1}^{2i-4} a_j + \frac{1}{n-i+2}a_{2i-3}$$
$$> \frac{n-i}{n-i+2}\sum_{k=1}^{i-2} a_{2k-1} + \frac{1}{n-i+2}a_{2i-3}$$

(by Property 2)

$$= \sum_{k=1}^{i-2}\frac{n-i}{n-i+2}a_{2k-1} + \frac{1}{n-i+2}a_{2i-3}$$

$$> \sum_{k=1}^{i-2} \frac{1}{n-k+1} a_{2k-1} + \frac{1}{n-i+2} a_{2i-3}$$

(since $n - i > 1$)

$$= \sum_{k=1}^{i-1} \frac{1}{n-k+1} a_{2k-1} = \sum_{k=1}^{i-1} l_k \sigma_k.$$

For $i = n$,

$$x_n = \frac{1}{2} a_{2n-3} + a_{2n-5} > \frac{1}{2} a_{2n-3} + (\frac{1}{3} a_{2n-5} + \cdots + \frac{1}{n} a_1)$$

(by Property 2)

$$= \frac{1}{2} a_{2n-3} + \sum_{k=1}^{n-2} \frac{1}{n-k+1} a_{2k-1} = \sum_{k=1}^{n-1} \frac{1}{n-k+1} a_{2k-1} = \sum_{k=1}^{n-1} l_k \sigma_k.$$

Thus, we have $\sum_{k=1}^{i-1} l_k \sigma_k < x_i$ for all $i = 2, \ldots, n$. Therefore, for all $i = 2, \ldots, n$,

$$\sum_{k=1}^{i-1} p_{2k} = \sum_{k=1}^{i-1} [p_{2k-1} + (l_k - 1)\sigma_k] < \sum_{k=1}^{i-1} p_{2k-1} + x_i.$$

□

We now prove assertions (a) and (b) through Lemmas 4 and 5. These two lemmas characterize the basic structure of a feasible schedule for instance I.

Lemma 4: *In a feasible schedule:*

(4.1) *there are exactly n tardy jobs;*

(4.2) *the R-job is non-tardy and scheduled after all other non-tardy jobs;*

(4.3) *at least one job from* $\{P_{2i-1}, P_{2i}\}$, $i = 1, 2, \ldots, n$, *must be non-tardy.*

Proof: (4.1) It is sufficient to consider the jobs of Agent B only since the jobs of Agent A have nothing to do with the feasibility of a schedule. Recall that the Hodgson-Moore algorithm yields

a schedule with the minimum number of tardy jobs. Thus, it is sufficient to show that there are exactly n tardy jobs when the Hodgson-Moore algorithm is applied to instance I. The Hodgson-Moore algorithm scans jobs in increasing order of their due dates. In the course of scheduling, if a job misses its due date, then the job with the largest processing time among all jobs currently in the schedule (including the job that misses its due date), will be chosen as a tardy job and deleted from the schedule. The algorithm then continues to scan the next job until all jobs have been processed. Finally, the tardy jobs (that were deleted from the schedule) will be scheduled after the non-tardy jobs, in any order.

For instance I, we have $d_{P_1} < d_{P_2} < \cdots < d_{P_{2n-1}} < d_{P_{2n}} < d_R$. Therefore, the jobs would be scheduled in the order of $P_1, P_2, P_3, P_4, \ldots, P_{2n-1}, P_{2n}, R$ by the Hodgson-Moore algorithm. It is easy to see that P_1 meets its due date, but P_2 will not. Since $p_1 < p_2$, P_2 will be chosen as a tardy job. Suppose we have scheduled all the jobs P_{2j-1} and P_{2j}, $1 \leq j \leq i-1$, and all the even P-jobs had been chosen as tardy jobs and discarded from the current schedule. If we now schedule P_{2i-1} and P_{2i} sequentially, then the completion times will be

$$C_{P_{2i-1}} = \sum_{j=1}^{i-1} p_{2j-1} + p_{2i-1} < \sum_{j=1}^{i-1} p_{2j} + \sum_{j=1}^{i-1} x_j + p_{2i-1} = d_{P_{2i-1}}.$$

For $i < n$, we have (by Lemma 2.1)

$$C_{P_{2i}} = \sum_{j=1}^{i-1} p_{2j-1} + p_{2i-1} + p_{2i} > p_{2i-1} + p_{2i} > \sum_{j=1}^{i-1} p_{2j} + \sum_{j=1}^{i} x_j + p_{2i} = d_{P_{2i}},$$

and for $i = n$, we have (by Lemma 2.2),

$$C_{P_{2n}} = \sum_{j=1}^{n-1} p_{2j-1} + p_{2n-1} + p_{2n} > p_{2n-5} + p_{2n-1} + p_{2n} > \sum_{j=1}^{n-1} p_{2j} + \sum_{j=1}^{n} x_j + p_{2n} = d_{P_{2n}}.$$

Since P_{2i} misses its due date and has the largest processing time among all the jobs currently in the schedule, P_{2i} will be chosen as a tardy job. Therefore, the Hodgson-Moore algorithm will choose all the even P-jobs as tardy jobs.

For the R-job, we have

$$C_R = \sum_{j=1}^{n} p_{2j-1} + L < A + L < \sum_{i=1}^{n} x_i + [A + \frac{1}{2}\sum_{i=1}^{n}(l_i - 1)\sigma_i] + L = d_R.$$

So it is on time. Hence, the total number of tardy jobs is n. Thus, any feasible schedule for instance I must have exactly n tardy jobs.

(4.2) Since the R-job has the large processing time, a job scheduled after the R-job must miss its due date. Hence, all the other on-time jobs must be scheduled before the R-job. Thus, (4.2) also holds.

(4.3) We now prove (4.3) by contradiction. Suppose $\{P_{2i-1}, P_{2i}\}$ is the first pair that are both tardy in a feasible schedule S. Consider now applying the Hodgson-Moore algorithm to the job set consisting of all the P-jobs except $\{P_{2i-1}, P_{2i}\}$, as well as the R-job. As shown in the proof of (4.1), all the even P-jobs, $P_{2j},\ j < i$, will be chosen as the tardy jobs by the Hodgson-Moore algorithm. If we now schedule the jobs P_{2i+1} and P_{2i+2}, then P_{2i+1} will still be on time. However, the completion time of P_{2i+2} is:

For $i < n - 1$ (by Lemma 2.1),

$$C_{P_{2i+2}} = \sum_{j=1}^{i-1} p_{2j-1} + p_{2i+1} + p_{2i+2} > p_{2i+1} + p_{2i+2} > \sum_{j=1}^{i} p_{2j} + \sum_{j=1}^{i+1} x_j + p_{2i+2} = d_{P_{2i+2}},$$

and for $i = n - 1$ (by Lemma 2.2),

$$C_{P_{2n}} = \sum_{j=1}^{n-1} p_{2j-1} + p_{2n-1} + p_{2n} > p_{2n-5} + p_{2n-1} + p_{2n} > \sum_{j=1}^{n-1} p_{2j} + \sum_{j=1}^{n} x_j + p_{2n} = d_{P_{2n}}.$$

Thus, P_{2i+2} will miss its due date. Since P_{2i+2} has the largest processing time among all jobs currently in the schedule, it will be chosen as a tardy job. By mathematical induction, we know that all even P-jobs are tardy. By assumption, P_{2i-1} is also a tardy job. Thus, the total number of tardy jobs will be $n+1$, contradicting our assumption that S is a feasible schedule. □

Lemma 5: *In a feasible schedule, exactly one job from* $\{P_{2i-1},\ P_{2i}\},\ i = 1, 2, \ldots, n$, *must be tardy.*

Proof: By contradiction. From Lemma 4 we know that at least one job from $\{P_{2i-1}, P_{2i}\}$, $i = 1, 2, \ldots, n$, must be non-tardy. Suppose P_{2i-1} and P_{2i} are both non-tardy, then by Lemma 2.1, for $i = 1, 2, \ldots, n-1$, we have

$$C_{P_{2i}} > p_{2i-1} + p_{2i} > \sum_{j=1}^{i-1} p_{2j} + \sum_{j=1}^{i} x_j + p_{2i} = d_{P_{2i}},$$

and for $i = n$, since at least one of the jobs in $\{P_{2n-3},\ P_{2n-2}\}$ is non-tardy (Lemma 4), we have (by Lemma 2.2)

$$C_{P_{2n}} > p_{2n-3} + p_{2n-1} + p_{2n} > p_{2n-5} + p_{2n-1} + p_{2n} > \sum_{j=1}^{n-1} p_{2j} + \sum_{j=1}^{n} x_j + p_{2n} = d_{P_{2n}}.$$

So one of the jobs in $\{P_{2i-1},\ P_{2i}\}$ must be tardy. Hence, Lemma 5 holds. □

We now prove assertions (c) and (d) through Lemmas 6, 7 and 8. These three lemmas state how Q-jobs must be scheduled in a feasible schedule with minimum $\sum C_j^a$ in instance I.

Lemma 6: *In a feasible schedule with minimum total completion time of Agent A, all the Q-jobs must be scheduled in the SPT order before the R-job.*

Proof: First, recall that by the Hodgson-Moore algorithm, we have a feasible schedule with all the odd P-jobs scheduled before the R-job and all the even P-jobs scheduled after the R-job; i.e., the schedule before the R-job (and including the R-job) is $P_1, P_3, \ldots, P_{2n-1}, R$. Note that in this

schedule, the completion time of job P_{2i-1} $(i = 1, 2, \ldots, n)$ is $\sum_{k=1}^{i} p_{2k-1}$ and the completion time of the R-job is $(A - \frac{1}{2}\sum_{i=1}^{n} \sigma_i) + L$. Through a simple calculation of the due dates, we can see that if we insert the Q-jobs in this schedule to obtain a schedule $P_1, Q_1, \ldots, P_{2n-1}, Q_n, R$, it is still a feasible schedule. We call this schedule S. Now, if we schedule any of the Q-jobs after the R-job, the total completion time will be larger than that of S. Thus, in a feasible schedule with minimum $\sum C_j^a$, all the Q-jobs must be scheduled before the R-job.

Next, we prove that all the Q-jobs must be scheduled in SPT order. Recall that the SPT rule minimizes total completion time of a schedule. Suppose in an feasible schedule S' that has minimum $\sum C_j^a$, Q_i is scheduled before Q_j, where $j < i$. Now we interchange these two jobs and schedule all the other jobs in the same order without idle time in between any two jobs. Call the new schedule S''. Since $p_j < p_i$, it is easy to see that S'' is still feasible, since the completion times of all the P-jobs and the R-job in S'' are no larger than their completion times in S'. However, it is easy to see that the total completion time of S'' is smaller than that of S', contradicting the fact that S' is a feasible schedule with minimum $\sum C_j^a$. □

Lemma 7: *In a feasible schedule with minimum total completion time of Agent A, no two Q-jobs can be scheduled between $d_{P_{2i-2}}$ and $d_{P_{2i}}$ $(i = 1, \ldots, n)$, where $d_{P_0} = 0$.*

Proof: By Lemma 6, we know that in a feasible schedule with minimum $\sum C_j^a$, all the Q-jobs must be scheduled in the SPT order before the R-job. So the Q-jobs must be scheduled in the order of $Q_1, Q_2, \ldots, Q_n$.

By Lemma 5, we know that exactly one job from $\{P_{2i-1}, P_{2i}\}$ is on time. Let $H_i \in \{P_{2i-1}, P_{2i}\}$ $(i = 1, \ldots, n)$ be the on-time P-job. Since

$$d_{P_2} = x_1 + p_2 \geq x_1 + p_{H_1},$$

Q_1 can be scheduled between 0 and d_{P_2} (either before or after H_1) without causing $H_k,\ k = 2, \ldots, n$, to be overdue. However, by Lemma 3,

$$x_1 + x_2 + p_{H_1} > x_1 + p_2 = d_{P_2}.$$

Therefore, Q_2 cannot be scheduled between 0 and d_{P_2}.

Suppose we have scheduled each Q_j between $d_{P_{2j-2}}$ and $d_{P_{2j}}$ (either before or after H_{j-1}) without causing H_i to be overdue, where $j < i$. Now we schedule Q_i. Since

$$d_{P_{2i}} = \sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i} x_k + p_{2i} \geq \sum_{k=1}^{i-1} p_{H_k} + \sum_{k=1}^{i-1} x_k + x_i + p_{2i},$$

Q_i can be scheduled between $d_{P_{2i-2}}$ and $d_{P_{2i}}$ (either before or after H_i) without causing $H_k,\ k = i+1, \ldots, n$ to be overdue. However, by Lemma 3,

$$\begin{aligned}\sum_{k=1}^{i-1} p_{H_k} + \sum_{k=1}^{i-1} x_k + x_i + x_{i+1} + p_{H_i} &= (\sum_{k=1}^{i} p_{H_k} + x_{i+1}) + \sum_{k=1}^{i-1} x_k + x_i \\ &> \sum_{k=1}^{i} p_{2k} + \sum_{k=1}^{i-1} x_k + x_i = \sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i-1} x_k + x_i + p_{2i} = d_{P_{2i}}.\end{aligned}$$

Thus, Q_{i+1} cannot be scheduled between $d_{P_{2i-2}}$ and $d_{P_{2i}}$.

By mathematical induction, we conclude that Lemma 7 holds. □

Lemma 8: *In a feasible schedule with minimum total completion time of Agent A, there are only two possible configurations for each set of jobs $\{P_{2i-1}, P_{2i}, Q_i\}$. We either have P_{2i-1} on time together with Q_i and scheduled as P_{2i-1} followed by Q_i before the R-job, or we have P_{2i} on time together with Q_i and scheduled as Q_i followed by P_{2i} before the R-job.*

Proof: By Lemma 5, we know that in a feasible schedule, exactly one job from $\{P_{2i-1}, P_{2i}\}$ is on time.

By Lemma 7, we know that in a feasible schedule with minimum $\sum C_j^a$, only Q_i can be scheduled between $d_{P_{2i-2}}$ and $d_{P_{2i}},\ i = 1, \ldots, n$.

Now consider Q_1 first. It is easy to see that if P_1 is on time (while P_2 is tardy), then P_1 must be scheduled before Q_1. On the other hand, if P_2 is on time (while P_1 is tardy), then Q_1 must be scheduled before P_2 so as to minimize the completion time of Q_1.

Suppose we have scheduled $\{P_{2j-1}, P_{2j}, Q_j\}$ and we have either P_{2j-1} followed by Q_j, or Q_j followed by P_{2j}, where $j < i$. Now consider job Q_i. Since

$$\sum_{k=1}^{i-1} p_{2k-1} + \sum_{k=1}^{i-1} x_k + x_i + p_{2i-1} = (\sum_{k=1}^{i-1} p_{2k-1} + x_i) + \sum_{k=1}^{i-1} x_k + p_{2i-1} > \sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i-1} x_k + p_{2i-1} = d_{P_{2i-1}},$$

Q_i can only be scheduled after P_{2j-1}. On the other hand, since

$$\sum_{k=1}^{i-1} p_{H_k} + \sum_{k=1}^{i-1} x_k + x_i + p_{2i} \leq \sum_{k=1}^{i-1} p_{2k} + \sum_{k=1}^{i} x_k + p_{2i} = d_{P_{2i}},$$

Q_i can be scheduled before P_{2i}.

By Lemma 5, we know that in a feasible schedule, exactly one job from $\{P_{2i-1}, P_{2i}\}$ is on time. Hence, if P_{2i-1} is on time, then we have P_{2i-1} followed by Q_i. On the other hand, if P_{2i} is on time, then we have Q_i followed by P_{2i}. By mathematical induction, we conclude that Lemma 8 holds. □.

Next, we show that in order to minimize the total completion time of the Q-jobs, we should schedule more even P-jobs to be on time.

Lemma 9: *In order to minimize the total completion time of Agent A, it is always better to choose P_{2i} on time together with Q_i and scheduled as Q_i followed by P_{2i}.*

Proof: By Lemma 8, in a feasible schedule with minimum $\sum C_j^a$, we either have $P_{2i-1}Q_i$ or Q_iP_{2i}. By interchanging $P_{2i-1}Q_i$ with Q_iP_{2i} in a schedule, the total completion time decreases by

$$a_{2i-1} - (n-i)l_i\sigma_i = (n-i+1)l_i\sigma_i - (n-i)l_i\sigma_i = l_i\sigma_i > 0.$$

Therefore, to minimize the total completion time, it is always better to choose P_{2i} on time together with Q_i and to schedule them as Q_i followed by P_{2i}.

Note that at the same time, the total processing time before the R-job will increase by exactly the same amount $l_i\sigma_i$. □

Lemma 10: *If there is a solution to an instance of the problem* $1 \mid\mid \sum C_j^a \leq TC : \sum U_j^b \leq n$, *then there exists a solution to the corresponding instance of the Even-Odd Partition problem.*

Proof: From Lemma 4, we know that in a feasible schedule, the R-job must be on time. In order to ensure that the R-job is on time, we cannot choose all the even P-jobs to be on time. Suppose, on the contrary, we pick all the even P-jobs to be on time. Note that all the Q-jobs must be scheduled before the R-job in a feasible schedule with minimum $\sum C_j^a$ (Lemma 8). Then, the completion time of the R-job will be

$$\sum_{j=1}^{n} p_{2j} + \sum_{j=1}^{n} x_j + L = \sum_{j=1}^{n} x_j + [A + \sum_{i=1}^{n}(l_i - \frac{1}{2})\sigma_i] + L > d_R.$$

Thus, the R-job will be tardy. So, we must choose some even P-jobs as tardy jobs. As we have shown above, each time we interchange P_{2i} with P_{2i-1}, the total completion time will increase by $l_i\sigma_i$. At the same time, the completion time of the R-job will decrease by exactly $l_i\sigma_i$. So, if we can schedule the jobs such that the R-job completes exactly at its due date, then the total completion time has the minimum value among all feasible schedules, and the total processing time of all the on-time P-jobs is *exactly* $TC = A + \frac{1}{2}\sum_{k=1}^{n}(l_i - 1)\sigma_i$. This means that there is a solution to the instance of the Even-Odd Partition problem if there is a solution to the instance I of the scheduling problem. □

From Lemmas 1 and 10, we know that there is a solution to the instance of the Even-Odd Partition problem if and only if there is a solution to the corresponding instance I of the scheduling

problem. Therefore, we have the following theorem.

Theorem 1: *The problem* $1||\sum C_j^a : \sum U_j^b$ *is binary NP-hard.* □

References

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

Garey M. R., R. E. Tarjan, G. T. Wilfong. 1988. One-Processor Scheduling with Symmetric Earliness and Tardiness. *Mathematics of Operations Research* **13**, 330-348.