Journal of Scheduling **7:** 333–348, 2004.

MINIMIZING MAKESPAN IN A TWO-MACHINE FLOW SHOP WITH DELAYS AND UNIT-TIME OPERATIONS IS NP-HARD

WENCI YU[1,†], HAN HOOGEVEEN[2,*], AND JAN KAREL LENSTRA[3]

[1] *Applied Mathematics Institute, East China University of Science and Technology, Shanghai 200237, China*
[2] *Department of Computer Science, Utrecht University, P.O. Box 80089, 3508 TB Utrecht, The Netherlands*
[3] *CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*

ABSTRACT

One of the first problems to be studied in scheduling theory was the problem of minimizing the makespan in a two-machine flow shop. Johnson showed that this problem can be solved in $O(n \log n)$ time. A crucial assumption here is that the time needed to move a job from the first to the second machine is negligible. If this is not the case and if this 'delay' is not equal for all jobs, then the problem becomes NP-hard in the strong sense. We show that this is even the case if all processing times are equal to one. As a consequence, we show strong NP-hardness of a number of similar problems, including a severely restricted version of the Numerical 3-Dimensional Matching problem.

KEY WORDS: flow shop scheduling, intermediate delays, makespan, computational complexity, strong NP-hardness

1980 MATHEMATICS SUBJECT CLASSIFICATION (REVISION 1991): 90B35.

1. INTRODUCTION

Until recently, one of the standard assumptions in scheduling theory was that the time needed to move a job from one machine to another is negligible. Although this assumption is often justified, there are many situations in which it must be abandoned as being unrealistic. For example, in manufacturing there may be a transportation time from one production facility to another, and in computer systems the output of a task on one processor may require a communication time so as to become the input to a succeeding task on another processor. We consider the effect of introducing such *delays* on the computational complexity of the problem under consideration.

We are interested in the complexity of the two-machine flow shop scheduling problem with *minimum* delays. There are two machines, M_1 and M_2, that are continuously available from time zero onwards for processing n independent jobs $j (j = 1, \ldots, n)$. Each machine can handle no

---

[†]The results reported in this paper form part of the PhD thesis of Wenci Yu (1996). Professor Yu visited the combinatorial optimization group in Eindhoven from September 1995 until June 1996, supported by EIDMA, the Euler Institute of Discrete Mathematics and its Applications. Within eight months, he obtained deep results on flow shop scheduling with delays, wrote a thesis, and defended it. We, his present co-authors, and Peter Brucker (Universität Osnabrück) acted as his supervisors.
Wenci Yu pursued his research with great intensity and commitment, showing an amount of energy rarely seen in students of half his age. He was a true scholar and a great friend. To know him and to work with him has enriched our life. We are saddened by his unexpected death in October 2002.

*Correspondence to: Han Hoogeveen. E-mail: slam@cs.uu.nl

more than one job at a time. Each job consists of two operations with an intermediate delay; after the completion of the first operation of job j, *at least* l_j time units must elapse before the second operation of j can start. The first (second) operation of each job j has to be executed by machine M_1 (M_2); processing the first (second) operation of job j takes time p_{1j} (p_{2j}). A *schedule* σ specifies a completion time $C_{ij}(\sigma)$ for the ith operation ($i = 1, 2$) of each job $j (j = 1, \ldots, n)$ such that the above conditions are met. The completion time $C_j(\sigma)$ of job j is then equal to $C_{2j}(\sigma)$. We omit the argument σ if there is no confusion possible as to the schedule we are referring to. Our objective is to minimize the *makespan*, or schedule length, which is defined as $\max_{j=1,\ldots,n} C_j$. Following and extending the three-field notation scheme introduced by Graham et al. (1979), we denote this problem by $F2|l_j|C_{\max}$.

Johnson (1954) presents an $O(n \log n)$ algorithm to solve the problem without delays. Johnson (1958) and Mitten (1958) consider the case with minimum delays and show that a variant of Johnson's algorithm for $F2||C_{\max}$ can be used to find an optimal *permutation* schedule, that is, a schedule in which both machines execute the jobs in the same order. If all delays are equal, then there exists an optimal schedule that is a permutation schedule; this is, however, not the case if there are two or more distinct delay values. Kern and Nawijn (1991) study a single-machine scheduling problem with two operations per job and intermediate minimum delays; in Section 9 we will show that this problem is equivalent to the two-machine flow shop problem with delays. They show that the single-machine problem is NP-hard in the ordinary sense if the solution space is not restricted to permutation schedules. This result is strengthened to NP-hardness in the strong sense for $F2 \mid l_j \mid C_{\max}$ (Lenstra, 1991), for $F2 \mid l_j, p_{1j} = p_{2j} \mid C_{\max}$ (Dell'Amico and Vaessens, 1996), and, finally, for $F2 \mid l_j \in \{0, l\}, p_{1j} = p_{2j} \mid C_{\max}$ (Yu, 1996); Yu further shows that in case of *unit* processing times the latter problem is solvable in polynomial time. Dell'Amico (1996) proposes several lower bounds, which are used in the derivation of several polynomial 2-approximation algorithms. He further presents a tabu search algorithm that gives good results. Orman and Potts (1997) study the problem of minimizing the idle time of a radar, which they formulate as a single-machine scheduling problem with *exact* rather than minimum delays. They identify some special cases that are polynomially solvable and show that the problem is already strongly NP-hard if the processing times of all operations are *identical.*

The complexity status of the special case with *unit* processing time tasks has been open for both minimum and exact delays. The complexity status of the one with minimum delays is posed as an open question by Kern and Nawijn (1991). In this paper, we show that the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ is NP-hard in the strong sense by a reduction from 3-PARTITION. As a corollary, we show that the special cases of unit processing times of the single-machine problems with minimum delays studied by Kern and Nawijn (1991) and with exact delays studied by Orman and Potts (1997) are strongly NP-hard as well. For completeness, we mention the problem of scheduling parallel machines subject to communication delays. Here, a delay between two precedence-constrained jobs occurs only if both jobs are allocated to different machines. We refer to Hoogeveen, Lenstra, and Van de Velde (1997) for some of the main references.

The paper is organized as follows. In Section 2, we present the reduction and show that a 'yes' answer to a 3-PARTITION instance implies a 'yes' answer to our instance of the decision variant of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. The proof of the converse statement is much more difficult and requires many steps. In Section 3 we sketch the steps in the proof; this will be worked out in Sections 4–7. In Sections 4–6 we derive several properties that a schedule that leads to a 'yes' answer to our decision instance of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ must satisfy; we use these properties in Section 7 in which we prove that 'yes' to our decision instance implies 'yes' to the 3-PARTITION instance.

As a corollary to our strong NP-hardness proof for $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$, we show in Section 8 that the special case of NUMERICAL 3-DIMENSIONAL MATCHING in which two sets are equal to the set $\{1, 2, \ldots, n\}$ is strongly NP-hard as well. Finally, in Section 9 we prove that the two-machine open shop scheduling problem with delays is strongly NP-hard in case of unit processing times.

2. A REDUCTION FROM 3-PARTITION

We prove strong NP-hardness of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ through a reduction from the problem 3-PARTITION, which is known to be NP-complete in the strong sense (Garey and Johnson, 1979).

3-PARTITION
Given a multiset of $3m$ positive integers $X = \{x_1, \ldots, x_{3m}\}$ and a positive integer b such that $b < x_j < 2b$ for $j = 1, \ldots, 3m$ and $\sum_{j=1}^{3m} x_j = 4mb$, does there exist a partition of the set X into m disjoint subsets $X_1, \ldots, X_m$ such that $\sum_{x_j \in X_i} x_j = 4b$?

Consider any instance of 3-PARTITION. Since an equivalent problem is obtained by multiplying each $x_j (j = 1, \ldots, 3m)$ and b by a factor m, we may assume without loss of generality that all x_j and b are divisible by m. Given any such instance of 3-PARTITION, we construct the following instance of the decision variant of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$, which we denote by $\mathcal{I}$. It contains three types of jobs, which only differ with respect to their delay value l_j, as the processing times are all equal to one; we therefore mention for each job its delay value only. Instance $\mathcal{I}$ consists of the following jobs:

- The jobs $1, \ldots, 3m$ are partition jobs, or P-jobs: the delay value of job j is equal to x_j, for $j = 1, \ldots, 3m$.
- The jobs $3m + 1, \ldots, 4mb$ are zero delay jobs, or Z-jobs: their delay value is equal to zero.
- The jobs $4mb + 1, 4mb + 2, \ldots, mu$, where $u = 4(m + 1)b$, are large delay jobs, or L-jobs: their delay value is equal to $u + 1$.

The threshold value on the makespan is $y = n + 4mb + 2$, where $n = 4m(m + 1)b$ is the number of jobs.

Lemma 1. If the answer to the 3-PARTITION instance is 'yes', then there exists a feasible schedule for instance $\mathcal{I}$ of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with makespan no more than y.

Proof. Suppose that $\{X_1, \ldots, X_m\}$ is a partition of X that leads to 'yes' to 3-PARTITION. Let the three elements in X_i be $x_{\xi(i)}$, $x_{\eta(i)}$, and $x_{\zeta(i)}$, for $i = 1, \ldots, m$. We construct for instance $\mathcal{I}$ a schedule with makespan equal to y in the following way (see Figure 1). The schedule consists of m blocks

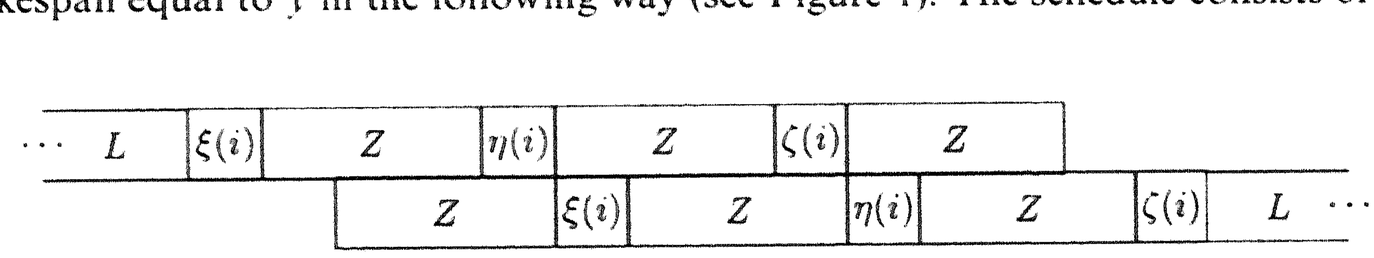

Figure 1. Block B_i in a schedule for a 'yes'-instance

B_i ($i = 1, \ldots, m$), where block B_i corresponds to the set X_i. Block B_i contains $4mb$ L-jobs, $4b - 3$ Z-jobs, and the P-jobs $\xi(i)$, $\eta(i)$, and $\zeta(i)$. The jobs in B_i are executed in the following order on M_1: first all L-jobs, then job $\xi(i)$ followed by $x_{\xi(i)} - 1$ Z-jobs, job $\eta(i)$ followed by $x_{\eta(i)} - 1$ Z-jobs, and job $\zeta(i)$ followed by $x_{\zeta(i)} - 1$ Z-jobs. On M_2 the relative order of the L-jobs and the Z-jobs is the same as on M_1. The order of the jobs in B_i on M_2 is then: first $x_{\xi(i)} - 1$ Z-jobs followed by job $\xi(i)$, then $x_{\eta(i)} - 1$ Z-jobs followed by job $\eta(i)$, after that $x_{\zeta(i)} - 1$ Z-jobs followed by job $\zeta(i)$, and finally the $4mb$ L-jobs. The blocks B_i ($i = 1, \ldots, m$) are patched together in any order without any unnecessary idle time. It is easily verified that the resulting schedule is feasible with makespan equal to y. ■

To prove the converse statement of Lemma 1, we need to introduce some new concepts. This is done in the Sections 4–6. We start, however, with providing an overview of the proof of the converse statement.

3. A SKETCH OF THE NP-HARDNESS PROOF

The proof that the existence of a schedule with makespan no more than y implies that the answer to 3-PARTITION is 'yes' consists of the following steps. Let σ denote a schedule with $C_{\max} \leq y = n + 4mb + 2$. In Section 4 we prove that in σ the time that elapses between the completion of the first operation and the start of the second operation of job J_j must be equal to l_j, for $j = 1, \ldots, n$. In Section 5 we show that this implies that in σ at each time t ($t = 4mb + 2, 4mb + 3, \ldots, n + 4mb + 1$) exactly one job must become available for processing on M_2 (Corollary 6), where J_j becomes available at time t if by then the minimum delay after the completion of its first operation on M_1 has elapsed. From here, we show in Section 6 that σ must possess the special structure defined in Lemma 9: there are m groups consisting of L-jobs only, which are separated by $(m - 1)$ groups containing $4b$ P-jobs and Z-jobs together with m L-jobs, and after the last pure L-subset there is one subset containing only P-jobs and Z-jobs. After having established this result, we disregard the Z-jobs and group the L-jobs and P-jobs in so-called job chains (Definition 11), where the different chains are formed by linking the job with arrival time t on M_2 with the job that starts at time t on M_1, and so on. We then show that $4mb + 1$ job chains exist in σ, of which there are only m that contain P-jobs (the so-called mixed job chains, see Lemma 19). These m mixed job chains all contain exactly 3 different P-jobs, and the total delay of these 3 P-jobs amounts to $4b$; hence, partitioning the $3m$ integers of 3-PARTITION according to the partitioning of the P-jobs in the m mixed job chains yields a 'yes' to 3-PARTITION.

4. THE CONCEPT OF TIGHT SCHEDULES

To prove the validity of our reduction, we first discuss a lower bound on the optimum solution value of the $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ problem. As a consequence, we come to the concept of a tight schedule. This lower bound is a special case of the lower bound derived by Dell'Amico and Vaessens (1996).

The lower bound is derived as follows. Consider any schedule σ; let σ_1 and σ_2 denote the order in which the jobs are executed on M_1 and M_2, respectively. As we do not incur unnecessary idle time in a schedule, the sequences σ_1 and σ_2 completely determine the schedule σ. Therefore, we denote from now on a schedule σ as $S(\sigma_1, \sigma_2)$; we denote its makespan by $C(\sigma_1, \sigma_2)$. For any job k, we denote its position in σ_1 and σ_2 by $\sigma_1^{-1}(k)$ and $\sigma_2^{-1}(k)$, respectively.

Lemma 2. For any schedule $S(\sigma_1, \sigma_2)$, *we have that*

$$C(\sigma_1, \sigma_2) \geq n + 1 + \left\lceil \sum_{j=1}^{n} l_j / n \right\rceil.$$

Proof. Consider any job k. Since it has $n - \sigma_2^{-1}(k)$ successors in σ_2 and all processing times are equal to one, we have that

$$C(\sigma_1, \sigma_2) \geq \sigma_1^{-1}(k) + l_k + \left(n + 1 - \sigma_2^{-1}(k)\right), \quad \text{for } k = 1, \ldots, n. \tag{1}$$

If we add up these inequalities for $k = 1, \ldots, n$, then we find that

$$nC(\sigma_1, \sigma_2) \geq \sum_{k=1}^{n} \left[\sigma_1^{-1}(k) + l_k + \left(n + 1 - \sigma_2^{-1}(k)\right)\right]$$

$$= \sum_{k=1}^{n} \sigma_1^{-1}(k) - \sum_{k=1}^{n} \sigma_2^{-1}(k) + n(n+1) + \sum_{k=1}^{n} l_k = n(n+1) + \sum_{k=1}^{n} l_k,$$

since σ_1 and σ_2 are both permutations that map the n jobs to the positions $1, \ldots, n$. As the delay values are all integral, the makespan must be integral, and the desired result follows immediately by dividing both sides of the above inquality by n and rounding up the right-hand side. ■

Given an instance of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$, we can compute the above lower bound for each subset of jobs; the maximum over all outcomes then yields a valid lower bound on the makespan of the entire instance. The lower bound found through this procedure was suspected to be tight until Coster (1993) pointed out that, for the instance with six jobs and delay values (0, 0, 0, 4, 4, 4), the lower bound is 9, but the minimum makespan is 10.

Definition 3. A schedule $S(\sigma_1, \sigma_2)$ for $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ is called a tight schedule and σ_1 is called a tight sequence, if all inequalities in (1) are equalities, that is, if

$$\sigma_1^{-1}(k) - \sigma_2^{-1}(k) = C(\sigma_1, \sigma_2) - (n + 1 + l_k), \quad \text{for } k = 1, \ldots, n. \tag{2}$$

Observation 4. Consider any instance of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. *Define* $\bar{l}$ *as the average delay. The following statements concerning tight schedules all follow immediately from Lemma* 2.

(i) *Schedule* $S(\sigma_1, \sigma_2)$ *is a tight schedule if and only if* $\sum_{k=1}^{n} l_k$ *is a multiple of* n *and* $C(\sigma_1, \sigma_2) = n + 1 + \bar{l}$.
(ii) *Schedule* $S(\sigma_1, \sigma_2)$ *is a tight schedule if and only if* $\sigma_1^{-1}(k) - \sigma_2^{-1}(k) = \bar{l} - l_k$, *for* $k = 1, \ldots, n$.
(iii) *Any tight schedule is an optimal schedule.*
(iv) *The instance of* $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ *has a tight schedule or a tight sequence if and only if the average delay is an integer and* $C^*_{\max} = n + 1 + \bar{l}$, *where* $C^*_{\max}$ *stands for the optimal value of the problem.*

Now consider the instance $\mathcal{I}$ of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. We have that $\sum_{k=1}^{n} l_k = n(4mb+1)$, which is a multiple of n. Moreover, $y = mu + 4mb + 2 = n + 1 + 4mb + 1 = n + 1 + \bar{l}$. Hence, part (i) of Observation 4 implies that any schedule $S(\sigma_1, \sigma_2)$ with $C(\sigma_1, \sigma_2) \leq y$ must be a tight schedule.

5. THE ONE-TO-ONE PROPERTY OF TIGHT SEQUENCES

As we have observed above, any schedule with makespan no more than y is a tight schedule. We need a further characterization of tight schedules, before we are able to prove the converse of Lemma 1. We derive these characterizations in this and the following two sections. The first characterization is derived by looking at the times at which the jobs become available for processing on M_2. Given these arrival times, we can construct a 'best possible' schedule by scheduling the jobs on M_2 in order of their arrival times, where ties are settled arbitrarily; hence, a schedule is completely specified by the job sequence that denotes the order in which the jobs are executed by M_1. In the following lemma, we show that a job sequence is a tight sequence if and only if the corresponding arrival times of jobs on M_2 make up an interval without overlaps and gaps. For short, we call this characterization the *one-to-one property of tight sequences.*

Lemma 5. Consider any instance of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. *A schedule* σ *with job sequence* σ_1 *on* M_1 *and* σ_2 *on* M_2 *is a tight sequence if and only if*

$$\bigcup_{k=1}^{n} \{\sigma_1^{-1}(k) + l_k\} = \{\bar{l} + 1, \ldots, \bar{l} + n\}.$$

Proof. First, we prove the 'only if' part. Without loss of generality, we may assume that on M_2 the jobs arrive in the order σ_2. As σ is a tight schedule, we have, according to the second part of Observation 4, that

$$\sigma_1^{-1}(k) + l_k = \bar{l} + \sigma_2^{-1}(k) \quad \text{for } k = 1, \ldots, n. \tag{3}$$

Since $\{\sigma_2^{-1}(1), \ldots, \sigma_2^{-1}(n)\} = \{1, \ldots, n\}$, the set of the right-hand side values of (3) is equal to the set $\{\bar{l} + 1, \ldots, \bar{l} + n\}$, from which we get the desired result.

Conversely, if the set of arrival times is equal to $\{\bar{l} + 1, \ldots, \bar{l} + n\}$, then the sequence σ_1 gives rise to a schedule with makespan $\bar{l} + n + 1$. Moreover, adding up the elements in the left and right set yields that

$$\sum_{k=1}^{n} (\sigma_1^{-1}(k) + l_k) = \sum_{k=1}^{n} (\bar{l} + k),$$

from which we derive that $\sum_{k=1}^{n} l_k = n\bar{l}$, which implies that $\sum_{k=1}^{n} l_k$ is a multiple of n. Hence, applying the first part of Observation 4 shows that σ_1 is a tight sequence. ■

As the instance $\mathcal{I}$ needs a tight schedule for a 'yes', we can use Lemma 5 to obtain the following corollary.

Corollary 6. The answer to the decision variant of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ *is 'yes' for the instance* $\mathcal{I}$ *if and only if*

$$\bigcup_{k=1}^{n} \{\sigma_1^{-1}(k) + l_k\} = \{4mb + 2, 4mb + 3, \ldots, n + 4mb + 1\}.$$

6. THE SEPARATION STRUCTURE OF A TIGHT SEQUENCE FOR $\mathcal{I}$

There are many NP-hardness proofs in scheduling theory that are based on a reduction from 3-PARTITION. The common structure of a successful ('yes') schedule is then that there are m subsets of the partition jobs, which are separated by the separation jobs. As indicated by Figure 1, in our case the L-jobs are the separation jobs, whereas the P-jobs and Z-jobs play the role of the partition jobs. We will show in Lemma 9 that such a structure can also be detected in any tight schedule for $\mathcal{I}$, but that the partition and separation jobs may get tangled up a bit. But before we are able to state and prove Lemma 9, we need a preliminary lemma and a definition to facilitate notation.

Definition 7. Let $\sigma = (\sigma(1), \sigma(2), \ldots, \sigma(n))$ be any sequence of jobs, and let r, s be two indices such that $1 \leq r < s \leq n$. We use $\sigma[r, s]$ and $[r, s]$ as a short notation for the job subsequence $(\sigma(r), \sigma(r+1), \ldots, \sigma(s))$ and the set $\{r, r+1 \ldots, s\}$, respectively. The set $[r, s]$ is also called a position interval or interval for short. Furthermore, we denote by L, P, and Z the sets containing all L-jobs, P-jobs, and Z-jobs, respectively.

Lemma 8 (L-job propagation). *Let σ be any tight sequence for $\mathcal{I}$ that contains a job subsequence $\sigma[r, s] \subset L$, with $2b - 1 \leq s - r \leq u = 4(m+1)b = n/m$. Then the following propagation rules hold:*

(i) *Forward propagation of L-jobs: if $s + (u + 2 - 2b) \leq n$, then $\sigma[r + (u+1), s + (u+2-2b)] \subset L$.*
(ii) *Backward propagation of L-jobs: if $r - (u+2-2b) \geq 1$, then $\sigma[r - (u+2-2b), s - (u+1)] \subset L$.*

Proof. The proof is based on Corollary 6. For any job k ($k = 1, \ldots, n$), let a_k denote its arrival time on M_2; we have $a_k = \sigma^{-1}(k) + l_k$. Since $\sigma[r, s]$ consists of L-jobs only, which all have delay value equal to $u + 1$, the arrival times of the jobs in $\sigma[r, s]$ form the set $[r + u + 1, s + u + 1]$.

We only prove part (i); the proof of part (ii) follows along the same lines. Suppose that part (i) is not true, that is, M_1 processes a P-job or Z-job on some position j such that $j \in [r + (u + 1), s + (u + 2 - 2b)]$. Since the delay of any P-job or Z-job is less than $2b$, we have for the arrival time $a_{\sigma(j)}$ of job $\sigma(j)$ that

$$r + u + 1 \leq a_{\sigma(j)} = j + l_{\sigma(j)} \leq s + u + 2 - 2b + 2b - 1 = s + u + 1.$$

Because of the assumption $s - r \leq u$, we have that $s < r + u + 1$, which implies that the intervals $[r, s]$ and $[r + (u + 1), s + (u + 2 - 2b)]$ are disjoint. Hence, we have that there are two jobs that arrive on M_2 at time $a_{\sigma(j)}$: job j and one of the L-jobs in $\sigma[r, s]$ (since $a_{\sigma(j)}$ belongs to the set $[r + u + 1, s + u + 1]$, which is the set of arrival times of the jobs in $\sigma[r, s]$). This implies, however, that σ does not satisfy the one-to-one property, which contradicts the assumption that σ is a tight sequence. ■

We give an intuitive explanation of Lemma 8. According to rule (i), any L-job subsequence propagates forward in any tight sequence σ for $\mathcal{I}$ in the following way: it moves forward at a distance $u = n/m$ at first, and each time it is decreased by 1 and by $2b - 2$ at the left and right part, respectively. The intuition behind rule (ii) is similar.

Lemma 9 (Separation Structure). *Let σ be any tight sequence for $\mathcal{I}$. Then σ must have the structure*

$$\sigma = (\alpha_1, \beta_1, \alpha_2, \beta_2, \ldots, \alpha_m, \beta_m),$$

where α_i and β_i are such that

(i) *each job subsequence $\alpha_i (i = 1, \ldots, m)$ consists of exactly $4mb - m + 1$ jobs, which are all L-jobs;*
(ii) *each job subsequence $\beta_i (i = 1, \ldots, m-1)$ consists of $4b + m$ jobs, whereas β_m consists of $4b$ jobs;*
(iii) *the job subsequences $\beta_i (i = 1, \ldots, m-1)$ all contain exactly m L-jobs, whereas β_m contains no L-jobs.*

Proof. We first determine the positions that must be occupied by L-jobs in any tight sequence σ. Corollary 6 states that the earliest time that a job arrives on M_2 is $4mb + 2$. As the delay value of any P-job or Z-job is at most equal to $2b - 1$, no P-job or Z-job can occupy a position j in σ with $j \leq 4mb + 2 - 2b$. Hence, the job subsequence $\sigma[1, 4m + 2 - 2b]$ must consist of L-jobs only. If we apply the forward propagation rule of L-jobs from Lemma 8 to $\sigma[1, 4m + 2 - 2b]$, then we find that the job subsequences $\sigma[r_i, r_i'] (i = 1, \ldots, m)$ consist of L-jobs only, where we define

$$r_i \equiv 1 + (i-1)(u+1) \quad \text{and} \quad r_i' \equiv 4mb + 2 - 2b + (i-1)(u + 2 - 2b), \quad \text{for } i = 1, \ldots, m.$$

Now we look at the back of the schedule. If a P-job or Z-job is completed last on M_1, then it will arrive on M_2 at time $n + 2b - 1 = mu + 2b - 1$ at the latest. Hence, the arrival times $mu + 2b, \ldots, mu + 4mb + 1$ must be covered by L-jobs, which must be processed on M_1 from time $(m-1)u + 2b - 1, \ldots, (m-1)u + 4mb$. This implies that the job subsequence $\sigma[(m-1)u + 2b - 1, (m-1)u + 4mb]$ is built solely of L-jobs. If we apply the backward propagation rule of L-jobs stated in Lemma 8, then we see that the job subsequences $\sigma[s_i', s_i]$ consist of L-jobs only, where we define

$$s_i' \equiv (m-1)u + 2b - (m-i)(u + 2 - 2b) \quad \text{and}$$

$$s_i \equiv (m-1)u + 4mb - (m-i)(u+1), \quad \text{for } i = 1, \ldots, m.$$

It is easily verified that for each i $(i = 1, \ldots, m)$ we have that $r_i < s_i' < r_i' < s_i$. This implies that the intervals $\sigma[r_i, s_i] (i = 1, \ldots, m)$ are pure L-job intervals. We define $\alpha_i \equiv \sigma[r_i, s_i]$, for $i = 1, \ldots, m$. As $s_i - r_i = 4mb - m$, we have proven part (i).

We further define $\beta_i \equiv \sigma[s_i + 1, r_{i+1} - 1] (i = 1, \ldots, m-1)$ and $\beta_m \equiv \sigma[s_m + 1, n]$. Because of the definitions of r_i and s_i $(i = 1, \ldots, m)$, the cardinality constraints of part (ii) are satisfied.

What is left to prove is that each job subsequence β_i $(i = 1, \ldots, m)$ has the right composition. Again, we look at the set of arrival times at M_2; we denote this set by $A = [4mb + 2, mu + 4mb + 1]$. As the job subsequences α_i $(i = 1, \ldots, m)$ consist of L-jobs, we know that the subsets $[r_i + u + 1, s_i + u + 1] (i = 1, \ldots, m)$ of A are occupied by the L-jobs from the subsequences α_i. We will prove part (iii) by a careful examination of the remaining subsets.

We first consider the subset $[4mb + 2, r_1 + u]$ of A. As $r_1 = 1$ and each L-job has delay value $u + 1$, these $4b$ arrival times must be occupied by P-jobs and Z-jobs, which must belong to β_1. The other m jobs in β_1 must be L-jobs, because these remaining jobs must arrive at M_2 after time $s_1 + u + 1$ (the set of arrival times $[r_1 + u + 1, s_1 + u + 1]$ is taken by the L-jobs from α_1). Next, we look at the subset $[s_1 + u + 2, r_2 + u]$. Each L-job that arrives in this interval must be executed in

the interval $[s_1 + 1, r_2 - 1]$, which is exactly β_2. As there are exactly m L-jobs in β_2, exactly m of the arrival times in $[s_1 + u + 2, r_2 + u]$ are occupied by L-jobs and the remaining $4b$ are filled with P-jobs and Z-jobs. Hence, $4b$ of the jobs in β_2 are P-jobs and Z-jobs, and by the same argument as before, we know that the remaining m jobs in β_2 are L-jobs. We can repeat this proof for job subsequences $\beta_3, \ldots, \beta_m$. ■

Definition 10. Let σ be any tight sequence for $\mathcal{I}$. The job subsequences α_i and $\beta_i (i = 1, \ldots, m)$ are called *separation subsequences* and *grouping subsequences*, respectively.

7. JOB CHAINS IN A TIGHT SEQUENCE

We need one more concept before we are able to show that the existence of a schedule with makespan no more than y for instance $\mathcal{I}$ implies that the answer to 3-PARTITION is 'yes'. It is called the concept of *job chains in a tight sequence*.

Definition 11. Let $\sigma = (\sigma(1), \ldots, \sigma(n))$ be any tight sequence for $\mathcal{I}$. A job subsequence $\pi = (\pi(1), \ldots, \pi(q))$ is called a *job chain* if it satisfies the following four conditions:

(i) there are no Z-jobs in π;

(ii) the position of each job $\pi(i+1)$ on M_1 is equal to the arrival time of job $\pi(i)$ on M_2, that is,

$$\sigma^{-1}(\pi(i+1)) = \sigma^{-1}(\pi(i)) + l_{\pi(i)} \quad \text{for } i = 1, \ldots, q-1;$$

(iii) the initial position $b(\pi)$ of π, which is defined as the position of the first job in π, is no more than $4mb + 1$, that is,

$$b(\pi) = \sigma^{-1}(\pi(1)) \leq 4mb + 1;$$

(iv) the terminal arrival time $e(\pi)$ of π on M_2, which is defined as the arrival time of the last job in π on M_2, is at least equal to $mu + 1$, that is,

$$e(\pi) = \sigma^{-1}(\pi(q)) + l_{\pi(q)} \geq mu + 1.$$

Given a job chain π, the jobs $\pi(1)$ and $\pi(q)$ are called the initial and terminal job of π. Moreover, $\pi(i+1)$ is called the chain successor of job $\pi(i)$, and job $\pi(i)$ is called the chain predecessor of job $\pi(i+1)$, for $i = 1, \ldots, q-1$.

Lemma 12 (Finding the chain successor). *Let σ be any tight sequence for $\mathcal{I}$; let job j be any L-job or P-job with arrival time no more than n. Let job k be the job that is completed on M_1 exactly at the arrival time of job j on M_2. Then job k is an L-job or a P-job.*

Proof. Suppose to the contrary that job k is a Z-job. Then $l_k = 0$, which implies that job k arrives on M_2 at the time at which it is completed on M_1, which time is equal to the arrival time of job j on M_2. But then there are two different jobs (jobs j and k are of different type and hence different) that arrive at the same time, which contradicts the one-to-one property of tight sequences. ■

Lemma 13 (Finding the chain predecessor). *Let σ be any tight sequence for $\mathcal{I}$; let j be any L-job or P-job that occupies a position $\sigma^{-1}(j) \geq 4mb + 2$. Let k be the job that arrives on M_2 at time $\sigma^{-1}(j)$. Then job k is an L-job or a P-job.*

Proof. The proof is similar to the proof of Lemma 12. ■

Lemma 14. Let σ be any tight sequence for $\mathcal{I}$. Each L-job and each P-job is contained in a unique job chain.

Proof. Given any L-job or P-job, we construct the corresponding chain by applying the steps of finding the chain predecessor and finding the chain successor in Lemmas 12 and 13. It is readily proven that the resulting job subsequence satisfies all properties stated in Definition 11; its uniqueness follows immediately from the one-to-one property of tight sequences. ■

Corollary 15. Let σ be any tight sequence for $\mathcal{I}$. The following observations hold true:

(i) *For any $k = 1, \ldots, 4mb + 1$, there exists a unique job chain π_k with initial position $b(\pi_k) = k$. Consequently, there are exactly $4mb + 1$ job chains.*
(ii) *For any $t = mu + 1, \ldots, mu + 4mb + 1$, there exists a unique job chain π with terminal arrival time $e(\pi) = t$.*

Proof. Part (i) follows immediately from Lemma 14, as Corrollary 6 implies that the job subsequence $\sigma[1, 4mb + 1]$ cannot contain Z-jobs. Corrollary 6 further implies that the arrival times $mu + 1, \ldots, mu + 4mb + 1$ cannot be due to Z-jobs, which in combination with Lemma 14 shows part (ii). ■

Definition 16. The unique job chain π_k, with initial position $b(\pi_k) = k$, is called the kth *job chain*. A *job chain* is called an L-job chain, if it consists of L-jobs only; it is called a *mixed job chain* if it contains both L-jobs and P-jobs.

Lemma 17 (L-Job Chains). *Let σ be any tight sequence for $\mathcal{I}$. Then for each $k = 1, \ldots, 4mb - m + 1$ job chain π_k, with initial position $b(\pi_k) = k$, is an L-job chain.*

Proof. Lemma 9 states that $\sigma[1, 4mb - m + 1] = \alpha_1$, which consists of L-jobs only. This implies that the initial job of each job chain $\pi_k (k = 1, \ldots, 4mb - m + 1)$ is an L-job. Using Lemma 12 we find that the second job in π_k is a job from α_2, and hence it is an L-job as well. Continuing in this fashion we find that each job chain $\pi_k (k = 1, \ldots, 4mb - m + 1)$ consists of L-jobs only. ■

Corollary 18. Let σ be any tight sequence for $\mathcal{I}$. Then the job chains $\pi_k (k = 1, \ldots, 4mb - m + 1)$ possess the following properties:

(i) *each $\pi(k)$ consists of m jobs, which are all L-jobs;*
(ii) $\bigcup_{k=1}^{4mb-m+1} \pi_k = \bigcup_{j=1}^{m} \alpha_j$;
(iii) $\bigcup_{k=1}^{4mb-m+1} \{e(\pi_k)\} = [mu + m + 1, mu + 4mb + 1]$.

Lemma 19 (Mixed Job Chains). *Let σ be any tight sequence for $\mathcal{I}$. Then for each $k = 4mb - m + 2, \ldots, 4mb + 1$ job chain π_k, with initial position $b(\pi_k) = k$, is a mixed chain with the following properties*:

(i) *the terminal arrival time of $\pi_k (k = 4mb - m + 2, \ldots, 4mb + 1)$ is equal to $e(\pi_k) = mu + k - (4mb - m + 1)$;*
(ii) *job chain $\pi_k (k = 4mb - m + 2, \ldots, 4mb + 1)$ consists of $(m - 1)$ L-jobs and three P-jobs, where the sum of the delay values of the P-jobs is equal to $4b$.*

Proof. Consider any job chain $\pi_k = (\pi_k(1), \ldots, \pi_k(q))$ with $k \in [4mb-m+2, 4mb+1]$. It follows immediately from the Corollaries 15 and 18 that the only terminal arrival times that are left for π_k are the times $mu + 1, \ldots, mu + m$ (recall that by definition of a job chain $e(\pi_k) \geq mu + 1$). Now consider the number of L-jobs in π_k. Recall that Lemma 17 implies that all jobs in $\alpha_i (i = 1, \ldots, m)$ are contained in the L-job chains. As $b(\pi_k) = k \in [4mb - m + 2, 4mb + 1]$, we have that job $\pi_k(1) \in \beta_1$. Moreover, since $e(\pi_k) \geq mu + 1$ and α_m is forbidden territory for π_k, job $\pi_k(q) \in \beta_m$. This implies that π_k has to 'jump' over the job subsequences $\alpha_2, \ldots, \alpha_m$, which requires an L-job per jump. Hence, π_k contains at least $(m - 1)$ L-jobs; as $e(\pi_k) - b(\pi_k) < mu$, there must be exactly $(m - 1)$ L-jobs in π_k.

This implies that the terminal arrival time of π_k is equal to

$$e(\pi_k) = k + (m - 1)(u + 1) + \sum_{j \in \pi_k \cap P} x_j. \tag{4}$$

Moreover, we know that $e(\pi_k) \in [mu + 1, mu + m]$. Since we have assumed in our choice of the instance of 3-PARTITION that all x_j's are divisible by m, we know that $e(\pi_k) \equiv k - 1 \pmod m$. Hence, through this relation we can determine the terminal arrival time $e(\pi_k)$ from among the m candidates in $[mu + 1, mu + m]$, and we find that $e(\pi_k) = mu + k - (4mb - m + 1)$.

If we substitute this result in (4), then we find that

$$\sum_{j \in \pi_k \cap P} x_j = mu + k - (4mb - m + 1) - (k + (m - 1)(u + 1)) = 4b.$$

It follows immediately from the assumptions made in the definition of 3-PARTITION that π_k must contain exactly three P-jobs. ■

Corollary 20. If there exists a feasible schedule for the instance $\mathcal{I}$ of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with makespan no more than y, then the answer to the 3-PARTITION *instance is 'yes'.*

Theorem 21. The problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ is strongly NP-hard.

Proof. The combination of Lemma 1 and Corollary 20 shows that the answer to 3-PARTITION is 'yes' if and only if for instance $\mathcal{I}$ there exists a schedule with makespan no more than y. All that is left to show is that the decision variant of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ belongs to NP, which is a triviality. ■

8. EXACT DELAYS AND NUMERICAL 3-DIMENSIONAL MATCHING

Theorem 21 can be applied to show strong NP-hardness of a number of other problems. The first problem has exact delays instead of minimum delays; in case of *exact* delays, the second operation of job j must start exactly l_j time units after the completion time of the first operation of job j. The computational complexity of this problem follows immediately from Theorem 21, and we therefore state the following corollary without proof.

Corollary 22. The two-machine flow shop problem of scheduling unit processing time jobs with exact delays is strongly NP-hard.

The next problem is a severely restricted variant of NUMERICAL 3-DIMENSIONAL MATCHING; we will show that it is NP-complete in the strong sense, too. The general problem is defined as follows (see Garey and Johnson, 1979):

NUMERICAL 3-DIMENSIONAL MATCHING. Given three multisets of integers

$$U = \{u_1, \ldots, u_n\}, \quad V = \{v_1, \ldots, v_n\}, \quad \text{and} \quad W = \{w_1, \ldots, w_n\}$$

and an integer e such that

$$\sum_{j=1}^{n} (u_j + v_j + w_j) = ne,$$

do there exist two permutations λ and μ of $\{1, \ldots, n\}$ such that

$$u_j + v_{\lambda(j)} + w_{\mu(j)} = e, \quad \text{for } j = 1, \ldots, n?$$

We consider the restricted version of this problem in which two of the three integer sets are index sets $\{1, \ldots, n\}$; we denote this problem as RN3DM. This problem is stated as follows:

RN3DM. Given a multiset $U = \{u_1, \ldots, u_n\}$ of integers and an integer e such that $\sum_{j=1}^{n} u_j + n(n+1) = ne$, do there exist two permutations λ and μ such that

$$u_j + \lambda(j) + \mu(j) = e, \quad \text{for } j = 1, \ldots, n?$$

Theorem 23. The problem RN3DM *is strongly NP-complete.*

Proof. In the previous sections, we have shown that given any instance of 3-PARTITION it is possible to construct an instance of the decision variant of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ that is answered affirmatively if and only if the answer to the instance of 3-PARTITION is 'Yes'. Let $\mathcal{I}$ be the instance of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ and y be the threshold. We will now show how to construct an instance of RN3DM that is answered affirmatively if and only if $\mathcal{I}$ admits a feasible schedule with makespan no more than y. This reduction from 3-Partition to RN3DM proves Theorem 23; RN3DM is a member of the class NP, as it is a special case of NUMERICAL 3-DIMENSIONAL MATCHING.

Given the instance $\mathcal{I}$ and threshold y defined in Section 2, define the following instance of RN3DM. Choose $V = \{1, \ldots, n\}$ and $W = \{1, \ldots, n\}$, where $n = 4m(m+1)b$, which is the number of jobs in $\mathcal{I}$. Choose u_j equal to the delay value l_j, for $j = \{1, \ldots, n\}$, and choose $e = y = n + 4mb + 2$.

First, suppose that there is a schedule σ for $\mathcal{I}$ with makespan no more than y; let σ_1 be the tight sequence on M_1 corresponding to σ. Corollary 6 states that

$$\bigcup_{k=1}^{n}\{\sigma_1^{-1}(k)+l_k\} = \{4mb+2,\ldots,n+4mb+1\} = \{e-n, e-n+1,\ldots,e-1\},$$

where $\sigma_1^{-1}(j)$ denotes the position of job j in σ_1. Hence, by choosing $\lambda = \sigma_1^{-1}$ and μ appropriately, we find that the answer to the constructed instance of RN3DM is 'Yes'.

Conversely, suppose that the answer to the instance of RN3DM, is 'Yes'. Then it is readily shown that $\sigma_1 = \lambda^{-1}$ is a tight sequence. ■

9. THE SINGLE-MACHINE AND THE TWO-MACHINE OPEN SHOP PROBLEM WITH MINIMUM DELAYS

The first problem in this section that we will show to be strongly NP-hard is the problem of minimizing the makespan on a *single machine* where each job consists of two operations with intermediate delays between the execution of these operations; it was studied by Kern and Nawijn (1991). This problem is strongly related to the two-machine flow shop problem with delays. Since we work with minimum delays, there exists an optimal schedule in which all first operations precede all of the second operations. If the delay values are large enough, that is, if all first operations have been completed at the time that the second operations can become available for processing, then the two problems are equivalent. Since the outcome of the decision variant of the problem $F2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$ does not change if we increase all delay values and y with the same positive constant, we obtain the following result, which we state without proof.

Theorem 24. The problem of minimizing the makespan on a single machine with two unit time operations per job with arbitrary intermediate delays is strongly NP-hard.

Next, we prove that the strong NP-hardness of $F2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$ can be used to show strong NP-hardness of the two-machine open shop problem with delays and unit processing times, which is denoted by $O2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$. The open shop environment differs from the flow shop environment in just one point: the order in which the operations within a job are to be executed is no longer fixed but is to be decided upon by the scheduler and may differ per job; the operations of the same job still cannot overlap in their execution. Our NP-hardness result improves upon a result by Dell'Amico and Vaessens (1996), who show that the problem $O2 \mid l_j,\ p_{1j}=p_{2j} \mid C_{\max}$ is strongly NP-hard.

Before we state our reduction, we need a preliminary result, which defines a relation between the kind of instance of the open shop problem $O2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$ that we are going to construct and the corresponding instance of the flow shop problem $F2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$.

Lemma 25. Consider an instance $\mathcal{J}$ of $O2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$ with the following properties:

(i) *the number of jobs n is even;*
(ii) *the average delay $\bar{l}=(n/2-1)$, or equivalently, $\sum_{j=1}^{n} l_j = (n/2-1)n$;*
(iii) *there exists a schedule π for the instance $\mathcal{J}$ of $O2 \mid l_j,\ p_{ij}=1 \mid C_{\max}$ with $C_{\max}(\pi)=n$.*

Let $\mathcal{K}$ be the same instance as $\mathcal{J}$, but then for the flow shop problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. Then there exists a schedule for $\mathcal{K}$ with makespan equal to $3n/2$.

Proof. Given a schedule π for $\mathcal{J}$ with makespan n, partition the job set $N = \{1, 2, \ldots, n\}$ into subsets N_1 and N_2 as follows:

$$N_1 = \{j \mid O_{1j} \text{ precedes } O_{2j} \text{ in } \pi\};$$

$$N_2 = \{j \mid O_{2j} \text{ precedes } O_{1j} \text{ in } \pi\}.$$

Define $n_1 = |N_1|$ and $n_2 = |N_2|$; we have that $n_1 + n_2 = n$. We start by showing that $n_1 = n_2 = n/2$.

Let $\mathcal{K}_1$ denote the instance of the flow shop problem that we create by removing all jobs in N_2 from the instance $\mathcal{K}$. If we remove all jobs in N_2 from π, then we obtain a feasible schedule for $\mathcal{K}_1$ with makespan at most equal to n. Hence, the lower bound of Lemma 1 for $\mathcal{K}_1$ can be at most equal to n, that is,

$$\sum_{j \in N_1} l_j/n_1 + 1 + n_1 \leq n, \quad \text{or} \quad \text{equivalently, } n_1 n \geq \sum_{j \in N_1} l_j + n_1 + n_1^2.$$

Similarly, we derive that

$$\sum_{j \in N_2} l_j/n_2 + 1 + n_2 \leq n, \quad \text{or} \quad \text{equivalently, } n_2 n \geq \sum_{j \in N_2} l_j + n_2 + n_2^2.$$

Adding up these relations and noting that $N_1 \cup N_2 = N$, we find that

$$n^2 = n_1 n + n_2 n \geq \sum_{j=1}^{n} l_j + n_1 + n_2 + n_1^2 + n_2^2 = n^2/2 + n_1^2 + n_2^2,$$

where the last equality sign comes from Property (ii). Hence, we have that $n_1^2 + n_2^2 \leq n^2/2$, which together with $n_1 + n_2 = n$ implies that $n_1 = n_2 = n/2$.

Through a simple interchange argument, we can show that there exists an optimal schedule in which all jobs in $N_1(N_2)$ precede the jobs in $N_2(N_1)$ on $M_1(M_2)$. As π has makespan n and $n_1 = n_2 = n$, π must have the form depicted in Figure 2. In this figure, the subsequences of the operations executed by M_1 are denoted by σ_1 and σ_2, depending on whether they belong to N_1 or N_2; similarly, the subsequences of the M_2-operations of the jobs in N_1 and N_2 are denoted by τ_1 and τ_2, respectively.

Given this schedule π, we can easily determine a feasible schedule π' for the flow shop problem with makespan no more than $3n/2$, as indicated in Figure 3. ■

We establish strong NP-hardness of $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ by a reduction that is strongly related to the reduction used to prove strong NP-hardness of $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$. The reduction is again

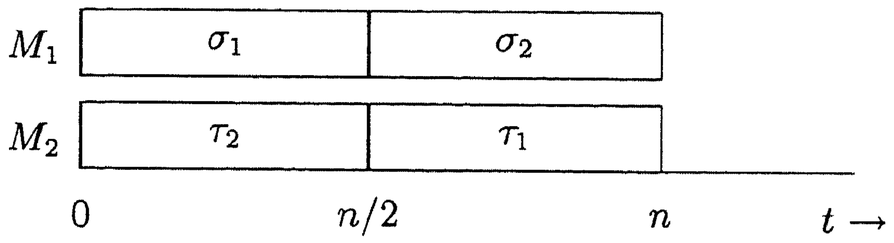

Figure 2. Schedule π

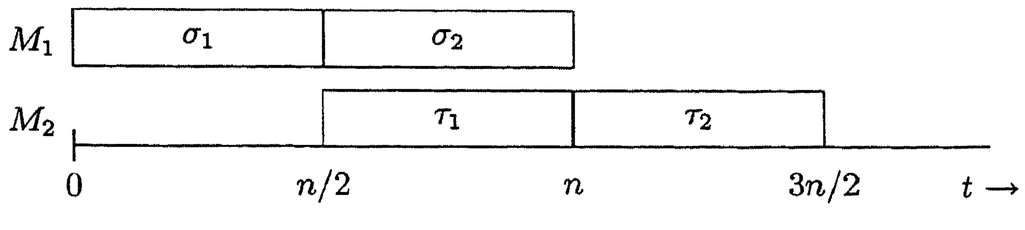

Figure 3. Schedule π'

from 3-PARTITION, but we need an even number of triples. We take care of this by adding, if m is odd, the triple of integers $(2b-2, b+1, b+1)$ to X. It is readily proven that the addition of this triple does not change the outcome of 3-PARTITION. If necessary, we multiply all x_j-values and b by m.

Given an instance of 3-PARTITION with an even number of triples, we construct the instance $\mathcal{I}'$ of the problem $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ by copying the instance $\mathcal{I}$ that we defined in Section 2, but with one difference: all delay values are increased by

$$\Delta = n/2 - (4mb + 2) = 2m(m+1)b - (4mb + 2).$$

The threshold is equal to the number of jobs n. We are asked to decide whether there exists a feasible schedule for the instance $\mathcal{I}'$ of $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with makespan no more than n.

*Lemma 26. If the answer to the 3-*PARTITION *instance is 'yes', then there exists a schedule for the instance $\mathcal{I}'$ of the problem $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with $C_{\max} \leq n$.*

Proof. Suppose that $\{X_1, \ldots, X_m\}$ is a partition of X that leads to 'yes' to 3-PARTITION. From $X_1, \ldots, X_{m/2}$ we construct the blocks $B_1, \ldots, B_{m/2}$ in the same way as described in the proof of Lemma 1 and depicted in Figure 1, albeit that the execution of the operations on M_2 is shifted Δ units to the right, because of the increased delays. Note that in such a block the first operation on M_2 starts $n/2$ time units after the start time of the first operation on M_1. From $X_{m/2+1}, \ldots, X_m$ we construct the blocks $B_{m/2+1}, \ldots, B_m$ in the same way, but with the roles of M_1 and M_2 reversed, that is, the operations are first processed by M_2 and then by M_1.

The blocks can be patched together to form a schedule as shown in Figure 2: the blocks $B_1, \ldots, B_{m/2}$ form the sequences σ_1 and τ_1, whereas the blocks $B_{m/2+1}, \ldots, B_m$ constitute the sequences τ_2 and σ_2. ■

*Lemma 27. If the instance $\mathcal{I}'$ admits a schedule for the problem $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with makespan no more than n, then the answer to the 3-*PARTITION *instance is 'yes'.*

Proof. We prove this result by using the flow shop problem as an intermediate. We copy the instance $\mathcal{I}$ defined in Section 2 and increase all delay values by Δ; note that this is exactly instance $\mathcal{I}'$. Then we ask ourselves the question: does there exist a schedule for the instance $\mathcal{I}'$ of the problem $F2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ with $C_{\max} \leq 3n/2$?

We have assumed that for the instance $\mathcal{I}'$ of the open shop problem there exists a schedule π with $C_{\max}(\pi) \leq n$. As in $\mathcal{I}'$ the total processing time of the jobs on M_1 is equal to n, we must have that $C_{\max}(\pi) = n$. A simple check shows that all properties of Lemma 25 are fulfilled, and hence applying Lemma 25 implies that for the instance $\mathcal{I}'$ of the flow shop problem there exists a schedule

π' for it with makespan no more than $3n/2$. This answers the question that we asked ourselves above.

Recall that the instances $\mathcal{I}$ and $\mathcal{I}'$ are identical except for the delay values, which all are Δ bigger in $\mathcal{I}'$ than in $\mathcal{I}$. If we decrease all delay values by Δ, then the schedule σ that we obtain from π' by starting all jobs on M_2 Δ time units earlier is a feasible schedule for the instance $\mathcal{I}$ for the flow shop problem with $C_{\max}(\sigma) \leq 3n/2 - \Delta \equiv n + 4mb + 2$, which value we recognize as the threshold defined in Section 2. Hence, Corollary 20 implies that the answer to 3-PARTITION is 'yes'. ■

Theorem 28. The problem $O2 \mid l_j, p_{ij} = 1 \mid C_{\max}$ *is strongly NP-hard.*

ACKNOWLEDGEMENTS

This research was conducted when the first author visited the Technische Universiteit Eindhoven. The visit was made possible by a grant from EIDMA, the Euler Institute of Discrete Mathematics and its Applications.

REFERENCES

Coster, M. J., Private communication, 1993.

Dell'Amico, M., "Shop problems with two machines and time lags," *Operations Research*, **44**, 777–787 (1996).

Dell'Amico, M. and R. J. M. Vaessens, "Flow and open shop scheduling on two machines with transportation times and machine-independent processing times is NP-hard," *Materiali di discussione* 141, Dipartimento di Economia Politica, Università di Modena, 1996.

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

Graham, R. L., E. L. Lawler, J. K. Lenstra, and A. H. G. Rinnooy Kan, Optimization and approximation in deterministic sequencing and scheduling: A survey. *Annals of Discrete Mathematics*, **5**, 287–326 (1979).

Hoogeveen, J. A., J. K. Lenstra, and S. L. van de Velde, "Sequencing and scheduling," in M. Dell'Amico, F. Maffioli, and S. Martello (eds.), *Annotated Bibliographies in Combinatorial Optimization*, Wiley, Chichester, 1997, Ch. 12.

Johnson, S. M., "Optimal two- and three-stage production schedules with setup times included," *Naval Research Logistics Quarterly*, **1**, 61–68 (1954).

Johnson, S. M., "Discussion: Sequencing n jobs on two machines with arbitrary time-lags." *Management Science*, **5**, 299–303 (1958).

Kern, W. and W. M. Nawijn, "Scheduling multi-operation jobs with time lags on a single machine," in U. Faigle and C. Hoede (eds.), *Proceedings 2nd Twente Workshop on Graphs and Combinatorial Optimization*, Enschede, 1991.

Lenstra, J. K. Private communication, 1991.

Mitten, L. G., "Sequencing n jobs on two machines with arbitrary time-lags," *Management Science*, **5**, 293–298 (1958).

Orman, A. J. and C. N. Potts, "On the complexity of coupled-task scheduling," *Discrete Applied Mathematics*, **72**, 141–154 (1997).

Yu, W., *The Two-Machine Flow Shop Problem with Delays and the One-Machine Total Tardiness Problem*, PhD Thesis, Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, 1996.