Alcom-FT Technical Report Series
ALCOMFT-TR-01-122

The three-machine proportionate flow shop problem with unequal machine speeds

Sixiang Hou
Institue of Policy and Management
Chinese Academy of Sciences
P.O.Box 8712, Beijing 100080, China
Email: housixin@public.east.cn.net

Han Hoogeveen
Department of Computer Science
Utrecht University
P.O. Box 80089, 3508 TB Utrecht, The Netherlands
Email: slam@cs.uu.nl

May 31, 2001

Abstract

In flow shop scheduling, it is usually assumed that the processing times of the operations belonging to the same job are unrelated. But when a job corresponds to producing a certain quantity of some good, then it is likely that the processing times are related to this quantity, and hence are constant except for some factor that depends on the speed of the machine. In this paper, we consider this special case of the flow shop problem, which we call the proportionate flow shop problem with unequal machine speeds. We study the problem with three machines, where we assume that the second machine is slowest, with the objective of minimizing the makespan. We show that then there exists an optimum schedule that is V-shaped. As a consequence, this problem can be solved in pseudopolynomial time, and we show that this is best possible by establishing $\mathcal{NP}$-hardness in the ordinary sense.

1980 Mathematics Subject Classification (Revision 1991): 90B35.
Keywords and Phrases: proportionate flow shop, machine speeds, V-shaped, makespan, dynamic programming, $\mathcal{NP}$-hardness.

1 Introduction

In many manufacturing and assembly facilities a number of operations have to be done on every job. Often, these operations have to be done on all jobs in the same order, which implies that the jobs have to follow the same route through the machines. Such an environment is referred to as a *flow shop*. Mathematically, our model is described as follows. There are m machines M_i $(i = 1, \ldots, m)$ that have to process n jobs. The machines are continuously available from time zero onwards and can handle at most one job at a time. Each job j consists of a *chain* of m

operations O_{ij} ($i = 1, \ldots, m; j = 1, \ldots, n$), which implies that the execution of O_{ij} cannot start before the execution of $O_{i-1,j}$ has been completed, for $i = 2, \ldots, m$. Operation i ($i = 1, \ldots, m$) of job j ($j = 1, \ldots, n$) has to be processed by M_i, which requires an uninterrupted time of length p_{ij}.

In this paper we consider a special case of a flow shop problem with three machines, that is, $m = 3$. Our objective is to minimize the makespan or length of the schedule, that is, the time at which machine 3 completes the last job. The specialty lies in the processing times of the operations. The standard assumption is that there is no relation between the processing times, but it is easy to imagine a situation in which a job j models the event of a costumer ordering a quantity q_j that has to be produced by the machines; it may be expected that the processing time of this job on any one of the machines is then proportionate to this quantity q_j. We assume that the processing time p_{ij} of operation O_{ij} ($i = 1, 2, 3; j = 1, \ldots, n$) is equal to $p_{ij} = p_j/s_i$, where s_i ($i = 1, 2, 3$) denotes the speed of M_i. Using the classification scheme of Graham, Lawler, Lenstra, and Rinnooy Kan (1979), we denote this problem as $F3|p_{ij} = p_j/s_i|C_{\max}$. We call this type of flow shop a *proportionate flow shop with unequal machine speeds.* This is a generalization of the problem with equal machine speeds, that is, $p_{ij} = p_j$. This special case is known in the literature as the *proportionate* flow shop, see Ow (1985) and Pinedo (1985, 1995). Pinedo observed that proportionate flow shop models in a number of cases are very similar to their single machine counterparts (see Pinedo (1995), page 103): the rule that solves the single-machine scheduling problem needs only a very slight modification to be applicable to the corresponding proportionate flow shop, except for the problem of minimizing the total weighted completion, which needs a more intricate approach (Shakhlevich, Hoogeveen, and Pinedo, 1998).

In this paper, we will consider the problem $F3|p_{ij} = p_j/s_i|C_{\max}$ with the additional assumption that $s_1 \geq s_3 > s_2$, that is, the second machine is the slowest. We will show that there exists an optimal schedule that is V-shaped, that is, the jobs in front of the job with the largest processing time are scheduled in shortest-processing-time (SPT) order, whereas the jobs after the largest job are in longest-processing-time (LPT) order. Using this result, we derive a dynamic processing time algorithm that runs in pseudo-polynomial time, and we show that this is the best we can hope for by establishing $\mathcal{NP}$-hardness of this problem.

2 Preliminary results

The $F3|p_{ij} = p_j/s_i|C_{\max}$ problem is a special case of the $F3||C_{\max}$ problem, which implies that we can use the fundamental results that have been obtained for this problem by Conway, Maxwell, and Miller (1967), who show the following lemma.

Lemma 1 *There exists an optimal schedule for the $F3||C_{\max}$ problem in which all machines execute the jobs in the same order.* ∎

Hence, from now on, we restrict our attention to this kind of schedules, which are called *permutation schedules.* Another well-known fact is that the $F3||C_{\max}$ problem with instance $\mathcal{I}$ is equivalent to the $F3||C_{\max}$ problem with the instance $\mathcal{I}'$ that is obtained by switching the processing times p_{1j} and p_{3j} for each $j = 1, \ldots, n$.

Lemma 2 *Let π be an optimal schedule for the instance $\mathcal{I}$ of the $F3||C_{\max}$ problem. Then the $F3||C_{\max}$ problem with instance $\mathcal{I}'$ is solved by executing the jobs in the reverse order of π.* ■

The makespan of a permutation schedule π can be obtained as follows. Renumber the jobs according to their order in π, that is, in π the machines execute the jobs in the order $1, 2, \ldots, n$. Since the objective criterion is regular, each operation i ($i = 1, 2, 3$) of job j ($j = 1, \ldots, n$) starts as soon as possible, that is, as soon as both machine i and job j are available. Since operation 1 of each job is available at time zero, the processing of the jobs on machine 1 is contiguous. The makespan of schedule π, denoted by $C_{\max}(\pi)$, is equal to the length of the longest path in the Gantt chart depicting π from the first operation of job 1 to the last operation of job 3; this path does not contain idle time. It first follows the jobs on M_1 until some job k whose second operation starts on M_2 as soon as the first operation of k gets completed at M_1; next it follows M_2 until some job q whose third operation starts on M_3 as soon as the second operation of job q gets completed at M_2; finally it follows the jobs $q, \ldots, n$ on M_3. We find that

$$C_{\max}(\pi) = \max_{1 \le k \le q \le n} \left\{ \frac{1}{s_1} \sum_{l=1}^{k} p_l + \frac{1}{s_2} \sum_{l=k}^{q} p_l + \frac{1}{s_3} \sum_{l=q}^{n} p_l \right\}. \tag{1}$$

Suppose that the maximum is attained by choosing $k = w$ and $q = z$. The jobs w and z are called the *pivots* or *pivotal jobs*; the length of this path is denoted by $l(w, z)$. Hence, we have that

$$C_{\max}(\pi) = l(w, z) = \frac{1}{s_1} \sum_{l=1}^{w} p_l + \frac{1}{s_2} \sum_{l=w}^{z} p_l + \frac{1}{s_3} \sum_{l=z}^{n} p_l.$$

Lemma 3 *If w and z are the pivots in π, then the following conditions hold:*

$$\frac{1}{s_1} \sum_{l=j+1}^{w} p_l \ge \frac{1}{s_2} \sum_{l=j}^{w-1} p_l \qquad \forall\, j = 1, \ldots, w-1; \tag{2}$$

$$\frac{1}{s_2} \sum_{l=w}^{j-1} p_l \ge \frac{1}{s_1} \sum_{l=w+1}^{j} p_l \qquad \forall\, j = w+1, \ldots, z; \tag{3}$$

$$\frac{1}{s_2} \sum_{l=j+1}^{z} p_l \ge \frac{1}{s_3} \sum_{l=j}^{z-1} p_l \qquad \forall\, j = w, \ldots, z-1; \tag{4}$$

$$\frac{1}{s_3} \sum_{l=z}^{j-1} p_l \ge \frac{1}{s_2} \sum_{l=z+1}^{j} p_l \qquad \forall\, j = z+1, \ldots, n. \tag{5}$$

Proof. We only prove condition 2; the conditions 3 - 5 can be proven in a similar fashion. Suppose to the contrary that condition 2 does not hold for some $j \in \{1, \ldots, w-1\}$. We will show that the choice of $k = w$ and $q = z$ does not solve the maximization problem of equation 1 then:

$$l(w, z) = \frac{1}{s_1} \sum_{l=1}^{w} p_l + \frac{1}{s_2} \sum_{l=w}^{z} p_l + \frac{1}{s_3} \sum_{l=z}^{n} p_l = \frac{1}{s_1} \sum_{l=1}^{j} p_l + \frac{1}{s_1} \sum_{l=j+1}^{w} p_l +$$

$$\frac{1}{s_2}\sum_{l=w}^{z} p_l + \frac{1}{s_3}\sum_{l=z}^{n} p_l < \frac{1}{s_1}\sum_{l=1}^{j} p_l + \frac{1}{s_2}\sum_{l=j}^{w-1} p_l + \frac{1}{s_2}\sum_{l=w}^{z} p_l + \frac{1}{s_3}\sum_{l=z}^{n} p_l = l(j,z).$$

■

Lemma 3 identifies necessary conditions for a pair of jobs to be the pivots. For the $F3||C_{\max}$ problem with arbitrary processing times, these conditions are not sufficient. Consider for instance the following two-job example with processing times $(0,1,1)$ for the operations of job 1 and $(3,0,0)$ for the operations of job 2. It is optimal to process job 1 first and then job 2; job 2 is the pivot twice (that is, $w = z = 2$). The choice $w = z = 1$ satisfies the conditions of Lemma 1 as well, however, but the length of this path is only 2, whereas the minimum makespan is 3. In case of proportionate processing times with $s_1 \geq s_3 > s_2$, the conditions of Lemma 3 are also sufficient, as we will prove below.

Lemma 4 *If in the schedule $\pi = (1, 2, \ldots, n)$ the indices w and z with $w \leq z$ satisfy the conditions 2 - 5, then $C_{\max}(\pi) = l(w,z)$.*

Proof. Suppose to the contrary that $C_{\max}(\pi) = l(x,y) > l(w,z)$; hence, the combination (x,y) satisfies the conditions 2 - 5 as well. The proof consists of a number of case distinctions, depending on the order between w, x, y, z. Without loss of generality, we assume that $w \leq x$; the case that $x \leq w$ is proven in a similar way. Since $x \leq y$, we cover all possibilities by distinguishing between $w \leq z < x \leq y$, $w \leq x \leq z \leq y$, and $w \leq x \leq y \leq z$.

(1) Suppose that $w \leq z < x \leq y$. In that case, we find that

$$\frac{1}{s_1}\sum_{l=z+1}^{x} p_l \geq \frac{1}{s_2}\sum_{l=z}^{x-1} p_l \quad \text{and} \quad \frac{1}{s_3}\sum_{l=z}^{x-1} p_l \geq \frac{1}{s_2}\sum_{l=z+1}^{x} p_l,$$

where the first inequality comes from condition 2 with pivot x and $j = z$; the second inequality comes from condition 5 with pivot z and $j = x$. Combining these two leads to

$$\frac{1}{s_1}\sum_{l=z+1}^{x} p_l \geq \frac{1}{s_2}\sum_{l=z}^{x-1} p_l = \frac{s_3}{s_2}\frac{1}{s_3}\sum_{l=z}^{x-1} p_l \geq \frac{s_3}{s_2}\frac{1}{s_2}\sum_{l=z+1}^{x} p_l,$$

which implies that $s_2^2 \geq s_1 s_3$, which contradicts the assumption that $s_1 \geq s_3 > s_2$.

(2) The case $w \leq x \leq z \leq y$. Applying condition 3 with pivot w and $j = x$ and condition 5 with pivot z and $j = y$ yields

$$\frac{1}{s_2}\sum_{l=w}^{x-1} p_l \geq \frac{1}{s_1}\sum_{l=w+1}^{x} p_l \quad \text{and} \quad \frac{1}{s_3}\sum_{l=z}^{y-1} p_l \geq \frac{1}{s_2}\sum_{l=z+1}^{y} p_l.$$

With these inequalities it is straightforward to show that $l(w,z) \geq l(x,y)$.

(3) The case $w \leq x \leq y \leq z$. Applying condition 3 with pivot w and $j = x$ and condition 4 with pivot z and $j = y$ yields

$$\frac{1}{s_2}\sum_{l=w}^{x-1} p_l \geq \frac{1}{s_1}\sum_{l=w+1}^{x} p_l \quad \text{and} \quad \frac{1}{s_2}\sum_{l=y+1}^{z} p_l \geq \frac{1}{s_3}\sum_{l=y}^{z-1} p_l.$$

It is now readily proven that $l(w,z) \geq l(x,y)$. ■

Lemma 4 provides a useful mathematical characterization of the pivots, which determine $C_{\max}(\pi)$. From now on, we will use the index w to denote the pivot from M_1 to M_2, and we will let z denote the pivot from M_2 to M_3. The combination (w,z) does not have to be unique, but each pivot pair leads to the same makespan. The following two lemmas reveal some further properties of the pivots.

Lemma 5 *Consider any schedule $\pi = (1,2,\ldots,n)$ with pivots w and z. If for some index $q < w \leq z$ holds that*

$$\frac{1}{s_2}\sum_{l=q}^{k-1} p_l \geq \frac{1}{s_1}\sum_{l=q+1}^{k} p_l \qquad \forall\, k = q+1, q+2, \ldots, z,$$

then (q,z) is a pivot pair as well.

Proof. Since $q < w \leq z$, we obtain by substituting $k = w$ that $\frac{1}{s_2}\sum_{l=q}^{w-1} p_l \geq \frac{1}{s_1}\sum_{l=q+1}^{w} p_l$. But condition 2 with pivot w and $j = q$ just states the reverse, which implies that $\frac{1}{s_2}\sum_{l=q}^{w-1} p_l = \frac{1}{s_1}\sum_{l=q+1}^{w} p_l$. Hence, we find that $l(q,z) = l(w,z) = C_{\max}(\pi)$, which implies that (q,z) is a pivot pair as well. ■

Lemma 6 *Consider any schedule $\pi = (1,2,\ldots,n)$ with pivots w and z. If for some index $w \leq q < z$ holds that*

$$\frac{1}{s_3}\sum_{l=q}^{k-1} p_l \geq \frac{1}{s_2}\sum_{l=q+1}^{k} p_l \qquad \forall\, k = q+1, q+2, \ldots, n,$$

then (w,q) is a pivot pair as well.

Proof. The proof proceeds along the same lines as the proof of the previous lemma. ■

3 Proving the V-shape of an optimum schedule

We are now ready to prove the key result of this paper: if $s_1 \geq s_3 > s_2$, then there exists an optimal schedule for the three-machine proportionate flow shop problem that is V-shaped. Our proof proceeds by modifying an optimal schedule until it is V-shaped, while taking care that at each step the makespan does not increase. We start with the part before pivot w.

Lemma 7 *Let π denote an optimal schedule with pivots w and z. Putting the jobs $1,\ldots,w$ in SPT-order does affect neither the pivots nor the makespan.*

Proof. We modify π by finding the first pair of adjacent jobs in π before w that are not in SPT-order; let these jobs be h and $h+1$ and suppose that $h+1 < w$. Interchanging these jobs leads to a schedule π'. We will show that w and z satisfy the pivot conditions of Lemma 3 in π'. It turns out that conditions 3 - 5 follow immediately from the fact that w and z are pivots in π. Moreover, the conditions 2 also follow from w being pivot in π, except for $j = h, h+1$. Hence, we only have to prove that

$$\frac{1}{s_1}\sum_{l=h+2}^{w} p_l + \frac{p_h}{s_1} \geq \frac{1}{s_2}\sum_{l=h}^{w-1} p_l \quad \text{and} \quad \frac{1}{s_1}\sum_{l=h+2}^{w} p_l \geq \frac{1}{s_2}\sum_{l=h+2}^{w-1} p_l + \frac{p_h}{s_2}.$$

As to the first one, we have that $p_h > p_{h+1}$, and hence

$$\frac{1}{s_1}\sum_{l=h+2}^{w} p_l + \frac{p_h}{s_1} > \frac{1}{s_1}\sum_{l=h+2}^{w} p_l + \frac{p_{h+1}}{s_1} = \frac{1}{s_1}\sum_{l=h+1}^{w} p_l \geq \frac{1}{s_2}\sum_{l=h}^{w-1} p_l.$$

The inequality for $j = h+1$ follows from adding up the inequalities

$$\frac{1}{s_1}\sum_{l=h+1}^{w} p_l \geq \frac{1}{s_2}\sum_{l=h}^{w-1} p_l \quad \text{and} \quad \frac{p_{h+1}}{s_2} > \frac{p_{h+1}}{s_1},$$

where the last inequality follows from $s_1 > s_2$.

Hence, we can put the jobs $1, \ldots, w-1$ into SPT-order in π, while the pivot pair (w, z) remains valid. Then condition 2 with $j = w-1$ states that

$$\frac{p_w}{s_1} \geq \frac{p_{w-1}}{s_2},$$

which implies that $p_w > p_{w-1}$, since $s_1 > s_2$. Hence, the jobs $1, \ldots, w$ are in SPT-order, and the identity of the pivots does not change. This implies that $l(w, z) = C_{\max}(\pi)$. ■

Since we can reverse the schedule (Lemma 2), it follows immediately from the previous lemma that we can reschedule the part of π containing the jobs $z, \ldots, n$ in LPT-order such that both the pivots and the length of the schedule remain the same. Therefore, we state the following lemma without proof.

Lemma 8 *Let π denote an optimal schedule with pivots w and z. Putting the jobs $z, \ldots, n$ in LPT-order does affect neither the pivots nor the makespan.* ■

Corollary 9 *Let π be any permutation schedule for the three machine proportionate flow shop problem, and let the job with longest processing time be scheduled on position y. Then $w \leq y \leq z$.*

Proof. We denote the longest job by job 0 with processing time p_0. Suppose to the contrary that $y < w$. Lemma 7 states that after putting job 0 on position $w-1$ job w is still the pivot. But then condition 2 is no longer satisfied for $j = w-1$, since

$$\frac{p_0}{s_2} > \frac{p_0}{s_1} \geq \frac{p_w}{s_1},$$

which is a contradiction. The case $y > z$ is dealt with similarly. ■

All that is left to do is to rearrange the jobs between job w and job 0 in π in SPT-order. We will show in the proof of the next lemma that two adjacent jobs h and $h+1$ with $h+1 < y$ and $p_h > p_{h+1}$ can be interchanged without increasing the makespan. Hence, after a finite number of interchanges the jobs before job 0 are in SPT-order, and the makespan has not been increased.

Lemma 10 *Let π denote an optimal schedule with pivots w and z in which the job with maximum processing time is scheduled on position y with $w \leq y \leq z$. Then the jobs $1, \ldots, y-1$ can be put in SPT-order without increasing the makespan.*

Proof. We modify π by interchanging two adjacent jobs h and $h+1$ with $h \geq w$ and $h+1 < y$ that are not in SPT-order, that is, $p_h > p_{h+1}$. Let π' denote the corresponding schedule; let w' and z' denote the pivot jobs in π'; and let $l'(w', z')$ denote the length of the path in π' with pivots w' and z'. We divide our proof into two cases:

- Case 1: $h > w$;
- Case 2: $h = w$.

Proof of 1. We first show that the combination (w', z) is a pivot pair. First, suppose that $z > z'$. Since z' is the pivot in π' and since the part after position $y \leq z'$ has not been changed, we know that

$$\frac{1}{s_3} \sum_{l=z'}^{j-1} p_l \geq \frac{1}{s_2} \sum_{l=z'+1}^{j} p_l \qquad \forall\, j = z'+1, \ldots, n.$$

Since $w \leq y \leq z' < z$, we can apply Lemma 6, and we find that if $z' < z$, then (w, z') is a pivot pair in π. If $z < z'$, then we can show in the same way that (w', z) is a pivot pair in π'. Hence, we may assume that $z' = z$.

We proceed by showing that $w' \leq w$. Suppose to the contrary that $w' > w$. Then we will show that we can apply Lemma 5 with $q = w$, which then shows that (w, z) is a pivot pair too. Hence, we have to show that

$$\frac{1}{s_2} \sum_{l=w}^{k-1} p_{\pi'(l)} \geq \frac{1}{s_1} \sum_{l=w+1}^{k} p_{\pi'(l)} \qquad \forall\, k = w+1, \ldots, z,$$

where $\pi'(l)$ refers to the job on position l in π'. Since π and π' are identical except for the interchange of the jobs h and $h+1$, we derive the desired inequality for $k = w+1, \ldots, h-1$ and for $k = h+2, \ldots, z$ from condition 3 with pivot w in π. What is left is to show that

$$\frac{1}{s_2} \sum_{l=w}^{h-1} p_l \geq \frac{1}{s_1} \sum_{l=w+1}^{h-1} p_l + \frac{p_{h+1}}{s_1} \quad \text{and} \quad \frac{1}{s_2} \sum_{l=w}^{h-1} p_l + \frac{p_{h+1}}{s_2} \geq \frac{1}{s_1} \sum_{l=w+1}^{h+1} p_l.$$

As to the first one, we have that $p_h > p_{h+1}$, and hence

$$\frac{1}{s_2} \sum_{l=w}^{h-1} p_l \geq \frac{1}{s_1} \sum_{l=w+1}^{h} p_l > \frac{1}{s_1} \sum_{l=w+1}^{h-1} p_l + \frac{p_{h+1}}{s_1}.$$

The second inequality follows from adding up the inequalities

$$\frac{1}{s_2} \sum_{l=w}^{h-1} p_l \geq \frac{1}{s_1} \sum_{l=w+1}^{h} p_l \quad \text{and} \quad \frac{p_{h+1}}{s_2} > \frac{p_{h+1}}{s_1}.$$

By now, we have shown that $w' \leq w$. If $w' = w$, then the length of the longest path remains the same (recall that $w < h$), which implies that $C_{\max}(\pi) = C_{\max}(\pi')$. If $w' < w$, then

$$C_{\max}(\pi') = l'(w', z) = l(w', z) \leq l(w, z) = C_{\max}(\pi),$$

since the choice (w, z) solves the longest path problem; this settles the proof of Case 1.

Proof of 2. Just like above, we can show that $z' = z$. If $w' \notin \{w, w+1\}$, then

$$C_{\max}(\pi') = l'(w', z) = l(w', z) \leq l(w, z) = C_{\max}(\pi).$$

Hence, we only have to deal with the cases $w' = w$ and $w' = w + 1$. If $w' = w$, then we find that

$$C_{\max}(\pi') = \frac{1}{s_1}\sum_{l=1}^{w-1} p_l + \frac{p_{w+1}}{s_1} + \frac{1}{s_2}\sum_{l=w}^{z} p_l + \frac{1}{s_3}\sum_{l=z}^{n} p_l < C_{\max}(\pi),$$

since $p_{w'} = p_{w+1} < p_w$. On the other hand, if $w' = w + 1$, then

$$C_{\max}(\pi') = \frac{1}{s_1}\sum_{l=1}^{w+1} p_l + \frac{p_w}{s_2} + \frac{1}{s_2}\sum_{l=w+2}^{z} p_l + \frac{1}{s_3}\sum_{l=z}^{n} p_l < C_{\max}(\pi),$$

since $\frac{p_{w+1}}{s_1} < \frac{p_{w+1}}{s_2}$.

After a finite number of interchanges, none of which increases the makespan, a schedule results in which all jobs before J_0 are in SPT-order. ∎

Obviously, we have a similar result for the part after J_0 due to the argument of symmetry shown in Lemma 2. We therefore state the following lemma without proof.

Lemma 11 *Let π denote an optimal schedule with pivots w and z in which the job with maximum processing time is scheduled on position y with $w \leq y \leq z$. Then the jobs $y + 1, \ldots, n$ can be put in LPT-order without increasing the makespan.* ∎

Corollary 12 *There exists an optimal schedule for the proportionate flow shop problem with $s_1 \geq s_3 > s_2$ that is V-shaped.* ∎

4 Solving the problem to optimality

In this section we show that we can find the optimal V-shaped schedule in pseudo-polynomial time. We renumber the jobs such that

$$p_1 \geq p_2 \geq \ldots \geq p_n.$$

For notational convenience, we denote the processing time of operation i ($i = 1, 2, 3$) of job j ($j = 1, \ldots, n$) by p_{ij}; we assume that each p_{ij} is integral, which can be achieved by multiplying each p_j value by the smallest common multiple of s_1, s_2, s_3.

We take a closer look at V-shaped schedules. We call the part before job 1 the *left part*, and the part after job 1 is called the *right part*. In the left part of the schedule, we do not have to care about scheduling the third operation, since its processing time is always smaller than the processing time of the second operation of the succeeding job. Similarly, we can neglect the first operation of the jobs in the right part. Moreover, delaying the operations of the jobs on M_2 in the left part such that processing on the second machine becomes contiguous does not increase the makespan; therefore, we work with this kind of schedules, which we refer to as *contiguous* schedules. We refer to the time during which M_1 is busy and M_2 is idle as the *left-gap*; similary, we define the period during which M_2 is ready and M_3 is still processing jobs as the *right-gap*.

Our algorithm builds the schedule from the inside; given a subschedule for the jobs $1, \dots, j-1$, we add job j to either the left or the right of this subschedule. We use state-variables $f_j(t_1, t_2)$, which indicate the minimum makespan of any subschedule for the jobs $1, \dots, j$ with a left-gap of size t_1 and a right-gap of size t_2.

If we add a job j to the left of the subschedule π_{j-1} corresponding to $f_j(t_1, t_2)$, then we must distinguish between the following two situations.

- If $t_1 \geq p_{2j}$, then we start job j as late as possible before the first job in π_{j-1}. This yields a feasible schedule with a makespan that is p_{1j} bigger; the size of the left-gap has been reduced to $t_1 - p_{2j} + p_{1j}$.

- If $t_1 < p_{2j}$, then putting job j in front of π_{j-1} and starting the operations of job j as late as possible would lead to the situation that the second operation of job j starts $p_{2j} - t_1$ time units before the first operation of job j has been completed. To avoid this infeasibility and to end up with a contiguous schedule, we delay the jobs on M_2 and the jobs on M_3 by $p_{2j} - t_1$ time units. The resulting schedule is then feasible; its makespan has increased by $p_{1j} + p_{2j} - t_1$, and its left-gap is equal to p_{1j}.

We make a similar distinction for adding job j to the right part.

- If $t_2 \geq p_{2j}$, then job j fits in the right-gap, which yields an increase of the makespan by p_{3j} and which reduces the size of the right-gap to $t_2 - p_{2j} + p_{3j}$.

- If $t_2 < p_{2j}$, then the third operation of job j must wait $p_{2j} - t_2$ time units until the processing of the second operation of job j has been completed. This results in an increase of the makespan by $p_{3j} + p_{2j} - t_2$; the size of the right-gap reduces to p_{3j}.

We see that the size of the left-gap is at most equal to p_{11} and at least equal to p_{1j} in π_j; hence, t_1 can assume the values $p_{1j}, \dots, p_{11}$. Similarly, t_2 can assume the values $p_{3j}, \dots, p_{31}$. Working things out, we derive at the following dynamic programming algorithm. As the initialization, we define

$$f_1(t_1, t_2) = \begin{cases} p_{11} + p_{21} + p_{31} & \text{if } t_1 = p_{11} \text{ and } t_2 = p_{31} \\ \infty & \text{otherwise.} \end{cases}$$

For $j = 2, \dots, n$, $t_1 = p_{1j}, \dots, p_{11}$, and $t_2 = p_{3j}, \dots, p_{31}$, we compute $f_j(t_1, t_2)$ through the following recurrence relations:

$$f_j(t_1,t_2) = \begin{cases} \min\{p_{1j} + f_{j-1}(t_1 + p_{2j} - p_{1j}, t_2), & \text{if } t_1 > p_{1j}, t_2 > p_{3j} \\ \quad p_{3j} + f_{j-1}(t_1, t_2 + p_{2j} - p_{3j})\} & \\ & \\ \min\{\min_{t_1 \le p_{2j}}\{p_{1j} + p_{2j} - t_1 + f_{j-1}(t_1, t_2)\}, & \text{if } t_1 = p_{1j}, t_2 > p_{3j}, \\ \quad p_{3j} + f_{j-1}(p_{1j}, t_2 + p_{2j} - p_{3j})\} & \\ & \\ \min\{p_{1j} + f_{j-1}(t_1 + p_{2j} - p_{1j}), & \text{if } t_1 > p_{1j}, t_2 = p_{3j} \\ \quad \min_{t_2 \le p_{2j}}\{p_{3j} + p_{2j} - t_2 + f_{j-1}(t_1, t_2)\}\} & \\ & \\ \min\{\min_{t_1 \le p_{2j}}\{p_{1j} + p_{2j} - t_1 + f_{j-1}(t_1, p_{3j})\}, & \text{if } t_1 = p_{1j}, t_2 = p_{3j} \\ \quad \min_{t_2 \le p_{2j}}\{p_{3j} + p_{2j} - t_2 + f_{j-1}(p_{1j}, t_2)\}\}. & \end{cases}$$

The optimum makespan is determined by solving

$$\min_{t_1, t_2} f_n(t_1, t_2),$$

and the corresponding schedule is found by backtracking.

This algorithm runs in $O(np_{11}p_{13})$ time. Since the original processing times have been multiplied by the smallest common multiple of s_1, s_2, s_3 to ensure integer processing times, the algorithm runs in $O(ns_1 s_2^2 s_3 \max\{p_j\})$ time. The space requirement is the same.

5 Ordinary $\mathcal{NP}$-hardness

In this section we show that the $F3|p_{ij} = p_j/s_i|C_{\max}$ problem with $s_1 \ge s_3 > s_2$ is $\mathcal{NP}$-hard in the ordinary sense. We show this result through a reduction from the problem PARTITION, which is known to be $\mathcal{NP}$-hard in the ordinary sense (Garey and Johnson, 1979). The problem PARTITION is defined as follows.

PARTITION
Given n positive integers $\{a_1, \dots, a_n\}$, does there exist a subset S of the index set $\{1, \dots, n\}$ such that

$$\sum_{j \in S} a_j = \sum_{j=1}^{n} a_j/2 \equiv A?$$

Since the outcome of PARTITION does not change if we multiply all a_j values by 2, we assume that all a_j and A are even. Given such an instance of PARTITION, we construct the following instance $\mathcal{I}$ of our scheduling problem. There are $n+2$ jobs, which are numbered $0, 1, \dots, n, n+1$. The processing times are $p_j = a_j$ for $j = 1, \dots, n$; $p_0 = 1$ and $p_{n+1} = 2A^2 + A$. The machine speeds are $s_1 = 1$, $s_2 = \frac{1}{2A}$, and $s_3 = \frac{1}{2A-1/2}$. We will show that the answer to this instance of PARTITION is 'yes' if and only if there exists a schedule for the instance $\mathcal{I}$ of $F3|p_{ij} = p_j/s_i|C_{\max}$ with a makespan that amounts to no more than the threshold

$$y = 8A^3 + 7A^2 + A + 1 = p_0 + (p_0 + A + p_{n+1})/s_2 + (p_{n+1} + A)/s_3.$$

We start with two preliminary results that we need to show that y is a lower bound on the makespan of an optimal schedule, which is attained only if the answer to PARTITION is 'yes'.

Lemma 13 *Let π be any V-shaped schedule for instance $\mathcal{I}$; let job $\pi(1)$ denote the first job in π. Then the first pivot in π is either job $\pi(1)$ or job $n+1$; job $n+1$ is the second pivot.*

Proof. This follows immediately from the choice of the processing times: if some job j in between the jobs $\pi(1)$ and $n+1$ would be the pivot, then according to condition 2 $p_j \geq \frac{1}{s_2} = 2A$ should hold, which is obviously not the case. Similarly, no job after job $n+1$ can be the second pivot, since the processing time of job $n+1$ on M_3 is larger than the total processing time of all other jobs on M_2. ■

Lemma 14 *Job 0 is the first job in any optimal V-shaped schedule for $\mathcal{I}$.*

Proof. Suppose to the contrary that π is an optimal V-shaped schedule in which job 0 is scheduled at the end. We move job 0 over to the front of π, which results in schedule π'; we examine the difference in makespan between π and π'. Define Q as the set of jobs in front of job $n+1$ in π; we use $p(Q)$ as a shorthand notation of the total processing time of all jobs in Q.

If job $(n+1)$ is not the first pivot in π, then it cannot be the first pivot in π', since conditions 3 in π' include the conditions 3 in π. Hence, job 0 must be the first pivot in π'. This implies that $C_{\max}(\pi) - C_{\max}(\pi') = p_{\pi(1)} + \frac{1}{s_3} - 1 - \frac{1}{s_2} = p_{\pi(1)} - 3/2 \geq 1/2$, which contradicts the optimality of π.

If on the other hand job $n+1$ is the first pivot in π, then it must satisfy all conditions 3, the last one of which states that

$$p(Q) - p_{\pi(1)} + p_{n+1} \geq p(Q)/s_2$$

Upon inspecting this inequality, we find that $p(Q) \leq A$, where we use that the first possible value of $p(Q)$ bigger than A is $A+2$. But then job $n+1$ is the first pivot in π' too, since it satisfies all conditions 3 in π' as well, as can be easily proven. Therefore, we find that moving job 0 from the last to the first position in π reduces the makespan by $\frac{1}{s_3} - 1 > 0$. ■

Now consider any optimal V-shaped schedule π for the instance $\mathcal{I}$ of $F3|p_{ij} = p_j/s_i|C_{\max}$; the previous lemma states that job 0 must be the first job in π. Let S be the set of jobs that are scheduled in between job 0 and job $n+1$ in π. Let $\bar{S}$ be the set of jobs that are scheduled after job $n+1$; we have that the union of S and $\bar{S}$ contains all jobs $1, 2, \ldots, n$. We compare $p(S)$ to A and find throught a straightforward computation that:

- If $p(S) > A$, then job 0 is the unique first pivot in π;

  $$C_{\max}(\pi) = 1 + \frac{p(S) + p_{n+1}}{s_2} + \frac{p_{n+1} + p(\bar{S})}{s_3} > y,$$

  since $s_2 < s_3$ and $0 < p(S) - A = A - p(\bar{S})$.

- If $p(S) < A$, then job $n+1$ is the unique first pivot in π;

  $$C_{\max}(\pi) = 1 + p(S) + p_{n+1} + \frac{p_n + 1}{s_2} + \frac{p_{n+1} + p(\bar{S})}{s_3} > y,$$

  since $\frac{1}{s_3} > 1$ and $0 < A - p(S) = p(\bar{S}) - A$.

- If $p(S) = A$, then both job 0 and job $n+1$ can serve as the first pivot;

$$C_{\max}(\pi) = 1 + \frac{1 + p(S) + p_{n+1}}{s_2} + \frac{p_{n+1} + p(\bar{S})}{s_3} = y.$$

The proof of the following theorem has become trivial by now.

Theorem 15 *The problem $F3|p_{ij} = p_j/s_i|C_{\max}$ is $\mathcal{NP}$-hard in the ordinary sense.* ■

6 Conclusions

We have shown that, although the general $F3||C_{\max}$ problem is $\mathcal{NP}$-hard in the strong sense (Garey, Johnson, and Sethi (1976)), the special case with processing times $p_{ij} = p_j/s_i$ is solvable in pseudo-polynomial time if s_2 is smaller than s_1 and s_3. The situation becomes more complicated in case of more than 3 machines, since it is not even clear whether the optimal schedule is a permutation schedule. There is one exception, however: if the machines speeds are such that $s_1 \geq s_2 \geq \ldots \geq s_m$ or $s_1 \geq s_2 \leq \ldots \leq s_m$, then we have shown that the problem is solved by putting the jobs in SPT-order or LPT-order, respectively (Hou and Hoogeveen, 2000).

Acknowledgements

Part of this research was conducted when the first author visited Eindhoven University of Technology, which visit was made possible by a grant from Nuffic. The second author was partially supported by EC Contract IST-1999-14186 (Project ALCOM-FT).

References

[1] R.W. Conway, W.L. Maxwell, and L.W. Miller (1967). *Theory of scheduling*, Addison-Wesley, Reading, MA.

[2] M.R. Garey, D.S. Johnson, and R. Sethi (1976). The complexity of flowshop and jobshop scheduling. *Mathematics of Operations Research 13*, 330-348.

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

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

[5] S. Hou and J.A. Hoogeveen (2000). *Polynomially solvable cases of the proportionate flow shop problem with unequal machine speeds.* Unpublished manuscript.

[6] P.S. Ow (1985). Focused Scheduling in Proportionate Flow Shops. *Management Science 31*, 852-869.

[7] M.L. Pinedo (1985). A note on stochastic shop models in which jobs have the same processing requirements on each machine. *Management Science 31*, 840-845.

[8] M.L. Pinedo (1995) *Scheduling: Theory, Algorithms and Systems,* Prentice-Hall, Englewood Cliffs, New Jersey.

[9] N. Shakhlevich, J.A. Hoogeveen, and M.L. Pinedo (1998). Minimizing total weighted completion time in a proportionate flow shop. *Journal of Scheduling 1*, 157-168.