MINIMIZING MAKESPAN IN FLOWSHOPS WITH PALLET REQUIREMENTS: COMPUTATIONAL COMPLEXITY[1]

MICHAEL WANG, SURESH SETHI

Faculty of Management, University of Toronto, Canada
mwang@brno.com and sethi@fmgmt.mgmt.utoronto.ca

CHELLIAH SRISKANDARAJAH

Department of Mechanical and Industrial Engineering, University of Toronto, Canada
chellia@mie.utoronto.ca

STEEF VAN DE VELDE

Rotterdam School of Management, Erasmus University, The Netherlands
s.velde@fac.fbk.eur.nl

ABSTRACT

We establish the computational complexity of the problem of minimizing makespan in a flowshop, where each jobs requires a *pallet* the entire time, from the start of its first operation until the completion of the last operation. We prove that the problem is NP-hard in the strong sense for $m \geq 2$ and $K \geq 3$, and for $m \geq 3$ and $K \geq 2$, where m is the number of machines and K is the number of pallets in the system.

Keywords: flowshop scheduling; makespan; pallets; buffers; computational complexity.

RÉSUMÉ

Nous réglons ici le statut de la complexité algorithmique du problème de la durée minimum (makespan) d'exécution des tâches dans un atelier de type ligne de transfert (flow-shop) lorsque chaque pièce mobilise une *palette* pendant toute la durée de son traitement dans l'atelier. Nous montrons que ce problème est NP-difficile au sens fort pour m machines et K palettes, pour $m \geq 2$ et $K \geq 3$, ainsi que pour $m \geq 3$ et $K \geq 2$.

1. INTRODUCTION

Pallets are essential components in automated or flexible manufacturing environments, where they serve as interfaces between machines and work pieces. Work pieces are fixed on the pallets when they enter the system, and the pallets are transported by some type of material handling system to machine centers where the work pieces remain mounted on to the pallets while being processed. Usually, pallets wait in a storage area or pallet pool located near the machine, and are loaded into the machine by means of an automatic pallet changer (APC). There are two basic arrangements in which pallets are stored and loaded. In the first case, pallets from the pool are loaded into the machine in an arbitrary order; this is often done by a rotary indexing table. In the other case, pallets wait in a linear row and are fed into the machine in the same order as they are waiting (Viswanadham and Narahari, 1992). Figure 1 illustrates a system containing two machines and K pallets.

Note that if we see the pallet handling, the mounting and dismounting of work pieces on to and from pallets, and the actual processing as a chain of operations that each job needs to undergo, then a flexible manufacturing system with pallet requirements can be modeled as a classical flowshop, or a variant of it, with pallet requirements.

For a manufacturing system with pallet requirements, there are two main planning problems involved: the *design* problem and the *scheduling* problem. The design problem concerns the number of pallets needed to configure the system. Pallets are expensive equipment, and a trade-off has to be made between the number of pallets and the performance of the system. Especially

[1]Recd. June 1996; Final version June 1997

interesting and worth investigating is the marginal gain of an additional pallet in terms of a specified criterion. The scheduling problem concerns finding the best schedule for processing a given set of jobs, given the system's configuration and the performance criterion.

Pallets are scarce renewable resources, and there is a lot of literature on machine scheduling problems with additional scarce renewable discrete resources. The few available polynomial algorithms and complexity results mainly concern single-operation models, like single and parallel machine problems. As could be expected, even simple models are already intractable; see for instance Blazewicz *et al.* (1983, 1993, Chapter 7). The bulk of literature in this area mainly concerns parallel machine and jobshop problems with tools and operators; such resources are needed only to perform the operations. In contrast, we study a type of problem where each job needs an additional resource the *entire* time, from the start of its first operation till the completion of the last operation.

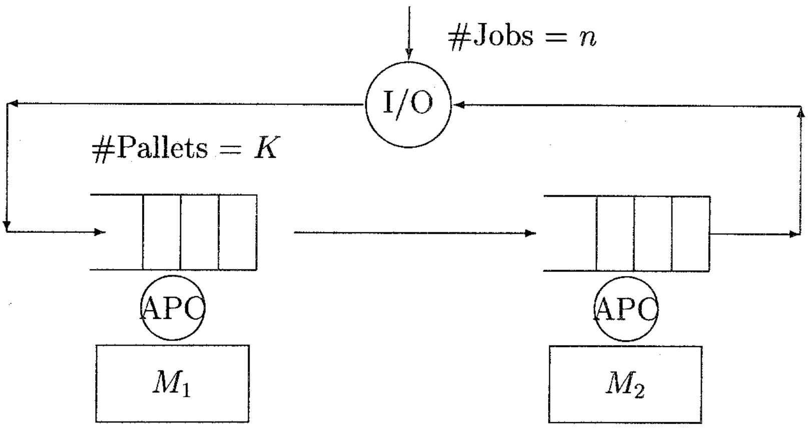


Figure 1: Illustration of a manufacturing cell with the pallet requirement.

More specifically, we study the problem of minimizing makespan in a classical flowshop with pallet requirements. In this paper, we are concerned with the computational complexity of the makespan problem for various values of m and K. This paper supplements our companion paper on the flowshop problem with pallet requirements (Wang, Sethi, Sriskandarajah, and Van de Velde, 1995) that focuses mainly on the impact of the number of pallets on makespan.

The plan of this paper is as follows. In Section 2, we formulate the problem and point out the relationship between the flowshop problem with pallet requirements and the flowshop problem with buffer restrictions and discuss some well-solvable cases. In Section 3, we prove that the makespan problem is NP-hard in the strong sense for $m \geq 2$ and $K \geq 3$, and for $m \geq 3$ and $K \geq 2$, where m is the number of machines and K is the number of pallets in the system. Section 4 concludes the paper.

2. PROBLEM DESCRIPTION

We consider the following problem. There are m machines $M_1, \ldots, M_m$ available from time zero onwards for processing a set $\mathcal{J}$ of n independent jobs $J_1, \ldots, J_n$. Each job J_j consists of a chain of m operations $O_{j1}, \ldots, O_{jm}$, which imply that operation O_{jk} can start only if its predecessor operation $O_{j,k-1}$ has been completed ($j = 1, \ldots, n$, $k = 2, \ldots, m$). Operation O_{jk} requires uninterrupted processing on machine M_k during a nonnegative time p_{jk}. Each machine is available from time zero onwards only and can process no more than one operation at a time. A job can be scheduled on the first machine only if a pallet is available; and once the job is started, the pallet will stay with the job until its last operation is completed. The pallet will then become available for the next job. Depending on the way the pallets are stored and transported, pallets may or may not pass each other in the system. In the case of no-passing, the pallet has to go through each machine even if the job it carries has zero processing time on some machine. Moreover, the number of pallets in the shop at any given time is limited by a positive integer K.

A *schedule* specifies a set of completion times C_{jk}, $\forall j,k$, such that the above conditions, including that no more than K pallets are used at any point in time, are met. The length of a schedule, denoted by $C_{\max}$, referred to as the *makespan*, is determined as $C_{\max} = \max_{1\leq j\leq n} C_{jm}$.

In the two-machine case, we let a_j and b_j denote the processing times of J_j on machines M_1 and M_2, respectively, use $\tau_j = \langle a_j, b_j\rangle$, and refer to the operations on M_1 and M_2 as *a-operations* and *b-operations*, respectively. Following the nomenclature for scheduling problems introduced in Graham *et al.* (1979), we refer to this problem as $Fm|K\text{-pallets}|C_{\max}$.

Obviously, the problem is solvable by the Johnson algorithm if $K \geq n$ (Johnson, 1954), since the pallet requirements are then not restrictive. In our companion paper (Wang *et al.*), we have established the strong connection between the K-pallet problem and the **p**-buffer problem, where $\mathbf{p} = \langle p_1, \ldots, p_{m-1}\rangle$. The **p**-buffer problem, designated as $Fm|\mathbf{p}\text{-buffer}|C_{\max}$ for $m \geq 2$, is a flow-shop problem with buffers of total size $p = \sum_{k=1}^{m-1} p_k$ with p_k being the size of the buffer between M_k and M_{k+1}. The problem $F2|K\text{-pallets}|C_{\max}$ is equivalent to $F2|(K-2)\text{-buffer}|C_{\max}$, $K \geq 2$, and a feasible solution to $Fm|\mathbf{p}\text{-buffer}|C_{\max}$ is also a feasible solution to $Fm|K\text{-pallets}|C_{\max}$ with $K = p+m$, but not vice versa. Accordingly, since the no-wait flowshop problem is equivalent to the two-machine no-buffer problem, we can solve the $F2|2\text{-pallets}|C_{\max}$ problem in $O(n \log n)$ time by a variant of the Gilmore-Gomory algorithm (Gilmore and Gomory, 1964).

3. THE COMPLEXITY

Here we show that the $Fm|K\text{-pallets}|C_{\max}$ problem is NP-hard. We shall first prove the NP-hardness for two machines and $K \geq 3$ pallets in Section 3.1 and for three machines and $K \geq 2$ pallets in Section 3.2 with passing allowed. Then we discuss the case when passing is not allowed in Section 3.3.

3.1 $F2|K$-pallets$|C_{\max}$

Papadimitriou and Kanellakis (1980) prove that the two-machine **p**-buffer problem is NP-hard in the strong sense for $p \geq 1$, which implies that the $F2|K\text{-pallets}|C_{\max}$ problem is NP-hard in the strong sense for $K \geq 3$. We present a direct proof here.

Consider the following problem which is known to be NP-complete in the strong sense (Garey and Johnson, 1979, p. 224).

Numerical Matching With Target Sums (NMWTS)

Let Y and Z be disjoint sets each containing t elements and $s(a)$ denotes a positive integer for $a \in Y \cup Z$. Let $X = \langle x_1, \ldots, x_t\rangle$ be a vector of positive integers.

Can $Y \cup Z$ be partitioned into t disjoint subsets $\langle A_1, \ldots, A_t\rangle$ each containing exactly one element from Y and one from Z such that $x_j = \sum_{a\in A_j} s(a)$, $j = 1, \ldots, t$?

In the following theorem, $F2|K\text{-pallets}|C_{\max}$ is shown to be NP-hard by a direct reduction from NMWTS.

Theorem 1 *The problem $F2|K$-pallets$|C_{\max}$ with $K \geq 3$ is* NP-*hard in the strong sense.*

Proof

We first establish the result for $K = 3$.

Consider any instance I of NMWTS. For ease of exposition, we use two vectors $\langle y_1, \ldots, y_t\rangle$ and $\langle z_1, \ldots, z_t\rangle$ to represent the sizes of elements in Y and Z, respectively, and without loss of complexity we may assume that $\sum_{j=1}^{t} x_j = \sum_{j=1}^{t}(y_j + z_j)$. Construct the following instance of $F2|3\text{-pallets}|C_{\max}$ with $n = 3t + 2$ jobs contained in the four job sets defined below:

- A set $\mathcal{P} = \{J_1, \ldots, J_t\}$ of t "pacesetting" jobs with $\tau_j = \langle 3L + x_j, 2\rangle$, for $j = 1, \ldots, t$; recall that τ_j is a vector of processing times for J_j.

- A set $\mathcal{M}_1 = \{J_{t+1}, \ldots, J_{2t}\}$ of t "matching" jobs with $\tau_j = \langle 1, 2L + y_{j-t}\rangle$, for $j = t+1, \ldots, 2t$.
- Another set $\mathcal{M}_2 = \{J_{2t+1}, \ldots, J_{3t}\}$ of t "matching" jobs with $\tau_j = \langle 1, L + z_{j-2t}\rangle$ for $j = 2t+1, \ldots, 3t$.
- A set consisting of two jobs, namely, a "heading" job J_{3t+1} with $\tau_{3t+1} = \langle 0, 2\rangle$, and a "tailing" job J_{3t+2} with $\tau_{3t+2} = \langle 2, 0\rangle$.

Question. Is there a schedule with $C_{\max} \leq C \equiv 2(t+1) + 3tL + \sum_{j=1}^{t} x_j$?

We prove that the answer to this question is 'yes' if and only if I is a 'yes' instance.

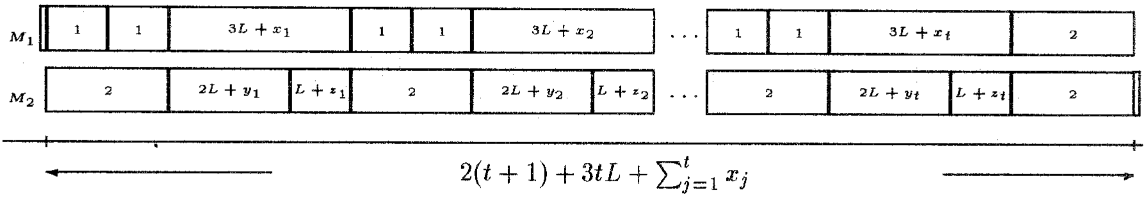


Figure 2: The Gantt Chart illustrating the NP-hardness proof for $F2|3\text{-pallets}|C_{\max}$.

First assume that there is a matching in I. Let us then relabel the elements of Y and Z vectors so that $x_j = y_j + z_j$, $j = 1, 2, \ldots, t$. It is now possible to obtain a schedule as shown in Figure 2, which has a makespan of C. Note that the first operation of the heading job and the second operation of the tailing job are shown as 'small' operations (actually of zero lengths) for convenience.

It remains to show the converse. We shall prove that if I is a 'no' instance, then every feasible schedule has a length strictly greater than C. In view of the assumption that $\sum_{j=1}^{t} x_j = \sum_{j=1}^{t}(y_j + z_j)$ and of the definition of the four job sets, C is exactly the sum of the operations on M_1 as well as those on M_2. So it is sufficient to show that there must be idle time on M_1 or M_2. Consider any feasible schedule σ. We observe that $K = 3$ implies that the shop admits at most three jobs at a time, or that during the time a job is processed on one machine, the other machine can process at most two jobs. In fact, we can establish that, during the time a job $J_j \in \mathcal{P}$ is processed on machine M_1, machine M_2 must be processing a job $J_j \in \mathcal{M}_1$ and a job $J_j \in \mathcal{M}_2$ one after another; otherwise, idle time will be incurred on M_2. This is true because $L \gg \max\{2, \sum_{j=1}^{t} x_j\}$, and there are exactly t jobs in $\mathcal{M}_1$ and t jobs in $\mathcal{M}_2$. Thus, without loss of generality, we can assume that $\{O_{j1}, O_{t+j,2}, O_{2t+j,2}\}, j = 1, 2, \ldots, t$, are processed together. Now if the answer to the NMWTS question is 'no', then there must be a j such that $x_j > y_{2,t+j} + z_{2,2t+j}$ because $\sum_{j=1}^{t} x_j = \sum_{j=1}^{t}(y_j + z_j)$, which implies an idle time on M_2. Since σ is arbitrary, the result is established.

It is then a straightforward generalization to prove the theorem for $K > 3$. The reduction proceeds from a K-dimensional NMWTS in which we are given sets $Y_1, \ldots, Y_{K-1}$, each containing t elements with size $s(a), a \in \bigcup Y_k$ and a vector of positive integers $X = \langle x_1, \ldots, x_t\rangle$. The question is whether $\bigcup Y_k$ can be partitioned into t disjoint subsets $A_1, \ldots, A_t$, each containing exactly one element from $Y_k, k = 1, 2, \ldots, K-1$ such that $x_j = \sum_{a \in A_j} s(a)$. An instance of this problem reduces to the following instance of $F2|K\text{-pallets}|C_{\max}$ with $n = Kt + 1$ jobs:

- A set $\mathcal{P} = \{J_0, J_1, \ldots, J_t\}$ with $\tau_0 = \langle 0, K-1\rangle$, $\tau_j = \langle KL + x_j, K-1\rangle$ for $j = 1, \ldots, t-1$, and $\tau_t = \langle KL + x_t, 0\rangle$, where $L \gg \max\{K, \sum_{j=1}^{t} x_j\}$;
- A set $\mathcal{M} = \{J_{t+1}, \ldots, J_{2t}, J_{2t+1}, \ldots, J_{Kt}\}$ with $\tau_j = \langle 1, kL + y_{j-kt}\rangle$, for $j = kt+1, \ldots, (j+1)t$, $k = 1, \ldots, K-1$, where y_{j-kt} is the size of the $(j-kt)$-th element of Y_k.

C is taken to be $(K-1)t + KtL + \sum_{j=1}^{t} x_j$. This completes the proof. •

3.2 F3|K-pallets|$C_{\max}$

We prove in this section that the pallet problem with three machines is NP-hard. We first consider the general case, which can be shown to be NP-hard in the strong sense, and then we impose the no-passing restriction and show that the problem is NP-hard. The proof for the former is by a reduction from 3-PARTITION (Garey and Johnson, 1979).

3-Partition

Given an integer B and a multiset $\mathcal{A} = \{a_1, \ldots, a_{3t}\}$ of $3t$ positive integers with $B/4 < a_i < B/2$, $i = 1, \ldots, 3t$, and with $\sum_{i=1}^{3t} a_i = tB$, is there a partition of $\mathcal{A}$ into t mutually disjoint subsets $\mathcal{A}_1, \ldots, \mathcal{A}_t$ such that for each $k = 1, \ldots, t$,

$$|\mathcal{A}_k| = 3 \text{ and } \sum_{a \in \mathcal{A}_k} a = B?$$

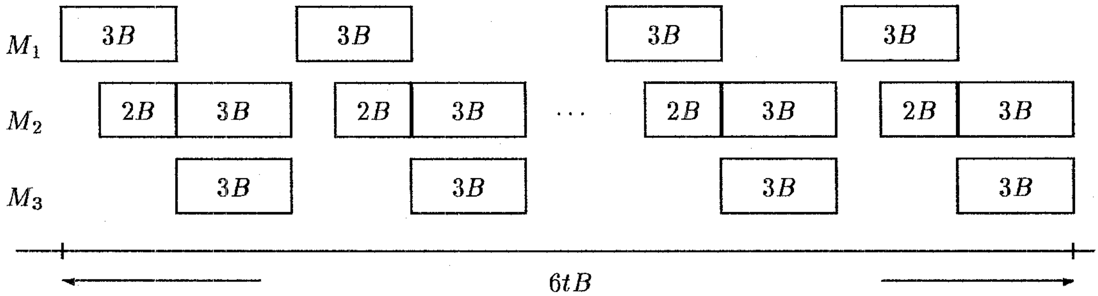


Figure 3: The Gantt Chart illustrating the NP-hardness proof for $F3|2$-pallets$|C_{\max}$.

Theorem 2 *The problem $F3|K$-pallets$|C_{\max}$ is NP-hard in the strong sense.*

Proof.

Consider any instance I of 3-PARTITION. We construct first an instance of the flowshop problem with $K = 2$ pallets and $n = 5t$ jobs contained in the three job sets as follows:

- A set $\mathcal{S} = \{J_1, \ldots, J_t\}$ of t "pacesetting" jobs with $\tau_j = \langle 3B, 3B, 0 \rangle$, for $j = 1, \ldots, t$;
- A set $\mathcal{H} = \{J_{t+1}, \ldots, J_{2t}\}$ of t "pallet-holding" jobs with $\tau_j = \langle 0, 2B, 3B \rangle$, for $j = t+1, \ldots, 2t$;
- A set $\mathcal{P} = \{J_{2t+1}, \ldots, J_{5t}\}$ of $3t$ "partition" jobs with $\tau_j = \langle 0, a_{j-2t}, 0 \rangle$, for $j = 2t+1, \ldots, 5t$.

Question. Is there a schedule with $C_{\max} \leq C \equiv 6tB$?

We prove that the answer to this question is 'yes' if and only if I is a 'yes' instance. Clearly, if the answer to I is 'yes', the partition jobs can be placed in the empty time slots on M_2 as shown in Figure 3, and we obtain a schedule with $C_{\max} = C$.

Now let σ be any schedule with length $C_{\max} = C$. Such a schedule requires that there is no idle time on M_2, since the total processing time on M_2 is exactly C. First we observe that the pacesetting and the pallet-holding jobs have to be 'glued' together as depicted in Figure 3, that is, the pacesetting and the pallet-holding jobs have to be processed in the same time interval on M_2 and on M_3, and the pallet-holding job must start on M_2 exactly B time units after the pace setting job starts on M_1. Otherwise, there would be a time gap on M_2 during which no job could be processed, since no pallet would be available.

Furthermore, we see that the next job can only be scheduled when the two glued jobs both finish. As a result, the schedule is divided into blocks of width $6B$. In each block, M_2 is left with a time slot of exactly B time units, and the partition jobs can only be processed in these slots. Due to the constraint $B/4 < a_i < B/2$, each slot can hold at most three partition jobs. Thus σ defines a 3-partition for I.

For $K > 2$, I simply reduces to the following instance of $F3|K\text{-pallets}|C_{\max}$ with $n = 4t + 1$ jobs:

- A set $\mathcal{P} = \{J_1, \ldots, J_{3t}\}$ of $3t$ "partition" jobs with $\tau_j = \langle 0, a_j, 0\rangle$, for $j = 1, \ldots, 3t$;
- A set $\mathcal{S} = \{J_{3t+1}, \ldots, J_{4t+1}\}$ of $t + 1$ "pacesetting" jobs with $\tau_j = \langle 3B, 2B, 3B\rangle$, for $j = 3t + 1, \ldots, 4t - 1$, $\tau_{4t} = \langle 0, 2B, 3B\rangle$, and $\tau_{4t+1} = \langle 3B, 2B, 0\rangle$.

Set $C \equiv 3tB + 2B$ to complete the proof. •

Remark

The makespan problems of three-machine flowshops and of three-machine, no-wait flowshops (without the pallet requirement) are both known to be NP-hard in the strong sense. However, the result of Theorem 2 cannot be inferred immediately. This is because when there are only two pallets, one of the three machines has to be idle at any given time, since a job can be scheduled only when a pallet is available. Therefore, the problem can be also viewed as one of processing jobs on two parallel pallets in three stages, with each stage requiring a unique machine. This perspective may prompt one to conjecture that the complexity of the problem is closer to that of the two-machine parallel jobshop problem, which is known to be NP-hard in the ordinary sense. While Theorem 2 confirms that the flowshop problem with the pallet requirement is as hard as that without the pallet requirement, our next theorem, proved by a reduction from 2-PARTITION, seems to suggest an NP-hardness in the ordinary sense when pallet passing is not allowed.

3.3 F3|2-pallets, no-passing|$C_{\max}$

In proving Theorems 1 and 2, we have implicitly assumed that when a pallet is idle, it is immediately available. This is not the case, however, in situations where pallets cannot pass each other during the time of a schedule. For example, pallets may be moved on a rail or a conveyor. Thus a pallet may not be available, even if the job placed on it has completed processing, because the pallet is not allowed to pass other pallets. Even if pallets can move around freely, *no-passing* schedules are still desirable because they are easier to control on the shop floor. An optimal schedule without the no-passing constraint may make it difficult to keep track of jobs. Therefore, no-passing is frequently imposed on schedules. The instance used in the proof for Theorem 1 is a no-passing schedule, so Theorem 1 still holds. But Theorem 2 will not hold because the partition jobs can no longer be scheduled into the empty time slots on M_2.

In the following theorem, we give a reduction from 2-PARTITION to prove the NP-hardness of the problem with $m \geq 3, K = 2$ and no-passing. In proving the following theorem, we look at the problem from the pallet's viewpoint. Figure 4 shows an example of a schedule looking both (a) from the pallet's viewpoint, and (b) from the machine's viewpoint. Clearly, the two perspectives are equivalent.

Theorem 3 *The problem of minimizing the makespan with $m \geq 3, K = 2$ and no-passing is* NP-*hard.*

Proof. The reduction proceeds from 2-PARTITION, which is NP-complete in the ordinary sense (Garey and Johnson, 1979).

2-PARTITION

Given a multiset $\mathcal{A} = \{\alpha_1, \ldots, \alpha_{2t}\}$ of $2t$ integers, does $\mathcal{A}$ include a subset $\mathcal{A}_1$ such that $|\mathcal{A}_1| = t$ and

$$\sum_{\alpha_j \in \mathcal{A}_1} \alpha_j = B = \sum_{j=1}^{2t} \alpha_j / 2?$$

For any instance I of 2-PARTITION, construct an instance of the decision variant of $F3|2\text{-pallets}|C_{\max}$ with $n = 2t + 2$ jobs as follows:

- a set of $2t$ "partition" jobs $\{J_j, j = 1, 2, \ldots, 2t\}$ with $\tau_j = \langle 2B, 2B, 2B + \alpha_j \rangle$;
- a "heading" job J_{t+1} with $\tau_{t+1} = \langle \epsilon, 2B, B \rangle$, where $0 < \epsilon \ll B$; and
- a "tailing" job J_{t+2} with $\tau_{t+2} = \langle 2B, B, \epsilon \rangle$.

Question. Is there a schedule with $C_{\max} \leq C \equiv (6t + 4)B + 2\epsilon$?

If I is a 'yes' instance, then without any loss of generality we may assume that $\mathcal{A}_1 = \{1, 3, \ldots, 2t-1\}$, and load the jobs onto the two pallets contiguously as depicted in Figure 4. Since job J_1 lags behind J_2 exactly $3B$ time units, leads J_4 more than $3B$ time units, and $\sum_{j \in \mathcal{A}_1} \alpha_j = B$, no two jobs will share the same machine, which also implies that no job will 'catch up' and pass another. That is, the schedule is feasible and $C_{\max} = C$.

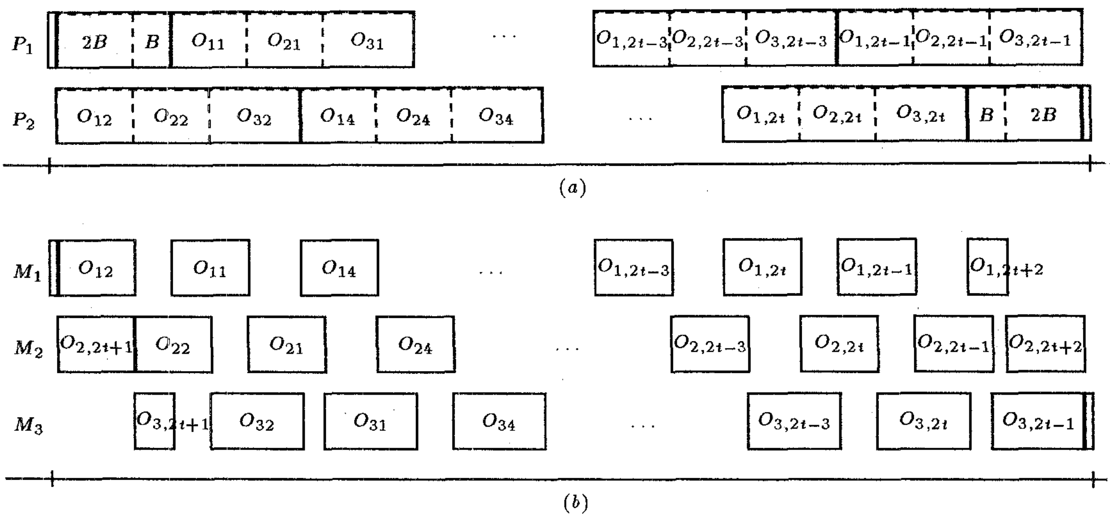


Figure 4: The Gantt Chart illustrating the proof of NP-hardness of $K = 3, m = 2$ with no-passing (a) from the pallet's viewpoint and (b) from the machine's viewpoint.

Conversely, if there is a schedule with $C_{\max} = C$, we argue that the schedule must have the same structure as that in Figure 4.

1. The heading job must be loaded first. Otherwise, if any other job is loaded first, one pallet will be idle for at least $2B$ time units, which is not allowed since the total processing time required is just $2C - 2\epsilon$, and any idle time greater than 2ϵ would extend the makespan beyond C.
2. Similarly, the tailing job must be loaded last.
3. There are exactly $t + 1$ jobs loaded on each pallet, and the heading and the tailing jobs are loaded on different pallets. To see the first point, note that the minimum makespan of any t jobs is no greater than $6tB + 2B < C$ and the minimum makespan of any $t + 2$ jobs is no less than $6tB + 6B > C$.

	$m=1$	$m=2$	$m=3$	$m\geq 4$
$K=1$	trivial	trivial	trivial	trivial
$K=2$	trivial	$O(n\log n)$	NPHS	NPHS
(no-passing)	trivial	$O(n\log n)$	NPH	NPH
$K=3$	trivial	NPHS	NPHS	NPHS
$K\geq 4$	trivial	NPHS	NPHS	NPHS

Note. NPHS: NP-hard in the strong sense; NPH: NP-hard.

Table 1: Complexity of $Fm|K$-pallets$|C_{\max}$.

Thus the partition jobs are scheduled as in Figure 4, except that they may be permuted. Then the two pallets represent the two partitions desired.

For the general $Fm|2$-pallets$|C_{\max}$, we simply construct an instance with $n = 2t+2$ jobs $\{J_1,\ldots,J_{2t+2}\}$ and the following processing times:

$$\tau_j = \langle 2B,\ldots,2B,2B+\alpha_j\rangle,\ j=1,\ldots,2t,$$

$$\tau_{2t+1} = \langle \epsilon,\ 2B,\ldots,2B,\ B\rangle,$$

and

$$\tau_{2t+2} = \langle B,\ 2B,\ldots,2B,\ \epsilon\rangle.$$

C is taken to be $(2mt+2m-2)B+2\epsilon$ and the theorem follows. •

4. CONCLUSIONS

We have established the complexity for all classes of the problem as shown in Table 1. The problem is trivial for one machine or one pallet. It is polynomial when $K=2, m=2$, and it is NP-hard in the strong sense when $K\geq 3, m\geq 2$, or when $K\geq 2, m\geq 3$. When the constraint of no-passing is imposed, the problem for $m\geq 3, K=2$ is NP-hard. The question of whether it is NP-hard in the strong sense or in the ordinary sense remains open.

ACKNOWLEDGEMENT

This research is supported in part by NSERC A4619, OGP0104900 and Manufacturing Research Corporation of Ontario

BIBLIOGRAPHY

[1] J. Blazewicz, J.K. Lenstra, and A.H.G. Rinnooy Kan. 1983. Scheduling subject to resource constraints: classification and complexity. *Discrete Applied Mathematics* **5**, 11-24.

[2] J. Blazewicz, K.H. Ecker, G. Schmidt, and J. Weglarz (1993). *Scheduling in Computer and Manufacturing Systems*, Springer-Verlag, Berlin.

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

[4] Gilmore, P.C. and R.E. Gomory. 1964. Sequencing a one-state-variable-machine: a solvable case of the traveling salesman problem. *Operations Research* **12**, 655-679.

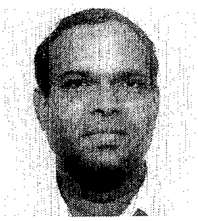
[5] Graham, R.L., E.L. Lawler, J.K.Lenstra, and A.H.G. Rinnooy Kan. 1979. Optimization and approximation in deterministic sequencing and scheduling: a survey, *Annals of Discrete Mathematics* **5**, 287-326.

[6] Johnson, S. M. 1954. Optimal two and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly* **1**, 61-68.

[7] Papadimitriou, C. H. and P.C. Kanellakis. 1980. Flowshop scheduling with limited temporary storage. *Journal of the Association of Computing Machinery* **27**, No. 3, 533-549.
[8] Viswanadham, N. and Y. Narahari. 1992. *Performance Modeling of Automated Manufacturing Systems,* Prentice-Hall, New Jersey.
[9] M.Y. Wang, S.P. Sethi, C. Sriskandarajah, and S.L. van de Velde (1995). *Minimizing makespan in flowshops with pallet requirements,* Working Paper, University of Toronto.

Yingmeng (Michael) Wang, Senior Analyst and Technical Advisor, Solutions and Applications, Treasury, Bank of Montreal, earned his PhD at the University of Toronto under the supervision of Dr. Suresh Sethi. He has published a number of papers in the area of scheduling, clustering analysis, and vehicle routing. He is now in charge of the system implementation of financial derivative products for Treasury, Bank of Montreal

Suresh Sethi is Professor of Operations Management and Director, Laboratory for Manufacturing Research at University of Toronto, and Ashbell Smith Professor of Operations Management at University of Texas at Dallas. He has published three books: *Optimal Control Theory:Applications to Management Science; Hierarchical Decision Making in Stochastic Manufacturing Systems*; and *Optimal Consumption and Investment Problems with Bankruptcy,* and over 200 papers on operations management, economics, finance, marketing, operations research and optimal control. Honors include: Connaught Senior Research Fellow; CORS Award of Merit; Honorary Professor, Zhejiang University of Technology; and Fellow, Royal Society of Canada. Membership: CORS, INFORMS, POMS and IEEE.



Chelliah Sriskandarajah is an Associate professor at the Mechanical and Industrial Engineering Department of the University of Toronto. He has a D.Sc degree from the Higher National School of Electrical Engineering of the National Polytechnic Institute of Grenoble (France) in the field of Production Automation and Operations Research. He received an M.Sc. degree in Industrial Engineering and Management from the Asian Institute of Technology (Bangkok) and a B.Sc. degree in Mechanical Engineering from the University of SriLanka. His research interests lie in the general area of production planning and scheduling, inventory control, machine scheduling theory, maintenance scheduling, supply chain management, group technology and performance evaluation of production systems. His articles on these topics have appeared in *Operations Research, Naval Research Logistics Quarterly, International Journal of Production Research, European Journal of Operational Research, Annals of Operations Research, International Journal of Flexible Manufacturing Systems, Journal of Operational Research Society,* and *INFOR.* He is a member of IIE, INFORMS and CORS.

Steef van de Velde is an Associate Professor of Quantitative Methods at the Rotterdam School of Management of Erasmus University. His research interests include machine scheduling theory, combinatorial optimization, production planning and control, production strategy and supply chain management. He is an editor for the Journal of Scheduling and an associate editor for Operations Research and for IIE Transactions, the focused issue on Scheduling and Logistics. He has published in such journals as Operations Research, Management Science, INFORMS Journal on Computing, and Mathematical Programming.