Operations Research Letters 44 (2016) 153–157

Contents lists available at ScienceDirect

Operations Research Letters

journal homepage: www.elsevier.com/locate/orl

Permutation schedules for a two-machine flow shop with storage

Joey Fung *, Yakov Zinder

School of Mathematical and Physical Sciences, University of Technology Sydney, PO Box 123, Broadway NSW 2007, Australia

ARTICLE INFO

Article history:
Received 28 February 2015
Received in revised form
23 November 2015
Accepted 16 December 2015
Available online 23 December 2015

Keywords:
Flow shop
Buffer
Permutation schedule
Makespan

ABSTRACT

This paper considers a two-machine flow shop problem with a buffer, arising in various applications, and addresses a fundamental question of the existence of an optimal permutation schedule. The paper proves that the problem of recognising whether an instance has an optimal permutation schedule is NP-complete in the strong sense, and estimates the deviation from the optimal makespan as a result of the restriction to permutation schedules only.

1. Introduction

This paper is motivated by the question, posed in [3,4], on whether or not any instance of the two-machine flow shop, considered in these publications, has an optimal schedule with the same order of jobs on both machines. For any flow shop model, a schedule with the same order of jobs on all machines is referred to as a permutation schedule. The answer to the question of whether there exists an optimal permutation schedule is fundamental in the analysis of any flow shop model and affects the design of scheduling algorithms (see for example [1,5,7]). The results in this paper are in the same spirit as those presented in [8] for a different flow shop model.

This paper proves that, for the considered flow shop model, the answer to the abovementioned question in general is negative. It is proven that the deviation from the optimum makespan, as a result of the restriction to only permutation schedules, can be made arbitrarily large by increasing the number of jobs, even if the maximum duration of operations and the buffer consumption by each job are bounded above by constants. These results are complemented by the proof that, unless $P = NP$, there is no polynomial-time algorithm which can detect whether or not a given instance has an optimal permutation schedule.

The considered flow shop model includes a specific type of buffer and was originally introduced in [6] for the purpose of optimising the object sequence in a prefetch-enabled TV-like presentation. Besides this application, the considered scheduling model is also relevant to various situations involving loading and unloading operations.

The problem considered in [6,3,4] can be stated as follows. The set of jobs $N = \{1, \dots, n\}$ is to be processed on two machines, M_1 and M_2. Each job j is processed on M_1 for $p_{1j} > 0$ units of time (first operation) and then on M_2 for $p_{2j} > 0$ units of time (second operation). Each job cannot start processing on M_2 before completing its first operation on M_1.

Besides the two machines, the processing of jobs requires a buffer. When job j begins processing on M_1, it consumes s_j units of buffer space. This space is released when the job completes processing on M_2. The capacity of the buffer is denoted by Ω. That is, the total size of all jobs which are either currently being processed on one of the two machines, or which have completed the first operation and are waiting in the buffer for the commencement of the second operation, cannot exceed Ω. The size s_j of each job j is proportional to p_{1j}, and therefore, without loss of generality, it is assumed that $s_j = p_{1j}$ (see [3]).

All jobs are available for processing from time $t = 0$. A schedule σ is defined by the starting times of all operations. The starting time of operation i of job j is denoted $S_j^i(\sigma)$. The completion of operation i of job j is $C_j^i(\sigma) = S_j^i(\sigma) + p_{ij}$. So, equivalently the schedule can be defined by the completion times of all operations. It is necessary to find a schedule which minimises the makespan, i.e. the objective function

$$C_{\max}(\sigma) = \max_{j \in N} C_j^2(\sigma). \tag{1}$$

The rest of this paper is structured as follows. Section 2 proves that, for $n \leq 4$, there exists an optimal permutation schedule.

* Corresponding author.
E-mail addresses: Joey.Fung@uts.edu.au (J. Fung), Yakov.Zinder@uts.edu.au (Y. Zinder).

http://dx.doi.org/10.1016/j.orl.2015.12.012
0167-6377/

Section 3 establishes bounds on the maximum deviation from the optimum under the restriction that only permutation schedules are considered. Section 4 proves that, unless $P = NP$, there is no polynomial-time algorithm which can detect whether or not a given instance has an optimal permutation schedule.

2. Case $n \le 4$: permutation schedules

The following lemma holds for all n.

Lemma 1. *For any instance, there exists an optimal schedule where the job processed first on M_1 is also the first on M_2, and the job processed last on M_1 is also the last on M_2.*

Proof. Suppose that in an optimal schedule σ^*, either the job h, which is processed first on M_2, is not the first on M_1, or that the job k, which is processed last on M_1, is not the last on M_2, or both. Since the processing on M_2 starts in σ^* only after the point in time $C_h^1(\sigma^*)$, all jobs which are processed on M_1 between points in time $t = 0$ and $t = C_h^1(\sigma^*)$ can be processed on M_1 in this time interval in any order without violating the feasibility. In particular, these jobs can be processed on M_1 in the order where h is first.

Similarly, since no jobs are processed on M_1 after $t = S_k^2(\sigma^*)$, all jobs which are processed on M_2 between $t = S_k^2(\sigma^*)$ and $t = C_{\max}(\sigma^*)$ can be processed on M_2 in this time interval in any order without violating the feasibility. In particular, these jobs can be processed on M_2 in the order where k is last. □

Two jobs are processed concurrently if and only if there exists a time period during which these jobs are processed simultaneously.

Theorem 1. *For any instance with $n \le 4$, there exists an optimal permutation schedule.*

Proof. By Lemma 1, there exists an optimal schedule σ^* where the first job and the last job on M_1 are also the first and the last, respectively, on M_2. Therefore, if $n \le 3$, then there exists an optimal permutation schedule.

Let $n = 4$ and assume that there exists a job j which is not processed in σ^* concurrently with any other job. In this case, there exists an optimal permutation schedule for the remaining three jobs, and the proof is concluded by appending job j to this schedule.

Assume that each job is processed concurrently with some other job. Taking into account Lemma 1, without loss of generality, assume that the jobs are processed in σ^* on M_1 in the sequence j_1, j_2, j_3, j_4, whereas on M_2 in the sequence j_1, j_3, j_2, j_4. Then,

$$C_{j_2}^1(\sigma^*) < C_{j_3}^1(\sigma^*) \le S_{j_3}^2(\sigma^*) < S_{j_2}^2(\sigma^*),$$

which implies that j_2 and j_3 are not processed in σ^* concurrently, since the first operations of these two jobs are completed before the start of their second operations.

Suppose that j_3 is processed in σ^* concurrently with j_1. Consider the permutation schedule σ' such that $S_{j_3}^1(\sigma') = S_{j_2}^1(\sigma^*)$ and $C_{j_2}^1(\sigma') = C_{j_3}^1(\sigma^*)$ and the completion times of all other operations in σ' are the same as in σ^*. So, σ' differs from σ^* only by the order of j_2 and j_3 on M_1. Then,

$$C_{j_3}^1(\sigma') < C_{j_2}^1(\sigma') = C_{j_3}^1(\sigma^*) \le S_{j_3}^2(\sigma^*) < S_{j_2}^2(\sigma^*),$$

and because $S_{j_3}^2(\sigma^*) = S_{j_3}^2(\sigma')$ and $S_{j_2}^2(\sigma^*) = S_{j_2}^2(\sigma')$,

$$C_{j_2}^1(\sigma') \le S_{j_2}^2(\sigma') \quad \text{and} \quad C_{j_3}^1(\sigma') \le S_{j_3}^2(\sigma'). \tag{2}$$

Before the point in time $S_{j_4}^1(\sigma')$, only j_1, j_2 and j_3 can use the buffer. The order in which jobs are processed in σ^* and the assumption that j_1 and j_3 are processed concurrently give $S_{j_3}^1(\sigma^*) < C_{j_1}^2(\sigma^*)$. Hence,

$$S_{j_1}^1(\sigma^*) < S_{j_2}^1(\sigma^*) < S_{j_3}^1(\sigma^*) < C_{j_1}^2(\sigma^*) < C_{j_3}^2(\sigma^*) < C_{j_2}^2(\sigma^*),$$

and between points in time $S_{j_3}^1(\sigma^*)$ and $C_{j_1}^2(\sigma^*)$, j_1, j_2 and j_3 use the buffer simultaneously. Given that the transformation of σ^* into σ' does not affect the consumption of the buffer from the point in time $C_{j_3}^1(\sigma^*)$ and that $C_{j_3}^1(\sigma^*) \le S_{j_4}^1(\sigma')$, this transformation cannot violate the buffer size. This, together with (2), gives the feasibility of σ', and the theorem follows from the observation that $C_{\max}(\sigma') = C_{\max}(\sigma^*)$.

Assume that j_3 is processed in σ^* concurrently with j_4. Consider the permutation schedule σ'' such that $S_{j_2}^2(\sigma'') = S_{j_3}^2(\sigma^*)$ and $C_{j_3}^2(\sigma'') = C_{j_2}^2(\sigma^*)$ and the completion time of all other operations are identical to σ^*. So, σ'' differs from σ^* only by the order in which j_2 and j_3 are processed on M_2 between the points in time $S_{j_3}^2(\sigma^*)$ and $C_{j_2}^2(\sigma^*)$. Observe that

$$C_{j_2}^1(\sigma^*) < C_{j_3}^1(\sigma^*) \le S_{j_3}^2(\sigma^*) = S_{j_2}^2(\sigma'') < S_{j_3}^2(\sigma''),$$

and because $C_{j_2}^1(\sigma^*) = C_{j_2}^1(\sigma'')$ and $C_{j_3}^1(\sigma^*) = C_{j_3}^1(\sigma'')$,

$$C_{j_2}^1(\sigma'') \le S_{j_2}^2(\sigma'') \quad \text{and} \quad C_{j_3}^1(\sigma'') \le S_{j_3}^2(\sigma''). \tag{3}$$

After the point in time $C_{j_1}^2(\sigma'')$, only j_2, j_3 and j_4 use the buffer. The order, in which the jobs are processed in σ^*, and the assumption that j_3 is processed concurrently with j_4 in σ^* gives $S_{j_4}^1(\sigma^*) < C_{j_3}^2(\sigma^*)$. Hence,

$$S_{j_2}^1(\sigma^*) < S_{j_3}^1(\sigma^*) < S_{j_4}^1(\sigma^*) < C_{j_3}^2(\sigma^*) < C_{j_2}^2(\sigma^*) < C_{j_4}^2(\sigma^*),$$

and between the points in time $S_{j_4}^1(\sigma^*)$ and $C_{j_3}^2(\sigma^*)$, j_2, j_3 and j_4 use the buffer simultaneously. Since $S_{j_3}^2(\sigma^*) \ge C_{j_1}^2(\sigma^*) = C_{j_1}^2(\sigma'')$ and since σ^* and σ'' differ only after the point in time $S_{j_3}^2(\sigma^*)$, schedule σ'' does not violate the buffer size. This, combined with (3), gives the feasibility of σ''. Finally, the theorem follows from the observation that $C_{\max}(\sigma'') = C_{\max}(\sigma^*)$. □

3. Case $n \ge 5$: difference in makespan

Let α and η be any positive integers and consider the instance $\mathcal{I}_0$ with $\Omega = 9\alpha$ and $N = G \cup H \cup J$, where the sets G, H and J are specified as follows.

- The set G contains 2η identical jobs. For each $g \in G$, $p_{1g} = 2\alpha$ and $p_{2g} = 5\alpha$.
- The set H contains 2η identical jobs. For each $h \in H$, $p_{1h} = 5\alpha$ and $p_{2h} = 1$.
- The set J contains η identical jobs. For each $j \in J$, $p_{1j} = 5\alpha$ and $p_{2j} = 4\alpha$.

Lemma 2. *For any optimal schedule σ^* for $\mathcal{I}_0$,*

$$C_{\max}(\sigma^*) = \eta(19\alpha + 2) \tag{4}$$

and, for each $0 \le t < C_{\max}(\sigma^*)$, there exists exactly one $e \in H \cup J$ and $i \in \{1, 2\}$, satisfying $S_e^i(\sigma^*) \le t < C_e^i(\sigma^*)$.

Proof. Since $s_e = 5\alpha$ for each $e \in H \cup J$ and since $\Omega = 9\alpha$, no two jobs from $H \cup J$ can be processed concurrently. Hence, for any schedule σ

$$C_{\max}(\sigma) \ge \sum_{e \in H \cup J} (p_{1e} + p_{2e}) = \eta(19\alpha + 2). \tag{5}$$

Observe that if (5) is equality, the corresponding σ is optimal and satisfies the lemma. It remains to prove that the right hand side in (5) is attainable.

Partition N into η disjoint sets, each containing two jobs from G, two jobs from H and one job from J. Let $N' = \{g_1, g_2, h_1, h_2, j\}$ be one of these sets, where $g_1 \in G$, $g_2 \in G$, $h_1 \in H$, $h_2 \in H$ and $j \in J$.