Journal of the Operations Research
Society of Japan
Vol. 34, No. 3, September 1991

ON AN AUTOMATED TWO-MACHINE FLOWSHOP SCHEDULING PROBLEM WITH INFINITE BUFFER

Hiroshi Kise
Kyoto Institute of Technology

(Received June 13, 1990)

Abstract A new flowshop scheduling problem related to automated manufacturing systems such as FMS's and FMC's is discussed. The problem is shown to be an extension of the two-machine flowship problem addressed by Johnson (or a special case of the three machine flowshop problem), and to be NP-hard. Some solvable cases are discussed.

1. Introduction

Recent innovations in manufacturing systems such as FMS's and FMC's aim not only at the realization of fully automation, but also at the realization of customizable production with high efficiency. Such systems consist of an automated warehouse, versatile machines such as machining centers, automated transportation and material handling systems such as AGV's (automated guided vehicles) and robot hands, all of which are controlled by computers. Wassenhove [11] and Jaikumar[6] surveyed over half the FMS's worldwide (93 in Japan, 35 in the USA and 27 in Europe), and concluded that there is a definite trend toward more integrated independent cells, each independent cell having a relatively small number of versatile machines, fairly large buffer and simple part routings (also see Kise et al [8] for the usefulness of such cells).

This paper considers a new flowship scheduling problem arising from such automated manufacturing systems. That is, there are two machines, an AGV (or a robot hand), a loading station and an unloading station. Jobs are processed on two machines in the same order. Each machine has sufficient capacity of buffer where partially finished jobs can temporarily be stored to utilize machines and the AGV efficiently. The AGV can send at most one job at a time between two buffers. The loading station where jobs are picked up for being processed on the first machine have sufficient capacity of storage. The unloading station where jobs finished on the second machine are deposited also have sufficient capacity of storage. For this system, we seek an optimal schedule of the jobs that minimizes the maximum completion time (i.e., makespan). If the machines have no buffer, the problem can be solved in polynomial time, [10]. However, if they have finite buffer, it is NP-hard (see Papadimitriou and Kanellakis [9]), strongly suggesting that there is no polynomial time algorithm for it. If the transportation time of the AGV is neglected, the problem is the classical two-machine flowshop scheduling problem addressed by Johnson [7]. There have been found some solvable cases that are extensions of the Johnson's problem (see Graham et al [5]). However, it will be shown that the new problem is a special case of the 3-machine flowshop scheduling problem, and NP-hard. Some solvable cases are discussed.

2. Description and Formulation of the System

The system considered here basically consists of two machines M_a and M_b, an AGV, a loading station S_l and an unloading station S_u. Machine M_a has sufficient capacity of buffer B_a where jobs finished on M_a are deposited until the AGV is ready to transport them. Machine M_b also has sufficient capacity of buffer B_b where jobs from B_a is deposited until M_b is ready to process them. Each machine can process at most one job at a time without pre-emption. The AGV repeatedly runs between B_a and B_b, carrying a job at a time from B_a to B_b. Loading and Unloading stations S_l and S_u have sufficient storage capacity where unfinished and finished jobs are deposited, respectively. In this system each of n jobs stored in S_l is processed on M_a, transferred to B_a, carried to B_b, processed on M_b and then deposited to S_u. Jobs flow in the same order as they are loaded from S_l. This order is referred to as a sequence of jobs.

The following notation is used to formulate the schedule which depicts the flow of each job.

$J = \{1, 2, \cdots, n\}$: the set of n jobs to be processed.
$p_a[j]$, $p_b(j)$: positive processing times of job j on machines M_a and M_b, respectively, including setup times for changing tools and loading and unloading job j from and to a machine, none of which depends on the schedule.
t_{ab} : nonnegative transportation time for the AGV to carry a job from B_a to B_b, including time for loading a job from B_a to the AGV and unloading it to B_b.
t_{ba} : nonnegative time for the empty AGV to travel from B_b to B_a.
$t_{rnd} = t_{ab} + t_{ba}$: the turnaround time of the AGV between B_a and B_b.

We define key time instants in a schedule as follows:

$S_a[j]$, $S_b[j]$: time instants when M_a and M_b start processing job j, respectively.
$T_a[j]$, $T_b[j]$: time instants when the AGV with job j starts from B_a and arrives at B_b, respectively.
$F_a[j]$, $F_b[j]$: time instants when job j is finished on M_a and M_b, respectively.

For a sequence $s = [j(1), j(2), \cdots, j(n)]$ where $j(k)$ represents the k-th job to be processed, these time instants are expressed by the following recursive equations:

$$S_a[j(1)] = 0. \tag{1}$$
$$F_a[j(1)] = S_a[j(1)] + p_a[j(1)]. \tag{2}$$
$$T_a[j(1)] = F_a[j(1)]. \tag{3}$$
$$T_b[j(1)] = T_a[j(1)] + t_{ab}. \tag{4}$$
$$S_b[j(1)] = T_b[j(1)]. \tag{5}$$
$$F_b[j(1)] = S_b[j(1)] + p_b[j(1)]. \tag{6}$$
$$S_a[j(k)] = F_a[j(k-1)], \qquad k = 2, 3, \cdots, n. \tag{7}$$
$$F_a[j(k)] = S_a[j(k)] + p_a(j(k)]. \qquad k = 2, 3, \cdots, n. \tag{8}$$
$$T_a[j(k)] = \max\{F_a[j(k)], T_b[j(k-1)] + t_{ba}\}. \qquad k = 2, 3, \cdots, n. \tag{9}$$
$$T_b[j(k)] = T_a[j(k)] + t_{ab}, \qquad k = 2, 3, \cdots, n. \tag{10}$$
$$S_b[j(k)] = \max\{T_b[j(k)], S_b[j(k-1)] + p_b[j(k-1)]\}, \quad k = 2, 3, \cdots, n. \tag{11}$$
$$F_b[j(k)] = S_b[j(k)] + p_b[j(k)], \qquad k = 2, 3, \cdots, n. \tag{12}$$

Let

$$F_{\max}(s) = F_b[j(n)], \tag{13}$$

then $F_{\max}(s)$ is referred to as the makespan for sequence s. Our objective is to find an optimal sequence s^* such that $F_{\max}(s^*) \leq F_{\max}(s)$ for any sequence s. It can easily be shown that these key time instants can be expressed as follows.

Lemma 1. For a sequence $s = [j(1), j(2), \cdots, j(n)]$,

$$S_a[j(k)] = \sum_{i=1}^{k-1} p_a[j(i)], \qquad k = 1, 2, \cdots, n. \quad (14)$$

$$F_a[j(k)] = \sum_{i=1}^{k} p_a[j(i)], \qquad k = 1, 2, \cdots, n. \quad (15)$$

$$T_a[j(k)] = \max_{1 \leq i \leq k} \{\sum_{h=1}^{i} p_a[j(h)] + (k-i)t_{rnd}\}, \qquad k = 1, 2, \cdots, n. \quad (16)$$

$$T_b[j(k)] = \max_{1 \leq i \leq k} \{\sum_{h=1}^{i} p_a[j(h)] + (k-i)t_{rnd}\} + t_{ab}. \qquad k = 1, 2, \cdots, n. \quad (17)$$

$$S_b[j(k)] = \max_{1 \leq g \leq i \leq k} \{\sum_{h=1}^{g} p_a[j(h)] + (i-g)t_{rnd} + \sum_{h=i}^{k-1} p_b[j(h)]\} + t_{ab}, \qquad k = 1, 2, \cdots, n. \quad (18)$$

$$F_b[j(k)] = \max_{1 \leq g \leq i \leq k} \{\sum_{h=1}^{g} p_a[j(h)] + (i-g)t_{rnd} + \sum_{h=i}^{k} p_b[j(h)]\} + t_{ab}, \qquad k = 1, 2, \cdots, n. \quad (19)$$

Thus we have

$$F_{\max}(s) = \max_{1 \leq i \leq k \leq n} \{\sum_{h=1}^{i} p_a[j(h)] + (k-i)t_{rnd} + \sum_{h=k}^{n} p_b[j(h)]\} + t_{ab}. \quad (20)$$

3. NP-Hardness

We show that our scheduling problem is NP-hard by reducing the following NP-complete knapsack problem to a special case of our problem.

The Knapsack Problem: Given a set of N positive integers $\{a(1), a(2), \cdots, a(N)\}$ where N is an even number, and an integer b such that

$$2b = \sum_{i=1}^{N} a(i) \text{ and } a(i) < b, \quad i = 1, 2, \cdots, N,$$

then decide whether there is a subset I of $\{1, 2, \cdots, N\}$ such that

$$\sum_{i \in I} a(i) = b, \text{ where } |I| = N/2. \quad (21)$$

For any instance of this knapsack problem, define an instance of our scheduling problem by

$$
\begin{aligned}
&n = N + 3,\\
&t_{ab} = b,\ t_{ba} = 1,\ t_{rnd} = b + 1,\\
&p_a[0] = 1,\ p_b[0] = N(b+1)/2 + 1,\\
&p_a[i] = p_b[i] = a(i),\ i = 1, 2, \cdots, N,\\
&p_a[N+1] = p_b[N+1] = N(b+1)/2 + 1,\\
&p_a[N+2] = N(b+1)/2 + 1, \text{ and } p_b[N+2] = 1.
\end{aligned}
$$

Before showing the NP-hardness, the following facts on an optimal sequence (denoted $s = [j(1), j(2), \cdots, j(n)]$) should be noted.

Fact 1. Any schedule s is optimal if it satisfies

$$F_{\max}(s) = n(b+1) + 1. \tag{22}$$

Proof. Note that $p_a[j],\ p_b[j] \geq 1,\ t_{ab} = b$ and $t_{rnd} = b + 1$. Then by letting $i = 1$ and $k = n$ in (20),

$$F_{\max}(s) \geq p_a[j(1)] + (n-1)t_{rnd} + p_b[j(n)] + t_{ab} \geq n(b+1) + 1.$$

The right hand side of this inequality does not depend on schedule s, and hence constitutes a lower bound of the minimum makespan. □

Fact 2. For any two jobs i and j such that

$$p_a[i] \leq p_a[j] \text{ and } p_b[i] \geq p_b[j],$$

there is an optimal sequence in which job i precedes job j.

Proof. It can easily be shown by (20) that a sequence in which job i precedes job j is never worse than the sequence obtained by exchanging the positions of two jobs i and j. This completes the proof. □

Fact 3. Job 0 and job $N + 2$ are optimally sequenced as the first and the last jobs, respectively.

Proof. Note that job 0 has the minimum and the maximum processing times on M_a and M_b, respectively, while job $N + 2$ has the maximum and the minimum processing times on M_a and M_b, respectively. Thus, it follows from Fact 2. □

Fact 4. Job $N + 1$ must be the $(n + 1)/2$-th job for an optimal sequence s to satisfy (22).

Proof. Assume that job $N+1$ is the p-th job in a sequence s and $p \neq (n+1)/2$. Then the following two cases are possible.
i) $p \leq (n-1)/2$: Let $i = p$ and $k = n$ in (20), then

$$
\begin{aligned}
F_{\max}(s) &\geq p_a[0] + \sum_{h=2}^{p-1} p_a[j(h)] + p_a[j(p)] + (n-p)t_{rnd} + p_b[n] + t_{ab}\\
&\geq n(b+1) + 2.
\end{aligned}
$$

ii) $p \geq (n+3)/2$: Let $i = 1$ and $k = p$ in (20), then

$$F_{\max}(s) \geq p_a[0] + (p-1)t_{rnd} + p_b[j(p)] + \sum_{h=p+1}^{n-1} p_b[j(h)] + p_b[n] + t_{ab}$$
$$\geq n(b+1) + 2.$$

This completes the proof. □

Lemma 2. There is an optimal sequence s satisfying (22), if and only if the knapsack problem has a solution I satisfying (21).

Proof. By the above four facts, an optimal sequence takes a form such that

$$s = [0, j(2), \cdots, j((n-1)/2), N+1, j((n+3)/2), \cdots, j(n-1), N+2]. \tag{23}$$

Let

$$A = \sum_{h=2}^{(n-1)/2} p_a[j(h)] = \sum_{h=2}^{(n-1)/2} p_b[j(h)] \text{ and}$$
$$B = \sum_{h=(n+3)/2}^{n-1} p_a[j(h)] = \sum_{h=(n+3)/2}^{n-1} p_b[j(h)] = 2b - A,$$

then the following three cases on the value of A (or B) are possible.

i) $A < b$ and $B > b$: By letting $i = 1$ and $k = (n+1)/2$ in (20),

$$F_{\max}(s) \geq p_a[0] + (n-1)t_{rnd}/2 + p_b[N+1] + B + p_b[N+2] + t_{ab}$$
$$> n(b+1) + 1.$$

ii) $A = B = b$: Note that

$$p_a[j(k)] < b + 1 = t_{rnd}, \qquad k = 1, \cdots, (n-1)/2, (n+3)/2, \cdots, n-1,$$

then by (16),

$$T_a[j(k)] = p_a[j(1)] + (k-1)t_{rnd} = 1 + (k-1)(b+1), \qquad k = 1, 2, \cdots, (n-1)/2. \tag{24}$$

Furthermore, by (9), (10), (15) and assumption ii),

$$T_a[j((n+1)/2)] = \max\{F_a[j((n+1)/2)], T_a[j((n-1)/2)] + t_{rnd}\}$$
$$= p_a[j(1)] + \max\{b + N(b+1)/2 + 1, N(b+1)/2 + b + 1\}$$
$$= 1 + (n-1)(b+1)/2. \tag{25}$$

By the same argument as the above,

$$T_a[j(k)] = p_a[j(1)] + (k-1)t_{rnd} = 1 + (k-1)(b+1). \qquad k = (n+3)/2, \cdots, n. \tag{26}$$

Note that by (19) and (21),

$$F_b[j(1)] = p_a[j(1)] + p_b[j(1)] + t_{ab} = (n-1)(b+1)/2 + 1, \tag{27}$$

then by (10), (11) and (24),

$$S_b[j(k)] = \max\{T_a[j(k)] + t_{ab}, F_b[j(k-1)]\} = \max\{k(b+1), F_b[j(k-1)]\}$$
$$= F_b[j(k-1)], \quad k = 2,3,\cdots,(n-1)/2.$$

This implies by (12) and (1) through (6) that

$$F_b[j(k)] = S_b[j(k)] + p_b[j(k)] = F_b[j(k-1)] + p_b[j(k)]$$
$$= F_b[j(1)] + \sum_{h=2}^{k} p_b[j(h)]$$
$$= (n-1)(b+1)/2 + 1 + \sum_{h=2}^{k} p_b[j(h)], \quad k = 2,\cdots,(n-1)/2. \tag{28}$$

Furthermore, by (9), (10), (11), (25), (28) and assumption ii),

$$S_b[j((n+1)/2)] = \max\{T_a[j((n+1)/2)] + t_{ab}, F_b[j((n-1)/2)]\}$$
$$= (n+1)(b+1)/2.$$

Thus by (12) and (23),

$$F_b[j((n+1)/2)] = S_b[j((n+1)/2)] + p_b[j((n+1)/2)] = (n-1)(b+1) + 1. \tag{29}$$

The same argument as the above leads by (26) and (29) that

$$S_b[j(k)] = F_b[j(k-1)] = F_b[j((n+1)/2)] + \sum_{h=(n+3)/2}^{k-1} p_b[j(h)]$$
$$= (n-1)(b+1) + 1 + \sum_{h=(n+3)/2}^{k-1} p_b[j(h)], \quad k = (n+3)/2,\cdots,n.$$

Thus by (23) and assumption ii),

$$F_{\max}(s) = F_b[j(n)] = (n-1)(b+1) + 1 + B + p_b[j(n)] = n(b+1) + 1.$$

Therefore sequence s satisfies (22).

iii) $A > b$ and $B < b$: By letting $i = (n+1)/2$ and $k = n$ in (20),

$$F_{\max}(s) \geq p_a[j(1)] + A + p_b[j((n+1)/2)] + ((n-1)/2)t_{rnd} + p_b[j(n)]$$
$$+ t_{ab} > n(b+1) + 1.$$

This completes the proof. □

Theorem 1. The automated two-machine flowshop scheduling problem with infinite buffer is NP-hard.

Proof. The knapsack problem is NP-complete [3], and can be reduced to a special case of the scheduling problem in a polynomial time of the problem size, as shown in Lemma 2. This implies the NP-hardness of the scheduling problem. □

In the following we show that our scheduling problem is equivalent to a special case of the classical three-machine flowshop scheduling problem where no transportation time is considered (e.g., see [7]).

The three-Machine Flowshop Problem: Let $A(j), B(j)$ and $C(j)$ be positive processing times of job j on the first, the second and the third machines, respectively. Let $s = [j(1), j(2), \cdots, j(n)]$ be a sequence of n jobs to be processed on these machines. The makespan $C_{\max}(s)$ for a sequence s is given by

$$C_{\max}(s) = \max_{1 \le i \le k \le n} \{\sum_{h=1}^{i} A[j(h)] + \sum_{h=i}^{k} B[j(h)] + \sum_{h=k}^{n} C[j(h)]\}.$$

Then, find a sequence s that minimizes $C_{\max}(s)$.

Here, assume that

$$A[j] = p_a[j], B[j] = t_{rnd} \text{ and } C[j] = p_b[j], \quad j = 1, 2, \cdots, n,$$

then

$$C_{\max}(s) = \max_{1 \le i \le k \le n} \{\sum_{h=1}^{i} p_a[j(h)] + (k - i + 1)t_{rnd} + \sum_{h=k}^{n} p_b[j(h)]\}. \tag{30}$$

Obviously, Minimizing (30) is equivalent to minimizing (20). The following corollaries can be obtained from this equivalency.

Corollary 1. The three-machine flowshop problem is NP-hard, even if the processing times of all jobs on the second machine are identical.

Corollary 1 had been open long time (see Szwarc [10]), and Garey, et al. [4] have shown the NP-hardness for a more general three machine flowshop scheduling problem where jobs are allowed to have different processing times on the three machines.

Corollary 2. The automated two-machine flowshop scheduling problem with infinite buffer has an optimal permutation schedule.

Proof. By the same property for the three-machine flowshop scheduling problem [7]. □

Corollary 3. The automated two-machine flowshop scheduling problem with infinite buffer can be solved in polynomial time, if one of the following conditions is satisfied:

i) $t_{rnd} \le \max\{\min_{1 \le j \le n} p_a[j], \min_{1 \le j \le n} p_b[j]\}$,

ii) $t_{rnd} \ge \min\{\max_{1 \le j \le n} p_a[j], \max_{1 \le j \le n} p_b[j]\}$,

iii) $p_a[i] \le p_a[j]$ implies $p_b[i] \ge p_b[j]$ for any pair of i and j, or

iv) $[\min\{p_a(i), t_{rnd}\} - \min\{p_a(j), t_{rnd}\}]$

$$\times [\min\{p_b(j), t_{rnd}\} - \min\{p_b(i), t_{rnd}\}] \ge 0. \quad i, j = 1, 2, \cdots, n.$$

Proof. The above four are sufficient conditions for the three-machine flowshop problem to be solved in polynomial time. Condition i) has been addressed by Johnson [7], condition ii) by Arthanari and Mukhopadhyay [1], condition iii) by Szwarc [10] and condition iv) by Burns and Rooker [2]. □

Acknowledgement: The author would like to thank Prof. Toshihide Ibaraki of Kyoto University for his valuable comments.

References

[1] T.S. Arthanari and A.C. Mukhopadhyay: A Note on a Paper by W. Szwarc, *Naval Research Logistics Quarterly*, Vol. 15 (1971), 135–138.

[2] F. Burns and J. Rooker: A Special Case of the $3 \times n$ Flow-Shop Problem, *Naval Research Logistics Quarterly*, Vol. 22 (1975), 811–817.

[3] M.R. Garey and D.S. Johnson: *Computers and Intractability*, W.H. Freeman and Company, San Francisco, 1979.

[4] M.R. Garey, D.S. Johnson and Ravi Sethi: The Complexity of Flowshop and Jobshop Scheduling, *Mathematics of Operations Research*, Vol. 1, No. 1 (1976), 117–128.

[5] R.L. Graham, 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*, Vol. 5 (1979), 287–326.

[6] R. Jaikumar: Postindustrial Manufacturing, *Harvard Business Review*, Nov.-Dec. 1986.

[7] S.M. Johnson: Optimal Two-and Three-Stage Production Schedules with Setup Times Included, *Naval Research Logistics Quarterly*, Vol. 1 (1954), 61–68.

[8] H. Kise, T. Shioyama and T. Ibaraki: Automated Two-Machine Flowshop Scheduling: A Solvable Case, *Transactions of IIE*, Vol. 23, No. 1 (1991), 10–16.

[9] C.H. Papadimitriou and P.C. Kanellakis: Flowshop Scheduling with Limited Temporary Storage, *Journal of the Association for Computing Machinery*, Vol. 27, No. 3 (1980), 533–549.

[10] W. Szwarc: Mathematical Aspects of the $3 \times n$ Job-Shop Sequencing Problem, *Naval Research Logistics Quarterly*, Vol. 21 (1974), 145–153.

[11] L.V. Wassenhove: A Planning Framework for a Class of FMS, *Operations Research Proceedings of the 17-th Annual Meetings for Deutsche Gesellshaft für Operations Research*, 524–532, Springer-Verlag, Berlin, Heidelberg, 1988.

Hiroshi Kise
Dept. of Mechanical and System Engineering
Kyoto Institute of Technology
Matsugasaki, Sakyouku
Kyoto, 606, Japan