9-1-2010

# A Flowshop Scheduling Problem With Transportation Times and Capacity Constraints

Abraham Mendoza
*Universidad Panamerica*

Jose A. Ventura
*The Pennsylvania State University*

Kwei Long Huang
*National Taiwan University*

# A FLOWSHOP SCHEDULING PROBLEM WITH TRANSPORTATION TIMES AND CAPACITY CONSTRAINTS

**Abraham Mendoza**
Department of Industrial Engineering
Universidad Panamericana, Zapopan, Jalisco 45010, México

**José A. Ventura**
Harold and Inge Marcus Department of Industrial and Manufacturing Engineering
The Pennsylvania State University, University Park, PA 16802, USA

**Kwei-Long Huang**
Institute of Industrial Engineering
National Taiwan University, Taipei 106, Taiwan - ROC

**Abstract**

Although there are numerous methodologies and research studies on machine scheduling, most of the literature assumes that there is an unlimited number of transporters to deliver jobs from one machine to another for further processing and that transportation times can be neglected. These two assumptions are not applicable if one intends to generate an accurate schedule for the shop floor. In this research, a flowshop scheduling problem with two machines, denoted as M1 and M2, and a single transporter with capacity $c$ is considered. The main focus is on the development of a dynamic programming algorithm to generate a schedule that minimizes the makespan. The transporter takes $t_1$ time units to travel with at least one job from machine M1 to machine M2, and $t_2$ time units to return empty to machine M1. When the processing times for all $n$ jobs on machine M1 are constant, denoted as $p_{j1} \equiv p_1$, and the capacity of the transporter $c$ is at least $\left\lceil \frac{2(t_1+t_2)}{p_1} \right\rceil - 1$, the computational complexity of the proposed algorithm is shown to be $O(n^3)$.

## 1. Introduction

During the past five decades, shop floor scheduling has been a topic intensively addressed in manufacturing systems planning and control [1], [6], [11]. Although there are numerous methodologies and research studies on machine scheduling, most of the literature assumes that there is an unlimited number of transporters to deliver jobs from one machine to another for further processing and that transportation times can be neglected, which means that jobs are transported to the next machine instantaneously. These two assumptions are not applicable if one intends to generate an accurate schedule for the shop floor. Furthermore, even though transportation times are considered and handled separately from the processing times, most existing models still assume an unlimited number of transporters. Integrated scheduling of processing and material handling operations involves two types of resources: machines and transporters. Either resource could become a bottleneck if not properly scheduled. Thus, incorporating transportation in classical machine scheduling will lead to more realistic and accurate models for practical implementation in the shop floor.

Extensive literature can be found on machine scheduling involving time lag which is the time between the completion of an operation and the beginning of the next operation of a job in a production system. It can be referred to as the transportation, cooling, or heating time. In our research, the time lag is considered to be the transportation time which is attributed to the actual transportation of a job between processing machines by transporters or automated guided vehicles (AGVs). In the classical models, it is assumed that jobs can be transported between machines instantaneously. This ideal assumption, however, is not applicable to most practical production environments. There are two types of transportation time considerations in the literature: one considers only the time lag, which implies that transporters are always available [2], [4], [12], [13], [15], [16]; the other explicitly takes both transportation time and availability of transporters into consideration [3], [5], [7], [8], [9], [10], [14]. In these models, several attributes can be configured according to real manufacturing environments: processing times on machines, transportation times between machines, number of transporters, and capacity of the transporters.

In this paper, a flowshop manufacturing environment involving processing and transportation of jobs is considered. The main focus is on the development of a dynamic programming (DP) algorithm to generate a schedule for a two-machine flowshop that minimizes the makespan. The two machines, M1 and M2, are available at time zero for processing $n$ independent jobs. All jobs begin their processing on machine M1 and complete it on machine M2. It is assumed that there is a single transporter of capacity $c$ in the flowshop to deliver jobs from M1 to M2. Jobs transported simultaneously in one trip from M1 to M2 are defined as a batch. Transportation times between these two machines are explicitly considered. Lee and Chen [5] have shown that, under the assumption that the processing times for all jobs on machine M1 are equal to a constant value, denoted as $p_{j1} \equiv p_1$, jobs can be pre-sequenced in the same LPT (longest processing time first) order on both machines and the problem becomes polynomially solvable. They developed a DP algorithm of order $O\left((cn)^3\right)$. In this research, an improved DP algorithm is proposed given that the capacity of the transporter $c$ is greater than or equal to $\left\lceil \frac{2(t_1+t_2)}{p_1} \right\rceil - 1$, where $(t_1+t_2)$ is the time that the transporter takes to move a batch of jobs from M1 to

M2 and return empty to M1. The computational complexity of the new algorithm is shown to be $O\left(n^3\right)$.

This paper is divided into five sections. Section 2 includes the problem statement. The DP algorithm and the proof of its computational complexity are presented in Sections 3 and 4, respectively. Conclusions and directions for future research are stated in Section 5.

## 2. Problem Description and Notation

In a two-machine flowshop, $n$ jobs need to be scheduled, first on machine M1 and later on machine M2. Each machine can only process one job at a time, and preemption is not allowed. All jobs are available at time zero and wait for processing in the input buffer of machine M1. The processing time on machine Ml for job $j$ is denoted as $p_{j1}$ and on machine M2 as $p_{j2}$. After the operation on machine M1 is completed, jobs are stored in the output buffer of machine M1 and wait to be transported to machine M2. We also assume that there is a single transporter in the system and its capacity is denoted as $c$. Jobs transported together in one shipment from machine M1 to machine M2 are defined as a batch. Let $u$ denote the maximum number of jobs to be transported in a batch. Then, $u \le c$. After being transported to machine M2, jobs wait to be processed in the input buffer of machine M2. The buffer sizes are assumed to be unlimited. The transporter takes $t_1$ time units to travel from machine M1 to machine M2, and $t_2$ time units to return to machine M1. The departure time of the $k^{th}$ batch from M1 to M2 is denoted as $d_k$. Loading and unloading times of jobs on machines are either negligible or assumed to be included in their processing times. Similarly, times to load and unload jobs on the transporter are either negligible or assumed to be included in the transportation times.

The objective of this study is to minimize the makespan, $C_{\max}$. The three-field notation $\alpha \mid \beta \mid \gamma$ is adopted to represent this machine scheduling problem. In the $\alpha$ field, $TF_2$ denotes the two-machine flowshop scheduling problem with transportation times between machines which is used by Lee and Chen [5]. In the $\beta$ field, $v$ denotes the number of transporters, and $c$ denotes the capacity of the transporter. In the $\gamma$ field, $C_{\max}$ is the objective of the problem. Hence, the scheduling problem to minimize the makespan in a two-machine flowshop with $x$ transporters and the capacity of each transporter equal to $y$ is can be represented as $TF_2 \mid v = x, c = y \mid C_{\max}$.

## 3. Dynamic Programming (DP) Model

According to the paper by Hurink and Knust [3], the two-machine flowshop problem with one transporter of capacity one ($TF_2 \mid v = 1, c = 1 \mid C_{\max}$) is strongly NP-hard.

However, Lee and Chen [5] have shown that, if the following assumption is considered, then a permutation sequence of jobs on the machines can be predetermined and, thus, the problem becomes polynomially solvable.
***Assumption 1***: The processing times for all jobs on machine M1 are job-independent, i.e., the processing times are equal to a constant value, denoted as $p_{j1} \equiv p_1$ for all $j$.

Under Assumption 1, Lee and Chen [5] proved that there exists an optimal schedule for the $TF_2 \mid p_{j1} \equiv p_1, v \geq 1, c \geq 1 \mid C_{\max}$ problem such that jobs are sequenced in the non-increasing order of $p_{j2}$ (the longest processing time first) on both machines. Based on this property, they develop a DP algorithm to solve the problem in polynomial time. In this research, an improved DP algorithm is proposed when there is only one transporter and the capacity of the transporter is greater than or equal to a threshold value *u*. For this special case, the number of jobs in a batch transported from machine M1 to machine M2 is always less than or equal to this threshold value in an optimal schedule. The threshold value will be derived later.

Lee and Chen [5] have proven several properties that hold for two-machine flowshop problems with transportation times. Those properties are also necessary conditions for deriving our algorithm.

***Property 1*** [5]: There exists an optimal schedule for the $TF_2 \mid v \geq 1, c \geq 1 \mid C_{\max}$ problem that satisfies the following conditions:

(i) Jobs are processed on machine M1 without idle time.
(ii) Jobs transported in the same batch are processed consecutively without idle time on both machines.
(iii) Jobs finished earlier on machine M1 are delivered earlier to machine M2. Furthermore, the sequence of jobs on machine M1 is the same as that on machine M2, namely, it is a permutation schedule.
(iv) The departure times of two consecutive batches delivered satisfy that either $d_{k+1} = d_k + t$ or $d_{k+1}$ is the completion time of the last job in the $(k+1)^{th}$ batch on machine M1, where $t = t_1 + t_2$ is the transportation time of a round trip between machines M1 and M2. When $d_{k+1}$ is equal to the completion time of the last job in the $(k+1)^{th}$ batch on machine M1, $d_{k+1}$ is referred to as an integer departure point; otherwise, it is called as the immediate departure point.

Given that the processing times for all jobs on machine M1 are identical (Assumption 1), a property regarding the threshold value of the transporter's capacity can be derived.

***Property 2*** [17]: There exists an optimal schedule for the $TF_2 \mid p_{j1} \equiv p_1, v=1, c \geq u \mid C_{\max}$ problem, where the number of jobs in a batch transported from machine M1 to machine M2 is always less than or equal to a threshold value $u = \left\lceil \frac{2(t_1+t_2)}{p_1} \right\rceil - 1$.

***Assumption 2***: The capacity of the transporter $c$ is greater than or equal to the threshold value $u = \left\lceil \frac{2(t_1+t_2)}{p_1} \right\rceil - 1$.

Based on Assumptions 1 and 2, a forward DP algorithm is proposed to solve the $TF_2 \mid p_{j1} \equiv p_1, v=1, c \geq u \mid C_{\max}$ problem. According to Property 2, when the size of a batch (denoted as *B*) is greater than the threshold value *u*, the batch can always be divided into two smaller batches of sizes $\left\lfloor \frac{t_1+t_2}{p_1} \right\rfloor$ and $B - \left\lfloor \frac{t_1+t_2}{p_1} \right\rfloor$ to yield a smaller makespan. If the number of jobs in the second batch ($B - \left\lfloor \frac{t_1+t_2}{p_1} \right\rfloor$) is still greater than *u*, this batch can be further split into two smaller batches $\left\lfloor \frac{t_1+t_2}{p_1} \right\rfloor$ and $B - 2\left\lfloor \frac{t_1+t_2}{p_1} \right\rfloor$ until none of the sizes of all these small batches is greater than *u*. Inspired by this idea, a forward DP algorithm is formulated below.

### 3.1. DP Algorithm for $TF_2 \mid p_{j1} \equiv p_1, v=1, c \geq u \mid C_{\max}$ [17]

**Optimal value function** (OVF): *F(k)* = minimum completion time of a partial schedule containing the first *k* jobs, given that the completion time of job *k* is an integer departure point.

**Arguments** (ARG): *k* = index of a job such that the completion time of the job is an integer departure point.

**Optimal policy function** (OPF): *j* = number of jobs from integer departure point *k* to the previous integer departure point.

**Recurrence relation** (RR):

$$F(k) = \min_{\left\lceil (t_1+t_2)/p_1 \right\rceil \leq j \leq k-1} \{F(k-j) + C(k-j+1, k)\}, \quad k = \left\lceil (t_1+t_2)/p_1 \right\rceil + 1, \ldots, n,$$

where $C(k-j+1, k)$ is the minimum increase in makespan due to jobs $k-j+1$ to $k$. It can be calculated by the following procedure:

Step 1: Let $g = 1,\ x = k - j + 1,\ x_0 = k - j,\ t_0 = p_1(k - j)$ and $C_0 = F(k - j)$.

Step 2: $C_g = \max\{C_{g-1}, t_0 + g(t_1 + t_2) + t_1\} + \sum_{i=x}^{x_0 + \left\lfloor \frac{g(t_1+t_2)}{p_1} \right\rfloor} p_{i2}$ .

Step 3: $x = x + \left\lfloor \dfrac{g(t_1 + t_2)}{p_1} \right\rfloor,\ g = g + 1$ .

Step 4: If $g = \left\lfloor \dfrac{jp_1}{t_1 + t_2} \right\rfloor$, stop and go to Step 5. Otherwise, go back to Step 2.

Step 5: $C(k - j + 1, k) = \max\{C_{g-1}, kp_1 + t_1\} + \sum_{i=x}^{k} p_{i2} - C_0$ .

**Boundary conditions** (BC):

$$F(1) = p_1 + t_1 + p_{1,2},$$

$$F(k) = \min\left\{\begin{array}{c} kp_1 + t_1 + p_{1,2} \\ p_1 + t_1 + \max\{t_1 + t_2,\ p_{1,2}\} \end{array}\right\} + \sum_{i=2}^{k} p_{i2},\quad k = 2, 3, ..., \left\lceil (t_1 + t_2)/p_1 \right\rceil.$$

**Answer** (ANS):

$$\hat{F}(n) = \min\left\{\begin{array}{c} \min_{\left\lceil (t_1+t_2)/p_1 \right\rceil \le j \le n-1} \left\{F(n - j) + \hat{C}(n - j + 1, n)\right\} \\ F(n) \end{array}\right\},$$

where the calculation of $\hat{C}(x, y)$ is similar to that of $C(x, y)$, but it requires modifications on the last four steps as follows:

Step 2: $C_g = \max\{C_{g-1}, t_0 + g(t_1 + t_2) + t_1\} + \sum_{i=x}^{\min\{n, x_0 + \left\lfloor \frac{g(t_1+t_2)}{p_1} \right\rfloor\}} p_{i2}$ .

Step 3: $x = \min\{n, x + \left\lfloor \dfrac{g(t_1 + t_2)}{p_1} \right\rfloor\},\ g = g + 1$ .

Step 4: If $(g - 1)(t_1 + t_2) \ge jp_1$ stop and go to Step 5, and go to Step 5. Otherwise, go back to Step 2.

Step 5: $\hat{C}(n - j + 1, n) = C_{g-1} - C_0$ .

The following property concerning departure points can be stated.

***Property 3*** [17]: Let $d_j$ and $d_k$ be two consecutive integer departure points corresponding to the completion times of jobs *j* and *k* on machine M1. Between these two integer departure points, once the transporter returns to machine M1, it will transport the completed jobs immediately to machine M2 until its returning time to machine M1 is greater than $d_k - (t_1 + t_2)$.

## 4. Complexity Analysis

To obtain the complexity of the proposed algorithm, the worst case ($k = 1,\dots,n$) is considered such that there are a total of *n* possibilities of *k*. For a given *k*, there are *k* possibilities of *j* since $j = 1,\dots,k$. Given *k* and *j*, there are at most *j* immediate departure points. In addition, in Boundary Condition (*BC*) we have $j = 1,\dots,n$ and there are at most *j* immediate departure points for a given *j*. Hence the overall complexity can be calculated as follows [17]:

$$\begin{aligned}\sum_{k=1}^{n}\sum_{j=1}^{k} j &= \sum_{k=1}^{n}\frac{k(k+1)}{2}\\ &= \sum_{k=1}^{n}\left(\frac{k^2}{2}+\frac{k}{2}\right)\\ &= \frac{1}{2}\frac{n(n+1)(2n+1)}{6}+\frac{1}{2}\frac{n(n+1)}{2}\\ &= \frac{n(n+1)(n+2)}{6}.\end{aligned}$$

Thus, the complexity of the proposed algorithm is $O\left(n^3\right)$ which is better than $O\left((c\,n)^3\right)$.

## 5. Concluding Remarks

A flowshop scheduling problem with two machines, one transporter with a specific capacity, and *n* jobs available at time 0 has been studied in this paper. The objective is to minimize the makespan. When processing times for all jobs on machine M1 are identical, a threshold value *u* for the transporter's capacity can be derived, as shown in Property 2. Under the assumptions of identical processing times on machine M1 and a transporter's capacity greater than or equal to *u*, the $TF_2 \mid p_{j1} \equiv p_1, v = 1, c \geq u \mid C_{\max}$ problem can be solved in polynomial time by the proposed DP algorithm. The computational complexity of the DP algorithm has been shown to be $O\left(n^3\right)$, which is better than complexity of the algorithm proposed by Lee and Chen [5]. Therefore, when the capacity of the transporter is not less than *u* ($c \geq u$), the problem can be solved more efficiently by using the proposed algorithm.

Many interesting topics on machine scheduling with transportation considerations remain for future exploration. Various polynomially solvable special cases need to be identified and more realistic models need to be investigated.

## Acknowledgements

This research has been funded by a grant from the PSU/Technion Marcus Funds and the Ridg-U-Rak Honors Scholarship from the Material Handling Education Foundation.

## References

[1] Askin, R. G. and J. B. Goldberg (2002). *Design and Analysis of Lean Production Systems*. New York: John Wiley and Sons.

[2] Dell'Amico, M. (1996). "Shop Problems with Two Machines and Time Lags," *Operations Research*, 44(5), pp. 777-787.

[3] Hurink, J. and S. Knust (2001). "Makespan Minimization for Flow-Shop Problems with Transportation Times and a Single Robot," *Discrete Applied Mathematics*, 112(1), pp. 199-216.

[4] Karuno, Y. and H. Nagamochi (2003). "A Better Approximation for the Two-Machine Flowshop Scheduling Problem with Time Lags," *The 14th Annual International Symposium on Algorithms and Computation*, Kyoto, Japan, pp. 309-318.

[5] Lee, C.-Y. and Z.-L. Chen (2001). "Machine Scheduling with Transportation Considerations," *Journal of Scheduling*, 4, pp. 3-24.

[6] Lee, C.-Y., L. Lei and M. Pinedo (1997). "Current Trends in Deterministic Scheduling," *Annals of Operation Research*, 70, pp. 1-41.

[7] Lee, C.-Y. and V. A. Strusevich (2005). "Two-Machine Shop Scheduling with an Uncapacitated Interstage Transporter," *IIE Transactions*, 37, pp. 725-736.

[8] Levner, E., K. Kogan and I. Levin (1995). "Scheduling a Two-Machine Robotic Cell: A Solvable Case," *Annals of Operations Research*, 5, pp. 217-232.

[9] Levner, E., K. Kogan and O. Maimon (1995). "Flowshop Scheduling of Robotic Cells with Job-Dependent Transportation and Set-up Effects," *Journal of Operational Research Society*, 46(12), pp. 1447-1455.

[10] Oulamara, A. and A. Soukhal (2001). "Flow Shop Scheduling Problems with Transportation and Capacity Constraints," *IEEE International Conference on Systems, Man, and Cybernetics*, 4, pp. 2540-2545.

[11] Pinedo, M. (2002). *Scheduling: Theory, Algorithms, and Systems*, 2nd Ed. Upper Saddle River, NJ: Prentice Hall, Inc.

[12] Rebaine, D. and V. A. Strusevich (1999). "Two-Machine Open Shop Scheduling with Special Transportation Times," *Journal of the Operational Research Society*, 50, pp. 756-764.

[13] Schutten, J. M. J. (1998). "Practical Job Shop Scheduling," *Annals of Operation Research*, 83, pp. 161-177.

[14] Soukhal, A., A. Oulamara and P. Martineau (2005). "Complexity of Flow Shop Scheduling Problems with Transportation Constraints," *European Journal of Operational Research*, 161, pp. 32-41.

[15] Strusevich, V. A. (1999). "A Heuristic for the Two-Machine Open-Shop Scheduling Problem with Transportation Times," *Discrete Applied Mathematics*, 93, pp. 287-304.

[16] Szwarc, W. (1983). "Flow Shop Problems with Time Lags," *Management Science*, 29(4), pp. 477-481.

[17] Ventura, J.A., K.L. Huang, and A. Mendoza (2010). "A Two-Machine Flow Shop Scheduling Problem with Transportation and Capacity Constraints," Working Paper, Department of Industrial and Manufacturing Engineering, The Pennsylvania State University, University Park, PA.