Heuristic procedures for minimizing makespan and the number of required pallets

Chengbin Chu, Jean-Marie Proth, Suresh Sethi

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Heuristic Procedures for Minimizing Makespan and the Number of Required Pallets

Chengbin CHU - Jean-Marie PROTH
Suresh SETHI

N° 1929
Juin 1993

PROGRAMME 5

Traitement du signal,
automatique et
productique

PROCEDURES HEURISTIQUES POUR MINIMISER LE TEMPS TOTAL DE FABRICATION ET LE NOMBRE DE PALETTES UTILISEES DANS UN FLOW-SHOP

Chengbin CHU(*), Jean-Marie PROTH(**) et Suresh SETHI(***)

RESUME :

Une procédure heuristique pour trouver l'ordonnancement qui minimise le temps total de fabrication d'un ensemble de produits dans un système linéaire est proposée. Ce problème est NP-complet. Comparé aux algorithmes existants, celui que nous proposons conduit à de meilleures performances.

Nous proposons également un algorithme pour minimiser le nombre de palettes nécessaires pour un ordonnancement et une performance donnée.

MOTS CLEFS :

Ordonnancement, système linéaire, optimisation des ressources, temps de fabrication.

(*) INRIA-Lorraine, CESCOM, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE.

(**) INRIA-Lorraine, CESCOM, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE, et Institute for Systems Research, University of Maryland, College Park, MD 20742, USA.

(***) Faculty of Management, University of Toronto, 246 Bloor Street West, Toronto, Ontario M5S 1V4, CANADA

Adresse de correspondance :
Jean-Marie PROTH, INRIA-Lorraine, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE, Tél: 87 20 35 00, Fax: 87 76 39 77, e-mail: proth@ilm.loria.fr

HEURISTIC PROCEDURES FOR MINIMIZING MAKESPAN AND THE NUMBER OF REQUIRED PALLETS

Chengbin CHU(*), Jean-Marie PROTH(**) and Suresh SETHI(***)

ABSTRACT:

A heuristic procedure is developed for minimizing makespan in flow-shop scheduling problems.

In comparison with current algorithms, our algorithm seems to result in an improved makespan with a small additional computational effort.

An algorithm is also developed to minimize the required number of pallets for a given makespan.

KEY WORDS:

Scheduling, flow-shop, resource optimization, makespan.

(*) INRIA-Lorraine, CESCOM, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE.

(**) INRIA-Lorraine, CESCOM, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE, and Institute for Systems Research, University of Maryland, College Park, MD 20742, USA.

(***) Faculty of Management, University of Toronto, 246 Bloor Street West, Toronto, Ontario M5S 1V4, CANADA

Correspondence address:
Dr. J.M. PROTH, INRIA-Lorraine, Technopôle Metz 2000, 4 rue Marconi, F-57070 Metz, FRANCE, Tel: (33) 87 20 35 00, Fax: (33) 87 76 39 77, e-mail: proth@ilm.loria.fr

1. INTRODUCTION

Flow-shop scheduling problems form an important class of scheduling problems. A flow-shop consists of m different machines arranged in series. A flow-shop scheduling problem considers a set of n jobs which must all be processed using the machines in the order of the series. On any given machine, two different jobs may require different processing times. Furthermore, on two different machines, the sequence of jobs proccessed is the same. A schedule is defined by specifying a sequence which the jobs must follow on the machines. The choice of a schedule depends on the type of the objective function of interest. Several different objective functions have been treated in the literature.

We are interested in minimizing the duration required to complete the given set of jobs. This duration, commonly known as the makespan, depends on the schedule. The problem is known as n/m/F/Cmax in the literature. It can be solved in O(n log n) time when m = 2 (see [9]) and is known to be NP-hard for $m \geq 3$ (see [11] and [17]). In the case of n jobs and m machines, we then have to consider, at least implicitly, n! different schedules. However, some results which reduce this number are known (see in particular [3]).

This problem has been extensively studied in the literature (see [2], [4], [6-8], [10], [12-16], [18] and [19]). Among these algorithms, the one proposed by CAMPBELL, DUDEK and SMITH (see [2]), commonly known as CDS, is used as a bench-mark to evaluate the heuristics. In this paper, we compare our algorithm with CDS and the most recently published one PHD (Procédure de HAN et DEJAX, see [7]).

In what follows, we propose a heuristic algorithm which provides a near-optimal solution to the problem. Another important goal is the minimization of the in-process inventory for a given value of the makespan. In the case of a Flexible Manufacturing System (FMS), the in-process inventory level is related to the amount of transportation facilities required (number of pallets or number of carts for instance). Moreover these facilities are also very expensive. We propose a fast algorithm which, for a given makespan and the job sequence, minimizes the required number of pallets by starting the processing of jobs on the first machine as late as possible.

This paper is organized in three sections. The second section shows how to compute the makespan for a given job sequence. We then point out the fact that the makespan can be interpreted as the longest path belonging to a particular set of paths. We also provide a proof of this result for the sake of completeness. This result leads to a heuristic algorithm. We finally give the procedure to obtain the minimal number of pallets for a given job sequence. In section 3 of the paper, we present some numerical examples.

2. STATEMENT OF THE PROBLEM AND THE HEURISTIC PROCEDURE

In this section, we specify the notation,state the problem along with a new representation, and develop a heuristic algorithm.

2.1. Definitions

We denote by n, the number of jobs and by m, the number of machines involved. Machine $j \in (1,2,...,m)$ denotes the j-th machine of the series.

We define order r as the permutation:

$r: \{1,2,...,n\} \rightarrow \{1,2,...,n\}$

where r(i) is the rank of the job i and r^{-1} (k) is job of rank k according to the order given by r. For instance, if five jobs 1,2,3,4,5 are ordered as {2,5,1,3,4}, then:

$r(1) = 3 ; r(2) = 1 ; r(3) = 4 ; r(4) = 5 ; r(5) = 2$

and:

$r^{-1}(1) = 2 ; r^{-1}(2) = 5 ; r^{-1}(3) = 1 ; r^{-1}(4) = 3 ; r^{-1}(5) = 4$

We are given the matrix:

$\Theta = \left[\theta_{i,j}\right]; i = 1,2,...,n; j = 1,2,...,m$

with $\theta_{i,j}$ denoting the processing time of the job i on machine j.

We denote by $\Theta^r = \left[\theta^r_{i,j}\right]; i = 1,2,...,n; j = 1,2,...,m$ the matrix Θ obtained after reordering the rows according to r.

Thus:

$$\left.\begin{array}{l} \theta_{i,j} = \theta^r_{r(i),j} \\ \text{and} \\ \theta^r_{i,j} = \theta_{r^{-1}(i),j} \end{array}\right\} i = 1,2,...,n; j = 1,2,...,m$$

It should be clear that $\theta^r_{i,j}$ is the processing time of the job of rank i under order r on machine j. Note also that $\theta = \theta^I$, where I denotes the identity permutation (I(i) = i).

Furthermore, we denote by:

$T^r = \left[t^r_{k,j}\right]; k = 1,2,...,n; j = 1,2,...,m$

the matrix of completion times with $t^r_{k,j}$ standing for the completion time on machine j, of the k-th job, i.e. the job of rank k under order r. Note that given r, the k-th job is the job $r^{-1}(k)$.

The goal consists of finding an r* such that:

$$t^{r^*}_{n,m} = \underset{r}{\text{Min}}\ t^{r}_{n,m}$$

where $t^{r}_{n,m}$ denotes the makespan corresponding to order r.

In other words, we are looking for an r* which minimizes the time at which the last job is completed, assuming that the first job starts at time 0.

2.2. Computation of the makespan

Two conditions have to hold before the k-th job (k>1) under a given order r begins on machine j (j>1):

a. The k-th job must be finished on machine j-1. This event occurs at time $t^{r}_{k,j-1}$.

b. The (k-1)-th job must be finished on machine j, which means that the machine j must be idle. This occurs at time $t^{r}_{k-1,j}$.

Hence, the time at which the k-th job begins on machine j is given by:

$$\text{Max}\ \left(t^{r}_{k,j-1}, t^{r}_{k-1,j}\right)$$

Then, we have:

$$\begin{aligned} t^{r}_{k,j} &= \text{Max}\ \left(t^{r}_{k,j-1}, t^{r}_{k-1,j}\right) + \theta^{r}_{k,j} \\ &= \text{Max}\ \left(t^{r}_{k,j-1}, t^{r}_{k-1,j}\right) + \theta^{I}_{r^{-1}(k),j} \end{aligned} \qquad (1)$$

with $t^{r}_{k,0} = 0$ and $t^{r}_{0,j} = 0$ for all k and j.

In order to illustrate the previous relations, we consider two examples with n = 4, m = 3. The processing times are given as follows:

$$\Theta = \begin{bmatrix} 9 & 2 & 1 \\ 1 & 7 & 6 \\ 5 & 9 & 8 \\ 1 & 2 & 7 \end{bmatrix}.$$

Example 1 (r = I). The related matrix $T^{I} = \left[t^{I}_{k,j}\right]$; k = 1,...,n; j = 1,...,m, is computed in four steps as shown in figure 1.

$$T^{I} = \begin{bmatrix} 9 & & \\ & & \\ & & \\ & & \end{bmatrix} \Rightarrow \begin{bmatrix} 9 & 11 & 12 \\ & & \\ & & \\ & & \end{bmatrix} \Rightarrow \begin{bmatrix} 9 & 11 & 12 \\ 10 & & \\ 15 & & \\ 16 & & \end{bmatrix} \Rightarrow \begin{bmatrix} 9 & 11 & 12 \\ 10 & 18 & 24 \\ 15 & 27 & 35 \\ 16 & 29 & 42 \end{bmatrix}$$

Fig. 1: Computation of a makespan: first example

In this example, therefore, the makespan corresponding to I is 42.

Example 2 (r(1) = 2 ; r(2) = 4 ; r(3) = 1 ; r(4) = 3). First we reorder the rows of Θ according to r to obtain:

$$\Theta^r = \begin{bmatrix} 5 & 9 & 8 \\ 9 & 2 & 1 \\ 1 & 2 & 7 \\ 1 & 7 & 6 \end{bmatrix}.$$

As for order r=I, the corresponding matrix T^r is computed in four steps (see figure 2).

$$T^I = \begin{bmatrix} 5 & & \\ & & \\ & & \\ & & \end{bmatrix} \Rightarrow \begin{bmatrix} 5 & 14 & 22 \\ & & \\ & & \\ & & \end{bmatrix} \Rightarrow \begin{bmatrix} 5 & 14 & 22 \\ 14 & & \\ 15 & & \\ 16 & & \end{bmatrix} \Rightarrow \begin{bmatrix} 5 & 14 & 22 \\ 14 & 16 & 23 \\ 15 & 18 & 30 \\ 16 & 25 & 36 \end{bmatrix}$$

Fig. 2: Computation of makespan: second example

Thus, the makespan corresponding to r is 36.

2.3. A basic result

From T^r, we derive the sequence $S^r = (s^r_1, s^r_2, \ldots, s^r_{n+m-1})$ of n + m - 1 elements as follows:

a. $s^r_{m+n-1} = t^r_{n,m}$ (2)

b. If $s^r_k = t^r_{i,j}$, then:

$$s^r_{k-1} = \begin{cases} t^r_{i,j-1} \text{ if } i = 1 \text{ and } 1 < j \le m \\ t^r_{i-1,j} \text{ if } 1 < i \le n \text{ and } j = 1 \\ \text{Max}\left(t^r_{i-1,j}, t^r_{i,j-1}\right) \text{ if } 1 < i \le n \text{ and } 1 < j \le m \end{cases} \quad (3)$$

Thus, sequence S^r is defined backward, starting from $t^r_{n,m}$. Obviously, $s^r_1 = t^r_{1,1}$.

Examples 1 and 2 continued. Applying relations (2) and (3), we obtain for Example 1 (see figure 1), S^I = {9, 11, 18, 27, 35, 42}. The sequence related to Example 2 (see figure 2) is: S^r = {5, 14, 22, 23, 30, 36}.

Starting from S^r, we build a sequence $\tilde{S}^r = \left\{\tilde{s}^r_1, \ldots, \tilde{s}^r_{m+n-1}\right\}$ with the elements of Θ^r by setting:

$$\tilde{s}^r_k = \theta^r_{i,j} = \theta^I_{r^{-1}(i),j} \quad \text{if } s^r_k = t^r_{i,j} \quad (4)$$

for k = 1, 2, ..., n+m-1.

More generally, we consider the set A^r of all the sequences $U^r = \{u_1^r, u_2^r, \ldots, u_{n+m-1}^r\}$ of elements of T^r, of length m+n-1, and satisfying the following properties:

if $U^r = \left\{u_1^r, \ldots, u_{m+n-1}^r\right\} \in A^r$, then:

a. $u_{m+n-1}^r = t_{n,m}^r$

b. If $u_k^r = t_{i,j}^r$ for $k > 1$, then:

$$u_{k-1}^r = \begin{cases} t_{i,j-1}^r \text{ if } i = 1 \text{ and } 1 < j \leq m \\ t_{i-1,j}^r \text{ if } 1 < i \leq n \text{ and } j = 1 \\ t_{i-1,j}^r \text{ or } t_{i,j-1}^r \text{ if } 1 < i \leq n \text{ and } 1 < j \leq m \end{cases} \quad (5)$$

Obviously, $S^r \in A^r$ and $u_1^r = t_{1,1}^r$.

We also consider the set $\tilde{A}^r$ of all the sequences $\tilde{U}^r = \{\tilde{u}_1^r, \tilde{u}_2^r, \ldots, \tilde{u}_{n+m-1}^r\}$ of elements of Θ^r derived from the sequences of A^r in the same way as $\tilde{S}^r$ has been derived from S^r (see relation (4)). Note that $S^r \in A^r$ leads to $\tilde{S}^r \in \tilde{A}^r$.

The following result allows us to call $\tilde{S}^r$ the critical sequence in $\tilde{A}^r$. Such a result appears in GRABOWSKI [5, 6]. Here we provide an alternate proof.

Result 1

The sum of the elements of $\tilde{S}^r$ is equal to $t_{n,m}^r$, the makespan corresponding to r. Furthermore, the sum of the elements of any sequence $\tilde{U}^r \in \tilde{A}^r$ is less than or equal to $t_{n,m}^r$. $\tilde{S}^r$ is referred hereafter as the critical sequence related to r.

Proof

The proof is by induction.

a. The result is obvious if $n \geq 1$ and $m = 1$, or if $n = 1$ and $m \geq 1$: in both cases, $\tilde{A}^r$ contains only one sequence, and the sum of the elements of this sequence is equal to $t_{n,m}^r$.

b. Let us now consider the case with $n > 1$ and $m > 1$, and assume that the result holds for every (n-1, m) and (n, m-1) matrix. We denote by $\Delta_{n-1,m}^r$ the sequence obtained by starting from the restriction of Θ^r to its first n-1 rows in the same manner as $\tilde{S}^r$ was obtained from Θ^r. We also denote by $\Delta_{n,m-1}^r$ the sequence obtained by starting from the restriction of Θ^r to its first m-1 columns in the same manner as $\tilde{S}^r$ was

obtained from Θ^r. Then $\tilde{S}^r$ is either $\Delta^r_{n-1,m} \circ \left\{\theta_{r-1(n),m}\right\}$ or $\Delta^r_{n,m-1} \circ \left\{\theta_{r-1(n),m}\right\}$, where o represents the concatenation operation. This property is also true for every sequence of $\tilde{A}^r$. Considering our assumption and relation (1), the property holds for dimensions (n, m).

This completes the proof. ❑

We now illustrate this result using Example 1.

Example 1 continued. In this example S^1 = {9, 11, 18, 27, 35, 42} and $\tilde{S}^1$ = {9, 2, 7, 9, 8, 7}, We observe that the sum of the elements of $\tilde{S}^1$ is equal to 42. We can also verify that, for each sequence belonging to $\tilde{A}^1$, the sum of the elements is less than 42. It is, for instance, the case of the following sequences:

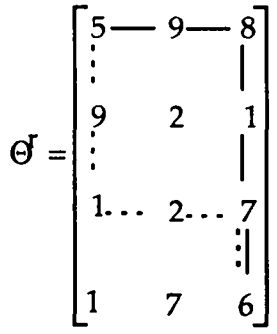
{9, 2, 1, 6, 8, 7} leads to the sum of 33, and

{9, 1, 7, 6, 8, 7} leads to the sum of 38.

2.4. The algorithm

Any element of $\tilde{A}^r$ can be represented by a path in Θ^r. Such a path starts from $\theta^r_{1,1}$, ends with $\theta^r_{n,m}$ and, if we consider one element (i, j) of the path, the next one (if any) is either the next on the right (i.e. (i, j+1)) or just below (i.e. (i+1,j)). Let $W(\tilde{U}^r)$ be the path related to $\tilde{U}^r \in \tilde{A}^r$. The path related to the critical sequence is hereafter referred to as the critical path.

Example 2 continued. Let us consider Example 2. In figure 3, we represent $W(\tilde{U}^r_1)$ corresponding to $\tilde{U}^r_1 = \{5,9,8,1,7,6\}$, and $W(\tilde{U}^r_2)$ corresponding to $\tilde{U}^r_2 = \{5,9,1,2,7,6\}$.



— $W(\tilde{U}^r_1)$ ··· $W(\tilde{U}^r_2)$

Fig. 3: Two paths of $\tilde{A}^r$ in Θ^r

Note that the sum of the elements of $\tilde{U}_1^r$ (also called the length of $W(\tilde{U}_1^r)$) is the makespan corresponding to the order r (i.e. $\tilde{U}_1^r$ is a dominant sequence in $\tilde{A}^r$).

Note also that there are (4! - 1 = 23) orders different from r.

However, according to result 1, we know that the makespan may be reduced only with orders which reduce the length of $W(\tilde{U}_1^r)$. In the previous example, every order r' derived from r by permutating rows {9,2,1}, {1,2,7} and {1,7,6} does not decrease the makespan. Thus, only (4! - 3! - 1) = 17) new orders may reduce the makespan.

In the algorithm proposed hereafter, we only permute pairs of rows. We thus reach a local optimum to the problem at hand. In the previous example, we would only try to permute rows 1 and 2 (which would reduce the critical path by 3), 1 and 3 (which would reduce the critical path by 11), and 1 and 4 (which would reduce the critical path by 6).

Path-Based Algorithm (PBA)

The algorithm can be described as follows:

1. Choose an initial order r and compute the corresponding makespan x.

The initial order may either be chosen at random, or be the order obtained by applying the CDS (CAMPBELL, DUDEK and SMITH, see [2]) algorithm or the PHD (Procédure HAN et DEJAX - HAN and DEJAX Procedure, see [7]).

2. Choose the pair of rows whose permutation leads to the maximal decrease in the length of the previous critical path. Let r^1 be the order derived from r by permuting these two rows.

For instance, if we consider figure 3, we can see that the permutation of rows 1 and 3 would lead to a decrease of 11 in the length of the critical path, which is greater than the decrease of 3 obtained by permutating rows 1 and 2, and the decrease of 6 obtained by permutating row 1 and 4.

3. Test

   3.1. If r^1 does not exist, then r is the (local) optimal solution. This ends the procedure.

   3.2. Otherwise:

      3.2.1. Compute x^1, the makespan corresponding to r^1.

      3.2.2. If $x^1 \geq x$, return to step 2 after removing from the list of feasible permutations the pair of rows which have been just tested, and continue.

      3.2.3. If $x^1 < x$, set $r = r^1$, set $x = x^1$, return to step 2 and continue.

Hereafter, this algorithm is referred to as Path Based Algorithm (PBA for short).

Example 2 continued. Let us consider Θ^r (see figure 3). The corresponding makespan has been computed in section 2.2 (see figure 2) and is equal to 36. Let us permute rows 1 and 3 which leads to the maximal decrease in the length of the critical path for r. We denote by r_1 the new order. Θ^{r_1}, T^{r_1}, and the critical path are given in figure 4.

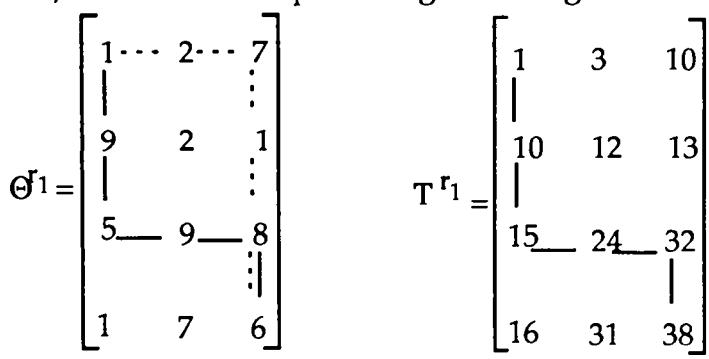


Fig. 4: First trial for changing order r

As we can see, the length of the critical path for r (see the dotted line in figure 4) decreases by 11, but a new critical path has appeared (see the continuous line), and the makespan is finally worse than the previous one.

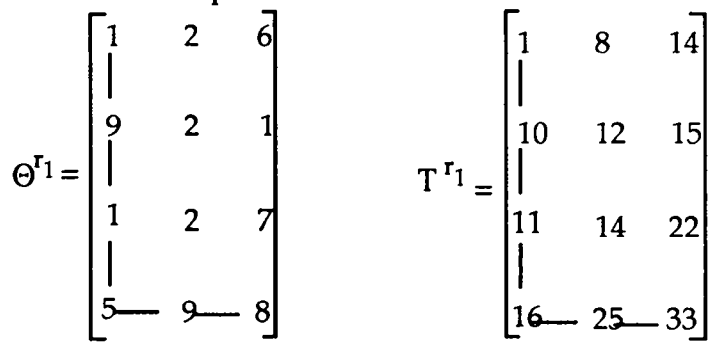


Fig. 5: Second trial for changing order r

We next consider r_1 obtained by permutating rows 1 and 4. It leads to Θ^{r_1} and T^{r_1} represented figure 5, along with a corresponding critical path.

This second trial reduces the makespan. Thus, we set $r = r_1$, and we try to permute two rows in order to reduce the length of the new critical path, and so on.

Note that the algorithm is based on result 1 which states that if we change the current order r, the makespan corresponding to the new order cannot be less than the current one if it does not result in decreasing the length of $W(\tilde{S}^r)$. This property allows us to avoid many computations of the makespan. In particular, we never try to permute two rows having only one element each on the critical path if these elements belong to the same column.

It is possible to improve the algorithm described above as explained in the next sub-section.

2.5. Improvement of the algorithm

In this sub-section, we show how to avoid some computations of makespans by introducing a minimal bound for the makespan associated with a given order.

$j^* \in \{1,2,...,m\}$ is said to be the slowest machine if:

$$z^* = \sum_{i=1}^{n} \theta^I_{i,j^*} = \underset{j\in\{1,...,m\}}{\text{Max}} \sum_{i=1}^{n} \theta^I_{i,j} \tag{6}$$

If various values of j realize the maximum, then choose anyone of them.

For a given order r, we also define:

$$v^*_r = \sum_{j=1}^{j^*-1} \theta^r_{1,j} \; ; \; v^*_r = 0 \text{ if } j^* = 1$$

$$w^*_r = \sum_{j=j^*+1}^{m} \theta^r_{n,j} \; ; \; w^*_r = 0 \text{ if } j^* = m$$

The following result holds:

Result 2

The makespan corresponding to r is greater than or equal to:

$$z^* + v^*_r + w^*_r = \tau^r \tag{7}$$

Proof

Note that v^*_r is the time required by the first job to reach the slowest machine, z^* is the total time required by all the jobs to pass through the slowest machine if they were already waiting for this machine, and w^*_r is the minimal time required by the last jobs to pass through all the machines which follow the slowest one. The sum of these values is obviously the lower bound of the makespan.

Finally, the algorithm given in section 2.4 is modified as follows:

Replace 3.2.1 by:

3.2.1. Compute τ^{r1}

- 3.2.1.1. If $\tau^{r1} \geq x$, return to step 2 and continue.
- 3.2.1.2. If $\tau^{r1} < x$, compute x^1, the makespan corresponding to r^1.

2.6. Minimizing the required number of pallets

Having found a good order, we propose a method to minimize the maximal in-process inventory, which is related to the number of transportation facilities like carts, pallets, etc.

In what follows, therefore, we provide a procedure to minimize the required number of pallets for a given makespan.

Let r be the order that we have found. Starting from it, equation (1) leads to the makespan and the corresponding schedule. Matrix T^r gives the completion times of each job on each machine. In this schedule, the beginning times of the jobs are lower bounded by 0 and each job starts as soon as possible.

The jobs whose completion times belong to S^r can neither be started earlier in time (because of the lower bound on the beginning times) nor be started later in time without increasing the makespan. These jobs are called the "fixed" jobs. The completion times of these jobs cannot be reduced. However, in some cases, the beginning times of the jobs which precede fixed jobs can be delayed without changing the makespan.

Starting from T^r, we build the matrix $T^{*r} = \left[t^{*r}_{i,j}\right]$; i = 1,...,n; j = 1,...,m as follows:

$t^{*r}_{i,j} = t^{r}_{i,j}$ for any pair (i,j) such that there exists a pair (s,j) with $s \geq i$ and $t^{r}_{s,j} \in S^r$

$t^{*r}_{n,j} = t^{*r}_{n,j+1} - \theta^{r}_{n,j+1}$ for any j such that there exists a pair (s,j) with $s < n$

and $t^{r}_{s,j} \in S^r$

$t^{*r}_{i,j} = \text{Min}\left\{t^{*r}_{i,j+1} - \theta^{r}_{i,j+1}, t^{*r}_{i+1,j} - \theta^{r}_{i+1,j}\right\}$ for any pair (i,j), $i < n$, such that there

exists a pair (s,j) with $s < i$ and $t^{r}_{s,j} \in S^r$ (8)

In other words, the elements of T^r remain unchanged in T^{*r} if they are on the critical path or above it.

The values given by (8) are the latest completion times of the jobs without changing the makespan, taking into account the fact that the first job begins at time 0. These values are the same as the corresponding values of T^r for each completion time of a fixed job or of a job which follows a fixed job.

Let b_k, k = 1,...,n, be the latest beginning times of the job of rank k:

$b_k = t^{*r}_{k,1} - \theta^{r}_{k,1}$

b_k is the instant job of rank k starts on the first machine.

For k = 1,...,n, we compute a(k) which is the largest integer belonging to {k,...n} and which satisfies:

$t^{*r}_{k,m} > b_{a(k)}$ (9)

a(k) is the rank of the last job which starts before the job of rank k ends.

We set:

$g(k) = a(k) - k + 1$ (10)

g(k) is the number of jobs in process just at the time the job of rank k ends.

Then:

$$g^* = \max_{k\in\{1,\dots,n\}} g(k) \qquad (11)$$

is the minimum required number of pallets for order r without affecting its makespan.

Consider the example specified by:

n = 6; m = 4; r(i) = i for i = 1,...,n

The processing times matrix is as follows:

$$\Theta^J = \begin{bmatrix} 2 & 8 & 3 & 2 \\ 4 & 2 & 5 & 3 \\ 3 & 9 & 6 & 9 \\ 6 & 1 & 7 & 6 \\ 2 & 4 & 1 & 7 \\ 3 & 2 & 4 & 4 \end{bmatrix}$$

The matrix T^r is computed by the method in section 2.2. We obtain:

$$T^r = \begin{bmatrix} 2 & 10 & 13 & 15 \\ 6 & 12 & 18 & 21 \\ 9 & 21 & 27 & 36 \\ 15 & 22 & 34 & 42 \\ 17 & 26 & 35 & 49 \\ 20 & 28 & 39 & 53 \end{bmatrix}$$

The critical circuit is represented in both matrices.

We can obtain $[T^{r*}]$ by using (8).

$t^{*r}_{6,3} = 53-4 = 49$; $t^{*r}_{6,2} = 49-4 = 45$; $t^{*r}_{6,1} = 45-2 = 43$;

$t^{*r}_{5,3} = \mathrm{Min}\ \{49-7, 49-4\} = 42$; $t^{*r}_{5,2} = \mathrm{Min}\ \{45-2, 42-1\} = 41$;

$t^{*r}_{5,1} = \mathrm{Min}\ \{43-3, 41-4\} = 37$; $t^{*r}_{4,3} = \mathrm{Min}\ \{42-6, 42-1\} = 36$;

$t^{*r}_{4,2} = \mathrm{Min}\ \{41-4, 36-7\} = 29$; $t^{*r}_{4,1} = \mathrm{Min}\ \{37-2, 29-1\} = 28$;

$t^{*r}_{3,1} = \mathrm{Min}\ \{21-9, 28-6\} = 12$; $t^{*r}_{2,1} = \mathrm{Min}\ \{12-2, 12-3\} = 9$

The other elements of T^{r*} are equal to the related element of T^r.

$$T^{r*} = \begin{bmatrix} 2 & 10 & 13 & 15 \\ 9 & 12 & 18 & 21 \\ 12 & 21 & 27 & 36 \\ 28 & 29 & 36 & 42 \\ 37 & 41 & 42 & 49 \\ 43 & 45 & 49 & 53 \end{bmatrix}$$

Finally,

$b_1 = 0 ; b_2 = 5 ; b_3 = 9 ; b_4 = 22 ; b_5 = 35 ; b_6 = 40$

Starting from (9),

$a(1) = 3 ; a(2) = 3 ; a(3) = 5 ; a(4) = 6 ; a(5) = 6 ; a(6) = 6$

The maximal in-process inventory is then given by:

$g^* = \text{Max } \{3,2,3,3,2,1\} = 3$

Remark

The required number of pallets tends to increase with decreasing makespan. But it is only a tendency, and not a rule, as it can be seen in the following counter-example.

Let us consider:

$$\Theta^I = \begin{bmatrix} 1 & 1 & 2 \\ 10 & 10 & 100 \\ 100 & 100 & 10 \end{bmatrix}$$

With the order I = {1,2,3}, we obtain:

$$T^I = \begin{bmatrix} 1 & 2 & 4 \\ 11 & 21 & 121 \\ 111 & 211 & 221 \end{bmatrix}$$

and the minimum number of required pallets is 2 with a makespan of 221.

We now consider r = {2,3,1}. It gives:

$$T^r = \begin{bmatrix} 10 & 20 & 120 \\ 110 & 210 & 220 \\ 111 & 211 & 222 \end{bmatrix}$$

for which the minimum number of required pallets is 3 with the makespan of 222.

3. COMPUTATIONAL RESULTS

Table 1: Significant results per matrix size

Size		Average value of the Makespan			Computation times (ms)			Average number of pallets			Improvement compared to PBA (in %)	
n	m	PHD	CDS	PBA	PHD	CDS	PBA	PHD	CDS	PBA	PHD	CDS
20	5	1353	1325	1247	29	22	293	7.95	7.95	8.15	7.84	5.89
40	5	2440	2394	2292	67	50	1427	12.25	12.25	12.40	6.06	4.26
60	5	3500	3459	3314	101	74	3738	16.15	16.15	16.75	5.34	4.23
80	5	4586	4505	4370	146	107	6614	20.50	20.50	20.75	4.70	2.98
100	5	5643	5595	5444	182	142	12235	26.10	26.10	24.75	3.51	2.69
20	10	1752	1717	1623	62	50	580	11.20	11.20	11.95	7.35	5.49
40	10	2913	2884	2652	134	114	4554	15.15	15.15	16.90	8.97	8.04
60	10	4047	4022	3733	216	183	13794	18.95	18.95	20.55	7.74	7.17
80	10	5112	5098	4730	304	257	31109	22.35	22.35	24.50	7.47	7.21
100	10	6210	6193	5813	394	329	46981	25.85	25.85	27.80	6.39	6.14
20	15	2126	2086	1967	99	83	842	13.25	13.25	14.65	7.49	5.71
40	15	3358	3301	3058	22	192	7451	18.60	18.60	20.40	8.95	7.37
60	15	4523	4445	4109	352	306	26666	22.10	22.10	24.45	9.14	7.56
80	15	5672	5592	5166	490	424	57554	27.45	27.45	28.40	8.92	7.62
100	15	6781	6729	6189	639	550	110638	27.40	27.40	31.90	8.73	8.03
20	20	2492	2455	2324	149	123	1083	15.75	15.75	17.05	6.74	5.35
40	20	3784	3715	3459	321	271	10325	22.20	22.20	24.20	8.57	6.89
60	20	4979	4906	4530	507	434	39497	26.15	26.15	29.30	9.02	7.66
80	20	6170	6082	5582	702	606	93606	29.75	29.75	32.85	9.53	8.23
100	20	7333	7233	6650	910	784	176010	32.95	32.95	36.55	9.32	8.07

The processing times (i.e. the elements of matrices Θ) have been generated randomly between 0 and 100. The matrices considered in this section are such that $n \in \{20, 40, 60, 80, 100\}$ and $m \in \{5, 10, 15, 20\}$. Recall that n and m are respectively the number of rows (i.e. the number of jobs) and the number of columns of Θ (i.e. the number of machines). Thus, $5 \times 4 = 20$ different sizes of matrices are used hereafter. For each size, 20 matrices have been generated, which means that the following results concern $20 \times 20 = 400$ Θ matrices.

For each example (i.e. for each matrix Θ), the initial solution is the best one among the solutions obtained by applying the CDS (see [2]) and the PHD (see [7]) procedures.

In table 1, we present significant average values for each matrix size.

These computations have been performed on a SUN SPARC 1+ workstation. An important remark must be made concerning the computational results presented in this section: because the elements of the Θ matrices have been generated at random, the makespan values for the n! feasible schedules are only slightly different from each other. It is then particularly difficult to find a solution significantly better than any solution generated at random and, as a consequence, it is even more difficult to find a better solution than the ones provided by the classical heuristic algorithms, namely PHD and CDS. We have to keep in mind that the results presented in table 1 have been obtained in an unfavorable context. Even so, the improvement is between 3% and 10% on the average; the better results being obtained, still on the average, for the matrices having larger sizes.

Furthermore, we have to outline that the heuristic algorithm presented in this paper needs more computation time than the PHD and CDS algorithms.

We also observe that a production line managed using a PBA would require more pallets or/and more transportation resources than if a PHD or a CDS policy is used. The explanation is that a reduction of the value of the makespan results in a higher work-in-process (WIP) which, in turn, results in increasing the transportation resources.

Recently, a new algorithm has been proposed in [8]. Simulation experiments presented by the authors show that the results obtained using this algorithm are never improved by more than 2% applying CDS, while the improvement obtained by using PBA is always greater than 3%.

5. CONCLUSION

In this paper, we have developed heuristic procedures to minimize makespan in flow-shop scheduling problems.

Our computational results show that, even in the unfavourable way chosen to generate the examples used in this paper, the improvement in relation to the results obtained by applying the most powerful existing algorithms, is significant.

Also, we have developed a procedure that computes the minimal number of required pallets required for a given makespan. This number is greater when using our PBA (Path-Based Algorithm) than when using PHD or CDS.

BIBLIOGRAPHY

[1] BAKER K.R., 1975
A comparative study of flowshop algorithms, *Ops. Res., 23, pp. 62-73.*

[2] CAMPBELL H.G., DUDEK R.A. and SMITH M.L., 1970
A heuristic algorithm for the n-job, m-machine sequencing problem, *Man. Sc., Vol. 16,, No. 10, pp. 630-637.*

[3] FRENCH S., 1982
Sequencing and scheduling. An introduction to the mathematics of the job-shop, ELLIS MURWOOD LIMITED, JOHN WILEY AND SONS.

[4] GELDERS L.F. and SAMBANDAM N., 1978
Four simple heuristics for scheduling a flowshop, *Int. J. Prod. Res., 16, 3, pp. 221-231.*

[5] GRABOWSKI J., 1980
On two-machine scheduling with release and due dates to minimize maximum lateness, *Ops. Res., 17, pp. 133-154.*

[6] GRABOWSKI J., 1982
A new algorithm of solving the flow-shop problem, in G. FEICHTINGER and P. KALL (eds.), *Operations Research in Progress, D. REIDEL Publishing Co. Dordrecht, pp. 57-75.*

[7] HAN W., 1992
Algorithmes de résolution exacte et heuristique pour les problèmes d'ordonnancement en flowshop, Thèse de Doctorat, Ecole Centrale des Arts et Manufactures, Grande Voie des Vignes, 92295 Chatenay Malabry Cédex, FRANCE.

[8] HO J.C. and CHANG Y.L., 1991
A new heuristic for the n-job, M-machine flow-shop problem, *EJOR, Vol. 52, No. 2, May, pp. 194-202.*

[9] JOHNSON S.M., 1954
Optimal two -and three- stages production schedules with setup times included, *Nav. Res. Log. Quart., 1, No. 1, pp. 61-68.*

[10] KING J.R. and SPACHIS A.S., 1980
Heuristic for flowshop scheduling, *Int. J. Prod. Res., 18, 3, pp. 345-357.*

[11] LENSTRA J.K., RINNOOY KAN A.H.G. and BRUCKER P., 1977
Complexity of machine scheduling problems, *Annals of Discrete Mathematics, 1, pp. 343-362..*

[12] McMAHON G.B. and BURTON P.G., 1967
Flowshop scheduling with the branch and bound method, *Ops. Res., 15, pp. 473-481.*

[13] OGBU F.A. and SMITH D.K., 1990

The application of the simulated annealing algorithm to the solution of the n/m/Cmax flow shop problem, *Comp. and Ops. Res., 17, 3, pp. 243-253.*

[14] OSMAN I.H. and POTTS C.N., 1989
Simulated annealing for permutation flow-shop scheduling, *OMEGA, 17, pp. 551-557.*

[15] PALMER D.S., 1965
Sequencing jobs through a multi-stage process in the minimum total time - a quick method of obtaining a near optimum, *Ops. Res., 16, pp. 101-107.*

[16] PARK Y., PEGDEN C. and ENSCORE E., 1984
A survey and evaluation of static flowshop scheduling heuristics, *Int. J. Prod. Res., Vol. 22, No. 1, pp. 127-141.*

[17] RINNOOY KAN A.H.G., 1976
Machine scheduling problems: classification, complexity and computations, Nijhoff, The Hague.

[18] TAILLARD E., 1980
Some efficient heuristic methods for the flow shop sequencing problem, *Eur. J. of Op. Res., 47, pp. 67-74.*

[19] WIDMER M. and HERTZ A., 1989
A new heuristic method for the flow shop scheduling problem, *Eur. J. of Op. Res., 41, pp. 186-193.*

Unité de Recherche INRIA Lorraine
Technopôle de Nancy-Brabois - Campus Scientifique
615, rue du Jardin Botanique - B.P. 101 - 54602 VILLERS LES NANCY Cedex (France)
Antenne de Metz
Technopôle de Metz 2000 - Cescom - 4, rue Marconi - 57070 METZ (France)

Unité de Recherche INRIA Rennes IRISA, Campus Universitaire de Beaulieu 35042 RENNES Cedex (France)
Unité de Recherche INRIA Rhône-Alpes 46, avenue Félix Viallet - 38031 GRENOBLE Cedex (France)
Unité de Recherche INRIA Rocquencourt Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)
Unité de Recherche INRIA Sophia Antipolis 2004, route des Lucioles - B.P. 93 - 06902 SOPHIA ANTIPOLIS Cedex (France)

EDITEUR
INRIA - Domaine de Voluceau - Rocquencourt - B.P. 105 - 78153 LE CHESNAY Cedex (France)

ISSN 0249 - 6399