Scheduling the arrivals in a no-wait flowshop with fuzzy processing times

V. Bourgade and A. Oulamara

Laboratoire Leibniz – Institut Imag

BP 53, 38041 Grenoble Cedex 9, France, Tel. +33-4-76574700

Fax +33-4-76574602 e-mail: vladimir.bourgade@imag.fr

Abstract

In this paper, we study the problem of the respect of due dates in no-wait flowshop, when processing times are fuzzy. We work in a context of synchronous production between orderers and suppliers or subcontractors. Our concern is to maximize the smallest common launching date to a set of tasks, while respecting each of their due dates.

Keywords:

Fuzzy systems, scheduling, flowshop.

1 INTRODUCTION

Consider the no-wait flowshop problem P whose objective is to maximize the smallest common launching date of a finite set $\{J_1, J_2, \dots, J_n\}$ of n tasks, while respecting their due dates. Each task has to be performed without interruption on all the m machines $M_1, M_2, \dots, M_m$ in this order (flowshop) and each machine can only perform one task at a time. Immediately after one task is completed on one machine, it has to begin on the next machine (no-wait). Note that these assumptions imply that the order in which the tasks are performed on one machine is the same on all the other machines.

The non-negative processing times t_{ik} of tasks J_i on machines $M_k \in M$ vary in fuzzy intervals $\tilde{t}_{ik}$. Our concern is to find robust scheduling methods in order to face uncertainties.

1.1 Definitions and notations

In this paper, we will use the following definitions and notations:

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: [10.1007/978-0-387-35390-6_58](https://doi.org/10.1007/978-0-387-35390-6_58)

L. M. Camarinha-Matos et al. (eds.), *Intelligent Systems for Manufacturing*

© IFIP International Federation for Information Processing 1998

- π : a sequence (permutation) of n tasks
- d_i : the crisp due date of task i
- t_{ij} : the effective processing time of task i on machine j
- $\tilde{t}_{ij}$: the fuzzy interval in which t_{ij} varies
- $S(\tilde{t}_{ij}) : S(\tilde{t}_{ij}) = [t_{ij}^-, t_{ij}^+]$ the support of the fuzzy interval $\tilde{t}_{ij}$
- S_{ij} : the effective launching date of task i on machine j
- S_i : the effective launching date of task i on the first machine

Let $R_{min}(\pi)$ be the greatest launching date of the first task $J_{\pi(1)}$ of the sequence π . $R_{min}(\pi)$ is such that each task is completed before its due date.

Our objective is to find, among the set Π of all possible sequences π , one that has the maximal production date of launching

$$R_{min}^* = \max_{\pi \in \Pi} R_{min}(\pi).$$

1.2 The restrained problem

Consider the restrained problem which consists of optimal task placement for a given sequence π . It expresses in these terms: how to determine, for a given sequence, the launching dates of the tasks with the result that there is no discontinuity in the completion of each of them?

If tasks are ordered in the natural order $(1, 2, \dots, n)$, our restrained problem can be expressed in the following linear program form:

$$\text{Maximize} \quad R_{min} \quad (1)$$

$$\text{s.t.} \quad R_{min} \leq S_{i1}, \forall i = 1..n \quad (2)$$

$$S_{ik} + t_{ik} \leq S_{(i+1)k}, \forall i = 1..(n-1), \forall k = 1..m \quad (3)$$

$$S_{ik} + t_{ik} = S_{i(k+1)}, \forall i = 1..n, \forall k = 1..(m-1) \quad (4)$$

$$S_{im} + t_{im} \leq d_i, \forall i = 1..n \quad (5)$$

$$t_{ik} \in S(\tilde{t}_{ik}), \forall i = 1..n, \forall k = 1..m \quad (6)$$

The constraint (2) implies that R_{min} is the maximal launching date for the sequence π . The constraint (3) expresses the fact that one machine can only perform one task at a time. The constraint (4) requires that the processing of each task is continuous. The constraint (5) means that each task must be processed before its due date, when the processing times t_{ik} vary in the support of the fuzzy interval $\tilde{t}_{ik}$, irrespective of their degree of possibility (constraint (6)).

This linear program allows us to determine recursively the launching dates

in reverse order of the tasks of the sequence (Bourgade 1998). Thus,

$$S_n = d_n - \sum_{k=1}^m t_{nk}^+ \quad (7)$$

$$S_{i-1} = \min \left\{ \begin{array}{l} d_{(i-1)} - \sum_{k=1}^m t_{(i-1)k}^+, \\ S_i - \max_{k=1}^m \left[\sum_{l=1}^k t_{(i-1)l}^+ - \sum_{l=1}^{k-1} t_{il}^- \right] \end{array} \right\} \quad (8)$$

Note: For the restrained problem, when the due dates are crisps, the optimistic (t_{ik}^-) and pessimistic (t_{ik}^+) processing times, constitute the only sufficient and necessary information to calculate the task schedule.

2 APPROACH OF RESOLUTION

The problem P is equivalent to that of the minimization of the maximal pessimistic lateness in no-wait flowshops (Bourgade 1998). In 1984, Rock (Röck 1984) proved that the latter was strongly NP-hard, when the processing times are crisps and when the workshop is composed of at least two machines. Thus, complexity of P remains the same.

As the problem P is strongly NP-hard, we have opted for a heuristic resolution approach. Consequently, we have elaborated three algorithms H_1 , H_2 and H_3 whose performances we test with respect to a taboo search heuristic (Glover 1989).

2.1 Principle of the heuristics

The heuristics H_1 , H_2 and H_3 are all based on the utilization of distinct preference relationships R_1 , R_2 and R_3 , and on a common task insertion method. More precisely, each heuristic H_i , $i = 1..3$ is made up of with the following steps:

Step 1: Create an ordered list, L , following the order imposed by R_i .

Step 2: Let π' be a partial schedule with $n' := 0$ initial tasks.

So that list L is not empty:

- Remove the first task J of L .

Insert J to the p^{th} position, $1 \leq p \leq n' + 1$, of π' beginning with the first position, retain the first sequence so that the value of $R_{min}(\pi')$ is the greatest.

- Update $n' := n' + 1$.

Step 3: Stop.

The complexity of the heuristic is $O(n^3m)$. Indeed, at the k^{th} iteration of step 2, for each of the k possible positions for the task to insert, we calculate values of the launching date for the k tasks of the partial sequence. As calculation of the launching date is in $O(m)$, the complexity of each of these iterations is in $O(k^2m)$, and that of step 2 is in $O(n^3m)$. Since the complexity of the sorting methods used in step 1 is at most in $O(n^2)$, we deduce the complexity of the heuristic.

Now, we shall clarify relationships of preference that will be used to create the ordered list L .

2.2 Relationship of preference R_1

We assume that task J_i is preferred to task J_j ($J_i R_1 J_j$) if its due date d_i is greater than the due date d_j of J_j . In this case, the relationship of preference R_1 induces a total order on the tasks. This order is such that tasks will be ordered following the decreasing order of their due dates.

2.3 Relationship of preference R_2

Let π' be a partial schedule and let $q(\pi', m)$ be the launching date of the first task of π' on the machine M_m . When a task J_i is scheduled directly before the first task of π' , the result is a partial schedule $J_i\pi'$. The greatest upper bound of the launching of $J_i\pi'$ is given by:

$$BS(J_i\pi') = \min\{d_i, q(\pi', m)\} - \sum_{k=1}^m t_{ik}^+.$$

We propose the following relationship of preference R :

$$J_i\pi' R J_j\pi' \text{ if } BS(J_i\pi') \geq BS(J_j\pi').$$

It induces the following relationship of preference R' :

$$J_i R' J_j \text{ if } \max\{d_i - d_j, 0\} - \sum_{k=1}^m t_{ik}^+ \geq - \sum_{k=1}^m t_{jk}^+.$$

In order to construct a relationship R_2 inducing a total order, we combine relationships R' and R_1 as follows:

$$J_i R_2 J_j \text{ si } (J_i R' J_j \wedge \overline{J_j R' J_i}) \vee (J_i R' J_j \wedge J_j R' J_i \wedge J_i R_1 J_j).$$

Thus, task J_i will be ordered before task J_j ($J_i R_2 J_j$) if $J_i R' J_j$ or if, in case of indecision, relationship R_1 is verified.

2.4 Relationship of preference R_3

The relationship of preference R_3 that we define below, will allow us to take into account the growing tendency of the processing times of the tasks following their order of passage on machines M_1 to M_m .

In his works on the two-machine flowshop scheduling problem, Johnson (Johnson 1954) has shown that tasks with growing processing time can be scheduled before those with decreasing processing time, when the objective is to minimize total production duration.

Given that in our problem processing time t_{ik} of tasks J_i on machines M_k varies in fuzzy intervals $\tilde{t}_{ik}$, the notion of growth becomes more difficult. Therefore, we use an average growth index:

$$P_F(J_i) = \sum_{k=1}^m (m+1-2.k) (t_{ik}^- + t_{ik}^+)/2.$$

Thus, if $P_F(J_i) \leq 0$, the task J_i will be said to be moderately growing. In the opposite case it will be said to be moderately decreasing.

Let $A = \{J_i / P_F \leq 0 \ i = 1..n\}$ be the set of moderately growing tasks and let $B = \{J_i / P_F > 0 \ i = 1..n\}$ be the set of moderately decreasing tasks.

Consider the following relationships of equivalence R_A , R_B and R_{BA} :

- $J_i R_A J_j$, si $J_i, J_j \in A$
- $J_i R_B J_j$, si $J_i, J_j \in B$
- $J_i R_{BA} J_j$, si $J_i \in B$ et $J_j \in A$

We define the relationship of preference R_3 as indicated below:

$$J_i R_3 J_j \text{ si } (J_i R_A J_j \wedge J_i R_2 J_j) \vee (J_i R_B J_j \wedge J_j R_2 J_i) \vee (J_i R_{BA} J_j).$$

The relationship of preference R_3 induces a total order on the set of tasks. This order is such that tasks of A will be scheduled in the order induced by R_2 , after those of B that will be ordered in the reverse order induced by R_2 .

3 NUMERICAL EXPERIENCES

We have generated 20 random instances for problems of different sizes (nxm), with $n \in \{10, 20, 30\}$ and $m = 10$.

For each instance, the fuzzy processing times of each task on all machines, as well as their due dates, have been generated according to the uniform law.

More precisely, we have:

$$\begin{aligned} t_{ik}^- &:= U(\inf, \sup), \text{ avec } i = 1..n \text{ et } k = 1..m \\ t_{ik}^+ &:= t_{ik}^- + U(\inf, \sup), \text{ avec } i = 1..n \text{ et } k = 1..m \end{aligned}$$

$d_i := U(2.m.sup, m.n.sup.ratio)$, avec $i = 1..n$ et $ratio \in [2/n, 1]$

where $U(inf, sup)$ is a determined random number according to the uniform law on a real interval $]inf, sup[$. In our tests we have taken $inf = 5$ and $sup = 10$.

The rates defined for the tests will allow us to control the amplitude of intervals in which the task due dates vary. All increase of the amplitude is in pair with that of the rate, and is accompanied by a greatest chance of dispersion of the due dates of tasks. Thus, we will be able to attempt to characterize the influence of distribution of these due dates on the tested heuristic.

For each instance of the problem, we have taken as an initial solution for the taboo search the algorithm, the best solution provided by the heuristics. So as to compare the performance of the heuristic, we have evaluated, according to the size of instances, average distances, D_{mo}^i , and maximal distances, D_{ma}^i that separate solutions of the heuristic H_i from the solution obtained by the taboo search technique. The distance between a solution π_i provided by the heuristic H_i , $i = 1..3$, and the improved solution π^* by the taboo search technique is given by:

$$D_i = \frac{R_{min}(\pi^*) - R_{min}(\pi_i)}{R_{min}(\pi^*)}.100$$

We have also compared the heuristic H_i , $i = 1..3$, by calculating, according to the size of instances, the number of times, f_i , when each of them provides the best result.

3.1 Analysis of results

Results of the tests are recorded in tables 1. These tests were performed on a *PENTIUM* 133 equipped computer.

An average performance analysis of the tests informs us that more the dispersion of due dates increases, the more it is important to schedule tasks according to the growing order of their due dates. Indeed, when the value of the rate tends to 1, the heuristic H_2 very often provides better results than H_1 and H_3 . However, probably H_1 is better than H_2 and H_3 if the dispersion of the due dates of tasks is more significant.

Conversely, the less the dispersion of due dates, the better it is to schedule tasks according to their average growth indexes, in the order induced by the relationship of preference R_3 , and incidentally by the heuristic H_3 . It is the observation that one can make with the results obtained when the value of the rate is equal to 0,25. In this case, we can observe that, whatever the generated problem size, the heuristic H_3 is most of the time better than the two others

regarding the quality of solutions that it provides. This fact appears all the more clearly when the value of the rate tends to $2/n$.

<i>Ratio</i>	<i>n x m</i>	D_{mo}^1	D_{ma}^1	f^1	D_{mo}^2	D_{ma}^2	f^2	D_{mo}^3	D_{ma}^3	f^3
2/n	10 x 10	0.06	0.52	11	0.05	0.440	17	0.04	0.35	17
	20 x 10	0.06	0.23	13	0.03	0.11	15	0.01	0.09	17
	30 x 10	0.01	0.14	18	0.01	0.08	18	0.00	0.06	19
0.25	10 x 10	0.11	0.46	14	0.11	0.46	14	0.10	0.50	15
	20 x 10	0.18	1.17	15	0.08	1.11	18	0.10	0.93	15
	30 x 10	0.07	0.34	14	0.06	0.34	16	0.03	0.29	18
0.50	10 x 10	0.29	1.30	15	0.27	1.66	16	0.71	3.40	8
	20 x 10	0.15	1.09	15	0.09	0.52	18	0.60	2.51	4
	30 x 10	0.23	0.87	15	0.24	0.79	15	0.62	1.98	3
0.75	10 x 10	0.43	2.21	14	0.44	2.21	13	0.76	2.36	8
	20 x 10	0.17	1.24	15	0.13	1.30	16	0.26	1.88	13
	30 x 10	0.20	1.00	14	0.16	1.05	15	0.18	1.98	13
1.00	10 x 10	0.05	0.51	19	0.05	0.51	19	0.14	2.16	19
	20 x 10	0.03	0.53	18	0.02	0.53	19	0.05	0.59	17
	30 x 10	0.01	0.13	20	0.01	0.13	20	0.09	0.58	15

Table 1 *Results of the tests*

As we can see for the rate 0,5, the heuristic H_2 provides better results than H_1 and H_3 , in most cases. This value of the rate characterizes cases where both the dispersion of due dates and processing times affect the results.

The heuristics H_1 , H_2 and H_3 that we have tested provide excellent results in view of the taboo search technique that we have computerized. Indeed, we have been able to observe that the improvement of the best result obtained with the heuristic by the taboo search procedure is very often inferior to 1,1 %. Furthermore, when we compare their speeds, the heuristics are far more rapid than the taboo search algorithm.

The CPU times of the heuristic are smaller than one second for instances whose size is smaller than (30 x 10), while those of the taboo search heuristic grow exponentially (see Table 2).

Unfortunately, it does not seem possible to characterize the influence of dispersion of due dates and processing times on heuristic results. We therefore suggest, for a given instance, to evaluate solutions with the help of the three heuristics and to take the best one. In the case where a better solution has to be found, we can use the taboo search heuristic.

Tailles	H_1	H_2	H_3	Tabu
10 x 10	0.02	0.05	0.06	121.46
20 x 10	0.27	0.27	0.28	390.76
30 x 10	0.98	0.99	0.99	725.75

Table 2 *Maximum CPU times observed (in seconde)*

4 CONCLUSION

We have proposed three heuristic approaches that we then compared to a taboo search heuristic. The performance of these heuristics has been evaluated and discussed. All three seem to produce good results relating to the level of solutions that CPU time takes to calculate them.

The results obtained show great potential for the utilization of these heuristics in real time in order to manage the arrival of orders in the workshop. This constitutes a means of useful and promising applied research.

REFERENCES

- Bourgade, V. (1998) *Sur l'ordonnancement d'ateliers en ligne en environnement incertain et imprécis*. Ph.D. Thesis, Institut National Polytechnique de Grenoble.
- Glover, F. (1989) Tabu search - Part I. *ORSA Journal on Computing*, **1**, 190-206.
- Johnson, S. (1954) Optimal two and three-stage production schedules with setup times included. *Naval Research Logistics Quarterly*, **1**, 61-68.
- Röck, H. (1984) Some new results in no-wait flowshop scheduling. *Zeitschrift für Opns. Res.*, **28**, 1-16.