A NEW CONSTRUCTIVE HEURISTIC METHOD FOR MINIMIZING MAKESPAN IN PERMUTATION FLOW SHOP SCHEDULING

Marcelo Seido Nagano
Faculdade de Economia, Administração e Contabilidade, Universidade de São Paulo
Av. dos Bandeirantes, 3900, CEP 14.040-900 Ribeirão Preto-SP
msnagano@sc.usp.br

João Vitor Moccellin
Escola de Engenharia de São Carlos, Universidade de São Paulo
Av. Dr. Carlos Botelho, 1465, CEP 13.560-250 São Carlos-SP
jvmoccel@prod.eesc.sc.usp.br

Abstract

This work is addressed to the Flow Shop Sequencing problem. As a result of an investigation on the problem characteristics it is presented a property of this scheduling problem, which is used for the development of a new constructive heuristic with the objective of minimizing the total time to complete the schedule (makespan). The new method, denoted by N&M, is compared with the best constructive heuristic reported in the literature, named NEH. Results from computational experience have shown that, for problems having up to 10 machines and 100 jobs, the new heuristic outperforms in solution quality the NEH one. There is no significant difference regarding computation effort for both NEH and N&M heuristics.

Keywords: production scheduling, flow shop sequencing, heuristics.

Resumo

Este trabalho é direcionado ao problema de Programação de Operações Flow-Shop Permutacional. Uma propriedade do problema, oriunda de estudo sobre a sua característica, é apresentada e utilizada para o desenvolvimento de um novo método heurístico construtivo, com o objetivo de minimizar a Duração Total da Programação (makespan). O novo método é comparado com o melhor heurístico construtivo reportado na literatura, conhecido por NEH. Os resultados da experimentação computacional efetuada mostram um melhor desempenho do método proposto,
denominado N&M, em comparação com o NEH, para problemas com até 10 máquinas e 100 tarefas, quanto à qualidade da solução, apresentando também eficiência computacional.

Palavras-chave: Programação da produção, programação flow-shop permutacional, métodos heurísticos.

1. Introduction

The general flow shop scheduling problem is a production problem where a set of \( n \) jobs have to be processed with identical flow pattern on \( m \) machines. When the sequence of job processing on all machines is the same we have the Flow Shop Sequencing production environment. Since there is no job passing, the number of possible schedules for \( n \) jobs is \( n! \). Usually the schedule performance measure is related to an efficient resource utilization looking for a job sequence that minimizes the makespan, that is the total time to complete the schedule.

This scheduling problem is generally modeled on the following assumptions: i) The operation processing times on the machines are known, fixed and some of them may be zero if some job is not processed on a machine; ii) Set-up times are included in the processing times and they are independent of the job position in the sequence of jobs; iii) At a time, every job is processed on only one machine, and every machine processes only one job; iv) The job operations on the machines may not be preempted.

A significant research effort has been devoted for sequencing jobs in a flow shop with the objective of finding a sequence that minimizes the makespan. For problems with 2 machines, or 3 machines under specific constraints on job processing times, the efficient JOHNSON's algorithm (1954) obtains an optimal solution for the problem. However, since this scheduling problem is NP-hard (GAREY et al., 1976) the search for an optimal solution is of more theoretical than practical importance. Having this in mind, since the 1960s a number of heuristic methods that provide near optimal or good solutions with limited computation effort have been proposed for flow shop sequencing.

The heuristic methods can be classified according to two major categories: constructive or improvement methods. The constructive one obtains directly a solution for the scheduling problem, i.e. a \( n \)-job sequence, by using some procedure which assigns to each job a priority index in order to
construct the solution sequence. An improvement method starts from a given initial solution, and looks for a better one usually by using some neighborhood search procedure.

In the last decades many constructive heuristics have been introduced with the objective of minimizing makespan in a permutation flow shop. For instance, PALMER (1965), GUPTA (1971), DANNENBRING (1977), NAWAZ, ENSCORE & HAM (1983), SEVAST’JANOV (1995), LOURENÇO (1996), and KOULAMAS (1998).

It is worth noting that the NEH heuristic is so far the best constructive one, in spite of those proposed by Sevast’janov, Lourenço, and Koulamas.

SEVAST’JANOV (1995) suggested a polynomial algorithm that yields a solution for the flow shop sequencing problem such that

$$C_{\text{max}} - C^*_{\text{max}} \leq \left( m^2 - 3m + 3 + \frac{1}{m-2} \right) \max_{i,j} (p_{ij}) \quad (1)$$

where:

$C_{\text{max}}$ = makespan of the solution sequence,

$C^*_{\text{max}}$ = minimal makespan for the flow shop sequencing problem,

$m$ = number of machines, and

$p_{ij}$ = processing time of job $i$ on machine $j$.

LOURENÇO (1996) implemented the algorithm of Sevast’janov by using linear programming. He has concluded that if the computation time is not a critical factor, the Sevast’janov’s algorithm is outperformed in solution quality by the NEH heuristic. However, if computation time is a restricted factor, the algorithm of Sevast’janov could be an useful alternative for large size problems concerning the number of jobs, specially when the flow shop has a relatively small number of machines.

KOULAMAS (1998) has proposed a constructive heuristic for the minimal makespan flow shop scheduling problem, whose principal feature is to be able of obtaining non-permutation schedules.
when it is expected that such kind of schedule can be optimal. However, for permutation scheduling the NEH heuristic still performs better.

Based on the literature examination we can highlight the following:

• Since the slope index heuristic of PALMER (1965) until the NEH one (1983) there was a regular and successful development of constructive heuristics for sequencing jobs in a flow shop in order to minimize the makespan;

• The last three aforementioned papers have reported interesting heuristic methods but all of them are not superior to the NEH heuristic for the permutation flow shop scheduling problem.

• It would not be advisable to conclude that the NEH superiority from 1983 so far is a consequence of an absence of interest for developing new constructive heuristics. Actually, the NEH is a powerful heuristic for flow shop sequencing.

According to previous experience concerning the NEH heuristic we could note that its principal feature is related to the initial arrangement for the jobs, from which the solution sequence is obtained by using an iterative job insertion procedure.

The heuristic procedure proposed by Nawaz, Enscore Jr. & Ham is based on the assumption that a job with more total processing time on all the machines should be given higher priority than a job with less total processing time. The algorithm can be stated as follows.

**Step 1.** For each job $v$ calculate

$$ P_v = \sum_{k=1}^{m} p_{kv} $$

where $p_{kv} =$ processing time of job $v$ on machine $k$, and

$$ m =$ number of machines.

**Step 2.** Arrange the jobs in descending order of $P_v$.

**Step 3.** Pick the two jobs from the first and second position of the list of Step 2, and find the best sequence for these two jobs by calculating makespan for the two possible sequences. Do not change the relative positions of these two jobs with respect to each other in the remaining steps of the algorithm. Set $i = 3$. 
Step 4. Pick the job in the $i$th position of the list generated in Step 2 and find the best sequence by placing it at all possible $i$ positions in the partial sequence found in the previous step, without changing the relative positions to each other of the already assigned jobs. The number of enumeration at this step equals $i$.

Step 5. If $n = i$, STOP, otherwise set $i = i + 1$ and go to Step 4.

The main objective of this paper is to introduce a new constructive heuristic for minimizing makespan in a flow shop production environment. In Section 2 we present a property concerning this scheduling problem, which is used for the development of the new heuristic that is presented in Section 3. In order to evaluate the proposed heuristic performance it was carried out a computational experience whose results are presented in Section 4, which finishes with some concluding remarks.

2. An useful property of the flow shop sequencing problem

For any $n$-job sequence $\sigma$, the makespan $M(\sigma)$ can be expressed by:

$$M(\sigma) = \sum_{j=1}^{n} p_{1,j} + \sum_{k=2}^{m} p_{k,n} + \sum_{k=1}^{m-1} Y_{n}^{k}$$

(2)

where

$p_{k,j}$ = processing time on machine $k$ of the job in the $j$th position of $\sigma$,

$Y_{j}^{k}$ = waiting time for the job in the $j$th position of $\sigma$, between the end of the operation on machine $k$ and the beginning on machine $(k+1)$. 
Based on an investigation made by NAGANO (1999), it can be stated the following:

Let $u$ and $v$ be any two jobs from the set of $n$ jobs. For an arbitrary sequence $\sigma$ with jobs $u$ and $v$ respectively scheduled in positions $j$ and $(j+1)$, $j = 1,2,...,n-1$, one has that

$$LBY_u^k = \max \left( 0, \left( p_{k+1,u} - p_{k,v} \right) - UBX_{u,v}^k \right) \quad (3)$$

where

$LBY_u^k$ = a LOWER BOUND for $Y_{u,v}^k$,

$Y_{u,v}^k$ = waiting time for the job $v$ between the end of its operation on machine $k$ and the beginning on machine $(k+1)$, when job $u$ immediately precedes job $v$,

$p_{k+1,u}$ = processing time of job $u$ on machine $(k+1)$,

$p_{k,v}$ = processing time of job $v$ on machine $k$,

$UBX_{u,v}^k = \max \left( 0, UBX_{u,v}^{k-1} + \left( p_{k-1,v} - p_{k,u} \right) \right)$,

$UBX_{u,v}^1 = 0$.

The above property, that we have named LBY, is illustrated in Figure 2.
3. The proposed constructive heuristic

The heuristic method we introduce in this paper, which is denoted by N&M heuristic, is similar to the NEH one. The significant difference from NEH is in the steps 1 and 2 concerning the initial arrangement for the jobs to be scheduled.

For the N&M heuristic such steps are:

*Step 1.* For each job $v$ calculate

$$I_v = \sum TP_v - \text{MAX} \sum LBY_{J_v}$$

where

$$\sum TP_v = \sum_{k=1}^{m} p_{k,v}$$

$$\text{MAX} \sum LBY_{J_v} = \max_{u=1}^{n} \left\{ \sum_{u \neq v}^{m-1} LBY_{u,v} \right\}$$

$p_{k,v}$ = processing time of job $v$ on machine $k$,

$n$ = number of jobs,

$m$ = number of machines, and
$\text{LBY}_k^{UV} = \text{the lower bound for } Y_k^{UV}, \text{ given by expression (3).}$

*Step 2.* Arrange the jobs in descending order of $I_v$.

The remaining *steps* are the same as those of NEH. Details regarding both the LBY property and the N&M heuristic can be found in NAGANO (1999).

## 4. Computational experience

The NEH and N&M heuristics have been evaluated on a total of 5700 problems with the number of machines $m \in \{4, 7, 10\}$ and the number of jobs $n \in \{10, 15, 20, 25, 30, 35, 40, 45, 50, 55, 60, 65, 70, 75, 80, 85, 90, 95, 100\}$. Each of the $m \times n$ combinations was replicated 100 times. The operation processing times were randomly generated integers uniformly distributed over the interval $[1, 10]$. In the computational tests the heuristics were coded in Turbo-Pascal and have been run on a microcomputer Pentium-S 166 Mhz.

The principal results from the computational experience are summarized in Tables 1 and 2.

Table 1 shows the *percentage of success* of each heuristic method. This percentage is defined as the total number of times the heuristic obtains the best makespan divided by the number of generated problems. Obviously, when the heuristics obtain the best makespan for the same problem their percentages of success are simultaneously improved. The results from Table 1 are illustrated in Figure 3.

Table 2 presents the *average computation times* in seconds, which are illustrated in Figure 4.
### Table 1 - percentage of success

<table>
<thead>
<tr>
<th>number of jobs</th>
<th>NEH</th>
<th>N&amp;M</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>83.00</td>
<td>84.33</td>
</tr>
<tr>
<td>15</td>
<td>70.66</td>
<td>74.66</td>
</tr>
<tr>
<td>20</td>
<td>66.33</td>
<td>78.67</td>
</tr>
<tr>
<td>25</td>
<td>65.33</td>
<td>71.00</td>
</tr>
<tr>
<td>30</td>
<td>66.33</td>
<td>69.33</td>
</tr>
<tr>
<td>35</td>
<td>63.33</td>
<td>73.00</td>
</tr>
<tr>
<td>40</td>
<td>66.66</td>
<td>73.66</td>
</tr>
<tr>
<td>45</td>
<td>63.33</td>
<td>74.67</td>
</tr>
<tr>
<td>50</td>
<td>68.33</td>
<td>72.33</td>
</tr>
<tr>
<td>55</td>
<td>68.33</td>
<td>70.33</td>
</tr>
<tr>
<td>60</td>
<td>66.99</td>
<td>71.66</td>
</tr>
<tr>
<td>65</td>
<td>69.33</td>
<td>74.00</td>
</tr>
<tr>
<td>70</td>
<td>69.00</td>
<td>72.33</td>
</tr>
<tr>
<td>75</td>
<td>69.66</td>
<td>70.66</td>
</tr>
<tr>
<td>80</td>
<td>67.66</td>
<td>72.99</td>
</tr>
<tr>
<td>85</td>
<td>68.33</td>
<td>71.00</td>
</tr>
<tr>
<td>90</td>
<td>69.66</td>
<td>71.99</td>
</tr>
<tr>
<td>95</td>
<td>72.67</td>
<td>74.33</td>
</tr>
<tr>
<td>100</td>
<td>65.66</td>
<td>74.66</td>
</tr>
<tr>
<td><strong>average</strong></td>
<td><strong>68.46</strong></td>
<td><strong>73.46</strong></td>
</tr>
</tbody>
</table>

### Figure 3 - percentage of success

![Figure 3 - percentage of success](image-url)
The results from Table 1 and Figure 3, related to percentages of success, show that the NEH heuristic is outperformed by the N&M heuristic for all problem classes according to the number of jobs. It is worthy to note that the differences between the percentages of success are not large: the
The greatest value is 12.34% for problems having 20 jobs, and the minimal difference is 1.0% for the problem class with 75 jobs. Such results could be expected because, as it is well-known, the NEH has been since 1983 the best constructive heuristic for sequencing jobs in a flow shop in order to minimize makespan. The general average percentage of success is 73.46% for N&M, and 68.46% for NEH.

The average computation times presented in Table 2 and Figure 4 substantiate an expected result. Obviously, the N&M computation time should be greater than the computation time for NEH as a consequence of the arithmetical process required for obtaining the job indexes in step 1. However, the differences between average computation times have not reached 0.1 seconds for the largest problems. Moreover, taking into account the computer we have used to run the heuristics, we can conclude that the computation effort for both NEH and N&M is practically the same.

References


NAGANO, M. S. (1999), Um Novo Método Heurístico Construtivo de Alta Performance para a Programação de Operações Flow-Shop Permutacional, Doctoral Thesis, School of Engineering of São Carlos, University of São Paulo, Brazil.

