5

Analyzing Different Production Times Applied to the Job Shop Scheduling Problem

Arthur Tórgo Gómez,
Antonio Gabriel Rodrigues and Rodrigo da Rosa Righi
Programa Interdisciplinar de Pós-Graduação em Computação Aplicada,
Universidade do Vale do Rio dos Sinos,
Brazil

1. Introduction

The classic objective of the Job Shop Scheduling Problem (JSSP) is to find a sequence of parts with minimal time to complete all parts (Nowicki and Smutnicki, 1996). The time spent to finalize all parts is known by makespan. In other words, the makespan is the total length of the schedule (when all the jobs have finished processing). Besides the makespan, other objectives can be considered, such as minimize the number of setups, the idle time of machines, the number of tool switches in a machining workstation and so on. The Scheduling Problem is considered hard to solve, with computational complexity defined as NP-Hard (Nowicki and Smutnicki, 1996). It can be applied in a variety of manufacturing systems, being specially studied in Flexible Manufacturing Systems (FMS) (Jha, 1991).

The objective of this study is to show the behavior of three different times in the context of Job Shop Scheduling Problem. The aforementioned times are: (i) tardiness time; (ii) setup time and; (iii) switching tool time. Objective functions were defined with this three production times, representing our decision variables. Cluster Analysis and Tabu Search (TS) Techniques are used to development the model.

The article is divided as follow. Section 2 introduces the concepts of JPSS and its mathematical formulation. Some resolutions methods are mentioned in Section 3. Moreover, Section 4 describes in details the metaheuristic denoted like Tabu Search. This section is responsible for presenting our proposal of a JSSP application. Considering this, this part of the text represents out main contribution regarding the JSSP field. Validation and some experimental results are showed in Section 5. Finally, Section 6 finalizes the chapter, emphasizing its main ideas and contributions.

2. The job shop scheduling problem

One of the classic problems of the area of combinatorial optimization is the Job Shop Scheduling Problem (JSSP), which is defined for minimizing the total production time of a specific system. Generally, JSSP is applied for a range of applications in manufacturing area.

Studied since the 60's, this problem is considered quite complex, and some of its instances waited about two decades to have reached the optimum result.

Since he JSSP comes from the area of Manufacturing, it is common to find variations that reflect different particularities of a production system. In addition, variations also occur when considering different objectives of the minimizing function. Basically, this function is applied to the total production time, but it can also be observed to minimize delivery times, as well as to minimize the number of stops of a machine. JSSP can be treated following two approaches: (i) stochastically and; (ii) deterministic mode. The former works with probability distributions for the arrival of requests for products and processing times, load and displacement within the manufacturing plant. The second approach assumes that the processing times of products are known in advance, as well as both the load times in the machines and the displacement within the factory.

Given the high complexity of JSSP, the accurate methods for solving the optimization combinatorial problems seem to be inefficient and computationally infeasible. This fact is explained because JSSP can demand an enormous amount of time and a high number of computational resources (such as memory and processing power). Therefore, studies involving heuristics and metaheuristics became more and more relevant in this context. Even without guaranteeing an optimal result, they present methods for getting good results with low computational costs. Currently, the JSSP problem serves as benchmark for new metaheuristics being studied by various fields such as engineering and computing. Classically, the Job Shop Scheduling Problem can be defined as a set of parts (or jobs), where each part has associated a set of operations to be performed. Furthermore, there is a set of machines that perform the operations of the aforementioned parts. Once an operation starts, it cannot be interrupted. A classical formulation of this problem is presented below (Blazewicz et al., 1996; Adams et al. 1988; Pezzella and Merelli, 2000):

$$\text{Minimize } tn \tag{1}$$

Subject:

$$tj - ti \geq pi \forall (i,j) \in A \tag{2}$$

$$t\,j - ti \geq pi \text{ or } ti - t\,j \geq pj \forall \{i,j\} \in E_k, \forall k \in M \tag{3}$$

$$ti \geq 0 \; \forall i \in V \tag{4}$$

Where, V= {0, 1,...,n} represents the set of operations, where "0" is the first operation and "n" will be the last operation for all jobs. The set of "m" machines is represented by "M" and "A" is the representation of the ordered pairs set of the constraints of operations by the precedence of the relation of each job. For each machine "k" the "Ek" set describes all the operation pairs given by the "k" machine. For each "i" operation, it is processed in a "pi" time (fixed) and the initial "i" process is denoted as "ti" , a variable that has been determinate during the optimization. The Job-shop objective function (1) is used to minimize the makespan. The constraint (2) assures that the sequence of the operation processing for each job corresponds to a pre-determinate order. The constraint (3) assures that there is only one job in each machine at a specific time,. Finally, constraint (4) assures completion of all jobs.

3. Resolutions methods

Many optimization methods have been proposed to the solution of Job Shop Scheduling Problem (Blazewicz et al., 1996) (Mascis and Pacciarelli, 2002)(Zoghby et al., 2004). They can be classified as optimization methods or approximation methods. Considering the optimization methods, we can mention the Integer Programming, Lagrangian Relaxation, Surrogate and Branch and Bound (Balas et al., 1979)(Fisher, 1976). On the other hand, iterative algorithms like Tabu Search, Neural Networks, Genetic Algorithms, Simulated Annealing and GRASP belong to an approach that works with approximation methods (Glover and Laguna, 1997)(Goncalves et al., 2005)(Jain and Meeran, 1998)(Hurink and Knust, 2004). Considering this scenario, Tabu Search is considered a good metaheuristic algorithm for treating problems with a high computational complexity, like JSSP one. Aiming to review some works regarding Tabu Search and others metaheuristics, we recommend some readings (Cordeau et al, 2002; Tarantilis et al, 2005).

3.1 Tabu search

Tabu Search (TS) was proposed by Glover (Glover, 1989) and had its concepts detailed by Glover and Laguna (Glover and Laguna, 1997). Tabu Search is a technique for solving optimization combinatorial problems that consists in iterative routines to construct neighborhoods emphasizing the prohibition of stopping in an optimum local. The main ideas of TS are: (i) It avoids to pass again by recently visited solution area and; (ii) It guides the search towards new and promising areas (Glover, 1986; Wu et al, 2009). Non-improving moves are allowed to escape from the local optima. Moreover, attributes of recently performed moves are stored in a tabu list and may be forbidden for a number of iterations to avoid cycling (Glover, 1986; Wu et al, 2009).

TS searches for the best solution by using an aggressive exploration (Glover and Laguna, 1997). This exploration chooses the best movement for each iteration, not depending on whether this movement improves or not the value of the current solution. In Tabu Search development, intensification and diversification strategies are alternated through the tabu attributes analysis. Diversification strategies drive the search to new regions, aiming to reach the whole search space. The intensification strategies reinforce the search in the neighborhood of a solution historically good (Glover and Laguna, 1997). The stop criterion may be applied to stop the search. It can be defined as the interaction where the best results were found or as the maximum number of iteration without an improvement in the value of the objective function. The tabu list is a structure that keeps some solution's attributes. The objective of this list is to forbid the use of some solutions during some defined time.

4. JSSP aplication

The proposed application considers the classical JSSP with due dates and tooling constraints (Hertz and Widmer, 1996). Each job has a due date in which all its operations shall be completed. Each operation requires a set of tools to processing. The problem is to minimize three production times, represented by decision variables in a objective function *f*. The production times are explained below.

- Makespan (*Ms*): the total time needed to complete the processing of all operations, considering production turns.
- Tardiness time (*At*): the positive difference between the date of completion and the due date of the part, expressed in minutes.
- Setup time (*Sp*): the time spent in preparation for processing new batches during the production of a set of parts, expressed in minutes. This time lasts α+βt, where α is the time to clean the work area; β is the time to replace one tool and t is the number of tools switched (Hertz and Widmer, 1996; Gómez, 1996).

The managing of the significance of these times is made through the definition of values for the weights of the objective function showed in equation 1.

Considering: d_j is the due date of the job j; x_{ko} is 1 if there is a setup after operation in the machine k, or 0 otherwise;

$$\text{Minimize} \quad W_1 Ms + W_2 At + W_3 St \tag{1}$$

Subject:

$$Ms = \max_{i \in O} \left(S_i + p_i \right) \tag{2}$$

$$At = \sum_{j=1}^{J} \max_{i \in O} (S_i + p_i) - d_j \quad \forall \max_{i \in O} (S_i + p_i) - d_j > 0 \tag{3}$$

$$St = \sum_{k=1}^{M} \sum_{o=1}^{M_k} x_{ko} (\alpha + \beta t) \tag{4}$$

$$Ms \geq 0,\ At \geq 0,\ St \geq 0, W_1 \geq 0, W_2 \geq 0, W_3 \geq 0. \tag{5}$$

The objective function is expressed by equation (1). Equation (2) represents the Makespan, *i. e.*, the total time to complete the last operation in the schedule. The equation (3) defines the tardiness time, as the sum of differences between the predefined due date (in minutes) and the part completion date. Equation (4) defines the setup time as the sum of all setups of all machines in the schedule. The Equation (5) shows the non-negativity constraints of the decision variables and of the weights.

The proposed application is based on the i-TSAB algorithm developed by Nowicki and Smutnick. (Nowicki and Smutinicki, 2002). It is based on the Tabu Search technique and presents two distinct phases: (i) firstly, the proposed application fills a list E of elite solutions to be explored and; (ii) secondly, using a modified Tabu Search algorithm and the path-relinking technique (Glover and Laguna, 1997), our application explores the solutions and updates E. The modified Tabu Search can reconstruct the best L visited neighborhoods to re-intensification. The algorithm stops when a measure of distance among the solutions in L reaches a threshold. In order to create the neighborhood of feasible results, we are using the Critical Path structure described by Nowicki and Smutniki (Nowicki and Smutnicki, 1996).

The model that represents the application and the proposed objective function was developed in four modules. They are described below.

1. Module 1: Instance Generator – It adapts classical JSSP instances, generating tools for the operations and due dates for the jobs.

2. Module 2: Family of Operations – This module organizes the operations in Family of Operations (FO), according to the tools required for each operation. It implements a Cluster Analysis Algorithm (Kusiak and Chow, 1987; Dorf and Kusiak, 1994).
3. Module 3: Initial Solution – It creates a feasible schedule, ordering the operations $O = 1,\ldots,o_j$ $j=1,\ldots n$, $j \in J$.
4. Module 4: Optimization – This module optimizes the initial solution based on the modified i-TSAB technique.

The architecture of the model represented by the information flow among the modules is shown in the Figure 1.

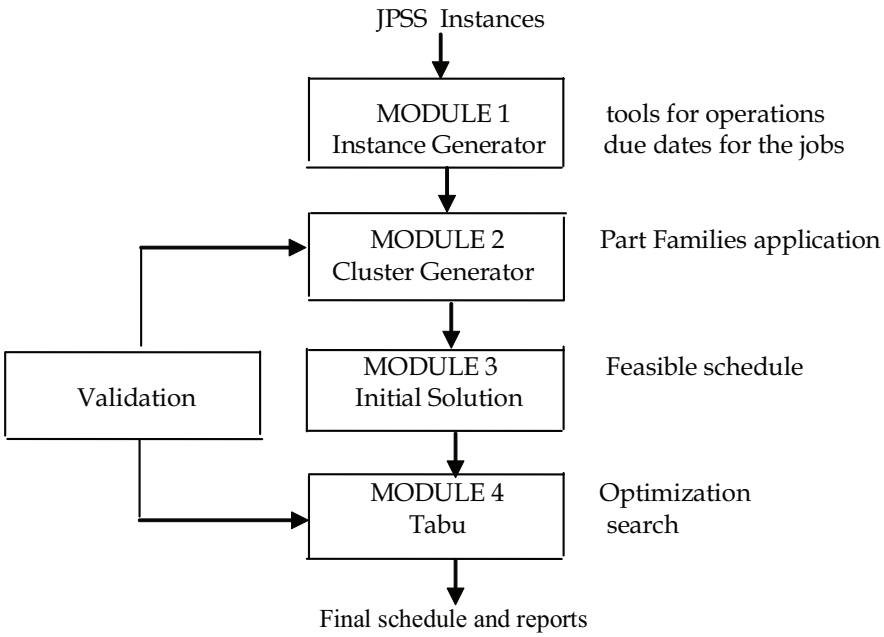

Fig. 1. Architecture of the model that describes the operation of the application

5. Performed experiments

The implementation of the model was made in C language. The source code was compiled by using the GCC compiler, which can be found in the GNU-Linux operational system. The model was validated in two phases: (i) validation of the module 2: generation of Family of Operations (FO) and (ii) validation of module 4: minimization of Ms and St decision variables. Both modules 2 and 4 are illustrated in Figure 1.

5.1 Validation of module 2

The module 2 was validated using a classical instance proposed by Tang and Denardo (Tang and Denardo, 1988). This instance is showed in Figure 2. It is composed by 10 parts and 9 tools. The optimal result for this instance is the generation of 5 Part Families.

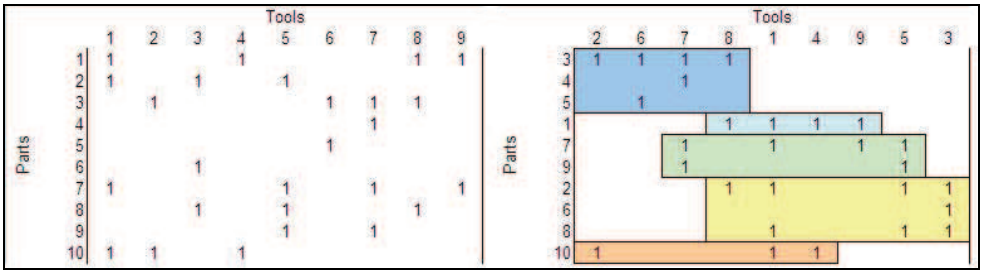

Fig. 2. Instance of 10 parts and 9 tools of Tang and Denardo.

5.2 Validation of module 4

The module 4 generates the schedule through the implementation of the modified i-TSAB technique. Firstly, it was validated the minimization of Setup time comparing results of the module 4 with the work of Hertz and Widmer (Hertz and Widmer, 1996). The authors used 45 benchmark problems provided by Lawrence and Adams *et al* (Jain and Meeran, 1999), adapted to tooling constraints. The search parameters of Tabu Search used in the module 4 were the same used by Hertz and Widmer. In the validation, Module 4 reached the same or better results as Hertz and Widmer. Some results are showed in the Table 1.

Instance	Hertz and Widmer	Module 4
LA16	963	961
LA17	793	789
LA18	876	863
LA19	870	859
LA21	1097	1091
ABZ5	1271	1261
ABZ6	970	963
ABZ7	691	685
ABZ8	701	697

Table 1. Objective function values of Hertz and Widmer and Module 4

The makespan validation was performed using classical JSSP instances proposed by Fisher and Thompson (Jain and Meeran, 1999). These instances are showed in the Table 2. The optimal know result for each instance was reached by Module 4.

Instance	Makespan
FT6	55
FT10	930
FT20	1165

Table 2. Makespan for JSSP benchmark instances.

After validating, there were performed experiments with three minimization politics: (1) minimization of Makespan, (2) Minimization of tardiness and (3) minimization of Setup.

5.3 Benchmark instances and TS parameters

The experiments were performed using 6 benchmark problems provided by Taillard (Taillard, 2006), showed in the Table 3, adapted to the due dates and tooling constraints.

Benchmark instance	Dimensions (job/machine/operation)
$TA1515_1$, $TA1515_2$	15 / 30 / 225
$TA3020_1$, $TA3020_2$	30 / 20 / 600
$TA5015_1$, $TA5015_2$	50 / 15 / 750

Table 3. Instance used in the experiments.

The parameters for the experiments were: production turn lasts 480 minutes; Setup α times lasts 5 minutes; Setup β lasts 4 minutes; machine magazine can hold at most 4 tools; total number of tools needed to process the all operations is 10; any operation requires more than 4 tools for its processing; the Tabu List stores 5 moves; the size of the list of elite solution E is 3; the size of the L list of best visited neighborhood is 1; the measure of distance among solution L is 5.

5.4 Non-tendentious solution

To perform the comparisons among the results obtained with the three minimization politics, it was defined a non-tendentious solution (NTS). It consists in a weight configuration in which any of the decision variables are not privileged. For each instance, it was performed 100 executions of the Module 4, where the values of weights of the objective function were varied in a 0-100 uniform distribution. The mean of the values obtained on each decision variable was calculated and a proportion was made. The Table 4 shows the values obtained for the weights of the decision variables (W_1 (*Ms*), W_2 (*At*) and W_3 (*St*)).

Dimension	W_1	W_2	W_3
15x15	5	1	14
30x20	15	1	29
50x15	21	1	46

Table 4. Weights of the decision variables of f for NTS solution.

Using these values of weights, the modules 3 and 4 were run, for each of the instances. The Tables 5 and 6 show the values obtained for the decision variables with the modules 3 and 4.

Instances	*Ms*	*At*	*St*
$TA1515_1$	10767	73041	2870
$TA1515_2$	10216	78214	2875
$TA3020_1$	27250	393643	7834
$TA3020_2$	27990	390054	7970
$TA5015_1$	33637	777600	10047
$TA5015_2$	31586	734498	9948

Table 5. Values of decision variables of *f* obtained with module 3.

Instances	*Ms*	*At*	*St*
$TA1515_1$	1580	6532	2563
$TA1515_2$	1563	10640	2576
$TA3020_1$	5957	112293	7466
$TA3020_2$	4466	72520	7624
$TA5015_1$	5061	129275	9442
$TA5015_2$	5226	145025	9459

Table 6. Values of decision variables of *f* obtained with module 4.

5.5 Minimization of makespan

In this experiments, the value of the weights of the decision variable *Ms* is increased, meanwhile the weights of the variables *At* and *St* remains the same as the NTS.

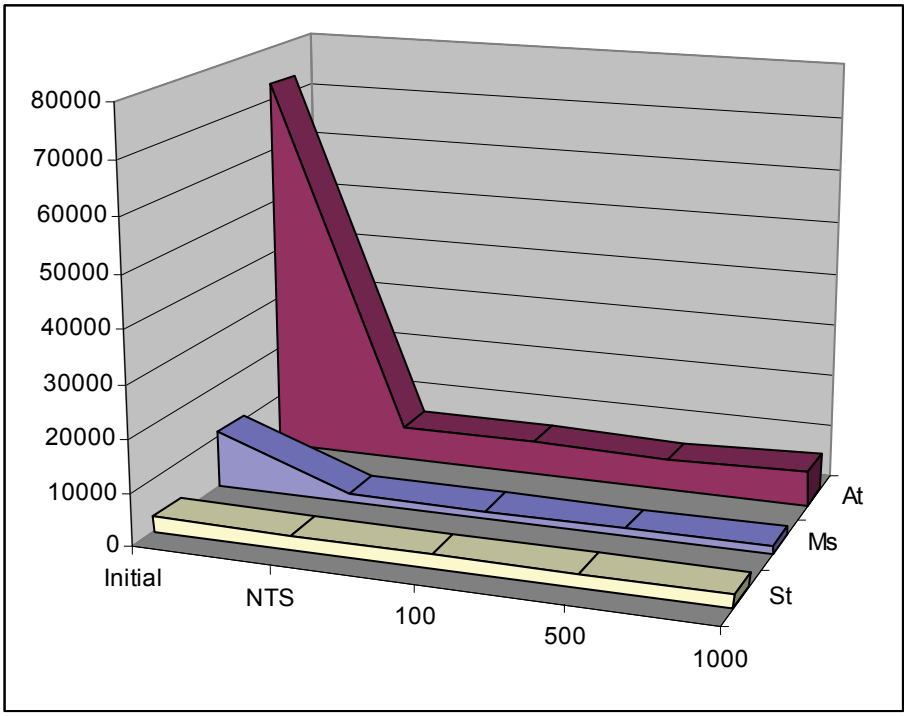

Fig. 3. Values for decision variables of f considering the increasing of Ms weight.

Figure 3 shows the comparison among the initial solution, the NTS and the values obtained with execution of module 4, increasing the value of weights of *Ms*. It can be noticed a reduction of 3% for the *St* decision variable, 87% for the *Ms* and *At* variables, compared to the initial solution. Compared to the NTS solution, the reduction is lower: 3.6% and 6.7% for the *Ms* and *At*. The variable St increases 5.8% compared to the NTS value. This increasing of *St* is due to the fact that the value of this variable depends of the

setup in all machines, not only the operations on the Critical Path used to generate neighborhoods.

5.6 Minimization of tardiness

In this experiment, the value of the weight of *At* is increased, while the *Ms* and *St* weights remain the same defined in the NTS previously. The Figure 4 shows the comparison of initial solution, NTS and values obtained with execution of the module 4, increasing the value of the *At* weight.

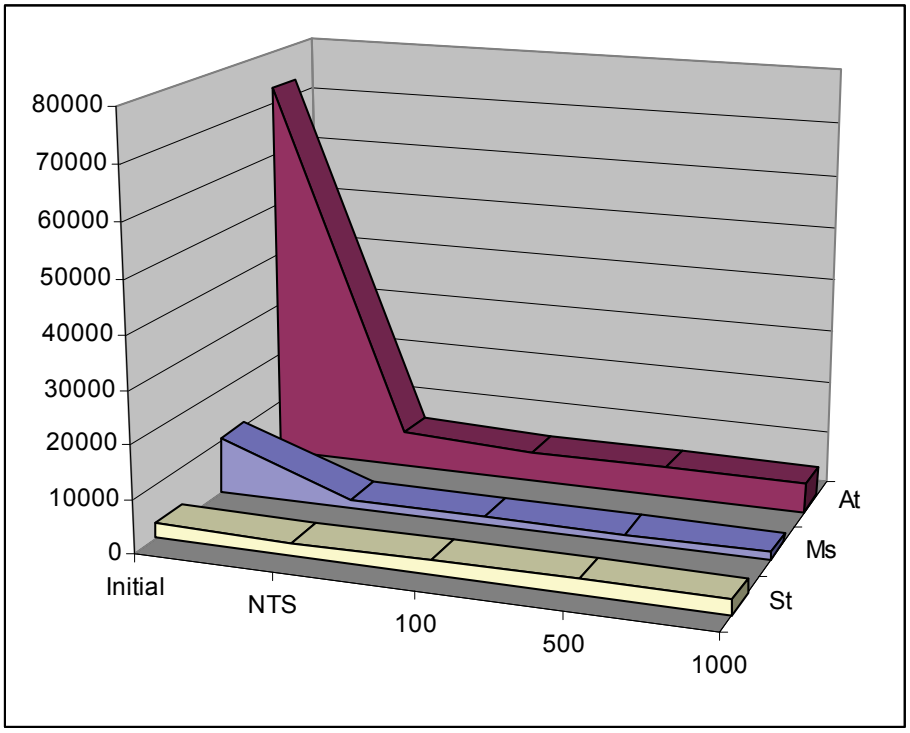

Fig. 4. Values of the decision variables of f considering increasing of At weight.

The increasing of *At* weight reduces 81% the *At* value when compared with the initial solution. The variables *Ms* and *St* had mean reductions of 82% and 2%, respectively. Comparing with NTS values, the values obtained for *At* for instances between 225 and 600 operations are at most 30% lowers. For instances of 750 operations, the *At* assumes higher values (at most 65%). This increasing of the obtained values for *At*, despite the privilege of this variable, is due to the fact that *At* depends of the last operation of each job. Considering that only operations on the Critical Path are moved to generate neighborhoods, *At* variable can be reduced only if the Critical Path contains the last operation of the jobs. Other factor in the comparison between the *At* minimization policy is that NTS reduces significantly the *At* value, compared to the initial solution.

5.7 Minimization of setup

It is considering that the value of *St* weight is increased and the values of *At* and *Ms* weights remains constants, having the NTS values. Figure 5 shows the comparison of the initial solution, NTS and the values obtained with the execution of the Module 4, increasing the value of the *St* weight.

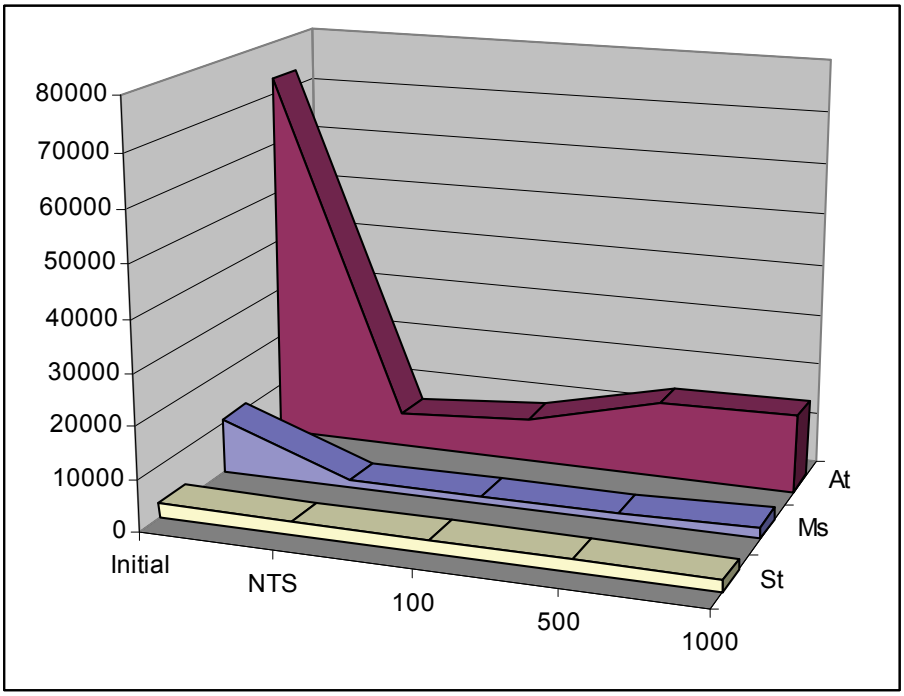

Fig. 5. Values of the decision variable of f considering increasing of St weight.

It can be observed the reduction of the values obtained for *Ms*, *At* and *St* decision variables in 79%, 78% and 14% respectively, compared with the initial solution. When the comparison is made with NTS, the *St* showed reduction of 8.4% in its value. Otherwise, *Ms* increased the obtained value in 24.2% and *At* almost doubled its value.

The setup time always increases when two operations of different FOs are processed in sequence. The machine must stop its processing to switching tools operation. When this operation occurs in the idle time of the machine (e. g. machine is waiting other machine release the product), it not increases the value of *f*. The occurrence of setup is represented, in this work, as a dummy operation that lasts $\alpha+\beta t$ minutes. The total setup time is the sum of all setups in the production, considering that only few of them compose the Critical Path. The setup reduction occurs when two operations of different FOs are swapped in the Critical Path, being operations of the same FO in sequence. Independently of the value assigned to the *St* weight, the operations that not compose the Critical Path will not be considered. The Figure 6 represents the classic instance FT6 adapted to tools

constrains. The nodes represent the operations and its processing times. The bold lines represent the operations in the Critical Path. The filled operation represents the setup. It can be noticed that the Critical Path concept does not contribute to the direct minimization of the *St*.

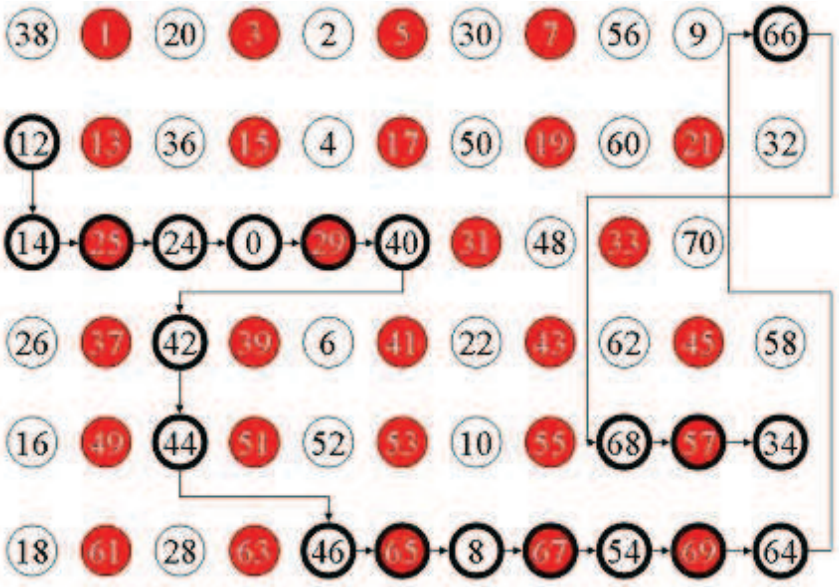

Fig. 6. Graph that represents the FT6 instance.

5.8 TS parameters variation

There were performed experiments where the Tabu Search parameters were changed, with the objective of verify the impact of this change in the generated schedules. Two sets of parameter were used:

1. nbmax = 20,000; Tabu List = 8; L = 5; E = 5.
2. nbmax = 20,000; Tabu List = 15 L = 7; E = 8.

The parameters used in these experiments were: nbmax is 20,000; Tabu List with sizes of 8 and 15; L, list of re-intensification with sizes 5 and 7; E, list of elite solutions with sizes 5 and 8.

The performed experiments were made considering the minimization politics showed in the previous items. There were performed too experiments where each of the decision variables was minimized separately, using the set (2) of parameters. This was performed assigning value 1 for the weight of the decision variable considered and value 0 for the weight of the other two variables.

Figure 7 shows the results of these experiments. The set (3) indicates the minimization of each decision variable separately. The legend *Ms* (1), for example, indicates the minimization of *Ms* policy, using set of parameters (1).

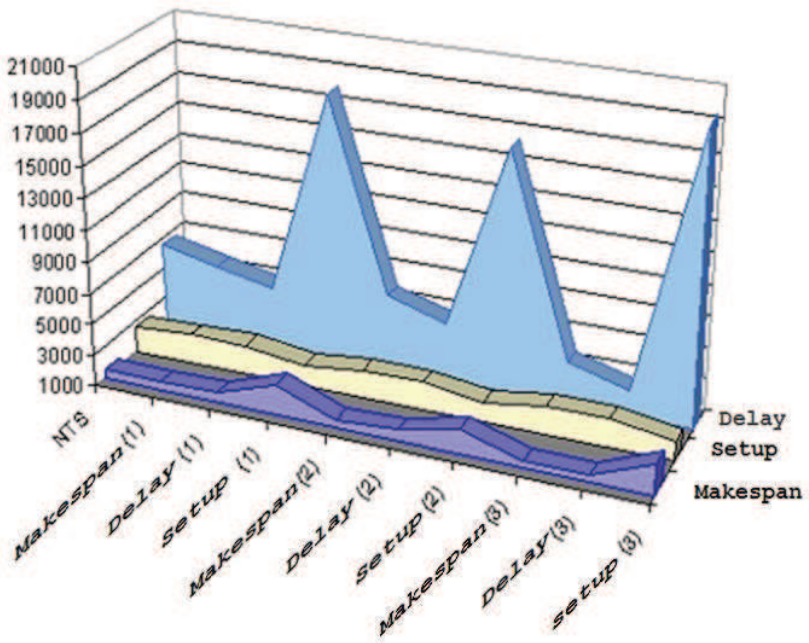

Fig. 7. Comparative graph considering NTS and the minimization politics with the new parameters.

It can be noticed that the increasing of the Tabu Search parameters had positive impact. For each minimization policy, the results obtained with the increased parameters are better than the obtained with NTS. When the decision variables are minimized separately, the reductions obtained for *Ms*, *At* and *St* was 18,4%, 58,1% and 11%, respectively, if compared with NTS. Comparing with initial solution, the reductions were about 88%, 96% and 20%.

In the performed experiments, it can be notices that the implemented model privileges the reduction of *Ms*, due to the use of the Critical Path concept.

6. Conclusion

This chapter proposes a computational model that considers the JSSP problem with due dates, production turns and tooling constraints. The approach used consists in a modification of the Tabu Search technique i-TSAB implemented by Nowicki and Smutinicki (Nowicki and Smutnicki, 2002). The computational model was validated with works of Tang and Denardo (Tang and Denardo, 1988), Hertz and Widmer (Hertz and Widmer, 1996) and using classical JSSP instances provided by Fisher and Thompson (Jain and Meeran, 1999).

There were performed experiments with the objective of verifying the behavior of the model considering three minimization politics: minimization of makespan, tardiness time and setup time. Aiming to compare the results obtained by each policy, it was generated a non-

tendentious Solution, in which the three decision variables have the same contribution for the value of *f*.

The implemented model generates good results for the minimization of *Ms*. This is due to the fact that the model is strongly based in the Nowicki and Smutnicki Critical Path concept. The minimization of *At* is not so significant as the *Ms*, considering that the reduction of this variable depends of the lasts operations of the jobs compose the Critical Path. In the *St* minimization, the reduction was not significant too, considering that only setup operations that compose the Critical Path can be reduced in the iterations of the technique.

The increasing of the parameters of the technique had good impact in the search results. A significant reduction of the decision variables can be noticed when each one is minimized separately.

Finally, the proposed model is an approach to the JSSP problem considering additional constraints. The use of the Critical Path concept turns difficult the search for good results when considering the three aforementioned decision variables simultaneously. Thus, the best results were obtained considering the decision variables separately.

7. References

Balas, E., Johnson, E. L., Korte, B., 1979. *Disjunctive programming*. In: Hammer, P.L. Discrete Optimization II, pages 49–55.

Blazewicz, J., Domschke, W., and Pesch, E., 1996. *The job shop scheduling problem: conventional and new solution techniques*. European Journal of Operational Research, 93:1–33.

Dorf, R., Kusiak, A., 1994. *Handbook of Design, Manufacturing and Automation*. John Wiley and Sons.

Fisher, M., 1976. *A dual algorithm for the one-machine scheduling problem*. Math Programming, 11:229–251.

Glover, F.,1989. *Tabu Search – parte1*. ORSA Journal on Computing v.1, n.3.

Glover F., Laguna, M., 1997. *Tabu Search*. Kluwer Academic, Boston.

Gómez, A. T., 1996. *Modelo para o seqüenciamento de partes e ferramentas em um sistema de manufatura flexível com restrições as datas de vencimento e a capacidade do magazine*. PhD Thesis. Instituto Nacional de Pesquisas Espaciais, Brasil.

Groover, M. P., 2001. *Automation, Production Systems, and Computer-Integrated Manufacturing*. United States, Prentice Hall, Second Edition, 856p.

Hertz, A,. Widmer, M.,1996. *An improved tabu search for solving the job shop scheduling problem with tooling constraints*. Discrete Applied Mathematics, 65, 319-345.

Hurink, J., Knust, S., 2004. *Tabu Search algorithms for job-shop problems with a single transport robot*. European Journal of Operational Research, 162:99–111.

Jain, A. S., Meeran, S., 1998. *Deterministic job-shop scheduling: Past, present and future*. European Journal of Operational Research, 113(2):390–434.

Jha, N. K., 1991. *Handbook of Flexible Manufacturing Systems.* Academic Press, 328p.

Kusiak, A., Chow, W. S., 1987. *Efficient Solving of the Group Technology Problem.* Journal of Manufacturing Systems, 6(2):117–124.

Mascis, A., Pacciarelli, D., 2002. *Job-shop scheduling with blocking and no-wait constraints*. European Journal of Operational Research, 143:498–517.

Nowicki, E., Smutnicki, C., 1996. *A fast tabu search algorithm for the permutation flow shop problem*. European Journal of Operational Research. Elsevier, v.91, p.160–175.

Nowicki, E., Smutnicki, C., 2002. *Some new tools to solve job shop problem.* Institute of Engineering Cybernetics, Wroclaw University, Poland.

Taillard, E. 2006. *Job Shop Scheduling Problem on-line instances. Available* in: http://ina2.eivd.ch/collaborateurs/etd/problemes.dir/ordonnancement.dir/ordonnancement.html.

Tang, C. S., Denardo, E. V., 1988. Models arising Flexible Manufacturing Machine Part II: minimization of the number of switching instants. Operations Research, 36(5).

Wu, T., Yeh, J., Chang, C., 2009. *A hybrid Tabu Search Algorithm to Cell Formation Problem and its Variants*. World Academy of Science, Engeneering and Technology, v. 53, 1090-1094.

Zoghby, J., Barnes, W. L., Hasenbein, J. J., 2004. *Modeling the Reentrant job shop problem with setups for metaheuristic searches.* European Journal of Operational Research, 167:336–348.

Production Scheduling

Edited by Prof. Rodrigo Righi

ISBN 978-953-307-935-6
Hard cover, 242 pages
Publisher InTech
Published online 11, January, 2012
Published in print edition January, 2012

Generally speaking, scheduling is the procedure of mapping a set of tasks or jobs (studied objects) to a set of target resources efficiently. More specifically, as a part of a larger planning and scheduling process, production scheduling is essential for the proper functioning of a manufacturing enterprise. This book presents ten chapters divided into five sections. Section 1 discusses rescheduling strategies, policies, and methods for production scheduling. Section 2 presents two chapters about flow shop scheduling. Section 3 describes heuristic and metaheuristic methods for treating the scheduling problem in an efficient manner. In addition, two test cases are presented in Section 4. The first uses simulation, while the second shows a real implementation of a production scheduling system. Finally, Section 5 presents some modeling strategies for building production scheduling systems. This book will be of interest to those working in the decision-making branches of production, in various operational research areas, as well as computational methods design. People from a diverse background ranging from academia and research to those working in industry, can take advantage of this volume.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821