Journal of Applied Sciences 9 (9): 1740-1745, 2009
ISSN 1812-5654

A Tabu Search Approach to Hybrid Flow Shops Scheduling with Sequence-Dependent Setup Times

[1]M.B. Abiri, [2]M. Zandieh and [2]A. Alem-Tabriz
[1]Department of Industrial Engineering, Mazandaran University of Science and Technology, Babol, Iran
[2]Department of Industrial Management, Faculty of Management and Accounting,
Shahid Beheshti University, G.C., Tehran, Iran

Abstract: This study describes a Tabu Search (TS) algorithm approach to the scheduling of a sequence-dependent setup times hybrid flow shop. The details of a TS approach are described and implemented. The results obtained are compared with those computed by Random Key Genetic Algorithm (RKGA) presented earlier. From the results, it was established that TS outperformed RKGA.

Key words: Scheduling, hybrid flow shops, sequence-dependent setup times, makespan, Tabu search

INTRODUCTION

A hybrid flow shop model, commonly known as flexible flow line, allows us to represent most of the production systems. The process industry such as chemical, pharmaceutical, oil, food, tobacco, textile, paper and metallurgical industry can be modeled as a hybrid flow shop. In the literature, the notion of hybrid flow shop has emerged in the 70s (Zandieh *et al*., 2006). A hybrid flow shop consists of a series of production stages, each of which has several facilities in parallel (Kurz and Askin, 2004). Some stages may have only one facility, but for the plant to be qualified as a hybrid flow shop, at least one stage must have several facilities. The flow of products in the plant is unidirectional. Each product is processed at only one facility in each stage and at one or more stages before it exits the plant. Each stage may have multiple parallel identical machines. These machines can be identical, uniform, or unrelated. Each job is processed by at most one machine at each stage.

Pinedo (2002) cited machine setup time is a significant factor for production scheduling in all flow patterns and it may easily consume more than 20% of available machine capacity if not well handled. Also the completion time of production and machine setups are influenced by production mix and production sequence. On the one hand, processing in large batches may increase machine utilization and reduce the total setup time. On the other hand, large batch processing increases the flow time. Scheduling problems with sequence-dependent setup times are among the most difficult classes of scheduling problems. A single-machine sequence-dependent setup scheduling problem is equivalent to a traveling-salesman problem and is NP-hard (Pinedo, 2002). Even for a small system, the complexity of this problem is beyond the reach of existing theories (Luh *et al*., 1998).

Sequence-dependent setup scheduling of a hybrid flow shop system is even more challenging. Although, there has been some progress reported, but the understanding of sequence-dependent setups, however, is still believed to be far from being complete (Luh *et al*., 1998).

Gupta and Tunc (1994) presented four heuristic algorithms to minimize makespan for a two stage hybrid flow shop problem with separable setup and removal times. In which, sequencing of jobs can be done using one of Sule's (1982) rule or Szwarc and Gupta's (1987) algorithm while assigning jobs to multiple machines at the second stage is done by attempting to minimize the job-waiting time at the second stage.

Robust local search improvement techniques for flexible flow-line scheduling were considered by Leon and Ramamoorthy (1997). Kochhar and Morris (1987) model flexible flow lines in a more complete manner in that they allow for setups between jobs, finite buffers which may cause blocking and starvation, machine down-time and current and subsequent state of the system.

Rios-Mercado and Bard (1998) also considered the sequence-dependent setup time flow shop and developed several valid inequalities for models based on the traveling salesman problem and the Srikar-Ghosh model.

Hung and Ching (2003) addressed a scheduling problem taken from a label sticker manufacturing company which is a two-stage hybrid flow shop with the

Corresponding Author: M. Zandieh, Department of Industrial Management, Faculty of Management and Accounting, Shahid Beheshti University, G.C., Tehran, Iran

characteristics of sequence-dependent setup time at stage 1, dedicated machines at stage 2 and two due dates. The objective was to schedule one day's mix of label stickers through the shop such that the weighted maximal tardiness is minimized. They proposed a heuristic to find the near-optimal schedule for the problem. The performance of the heuristic was evaluated comparing its solution with both the optimal solution for small-sized problems and the solution obtained by the scheduling method used in the shop.

While many study have been written in the area of scheduling hybrid and flexible flow lines, many of them are restricted to special cases of two stages, specific configurations of machines at stages and to simplify the problem, setups are seldom considered in the scheduling. For those ones addressing setups, the setup times are fixed and included in processing times. However, in most real world cases, the length of the setup time depends on both jobs, which is separable from processing. There seems to be published only three works addressing heuristics for flexible flow lines with sequence-dependent setup times. Kurz and Askin (2003) examined scheduling rules for SDST flexible flow lines. They explored three classes of heuristics. The first class of heuristics (cyclic heuristics) is based on simplistic assignment of jobs to machines with little or no regard for the setup times. The second class of heuristics is based on the insertion heuristic for the Traveling Salesman Problem (TSP). The third class of heuristics is based on Johnson's Rule. Note that the second class caters to setup aspects of the problem while the third derives from standard flow shops. They proposed eight heuristics (CH, RCH, SPTCH, FTMIH, CTMIH, MMIH, 1, g Johnson's Rule, g/2, g/2 Johnson's Rule) and compared the performance of those on a set of test problems. Moreover, Kurz and Askin (2004) formulated the SDST flexible flow lines as an integer programming model. Because of the difficulty in solving the IP model directly, they developed a Random Keys Genetic Algorithm (RKGA). Problem data was generated to evaluate the RKGA with other scheduling heuristics rules, which they proposed aforetime. They created a lower bound to evaluate the heuristics. Zandieh *et al.* (2006) proposed an immune algorithm and showed that this algorithm outperforms the random keys genetic algorithm of Kurz and Askin (2004).

PROBLEM DESCRIPTION

Let g be the number of workshops in series. Let n be the number of jobs to be processed and m^t be the number of machines in parallel at each stage t. We assume that machines are initially setup for a nominal job 0 at every stage. Job n+1 exists at every stage only to indicate the end of the process, if needed. We have the following definitions:

p_i^t = Processing time for job i at stage t
(i = 1, 2,....n; t = 1, 2....,g)
s_{ij}^t = Sequence-dependent setup time from job i to job j at stage t
(i = 1, 2,....n; j = 1, 2,....,n; t = 1, 2....,g)
$\tilde{p}_i^t$ = Modified processing time for job i at stage t
($\tilde{p}_i^t = p_i^t + \min_j s_{ij}^t$)
(i = 1, 2,....n; j = 1, 2,....,n; t = 1, 2....,g)
S^t = set of jobs that visit workshop stage t
(t = 1, 2,...,g)

The processing time of job 0 is set at 0. The setup time from job 0 indicates the time to move from the nominal set solution state. We assume that all jobs currently in the system must be completed at each stage before the jobs under consideration may begin setup. The completion times of job 0 at each stage are set to the earliest setup time may begin at that stage. The setup time for job n+1 is set at 0; this job only exists to indicate the end of the schedule. We also include the restriction that every stage must be visited by at least as many jobs as there are machines in that stage.

THE PROPOSED TABU SEARCH ALGORITHM

Tabu search algorithm in general: Tabu Search (TS) has been found to be a remarkably efficient approach for solving hard combinatorial problems, including traveling salesman problems, scheduling problems, product delivery and routing problems and manufacturing cell design problems. Starting from an initial solution, TS generates a new alternative S ' in the neighborhood of the original alternative S with a function that transforms S into S '. This is usually called a move, which can be made to a neighbor solution even though it is worse than the given solution. This makes a TS escape from a local optimum in its search for the global optimum. To avoid cycling, TS defines a set of moves that are tabu (forbidden) and these moves are stored in a set Λ, called tabu list. Elements of Ë define all tabu moves that cannot be applied to the current solution. The size of Λ is bounded by a parameter *l*, called tabu list size. If $|\Lambda| = l$, before adding a move to Λ, one must remove an element in it, the oldest one in general. Note that a tabu move can be always allowed to be chosen if it creates a solution better than the incumbent solution, the best objective value obtained so far. Flow chart for the algorithm is shown in Fig. 1. Glover (1989) explained a comprehensive description of various aspects of TS.

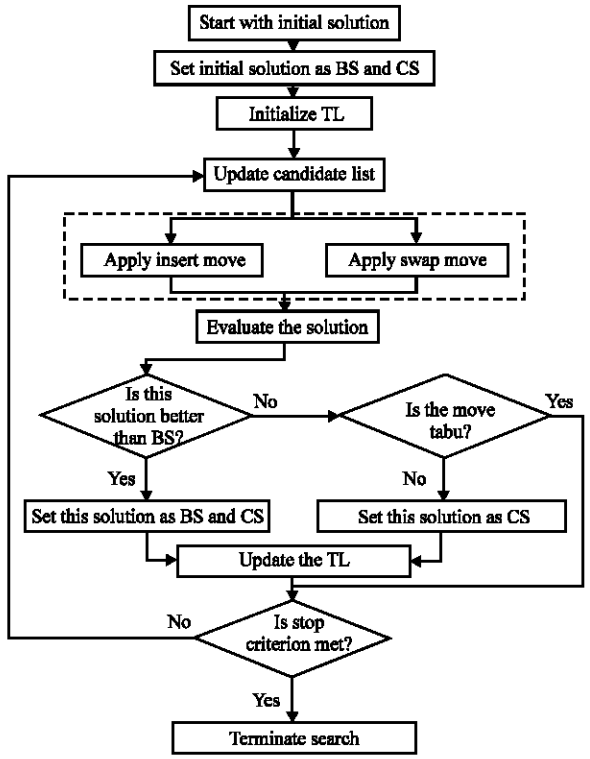

Fig. 1: Flow chart for tabu search. TL: Tabu list, BS: Best solution, CS: Current solution

A tabu search algorithm approach to hybrid flow shop scheduling: An application of TS is generally characterized by several factors. They are:

- Initial solution methods
- Neighborhood generation methods, i.e., set of possible moves applicable to the current solution
- Definition of tabu moves with the tabu list size
- Termination condition(s)

In this study, initial solution is created randomly and termination condition is a specified number of seconds.

First, the solution is represented by a random key representation. The advantage of this representation is its ease of implementation. This representation was proposed by Norman and Bean (1999) to avoid infeasible solution. They used the following solution representation for an identical multiple machine problem. Each job is assigned a real number whose integer part is the machine number to which the job is assigned and whose fractional part is used to sort the jobs assigned to each machine. For example solution shown in Fig. 2, is interpreted as Fig. 3.

Note that the represented solution schedules jobs only in first stage. For other stages we use the fallowing algorithm, witch is proposed by Kurz and Askin (2003, 2004).

1.41	2.13	1.23	1.65	2.01	3.27	1.19	1.87	2.61	3.05

Fig. 2: Representation of candidate solution in TS

M1	J_7	J_3	J_1	J_4	J_8
M2	J_5	J_2	J_9		
M3	J_{10}	J_6			

Fig. 3: Interpreted candidate solution (Fig. 2)

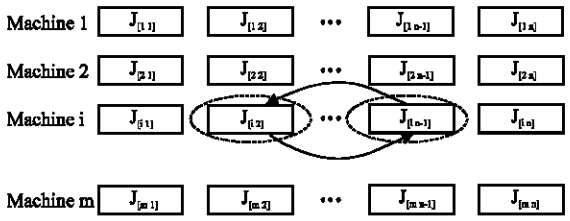

Fig. 4: Intra-machine move

For each stage t = 2,.. .,g:

(a) Update the ready times in stage t to be the completion times in stage t−1
(b) Arrange jobs in increasing order of ready times
(c) Let bestmc = 1
(d) For [i] = 1 to n, i € S^t:

- For mc = 1 to m^t
- Place job [i] last on machine mc
- Find the completion time of job [i]. If this time is less on mc than on bestmc
- let bestmc = mc
- Assign job [i] to the last position on machine bestmc

Neighborhood generation (move): Insert moves and pair wise exchanges (swaps) are two of the frequently used move types in permutation problems. An insert move identifies two particular jobs and places the first job in the location that directly precedes the location of the second job. A swap move, on the other hand, places each job in the location earlier occupied by the other. The neighborhood generation methods suggested in this study have a hybrid structure with swap and insertion methods. A schematic description of components of this hybrid structure is shown in Fig. 4 and 5. As can be shown from the Fig. 4 and 5, swap move is done for a pair of jobs on the same machine and the insertion move is done for jobs on different machines, so they are also referred as intra-machine and inter-machine moves by Bilge *et al.* (2004).

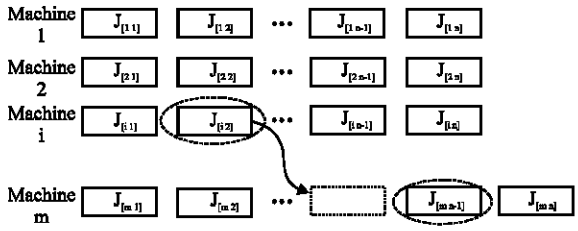

Fig. 5: Inter-machine move

Candidate list strategy: For situations where the neighborhood of a solution is large or its elements are expensive to evaluate, candidate list strategies are essential to restrict the number of solutions examined on a given iteration (Laguna *et al*., 1991). The purpose of these rules is to screen the neighborhood so as to concentrate on promising moves at each iteration. In this study, for better performance of TS, two selection strategies are considered. For insert move only the one job with highest makespan is chosen while the other needed job in this move is selected randomly. Selection strategy in the swap move is a roulette wheel sampling based on the sequence-dependant setup time of each job in first level, so the selection probability of the i-th job is as follows.

$$p_i = \frac{1/S^1_{i-1,i}}{\sum_{j=1}^{n} 1/S^1_{j-1,j}}$$

Tabu moves: Tabu moves are defined as follows. In the swap move, a pair of jobs that have been interchanged is defined as a tabu move. Also, the insertion method defines a tabu move as the job to be moved. As an exceptional case, a tabu move can be allowed to be chosen if it generates a solution better than the incumbent solution, the best objective value obtained so far.

For more effective search, in each iteration we perform n moves (starting from current position) and the one with best fitness function is selected.

EXPERIMENTAL DESIGN

Data generation and settings: An experiment was conducted to test the performance of the tabu search algorithm. Following Kurz and Askin (2003) data required for a problem consists of the number of jobs, number of stages, number of machines in each stage, range of processing times and the range of sequence-dependent setup times. The ready times for stage 1 are set to 0 for all jobs. The ready times at stage t+1 are the completion times at stage t, so there is no need this data to be generated. Processing times are distributed uniformly over two ranges with a mean of 60: [50-70] and [20-100]. Flexible flow lines are considered by allowing some jobs to skip some stages. Following Leon and Ramamoorthy (1997), the probability of skipping a stage is set at 0, 0.05, or 0.40. The setup times are uniformly distributed from 12 to 24 which are 20 to 40% of the mean of the processing time. The setup time matrices are asymmetric and satisfy the triangle inequality. The setup time characteristics follow Rios-Mercado and Bard (1998).

Table 1: Factor levels of the problems

Factor	Levels
No. of jobs	6-30-100
No. of machines	Constant:1-2-10
	Variable: Uniform [1-4]-Uniform [1-10]
No. of stages	2-4-8
Processing times	Uniform [50-70]-Uniform[20-100]
Skipping probability	0.00-0.05-0.40

The problem data can be characterized by 6 factors and each of these factors can have at least two levels. These levels are shown in Table 1.

In general, all combinations of these levels will be tested. However, some further restrictions are introduced. The variable machine distribution factor requires that at least one stage have a different number of machines than the others. Also, the largest number of machines in a stage must be less than the number of jobs. Thus, the combination with 10 machines at each stage and 6 jobs will be skipped and the combination of 1-10 machines per stage with 6 jobs will be changed to 1-6 machines per stage with 6 jobs. There are 252 test scenarios and 10 data sets are generated for each one.

Experimental results: Here, we are going to compare the proposed tabu search algorithm with the RKGA which proposed by Kurz and Askin (2003) for the SDST flexible flow lines. The heuristics were implemented in Borland C++ 5.02 and run on a PC with a Pentium IV 1.8 GHz processor with 1 GB of RAM. When the C_{max} of each algorithm has been obtained for its instances, the best solution obtained for each instance (which is named Min_{sol}) by any of the two algorithms is calculated. Relative Percentage Deviation (RPD) is obtained by given formula below:

$$RPD = \frac{Alg_{sol} - Min_{sol}}{Min_{sol}}$$

where, Alg_{sol} is the C_{max} obtained for a given algorithm and instance. RPD of 4% for a given algorithm means that this algorithm is 4% over the best obtained solution on average. Clearly, lower values of RPD are preferred.

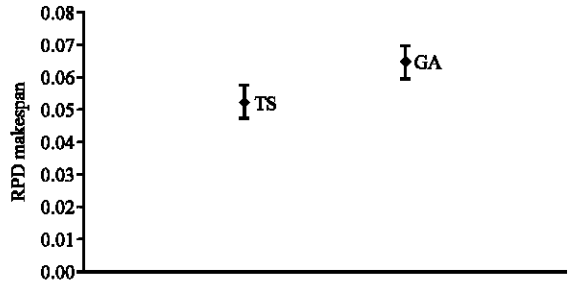

Fig. 6: Means plot and LSD intervals (at the 95% confidence level) for EM and GA algorithms

Table 2: Average RPD of makespan and solution time for the TS and GA algorithms by n and g

	RPD of makespan		
Instance	TS	GA	Solution time (sec)
6×2	0.156286	0.177086	5
6×4	0.079722	0.090125	
6×8	0.004437	0.006482	
6 Job	0.080149	0.091231	
30×2	0.17279	0.213516	20
30×4	0.020792	0.022644	
30×8	0.010624	0.014327	
30 Job	0.068069	0.083496	
100×2	0.006334	0.020888	45
100×4	0.00842	0.017256	
100×8	0.009062	0.016879	
100 Job	0.007939	0.018341	
Average	0.052052	0.064356	

Analysis of makespan and solution time base on RPD: The results of the experiments for two subsets, averaged for each one of the n and g configurations (15 data per average) are shown in Table 2. As it can be seen, TS algorithm provides better results than GA algorithms.

In order to verify the statistical validity of the results shown in Table 2 and to confirm which the best algorithm is, we have performed a design of experiments and an Analysis of Variance (ANOVA) where we consider the different algorithms as a factor and the response variable RPD.

The results demonstrate that there is a clear statistically significant difference between performances of the algorithms. The means plot and LSD intervals (at the 95% confidence level) for two algorithms are shown in Fig. 6.

ANALYSIS OF CONTROLLED FACTORS

Analysis of problem size factor (number of jobs): In order to see the effects of number of jobs on two algorithms, a two ways ANOVA is applied. Means plot and LSD intervals (at the 95% confidence level) for the interaction between the factors type of method and number of jobs are shown in Fig. 7.

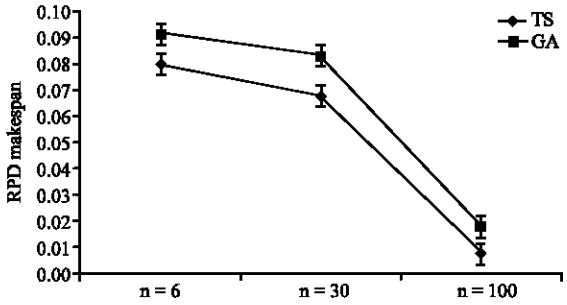

Fig. 7: Means plot and LSD intervals (at the 95% confidence level) for the interaction between the factors type of algorithm and number of jobs

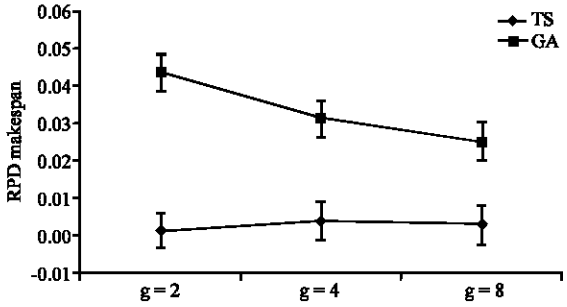

Fig. 8: Means plot and LSD intervals (at the 95% confidence level) for the interaction between the factors type of algorithm and magnitude of stages

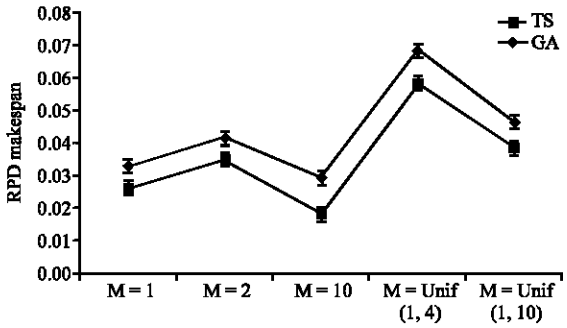

Fig. 9: Means plot and LSD intervals (at the 95% confidence level) for the interaction between the factors type of algorithm and magnitude of machines

As we can shown in Fig. 7, in the case of n = 100, n = 30 and n = 6 TS works better than GA.

Analysis of *g* factor (number of stages): Another Two ways ANOVA and LSD test are applied to see the effect of magnitude of stages on quality of the algorithms. The results are shown in Fig. 8.

Table 3: TS Makespan value versus GA

	Makespan value				
	Decrease		Similar	Increase	
Problem size	Problem (%)	Average decrease	Problem (%)	Problem (%)	Average increase
Small	42.83	12.07	49.25	7.93	24.99
Medium	66.81	21.78	2.22	30.96	7.76
Large	77.11	35.31	-	22.89	27.07

As we can shown in Fig. 8, in the total case TS works better than GA.

Analysis of m factor (number of machines): Another two ways ANOVA and LSD test are applied to see the effect of magnitude of machines on quality of the algorithms. The results are shown in Fig. 9.

As we can shown in Fig. 9, in the case of m = uniform (1, 10), m = uniform (1, 4), m = 10, m = 2 and m = 1 TS works better than GA.

Final analysis of makespan and solution time: Table 3 shows the makespan value for TS versus RKGA.

CONCLUSIONS AND FUTURE WORK

A TS approach for the scheduling of a hybrid flow shop has been successfully developed. The approach incorporates a new selection mechanism as well as a n-dimensional search mechanism to assist the search for a near optimal solution. An experiment was carried out to illustrate the effectiveness of tabu search algorithm in scheduling. Compared to past GA, the lower makespan values in many test problems, can be attributed to the fact that the TS tends to find better solutions.

There are potentially unlimited opportunities for research in scheduling to minimize makespan in hybrid flow shops with sequence-dependent setup times. In this study, we have addressed only a few areas.

In many researches such as present study, the lower bounds are typically used to evaluate the performance of heuristics for solving combinatorial optimization problems. In the absence of tight analytical lower bounds, optimal objective function values may be estimated statistically. Extreme value theory can be used to construct confidence-interval estimates of the minimum makespan.

Also by creating a general permutation schedule definition, we may be able to find a class of schedules that contains the optimal makespan schedule for some special cases, such as two stages with one machine at the first stage and two machines at the second, with sequence-dependent setup times at both.

REFERENCES

Bilge, U., F. Kirac, M. Kurtulan and P. Pekgun, 2004. A tabu search algorithm for parallel machine total tardiness problem. Comput. Operat. Res., 31: 397-414.

Glover, F., 1989. Tabu search: Part I. ORSA J. Comput., 1: 190-206.

Gupta, J.N.D. and E.A. Tunc, 1994. Scheduling a two-stage hybrid flowshop with separable setup and removal times. Eur. J. Operat. Res., 77: 415-428.

Hung, T.S.L. and J.L. Ching, 2003. A case study in a two-stage hybrid flow shop with setup time and dedicated machines. Int. J. Prod. Econ., 86: 133-143.

Kochhar, S. and R.J.T. Morris, 1987. Heuristic methods for flexible flow line scheduling. J. Manufact. Syst., 6: 299-314.

Kurz, M.E. and R.G. Askin, 2003. Comparing scheduling rules for flexible flow lines. Int. J. Prod. Econ., 85: 371-388.

Kurz, M.E. and R.G. Askin, 2004. Scheduling flexible flow lines with sequence-dependent setup times. Eur. J. Operat. Res., 159: 66-82.

Laguna, M., J.W. Barnes and F. Glover, 1991. Tabu search methods for a single machine scheduling problem. J. Intell. Manuf., 2: 63-74.

Leon, V.J. and B. Ramamoorthy, 1997. An adaptable problem-space-based search method for flexible flow line scheduling. IIE Trans., 29: 115-125.

Luh, P.B., L. Gou, Y. Zhang, T. Nagahora, M. Tsuji and K. Yoneda *et al*., 1998. Job shop scheduling with group-dependent setups, finite buffers and long time horizon. Ann. Operat. Res., 76: 233-259.

Norman, B.A. and J.C. Bean, 1999. A genetic algorithm methodology for complex scheduling problems. Naval Res. Logist., 46: 199-211.

Pinedo, M., 2002. Scheduling Theory, Algorithms and Systems. 2nd Edn., Prentice-Hall, Englewood Cliffs, New Jersey.

Rios-Mercado, R.Z. and J.F. Bard, 1998. Computational experience with a branch-and-cut algorithm for flowshop scheduling with setups. Comput. Operat. Res., 25: 351-366.

Sule, D.R., 1982. Sequencing n jobs on two machines with setup, processing and removal times separated. Naval Res. Logist. Q., 29: 517-519.

Szwarc, W. and J.N.D. Gupta, 1987. A flow-shop with sequance.dependent additive setup times. Naval Res. Logist., 34: 619-627.

Zandieh, M., S.M.T. Fatemi Ghomi and S.M. Moattar Husseini, 2006. An immune algorithm approach to hybrid flow shops scheduling with sequence-dependent setup times. J. Applied Math. Comput., 180: 111-127.