10.1515/acsc-2015-0012

Archives of Control Sciences
Volume 25(LXI), 2015
No. 2, *pages 189–199*

Permutation, no-wait, no-idle flow shop problems

MARIUSZ MAKUCHOWSKI

The paper compares the schedules of different variants of the flow shop problem, i.e. permutation, no waiting and no idle flow shop problems. It is assessed the impact of the constraints on the extension of the schedules and correlations of the length of the schedules for these variants. It is also examined the effectiveness of a set of insert type algorithms. The efficiency of the algorithms is tested on well-known literature benchmarks.

Key words: flow shop problem, permutation constraint, no-wait constraint, no-idle constraint, permutation-graph models.

1. Introduction

The flow shop problem is the key problem in a scheduling theory. For decades, it has been considered as a very interesting issue by both theorists and practitioners. The problem models many real industrial systems, as e.g. tape production lines. In the general flow shop problem one must perform a specified number of production tasks. Machines are set in the so-called 'production process' and each machine is responsible for the execution of a particular stage of production. Tasks are performed on all machines, whereas the technological route (order of machines used by tasks) is the same for all tasks. Scheduling of tasks in a flow shop system consists of determining the permissible moments of starting and ending of execution of all the tasks allocated to individual machines. The objective of the optimization is to select such a schedule that is the best in terms of a given criterion. One of the mostly studied, among the existing criteria, is the criterion of minimizing the length of the schedule, i.e., minimizing the time of completion of all the tasks. The schedule, in which the order of execution of tasks on each machine is the same, is called a permutation schedule. Sometimes, despite the fact that the production schedules allow to execute a non permutation schedule, it imposes an artificial constraint that the solution must be a permutation solution. This procedure has the advantages of reducing the solution space and increases the efficiency of algorithms. Unfortunately, in some cases the assumption deprives us of the possibility of finding the optimal solution.

M. Makuchowski is with Wroclaw University of Technology, 27 Wybrzeże Wyspiańskiego St., 50-370 Wroclaw, Poland. E-mail: mariusz.makuchowski@pwr.edu.pl

Received 14.11.2014.

2. Mathematical model

We are given a set of tasks $J = \{1,2,\ldots,n\}$ and a set of machines $M = \{1,2,\ldots,m\}$. Each task $j \in J$ must be performed sequentially on the each machines, in the order of numbering of machines. The process of execution of $j \in J$ task on machine $l \in M$ is called an operation and noted as a pair of (j,l). For each operation (j,l) there is given $p_{j,l} > 0$ time of its execution. The basic assumptions concerning the production are as follows:

- operations are performer without stops,
- the machine can perform, at most, one operation at a time,
- several operations of the same task cannot be executed simultaneously.

An acceptable schedule is called $S(j,l)$, beginning moments $C(j,l)$ and/or moments of operations completion (j,l), $l \in M$, $j \in J$ fulfilling all the above mentioned constraints. Between the starting and ending times of each operation there is: $C(j,l) = S(j,l) + p_{j,l}$. For each possible schedule there can be an evaluation function determined. In the this study the examined criterion is C_{max} i.e. a moment of completion of all operations; $C_{max} = \max_{j \in J} C(j,m)$. The problem relies in finding an acceptable schedule that minimizes the selected criterion.

2.1. Different variants of the flow shop problem

Taking into account the additional assumptions regarding production one imposes additional requirements (constraints) with respect to searched schedule. Due to the additional constraints the flow shop problem creates additional special cases. In this paper three special cases are going to be compared:

- permutation flow shop problem – the order of execution of tasks on all machines must be the same, [3, 4];
- no-wait flow shop problem – in which the start of tasks execution on the next machine must be initiated immediately after the completion of tasks on the previous machine, [5, 8];
- permutation no-idle flow shop problem – in which it is required that each machines must work without any breakdown, [2, 6].

These problems in Graham's notation [1] are denoted as $F|permu|C_{\max}$, $F|no-wait|C_{\max}$ and $F|permu,no-idle|C_{\max}$. It seems that the properties of no-wait problem enforce that a feasible solution is by definition a permutation solution. On the contrary, the no-wait constraint itself does not enforce the permutational character of the schedule. However, in the succeeding part of this paper with reference to the no-idle solution, we have in mind only permutation no-idle schedules. Most frequently, additional no-wait

and no-idle constraints extend the schedule, but this is not the case. Other very interesting properties of the above described problems are included in [7]. Exemplary schedules of the analyzed problems are shown in Fig. 1.

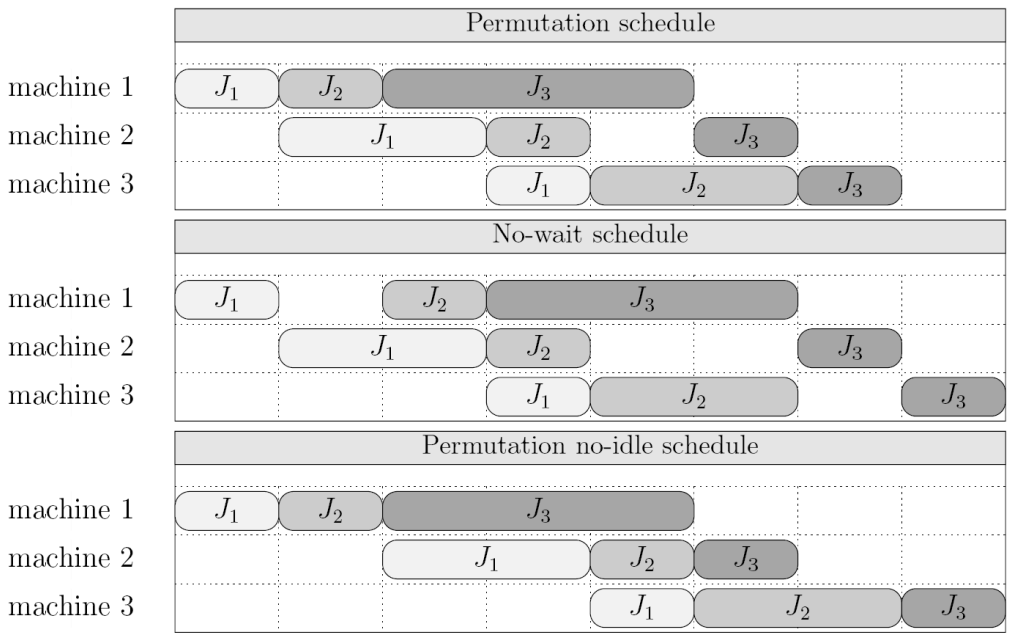

Figure 1: Schedules of the different variants of the flow shop problem.

2.2. Permutation-graph model

In every mentioned above cases, the feasible schedule can be uniquely defined by the sequence of the tasks execution. Therefore, it is convenient to apply the permutation-graph model, in which a decision variable is the permutation π of the set of tasks J; $\pi = (\pi(1), \pi(2), \ldots, \pi(n))$. The set of all feasible permutations is denoted by Π. The criterion is the length of the longest path in a directed graph:

$$G(\pi) = (J \times M, E^T \cup E^K(\pi)). \tag{1}$$

Vertex (j,l), $j \in J$, $l \in M$ is represented by the operation (j,l) and has a load of $p_{j,l}$. The set of unburdened arcs E^T represents a set of technological constraints

$$E^T = \bigcup_{j \in J} \bigcup_{l=2}^{m} \Big\{ \Big((j,l-1), (j,l) \Big) \Big\}. \tag{2}$$

A set of unburdened arcs $E^K(\pi)$ represents sequential constraints resulting from the determined sequence of tasks execution

$$E^K(\pi) = \bigcup_{i=2}^{n} \bigcup_{l \in M} \left\{ \Big((\pi(i-1), l), (\pi(i), l) \Big) \right\}. \tag{3}$$

Graph $G(\pi)$ for the permutation flow shop problem is shown in Fig. 2.a. Since the length of the longest path in the graph $G(\pi)$ being denoted by $C_{\max}(\pi)$ is equal to the moment of execution of all tasks, the analyzed problem is reduced to finding

$$\pi^* \in arg \min_{\pi \in \Pi} C_{\max}(\pi). \tag{4}$$

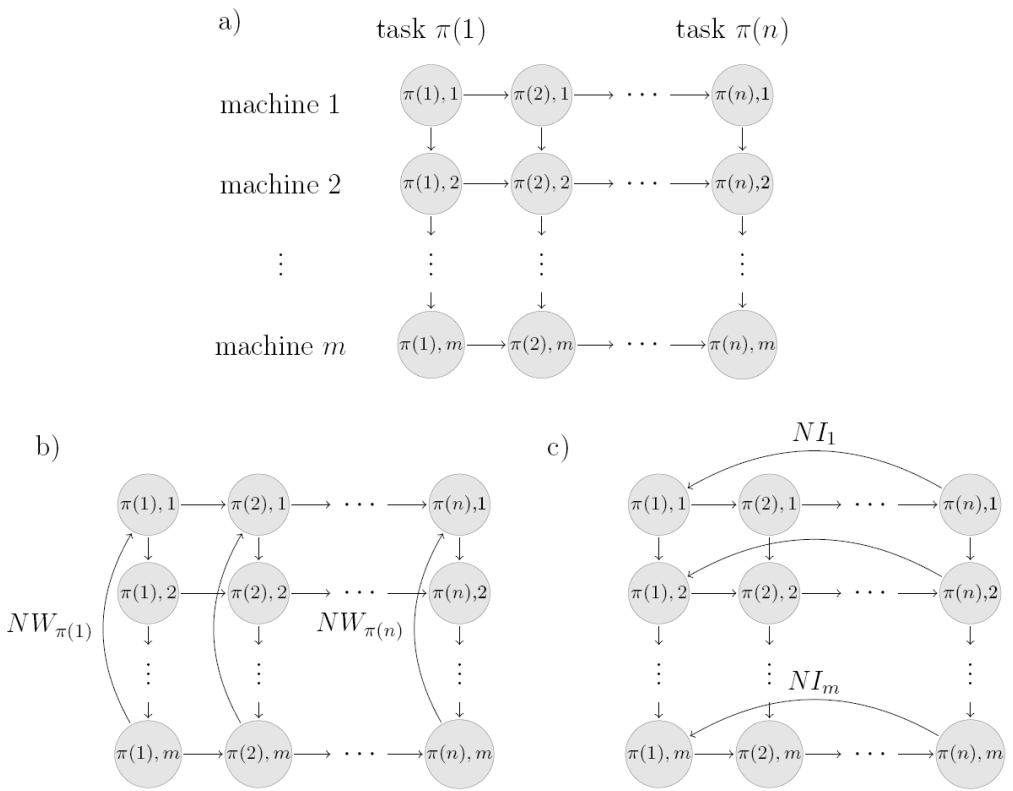

Figure 2: The schedule graph models: a) permutation model b) no-wait model c) permutation no-idle model.

When taking into account the additional no-wait constraint, it is convenient to transform the analyzed problem to the asymmetric traveling salesman problem [10]. However, it is also possible to consider the no-wait constraint by adding to the graph $G(\pi)$ the set

of loaded arcs

$$E^{NW} = \bigcup_{j \in J} \left\{ \Big((j,m),(j,1) \Big) \right\}. \tag{5}$$

The arc between the last and first operation of task $j \in J$ is loaded with negative sum of the duration of all operations of this task

$$NW_j = -\sum_{l \in M} p_{j,l}. \tag{6}$$

Graph representing the solution π of the problem with no-wait constraint is shown in Fig. 2.b. When considering the no-idle constraint, the graph model can be extended by the set of loaded arcs:

$$E^{NI} = \bigcup_{l \in M} \left\{ \Big((\pi(n),l),(\pi(1),l) \Big) \right\}. \tag{7}$$

Graph representing the sequence of tasks π with no-idle constraints is presented in Fig. 2.c. The arc between the last and the first operation performed on the machine $l \in M$ is charged with the sum of the duration of the operations performed on the machine

$$NI_l = -\sum_{j \in J} p_{j,l}. \tag{8}$$

There exist some similarities in the permutation-graph models of the analyzed problems. The structure of the graph presenting the flow shop schedule with no-wait constraint is the same as the structure of the graph of the permutation schedule with no-idle constraints, see Fig. 3. Having only the graph representing one said of schedules (with invisible names of vertices) it is not possible to recognize which of the schedules is modeled. Despite the similarity of graphs modeling the solutions, the properties of the problems are different. The difference results from the changes that occur in the graphs for the different permutations of tasks $\pi \in \Pi$. The problem with no-wait constraint could be equivalent to the permutation no-idle problem if the sequence of tasks execution was fixed and decision variable was the sequence of the machines. Similarly, if in the permutation no-idle problem it was possible to select a sequence of machines and the order of the tasks execution was fixed, then it would correspond exactly to the no-wait problem. The difference in the above mentioned problems is also visible in the acceleration of calculating the value of the objective function for the sequence of tasks $\pi \in \Pi$. In the case of no-wait constraint the preliminary calculations (with computational complexity $O(mn^2)$) can be done by designating the length of the schedule for each pair of tasks. Then, designation the length of the schedule would involve determination of the sum of corresponding values of each pair of adjacent tasks in the permutation π. The calculation complexity of determining the length of the schedule (not including single preliminary calculations) in this case is $O(n)$. A similar acceleration cannot be applied in the case of no-wait constraint, for which the complexity of determining the length of the schedule is $O(nm)$.

3. Experimental study

In this section the impact of additional no-wait and no-idle constraints on the length of the schedule is tested. The study is divided into three parts, in which the following issues are compared:

- the impact of additional constraint on the average length of the schedule,
- correlation of the length of the solutions in the flow shop problems with additional constraints,
- the quality of the generated solutions gathered by dedicated insert type algorithms.

The tests were carried out on well-known 120 examples proposed in [9]. These examples are divided into 12 groups of 10 instances. The groups vary in size instances, i.e., in the number of tasks n and/or the number of machines m. Each group is identified by the pair of $n \times m$.

3.1. Average length of schedules

This work studies the impact of additional constraints in the flow shop problem on the average length of schedules in each group instance. For each instance of the problem there was determined a set of $k = 10\,000$ of pseudo-random sequences of tasks execution. Then, on this basis there were determined 3 sets of schedules for the flow shop problems: permutation, no-wait constraint and no-idle constraint. More precisely, for each sequence there was created a schedule, one for each type, respectively. The sequence of the length of the permutation schedules was denoted by $X = (x_1, x_2, \ldots, x_k)$, the sequence of the length of the schedule with no-wait constraint as the sequence $Y = (y_1, y_2, \ldots, y_k)$, whereas the sequence of the length of the schedule with no-idle constraint as the sequence of $Z = (z_1, z_2, \ldots, z_k)$. Then, the reference length was determined as the length of the shortest generated permutation schedule $ref = \min_{i=1,\ldots,k} x_i$. For each schedule there was determined relative length F of the schedule with respect to the reference value of ref. For each instance and every constraint, with the set of relative length, there were determined: F_{min} – minimum, F_{avg} – average and F_{max} – maximum relative length as well as F_{dev} – standard deviation. The values of these parameters, averaged for each group, are given in Tab. 1.

3.2. Correlations of the length of schedules

The correlation between the lengths of the permutation, no-wait and no-idle schedules is now going to be tested. The sets of the relative length schedules, obtained in the previous experiment are used. For each instance, based on the sequences X, Y and Z of the relative length schedules the correlations between them is calculated. The averaged results for each group are shown in Tab. 2.

The length of schedules with two different constraints created for the same sequence of tasks forms a pair of numbers which may be marked on the chart as a point. A whole

Table 1: Average relative length of randomly schedules

Group	Type of schedule											
	permutation				no-wait				permu, no-idle			
$n \times m$	F_{min}	F_{avg}	F_{max}	F_{dev}	F_{min}	F_{avg}	F_{max}	F_{dev}	F_{min}	F_{avg}	F_{max}	F_{dev}
20×5	1.00	1.17	1.40	0.055	1.35	1.61	1.84	0.067	1.10	1.28	1.52	0.058
20×10	1.00	1.16	1.35	0.047	1.45	1.73	1.98	0.075	1.26	1.47	1.76	0.068
20×20	1.00	1.12	1.26	0.035	1.47	1.75	2.00	0.076	1.50	1.71	1.96	0.061
50×5	1.00	1.11	1.27	0.037	1.53	1.71	1.87	0.046	1.08	1.21	1.39	0.042
50×10	1.00	1.11	1.25	0.033	1.75	1.97	2.16	0.056	1.19	1.36	1.56	0.051
50×20	1.00	1.10	1.21	0.027	1.93	2.18	2.40	0.064	1.50	1.69	1.92	0.056
100×5	1.00	1.09	1.21	0.028	1.64	1.78	1.90	0.033	1.05	1.15	1.29	0.032
100×10	1.00	1.08	1.19	0.025	1.94	2.11	2.27	0.042	1.19	1.31	1.49	0.038
100×20	1.00	1.07	1.16	0.020	2.22	2.42	2.61	0.050	1.42	1.58	1.78	0.046
200×10	1.00	1.06	1.14	0.018	2.10	2.22	2.34	0.031	1.14	1.23	1.37	0.029
200×20	1.00	1.05	1.12	0.016	2.50	2.66	2.80	0.039	1.33	1.45	1.60	0.037
500×20	1.00	1.04	1.08	0.011	2.81	2.92	3.02	0.027	1.22	1.31	1.42	0.026
All	1.00	1.10	1.22	0.029	1.89	2.09	2.27	0.050	1.25	1.40	1.59	0.045

series of points generates a picture in a form of cloud. The correlation graphs for one instance is presented in Fig. 3. This instance (62-Taillard instance) from the group of 100×5 and has the highest correlation value (above 0.7). This correlation concerns Z lengths of no-idle time schedules and X lengths of permutation schedules. For other instances, the correlations were lower. Obtained graphs have a similar nature, but they are more blurred.

3.3. Insert type algorithm

The insertion sort algorithm *NEH* [3] is a dedicated design algorithm for the permutation flow shop problem with the criterion being the moment of completion of all performed tasks. Generally, *NEH* algorithm works iteratively, i.e., in each iteration it embeds another task into the schedule. Using the permutation-graph model, partial task sequence is build. Detailed description of the algorithm based on the permutation-graph model is presented in [4]. The choice of the position in the schedule in which the next task is inserted is made on the basis of the evaluation criteria for the created schedule. More precisely, the next task can be inserted in the k-th step in k positions. Thus, there are k sample schedules created by analyzing every possible insertion. The best of these

Table 2: Correlations of the relative length of schedules

Type of schedule: X - permutation; Y - no-wait; Z - permu, no-idle							
$n \times m$	$CorrXY$	$CorrXZ$	$CorrYZ$	$n \times m$	$CorrXY$	$CorrXZ$	$CorrYZ$
20×5	0.19	0.64	0.20	100×5	0.07	0.65	0.06
20×10	0.19	0.40	0.14	100×10	0.05	0.37	0.04
20×20	0.12	0.22	0.12	100×20	0.03	0.22	0.02
50×5	0.11	0.62	0.10	200×10	0.02	0.39	0.02
50×10	0.08	0.41	0.06	200×20	0.01	0.24	0.01
50×20	0.07	0.20	0.05	500×20	0.01	0.24	0.00
All	0.08	0.38	0.07	—	—	—	—

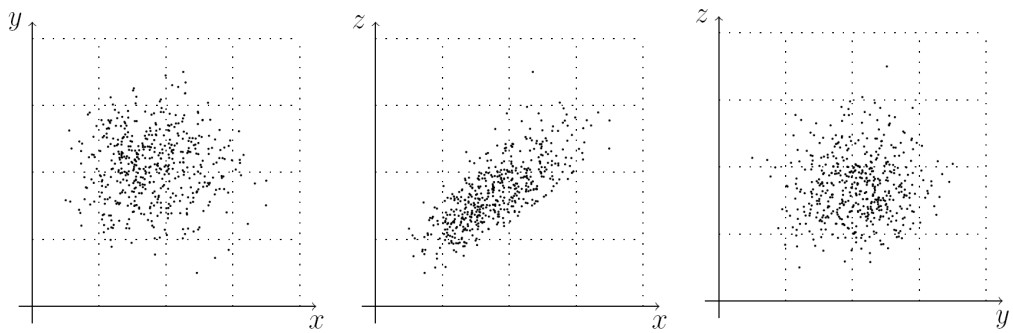

Figure 3: Visualization of the scaled correlations of the length: x permutation scheduling, y no-wait scheduling and z no-idle scheduling.

schedules determines the searched position for the added task. Since we analyze three cases of the flow shop problem with different constraints, it automatically generates three versions of algorithm: NEH, NEH^{NW} and NEH^{NI}. They differ one from another by created partial schedules. The classic NEH algorithm evaluates partial sequences of the length of the permutation schedule. Similarly, algorithms NEH^{NW} and NEH^{NI} evaluate the partial sequences of tasks on the basis of the length of schedules with no-wait and no-idle constraints. Let us assume that the final result of each of the algorithm is not the schedule itself, but the sequence of the tasks execution. Then, each sequence corresponds to the schedules with the imposed constraints (permutation schedule, no-wait and the permutation no-idle schedule). This means that it is possible to use any variant of NEH algorithm in each of the analyzed problems.

Table 3: Average relative length of schedules for NEH algorithms

Group	Type of schedule								
	permutation			no-wait			permu, no-idle		
$n \times m$	NEH	NEH^{NW}	NEH^{NI}	NEH	NEH^{NW}	NEH^{NI}	NEH	NEH^{NW}	NEH^{NI}
20×5	0.96	1.02	1.13	1.40	1.18	1.43	1.11	1.16	1.23
20×10	0.95	1.05	1.13	1.58	1.23	1.55	1.30	1.34	1.44
20×20	0.95	1.04	1.13	1.66	1.26	1.56	1.61	1.66	1.65
50×5	0.95	1.01	1.11	1.49	1.22	1.51	1.07	1.12	1.20
50×10	0.91	1.03	1.11	1.80	1.32	1.70	1.18	1.29	1.34
50×20	0.92	1.03	1.10	2.06	1.44	1.86	1.56	1.62	1.65
100×5	0.96	1.00	1.09	1.49	1.23	1.56	1.04	1.07	1.15
100×10	0.92	1.02	1.07	1.84	1.37	1.79	1.17	1.27	1.31
100×20	0.90	1.04	1.06	2.28	1.52	2.01	1.44	1.56	1.56
200×10	0.93	1.01	1.05	1.92	1.41	1.89	1.10	1.19	1.23
200×20	0.90	1.01	1.05	2.47	1.62	2.22	1.31	1.40	1.42
500×20	0.91	1.03	1.04	2.66	1.69	2.41	1.17	1.30	1.33
All	0.93	1.02	1.09	1.89	1.37	1.79	1.26	1.33	1.38

This paper studies generating 3 sequential executions of tasks, for each instance, with the help of algorithms NEH, NEH^{NW} and NEH^{NI}. Next, there are 3 schedules determined for each of the analyzed problem (generally 9 schedules for a given instance). There are relative length of each of the obtained solutions determined (relative to the reference solution from the first test). Averaged length of schedules with analyzed constraints derived from each of the versions of the algorithm are presented in Tab. 3.

4. Summary

Comparing the graph models of no-wait and no-idle constraints, one can expect that each of these constraints extend the production schedule in a similar way. Numerical study confirms this expectation. It can be seen in the examples where number of machines is the same as number of tasks. In the absence of symmetry, when the number of tasks exceeds the number of machines, no-wait restriction causes much greater increase of the schedule length then no-idle constraint.

Studies have shown a significant discrepancy in the correlation of the lengths of different types of schedules. The only significant correlation of the length of schedules is the length of the solution without constraints and with no-idle constraint. Even in the case of instances of size 20×20 significant correlation exists only between these types of schedules.

The no-idle constraint without breakdowns impedes a design of very efficient insertion sort algorithm. This is because the embedding of the next task in any place of a partial solution interferes with the whole schedule. Even inserting the task in the end of the schedule changes the moments of starting the operations from the initial positions of the schedule. For the rest of the analyzed problems such phenomenon does not occur.

In case of the problem with the no-idle time constrains numerical studies have confirmed ineffectiveness of the insert sort algorithm. For this problem, the algorithm NEH^{NI} generates the solution, on average, of 10% worse than the algorithm choosing the best solution from 10 000 random schedules. Similar algorithms for the permutation problem and the no-wait constraint problem provide solutions 7% and 28% better than the corresponding random algorithm, respectively.

Significant correlation between the quality of permutation and no-idle solution allows to propose solving the second problems with the use of the algorithm dedicated to the first problem. Then, the constraints from the second problem must be imposed on the obtained solution. In this way we can obtain the solution to the no-idle problem with the use of classical *NEH* algorithm. The quality of the obtained solution is improved approximately by 10% than in the case of using a dedicated NEH^{NW} algorithm and at the level of the best out of 10 000 random solutions.

References

[1] R. GRAHAM, E. LAWLER, J. LENSTRA and A. RINNOOY KAN: Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*, **5** (1979), 287-326.

[2] Y. GONCHAROV and S.SEVASTYANOV: The flow shop problem with no-idle constraints: A review and approximation. *European J. of Operational Research*, **196**(2), (2009), 450-456.

[3] M. NAWAZ, E.E. ENSCORE JR and I. HAM: A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem. *The Int. J. of Management Science*, **11** (1983), 91-96.

[4] E. NOWICKI and M. MAKUCHOWSKI: Insert method in classical schuduling problems. Part I. Flow shop problem. *Komputerowo Zintegrowane Zarządzanie*, WNT Warszawa, **2** (2001), 113-122, in Polish.

[5] H. RÖCK: The three-machine no-wait flow shop is NP-complete. *J. of Association for Computing Machinery*, **31**(2), (1984), 336-345.

[6] N. SAADANI, A. GUINET and M. MOALLA: A travelling salesman approach to solve the F/no-idle/Cmax problem. *European J. of Operational Research*, Elsevier, **161**(1), (2005), 11-20.

[7] F.C.R. SPIEKSMA and G.J. WOEGINGER: The no-wait flow-shop paradox. *Operations Research Letters*, **33**(6), (2005), 603-608.

[8] M. SVIRIDENKO: Makespan minimization in no-wait flow shops: A polynomial time approximation scheme. *J. on Discrete Mathematics*, **16**(2), (2003), 313-322.

[9] E. TAILLARD: Benchmarks for basic scheduling problems. *European J. of Operational Research*, **64** (1993), 278-285.

[10] D.A. WISMER: Solution of the flow shop scheduling problem with no intermediate queues. *Operational Research*, **20** (1972), 689-697.