Marek MAGIERA*

TWO-LEVEL OF PRODUCTION SCHEDULING FOR FLOW-SHOP SYSTEMS WITH INTERMEDIATE STORAGES

Abstract: The paper presents relaxation heuristics for production planning for multistage flow-shop with intermediate storages. The top-level is a stage loading, i.e., allocation of operations to the stages. The base-level is a task scheduling – allocation of operations to the stations. The time criterion is used in the minimisation function – the minimal production schedule is fixed. Results of computational experiments with the proposed method are presented.

Keywords: scheduling, heuristics, production planning, integer programming.

1. Introduction

Production schedules, which place the planned operations (tasks) in time and space, are developed on the basis of one of two possible approaches: monolithic or hierarchical (Sawik 1998). The former, a single-level approach, is characterised by simultaneous solution of all sub-problems (such as balancing station workloads, or task scheduling). Owing to global treatment of the problem, best solutions are obtained.

However, the enormous number of parameters and variables increases the amount of time required to perform computations, and in numerous cases precludes solving relatively large tasks. Therefore, in the case of significantly-sized tasks, a multilevel, or hierarchical, approach is employed. The global problem is divided into a series of sequentially solved partial tasks. Results obtained at a higher level provide data for a task solved at a lower level. One drawback of that concept, in comparison with the single-level approach, is a more significant deviation from the global optimum.

This paper discusses the latter, multilevel, approach as the developed production scheduling method serving the purpose of solving relatively large problems.

* AGH University of Science and Technology, Krakow, Poland

The task of building a production schedule has been divided into two consecutively solved problems, which are detailed in the subsequent section.

The consecutively solved problems have been expressed in the form of linear mathematical models. The condition that variables be integers has been ignored. The relaxed heuristics developed in such manner enables obtaining good results in a very short time. This has been confirmed by the findings of computational experiments described in Section 3. This is the reason why that algorithm is the preferred choice for rescheduling purposes. In the case of failure of one of the machines or emergence of another urgent order, it shall be possible to develop a new production schedule.

2. Two-level method of production scheduling

The method concerns unidirectional multistage flow-shop systems where operations with respect to many various types of products are performed simultaneously. Each stage is a set of machines operating in a parallel manner. A product passing through a given stage puts load only on a single machine. Some stages may be omitted. In between stages, there are intermediate buffers of limited capacities.

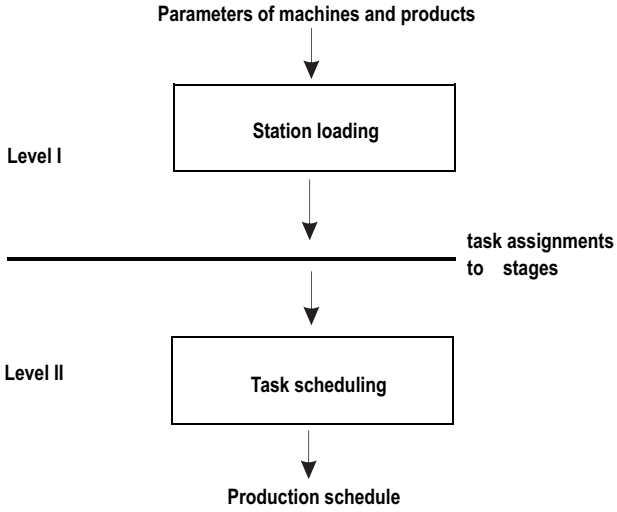

Fig. 1. *Block diagram of the method*

A block diagram of the developed two-level method is presented in Figure 1. At the top level, tasks (operations) are assigned to stages in such a way as to balance their loads. At the bottom level, tasks are separated in time and space – assigned to machines belonging to the stages to which the respective operations were assigned at the top level.

Tasks solved at the individual levels have been expressed in the form of mathematical relationships. They use symbols set forth in Table 1.

Table 1. *Specification of indices, input parameters and variables*

Indices:	e	– number of iteration;	k	–	product; $k \in K = \{1, ..., W\}$
	i	– machine; $i \in I = \{1, ..., M\}$	l	–	time interval; $l \in L = \{1, ..., H\}$
	j	– task (operation); $j \in J = \{1, ..., N\}$	v	–	stage; $v \in V = \{1, ..., \vartheta\}$

Input parameters:

a_{vj}	–	working space for task j at stage v;
b_v	–	working space of machine in stage v;
d_v	–	capacity of buffer located before stage v;
$g_{v\epsilon}$	–	transport time between machines in stage v and in stage ϵ;
m_v	–	number of machine in stage v;
p_{jk}	–	time for task j for product k;
φ_{il}	=	1, if machine i is accessible in time interval l; otherwise $\varphi_{il} = 0$;
D	–	the set of pairs, where machine i is assigned to stage v;
J_k	–	the set of tasks required for product k;
J_c	–	the set of tasks demanding part feeder $J_c \subset J$;
R_k	–	the set of pairs of tasks (j, r) for product k, such that task j must be performer immediately before task r;
V_j	–	the set of stages capable of performing task j;

Variables for iteration e:

$x^e_{vj} = 1$,	if task j assigned to stage $v \in V_j$; otherwise $x^e_{vj} = 0$ (for level I);
$z^e_{vjk} = 1$,	if product k is assigned to stage v to perform task j; otherwise $z^e_{vjk} = 0$ (for level I);
$q^e_{ikl} = 1$,	if product k is assigned to machine i in time interval l; otherwise $q^e_{ikl} = 0$ (for level II);
$y^e_{vkl} = 1$,	if product k in time interval l is assigned to intermediate buffer located before stage v; otherwise $y^e_{vkl} = 0$ (for level II);

s^e_{ik}, c^e_{ik} – times of start, completion of loading machine i by product k (for level II).

A production schedule with the shortest possible makespan is sought. The makespan has been divided into unit time intervals. The number of those intervals H is determined according to the following procedure (Magiera 2007a):

1. Using formula (1), determine δ_k – total operation time for product k.

$$\delta_k = \sum_{j \in J_k} p_{jk}; \quad k \in K \tag{1}$$

2. Using formula (2), calculate ψ – average time of machine load, rounded off to an integer.

$$\psi = \text{round}\left(\sum_{k \in K} \delta_k \Big/ M \right) \tag{2}$$

3. For each machine i determine $\tilde{l}_i$ – minimum number of time spans during which the machine is available and can be loaded during time ψ . Value of $\tilde{l}_i$ fulfils (3).

$$\tilde{l}_i = \sum_{\tau=1}^{\tilde{l}_i} \mu_{\tau l}; i = 1, \ldots, M \tag{3}$$

4. Using formula (4), determine the maximum number from among the values calculated in the previous step.

$$\lambda = \max\left\{\tilde{l}_1, \tilde{l}_2, \ldots, \tilde{l}_m\right\} \tag{4}$$

5. End the procedure by determining the number of individual time spans, using formula (5).

$$H = 1,2 \cdot \lambda \tag{5}$$

The value of the parameter H has been verified by means of computational experiments (see Section 3).

A description of two-level heuristics of production scheduling for flow-shop systems with intermediate storages follows:

Level I (stage load balancing)

Step 1. Assume iteration number $e := 1$ and solve relationship (6) ÷ (13) (Magiera 2007b):

Minimise:

$$P_{\max} \tag{6}$$

subject to:

$$\sum_{j\in J}\sum_{k\in K} p_{jk} z^e_{vjk} + \sum_{l\in L:\, l\leqslant\lambda}\ \sum_{i\in I:\,(i,v)\in D} (1-\varphi_{il}) \leqslant P_{\max};\ v \in V \tag{7}$$

$$\sum_{v\in V} z^e_{vjk} = 1; j \in J_k;\ \ k \in K \tag{8}$$

$$z^e_{vjk} \leqslant x^e_{vj}; v \in V;\ j \in J_k;\ \ k \in K \tag{9}$$

$$\sum_{v\in V} v \cdot z^e_{vjk} \leqslant \sum_{v\in V} v \cdot z^e_{vrk}; k \in K;\ \ (j,r) \in R_k \tag{10}$$

$$\sum_{v\in V_j} x^e_{vj} \geqslant 1; j \in J \tag{11}$$

$$x^e_{vj} = 0; j \in J;\ \ v \notin V_j \tag{12}$$

$$\sum_{j\in J_c} a_{vj} x^e_{vj} \leqslant b_v m_v; v \in V \tag{13}$$

In the linear mathematical model presented above, the load of the most loaded stage (6), determined according to (7), is minimised. The second addend at the left-hand side of inequality (7) allows for a limited availability of machines in the makespan estimated according to (4). The remaining relationships guarantee: (8) – allocation of all operations to the stages; (9) – allocation of products to stages to which relevant operations were allocated; (10) – maintenance of sequence limitations and unidirectional flow; (11) – allocation of each type of operations to at least one stage; (12) – elimination of assignment of tasks to inappropriate stages; (13) – maintenance of limited machine working space. Go to *Step 2*.

Step 2. Assume: $e := e + 1$. Determine allocations of tasks and products to stages according to (14).

$$x_{ij}^{e} = \text{ round}\left(x_{ij}^{e-1}\right), \;\; z_{vjk}^{e} = \text{ round}\left(z_{vjk}^{e-1}\right); \qquad i \in I,\; j \in J,\; k \in K,\; v \in V \tag{14}$$

If constraint (13) is fulfilled for each stage v, go to *Step 3*; otherwise, for each stage which did not fulfil condition (13) determine coefficients of additional cut-off constraints. Those coefficients: $C_v = \left\{j \;:\; \xi_{vj} = 1\right\} \subset J$ meet condition (15) (Sawik 1998). Return to *Step 1* and solve the task formulated there complemented with additional constraints (16), developed for stages v which has not met constraint (13).

$$\sum_{j \in J} a_{vj}\xi_j \geqslant b_v + 1 \;\wedge\; \sum_{j \in J} \left(1 - x_{vj}^{e}\right) \xi_{vj} < 1; \quad \xi_{vj} \in \{0, 1\} \tag{15}$$

$$\sum_{j \in C_v} x_{vj}^{e} \leqslant \overline{\overline{C}}_v - 1\,; \qquad i \in I, \quad e = 1 \tag{16}$$

Step 3. Using formula (17), determine the total time of performing a task for product k at stage v, and go to *Step 4*.

$$t_{vk} = \sum_{j \in J_k} p_{jk} z_{vjk}^{e}; v \in V; \;\; k \in K \tag{17}$$

Level II (distributing tasks in time and space)

Step 4. Solve the problem expressed as mathematical model (18) – (27).

Minimise:

$$\sum_{i \in I} \sum_{k \in K} \sum_{l \in L} l \cdot q_{ikl}^{e} \tag{18}$$

subject to:

$$\sum_{i \in I:\, (i,v) \in D} \sum_{l \in L} q_{ikl}^{e} = t_{vk}; v \in V; \;\; k \in K \tag{19}$$

$$\sum_{k\in K} q^e_{ikl} \leqslant \varphi_{il}; i \in I;\ \ l \in L \tag{20}$$

$$q^e_{ikl} + q^e_{\tau kf} \leqslant 1; k \in K;\quad (\tau, v), (i, v) \in D : i \neq \tau \tag{21}$$

$$\begin{gathered} l \cdot q^e_{ikl} - f \cdot q^e_{ikf} \leqslant t_{vk} - 1 + (1 - q^e_{ikf}) \cdot \alpha; \\ (i, v) \in D;\ \ l, f \in L : l > f;\ \ k \in K \end{gathered} \tag{22}$$

$$\frac{\sum\limits_{i\in I:\,(i,v)\in D} \sum\limits_{l\in L} l \cdot q^e_{ikl}}{t_{vk}} - \frac{\sum\limits_{\tau\in I:\,(\tau,\varepsilon)\in D} \sum\limits_{l\in L} l \cdot q^e_{\tau kl}}{t_{\varepsilon k}} - \frac{t_{vk} + t_{\varepsilon k}}{2} \geqslant g_{vk}; \tag{23}$$
$$k \in K;\ \ v, \varepsilon \in V :\ t_{vk},\ t_{\varepsilon k} > 0$$

$$\begin{gathered} \sum_{i\in I:\,(i,v)\in D} \sum_{l\in L} l \cdot q^e_{ikl} \Big/ t_{vk} - \sum_{\tau\in I:\,(\tau,\varepsilon)\in D} \sum_{l\in L} l \cdot q^e_{\tau kl} \Big/ t_{\varepsilon k} - \\ -0,5 \cdot (t_{vk} + t_{\varepsilon k}) - g_{vk} = \sum_{l\in L} y^e_{vkl} \\ k \in K;\ \ l, f \in L;\ \ v \in V - \{1\} :\ t_{vk} > 0;\ \ \varepsilon \in V\colon t_{\varepsilon k} > 0,\ v > \varepsilon, \\ \sum_{\psi=\varepsilon}^{v} t_{\psi k} = t_{vk} + t_{\varepsilon k} \end{gathered} \tag{24}$$

$$\begin{gathered} l \cdot y^e_{vkl} \geqslant l \cdot \sum_{f\in L} \sum_{\tau\in I:\,(\tau,\varepsilon)\in D} f \cdot q^e_{\tau kf} \Big/ t_{\varepsilon k} + \\ +0,5 \cdot (t_{\varepsilon k} + 1) + g_{vk} - \alpha \cdot (1 - y^e_{vkl}) \\ k \in K;\ \ l \in L;\ \ v \in V - \{1\} :\ t_{vk} > 0;\ \ \varepsilon \in V :\ v > \varepsilon,\ t_{\varepsilon k} > 0 \end{gathered} \tag{25}$$

$$\begin{gathered} \sum_{f\in L} \sum_{i\in I:\,(i,v)\in D} f \cdot q^e_{ikf} \Big/ t_{vk} - 0,5 \cdot (t_{vk} + 1) - l \cdot y^e_{vkl} \geqslant 1; \\ k \in K;\ l \in L;\ v \in V - \{1\} :\ t_{vk} > 0 \end{gathered} \tag{26}$$

$$\sum_{k\in K:\,t_{vk}>0} y^e_{vkl} \leqslant d_v; v \in V \setminus \{1\};\ \ l \in L \tag{27}$$

Parameter α, used in the notation of certain constraints of the models presented, is any integer which fulfils the following inequality: $\alpha > H$. The minimised sum (18) guarantees formation of shortest possible schedules. It also guarantees obtaining relatively short times of completing tasks for individual products. Further mathematical relationships guarantee: (19) – distribution of all tasks among machines; (20) – loading a machine during its availability in a given moment with a maximum of one task; (21) – product flow through a maximum of one machine at a given stage;

(22) – indivisibility of task performance in time and space; (23) – order of operation performance in an unidirectional flow-shop system in accordance with a given sequence and provision of time required for transport between stages; (24) – determination of time that given products spend in the individual buffers; (25), (26) – location of products in appropriate buffers at a given time (directly before performing subsequent operations); (27) – maintenance of the limited buffer capacities.
Assume: $e := e + 1$. Using (28) and (29), determine initial solution (s^e_{ik}, c^e_{ik} – times of start and completion of task performance for product k on machine i).

$$q^e_{ikl} = \text{ round}\left(q^{e-1}_{ikl}\right); \quad i \in I,\ k \in K,\ l \in L \tag{28}$$

$$s^e_{ik} = \min_{l \in L}\left(l q^e_{ikl}\right) \text{ for } q^e_{ikl} = 1, c^e_{ik} = s^e_{ik} + p_{jk} - 1; i \in I; k \in K \tag{29}$$

Having determined machine loads according to (30), go to *Step 5*.

$$q^e_{ikl} = \begin{cases} 1, \text{ if } s^e_{ik} \geqslant l \leqslant c^e_{ik} \\ 0, \text{ otherwise} \end{cases};\ i \in I;\ k \in K;\ l \in L \tag{30}$$

Step 5. a) In order to verify separation of task performance, assume $i := 0$ and go to *Step 5b*.
b) Assume $i := i + 1$ and $l := 0$, and go to *Step 5c*.
c) Let $l := l + 1$. If constraint (20) is fulfilled, go to *Step 6*, if not – go to *Step 5d*.
d) From among products which do not fulfil relationship (20), select one product k^* only according to the lexicographic order: 1 – product which did not fulfil those relationships in the previous iteration and thus was not selected, 2 – product of the smallest non-zero value s^e_{ik}, 3 – product of the smallest index k.

Assume $e := e + 1$. Let $\hat{j}$ mean a task performed in span l on machine i with respect to product k^*. Applying relationship (31), determine the number of time spans β during which product k^* requires machine load i (during a period from l until machine load i by that product ends) and elements of set $\tilde{K}$. Having modified the schedule according to (32) (Magiera 2005) and updated times $s^e_{\tau k}$, $c^e_{\tau k}$ $(\tau \in I,\ k \in K)$ according to (29), go to *Step 6*.

$$\begin{aligned} &\beta = c^{e-1}_{ik^*} - l + 1; \\ &\tilde{K} : \tilde{K} = \left\{k \in K \backslash \{k^*\} : q^{e-1}_{ikr} = 1; \quad r \in \langle l, l + \beta - 1\rangle;\ k \in K\right\} \end{aligned} \tag{31}$$

$$q^e_{\tau kr} = \begin{cases} q^{e-1}_{\tau kr} \text{for } \left((\tau \in I \backslash \{i\},\ k \in K) \vee \left(\tau = i,\ k \in K \backslash \tilde{K}\right)\right) \\ q^{e-1}_{\tau kf} \text{for } \tau = i,\ \ k \in \tilde{K},\ \ \text{where :} r \geqslant l + \beta,\ \ f = r - \beta \end{cases};\ r \in L \tag{32}$$

Step 6. In order to verify the order of task performance, check whether $s^e_{ik} = l$ for product k which loads machine i in time interval l. If not – proceed to *Step 7*, if yes – check whether relationship (33) holds. If that constraint is fulfilled, go to *Step 7*; otherwise, assume $e := e + 1$ and determine β – minimum number of time intervals by which s^e_{ik} needs to be increased in order to fulfil the relationship in question. Mark the analysed product $\bar{k}$ and having modified the schedule according to (34) and updated the times of task start and completion on the basis of (34), go to *Step 7*.

$$s^e_{ik} - c^e_{\tau k} \geqslant g_{vk} \text{ for } (i,v),(\tau,\varepsilon) \in D, \text{when} \sum_{\psi=\varepsilon}^{v} t_{\psi k} = t_{vk} + t_{\varepsilon k} \tag{33}$$

$$q^e_{\tau kr} = \begin{cases} q^{e-1}_{\tau kr} \text{ for } \left((\tau \in I \setminus \{i\},\ k \in K) \vee \left(\tau = i,\ k \in K \setminus \tilde{K}\right)\right) \\ q^{e-1}_{\tau kf} \text{ for } \tau = i,\ \ k = \bar{k},\ \ \text{when} : r \geqslant l + \beta,\ \ f = r - \beta \end{cases} ;\ \ r \in L \tag{34}$$

Step 7. Halt condition for the previous phases of schedule modification is checked here. If $l < \max\limits_{\tau \in I,\, k \in K} c^e_{\tau k}$, go to *Step 5c*, if not - check relationship: $i < m$. If the relation is fulfilled, go to *Step 5b*; otherwise, go to *Step 8*.

Step 8. Availability of buffers and machines is verified here. For that purpose, the loads of each machine and buffer in subsequent time intervals are reanalysed. The following are checked in succession:

a) availability of a buffer in time interval l, which is located before stage v – relationship (27) is checked;
b) availability of machine i in time interval l – on the basis on the value of parameter φ_{il}.

For each of the above items, the value of parameter β is determined, by which such schedule modification (operation "shift") should be made which would ensure non-disturbance of the limited buffer capacities (item a) or availability of a machine for loading by a given product, starting from time interval l (item b). Before each schedule modification, $e := e + 1$ shall be assumed. Modification should be made analogously to relationship (32), (34). After each modification, the start and completion times of loading individual machines with given products should be updated according to (29). Verification of the last time interval for a machine with the highest index shall end the algorithm. Start and completion times of loading individual machines with products and schedule length $C^h_{\max}$ are determined according to (35).

$$s^h_{ik} = s^e_{ik};\ \ c^h_{ik} = c^e_{ik} \ \text{ for } \ i \in I;\ \ k \in K;\ \ l \in L; C^h_{\max} = \max_{i \in I,\, k \in K} c^h_{ik} \tag{35}$$

3. Computational experiments

The presented two-level heuristics has been verified by means of computational experiments. A discrete optimisation package was used for computations (Fourer *et al.* 1993) The experiments covered 5 groups of tasks. Parameters of those groups and findings are presented in Figure 1. The findings include average values of two indices, which were determined for each test task. The indices serve the purpose of algorithm assessment, which is performed at two levels: the quality of findings (makespan) and the amount of time the computations require. The indices, which are defined below, enable comparison of the described heuristic conception of schedule formation with the two-level method which generates an optimal solution at each level. At each level of that method, a linear programming task is solved (level I: relationships (6) – (13), level II: relationships (18) – (27)), supplemented with decision-making variable binarity conditions, which are defined in Figure 1. Index h has been ascribed to he heuristic solution, and index $*$ to the optimal solution.

The indices are as follows:

- $\chi = (C^h_{\max} - C^*_{\max})/C^*_{\max}$ – value of relative error, where: $C^h_{\max}$, $C^*_{\max}$ – makespans determined by means of the following algorithms: heuristic (according to (35)), optimal.
- $\gamma = CPU^h/CPU^*$ – which serves the purpose of comparing calculation times: CPU^h – in the case of heuristics, CPU^* – in the case of the two-level method, which contains linear mathematical models with binary decision-making variables.

Table 2. *Specification of parameters of groups of task and computational results*

Gr.	Parameters of groups						χ [%]	γ [%]	Numbers of: machines: M, stages: ϑ, types of tasks: N, types of products: W, time intervals: H, ψ – sum of capacity of intermediate buffers.
	M	ϑ	N	W	H	ψ			
1	4	2	10	3	18	8	6.5	0.69	
2	5	2	12	4	20	10	6.4	0.64	
3	6	4	14	6	30	12	5.6	0.60	
4	6	4	16	7	40	12	5.2	0.58	
5	8	4	18	8	40	16	4.9	0.53	

Findings of computational experiments contained in Table 2 indicate significant reduction of computation time when using two-level heuristics with respect to the alternative conception - successively solved integer programming tasks, in the case of which optimum solutions are generated at every level. The reduction amounted to approximately $0.5 \div 0.7\%$, i.e. solutions were generated approximately $140 \div 200$ times faster than in the case of the two-level method with binary decision-making variables. However, that advantage is achieved at the expense of a minor deviation from the optimum. This is indicated by the average values of index χ, which did not exceed 6.5%, and the maximum value of the index reached 6.7%.

4. Conclusions

The presented two-level heuristics is particularly preferable in the case of relatively large task, characterised by a significant number of parameters and variables. Decomposition of a task into two sequentially solved problems reduces the number of constraints accounted for simultaneously, in which, when compared with the monolithic approach, there is a smaller number of parameters and variables. Introduction of relaxation into the algorithm had an effect on the possibility to solve complex problems in a short time. Thus, that algorithm is recommended to facilitate ongoing functioning of a production system - controlling the flow of goods through the system in the process of task execution.

Accounting for scheduled stoppages used for, among other things, maintenance, repairs, inspections, or refitting, resulted in better reflection of the production process. In the case of machine stoppages caused by damage, the developed heuristics can also be used to reschedule the production. It is possible owing to the very little amount of computation time required by that algorithm - in a very short time a new production schedule can be developed, which accounts for the changed properties of the machine park or a new urgent production order.

The two constructed linear mathematical models supplemented with decision-making variable binarity conditions, constitute a separate production planning method – quite time-consuming but one which enables generation of shorter schedules with respect to heuristic algorithms. The relationships contained in those models may be used to form a monolithic algorithm – in particular for relatively small problems.

References

Fourer R., Gay D., Kernighan B. 1993. *AMPL: A Modelling Language for Mathematical Programming*. Boyd & Fraser Publishing Company, Danvers, MA, USA.

Hall N.G., Sriskandarayah C. 1996. *A survey of machine scheduling problems with blocking and no-wait in process*. Operations Research, 44, pp. 510–525.

Magiera M. 2005. *Heurystyczne algorytmy planowania montażu dla elastycznej linii montażowej z maszynami równoległymi o ograniczonej dostępności*. [in:] Zastosowania teorii systemów, Monografie, Nr 3, Wydział Inżynierii Mechanicznej i Robotyki AGH, Kraków, pp. 185–194.

Magiera M. 2007a. *Analiza porównawcza trzech metod planowania produkcji dla systemów przepływowych bez magazynów*. Automatyka (półrocznik AGH), 11, 1–2, pp. 191–202.

Magiera M. 2007b. *Heurystyczne algorytmy szeregowania operacji dla systemów wytwarzania dokładnie na czas.* [in:] Waszkielewicz W. (Ed.), Zarządzanie organizacjami w gospodarce rynkowej, UWND AGH, Kraków, pp. 202–209.

Sawik T. 1998. *Badania operacyjne dla inżynierów zarządzania*. Wyd. AGH, Kraków.

Schmidt G. 2000. *Scheduling with limited machine availability*. European Journal of Operational Research, 121, pp. 1–15.

Tkindt V., Billaut J-C. 2002: *Multicriteria scheduling: theory, models and algorithms*. Springer.