BULLETIN OF THE POLISH ACADEMY OF SCIENCES
TECHNICAL SCIENCES, Vol. 61, No. 4, 2013
DOI: 10.2478/bpasts-2013-0100

A relaxation heuristic for scheduling flowshops with intermediate buffers

M. MAGIERA*

AGH University of Science and Technology, Faculty of Management, Department of Operations Research and Information Technology, 30 Mickiewicza Ave., 30-059 Cracow, Poland

Abstract. The paper presents a two-level relaxation heuristic for production planning for multistage flowshop systems with intermediate buffers. The method concerns unidirectional multistage systems where tasks with respect to many various types of products are performed simultaneously. The fixed and the alternative production routes are regarded in the method. The top-level is a stage loading, i.e., allocation of tasks among the stages. The base-level is a task scheduling – allocation of tasks among the stations. The linear mathematical models of mixed integer programming are used in the method. The time criterion is used in the minimization functions – the minimal schedule is fixed. The condition that variables are to be integers has been ignored in the heuristic. The relaxed heuristic developed in such a manner enables obtaining good results in a very short time. This paper discusses the multilevel approach as the developed production scheduling method serves the purpose of solving relatively large problems. Results of computational experiments with the proposed heuristic method are presented.

Key words: scheduling, heuristic, linear programming, production planning, decision making, flowshop.

1. Introduction

One of the basic tasks concerning production planning is building production schedules. These schedules, which take into consideration specific criteria for breaking down the tasks, are built for different time horizons. This paper is focused on short-term planning of product flow through production lines. Production schedules are built for different types of products. The issues presented in the paper, related to task scheduling theory, mathematical modelling, discrete programming, are included in the field of operational research.

Production schedules, which place the planned tasks (operations) in time and space, are developed on the basis of one of two possible approaches: monolithic or hierarchical. The former, a single-level approach, is characterized by simultaneous solution of all sub-problems (such as balancing station workloads, or task scheduling). Owing to global treatment of a problem, best solutions are obtained. However, the enormous quantity of parameters and variables increases the amount of time required to perform calculations, and in many cases precludes solution of relatively large tasks. Therefore, in the case of significantly-sized tasks, the 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 the more significant deviance from the global optimum.

This paper deals with the application of the hierarchical approach. A two-level task scheduling algorithm has been developed. It is used for building schedules of product flow through unidirectional production lines. The issues of task scheduling for production systems, related to the algorithm presented in the paper, was analysed in detailed in, among others, the works of [1] and [2]. These papers include not only the description of the issues related to task and resources distribution, but also the description of systems, applied criteria as well as concepts used in the scheduling algorithms built for various configurations of production systems.

The algorithm presented in this paper applies to flexible production systems. Flexibility is understood here as the capacity of the system for simultaneous short-series (including single-unit) production of a number of different products at high performance of the entire system [3]. The paper [3] specifies typical types of flexibility, both at the level of the machine as well as at the level of production system management. It referred to, among others, flexibility of production assortment and flexibility of production volume. These two types of flexibility specifically refer to production lines for which the task scheduling algorithm has been developed. These systems feature the capacity of quick and economic transition do production of new types of goods (flexibility of production assortment) and the related production profitability, even with small volumes (production volume flexibility).

Task scheduling methods may be broken down into two groups: those which allow determination of optimum solutions (in view of the criteria taken into account) and those which are used to determine approximate solutions which are slightly deviated from the optimum.

A large number of task scheduling methods for flow systems is based on mathematical programming. The developed algorithm takes advantage of this tool. The problem to be

*e-mail: mmagiera@zarz.agh.edu.pl

solved was formulated in the form of linear mathematical relationships. The art of building mathematical models is presented in, among others, [4] and [5].

The first of the models presented in this paper was inspired with [3] and [6]. These papers present models of discrete optimization tasks used for machine load-balancing. For the algorithm presented in this paper, a stage load-balancing model was built, in which every stage constitutes a set of machines working in parallel. The other works of the same author [7] and [8], in which integer programming has also been used, are similar in terms of the application of the two-level approach to product flow planning. At the top level of both works, machine workload is balanced, with scheduling of tasks at the lower level. These works, related to flexible assembly systems, cover the issues of the feeder layout. The tasks have been also emphasised in this paper which require using parts. Unlike with the concept of task scheduling described in the works [7] and [8], this paper presents the method in which the allocation of the tasks to the machines is done as late as at the level II. This results in better route flexibility. The presented method differs also with a number of other aspects, e.g. taking into consideration limited availability of the machines or setting intermediate buffers loads.

The algorithm presented in the paper is related, however, to the group of task scheduling methods which enables determination of approximate solutions. The grounds for building approximate algorithms and the related issues are described in [9]. In the mathematical models prepared for the algorithm presented here, the conditions for discrete decision variables were rejected. It clearly required building approximation procedures of the obtained results and verification and modification of the obtained solutions. The said activities are characteristic of relaxation heuristics which includes the developed algorithm.

With the application of the developed heuristic method, problems of significant scale may be solved in a relatively short time. Moreover, application of the hierarchical approach enables taking into consideration a large number of parameters and variables in mathematical relationships built for the problems solved one by one.

Taking into consideration limited availability of machines is a characteristic of the developed algorithm. It was inspired with [10]. Thus, planned demurrage of machines may be taken into account, for example allocated for maintenance or refitting.

The literature covering the issues of task scheduling for flow systems is very rich. The works [11] and [12] are dedicated to the presentation of the review of the used methods. The authors of the work [12] have classified these methods. They have described methods used for determination of optimum solutions, heuristics, hybrid approaches and simulation/decision support system (DSS). Higher importance of hybrid methods is indicated in the work, the methods which use combinations of two or more tools to find the solution. The classification of the methods used for task scheduling for multi-stage assembly systems is described in the work [13]. It has to be emphasised here that these methods include also designing systems, e.g. defining the locations of part feeders (which is also taken into account in the method described in this paper) or optimisation of the configuration of a production line. Among the developed methods for building schedules of product flow through production lines, heuristics play a major part, including those methods which are presented in this work.

The overview of the used heuristic methods may be found in [14]. A more recent article covering the review of task scheduling methods, not only for production systems, is [15]. A clear majority from among over two hundred articles in its bibliography refers to heuristic algorithms. The authors of the paper briefly reviewed task scheduling in production parallel machines or processor working in parallel. On the basis of the analyses of over 200 articles, many summaries were developed. For example, it was proven that mathematical programming is related to about 15% of the analysis methods. The overview of the used task scheduling criteria proved a major dominance of the scheduling length criterion (this criterion was also taken into account in the algorithm developed in this paper) over other criteria. A definite majority of the methods took into account only one criterion. Multicriteria methods constituted as little as 2% of all those taken into account.

Many developed algorithms which give very good results for the applied criteria are developed for a very limited, small number of machines (as with, for example, [16]) or a number of stages in flow systems. Examples of such algorithms for a two-stage system is provided in [17] (where an optimum solution is found), and heuristics described in [18] (in the first stage here only one machine may be located), whereas heuristics for three-stage systems are included in, among others, in [19]. The papers [17–19], just like in case of the algorithm described in the following sections of this paper, minimize scheduling length. The algorithm presented in this paper refers to multistage systems. However, in case of solving tasks of relatively large scale (including a major number of stages, machines and products, for which tasks are executed), a problem arises as regards limited capacity of the employed software of discrete optimisation packages. However, the current development of computer technology and software will allow solving tasks of ever increasing scale.

The algorithm presented here is also adjusted to functioning of assembly systems as a special case of production systems. The issues related to building assembly schedules are described in, among others, [3]. For assembly systems, apart from distribution of tasks among the machines, part feeders are also distributed. Execution of many assembly tasks requires setting up a corresponding part feeder near the assembly station. A separate group of variables was taken into account in the developed algorithm, used for distribution of feeders, just like in [6] and [20]. With this, the algorithm is also used for supporting assembly systems designing.

The detailed presentation of the developed algorithm is included in the following sections of the paper. Section 2 is dedicated to the description of the problem and the concept of its solution with the two-level approach. Section 3 features linear mathematical models used in relaxation heuristics described in Sec. 4. The results of calculation experiments used for verification of the developed mathematical models and the heuristic algorithm are given in Sec. 5.

2. The description of the problem and the concept of its solution

This section features the description of the problem as well as the hierarchical method idea used for solution of this problem. The description of the system configuration is presented here for which the method has been built. An example of such a configuration is given in Fig. 1. The markings used in the description of the problem correspond with the summary of all the used markings given in Table 1.

The method used for distribution of tasks in time and space was developed for multistage flow systems. Let $V = \{1, \ldots, \vartheta\}$ be the given set of stages. Every stage constitutes a set of machines working in parallel. It includes identical parallel machines, that is machines which perform the same functions. It means that all the machines within the given stage $v \in V$ are capable of execution of the same types of tasks and work at the same speeds. Each of the machines $i \in I$ belongs exactly to one stage. The allocation of the machines to the stages is known from the set D, which is a set of arranged pairs (i, v), where the machine i belongs to the stage v. With these machines, tasks of the type $j \in J$ shall be conducted which are assigned to the products $k \in K$. Obviously, for each product $k \in K$ not all types of tasks have to be performed. The set $J_k \subset K$ includes the list of types of tasks executed for the product $k \in K$. Execution of the tasks for individual products $k \in K$ requires also taking into account sequencing limitations saved in the set R_k. Execution of some tasks requires using auxiliary equipment, for example part feeders. Assembly systems in which part feeders are used are a good example. Many assembly tasks consist in adding parts or sets of parts (sub-assemblies) to the parts already installed. In this case, execution of the assembly task with a specific machine requires assigning to this machine a corresponding feeder from which the part to be added is collected. Execution of the first task, according to the given assembly sequence, which consists in fixing a base element in the grip, also requires a feeder from which the base part is collected. Examples of process tasks which do not require using part feeders are: welding, pressure welding, soldering, turning and cutting. Placing a specific feeder in the stage requires a pre-defined work space to be occupied. It means taking limited space in this stage, which is allocated for setting auxiliary equipment, each of which may take different space dimensions. Distribution of tasks which require the use of part feeders requires limited working space to be taken into account. For this purpose, the set $J_c \subset J$ has been defined which reflects the list of tasks which require the use of part feeders.

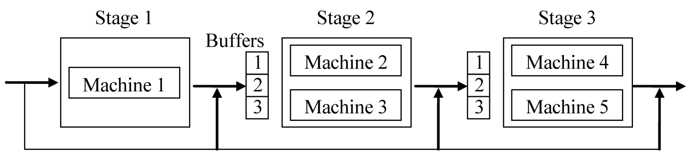

Fig. 1. Diagram of multistage system with intermediate buffers

Table 1

Specification of indexes, input parameters and variables

Indexes:			k	–	product; $K \in K = \{1, \ldots, W\}$
i	–	machine; $i \in I = \{1, \ldots, M\}$	l	–	time interval; $l \in L^* = \{1, \ldots, H^*\}$
j	–	task (type of task); $j \in J = \{1, \ldots, N\}$	v	–	stage; $v \in V = \{1, \ldots, \vartheta\}$

Input parameters:		
a_{vj}	–	working space of machine in stage v required for execution of task j;
b_v	–	total working space of the machine placed in stage v;
d_v	–	capacity of buffer located before stage v;
g_{vk}	–	transport time for product k from the stage in which tasks has been completed to stage v;
m_v	–	number of machines in stage v;
p_{jk}	–	processing time for task j of product k;
η_{il}	=	1, if machine i is accessible in time interval l, otherwise $\eta_{il} = 0$;
D	–	the set of arranged pairs (i, v), such that the machine i belongs to stage v;
J_k	–	the set of tasks required for product k, $J_k \subset J$;
J_c	–	the set of tasks which require using the feeder for the parts, $J_c \subset J$;
R_k	–	the set of pairs of tasks (j, r) for product k, such that task j is executed immediately before task r, $j \in J_k$, $r \in J_k$;
V_j	–	the set of stages in which the machines are capable of execution of task j;

Variables:	
$x_{vj} = 1$,	if type of task j is assigned to stage $v \in V_j$, otherwise $x_{vj} = 0$ (for the level I);
$z_{vjk} = 1$,	if product k is assigned to stage v to perform task j otherwise $z_{vjk} = 0$ (for the level I);
$q_{ikl} = 1$,	if product k is assigned to machine i in time interval l, otherwise $q_{ikl} = 0$ (for the level II);
$y_{vkl} = 1$,	if product k in time interval l is assigned to intermediate buffer located before stage v, otherwise $y_{vkl} = 0$ (for the level II);

The method has been developed for an unidirectional flow system. Passing through the given stage, the product only loads one machine from among those working in parallel. Thus, there is no necessity to move the product between parallel machines which belong to the same stage. Some stages may be omitted.

As Fig. 1. shows, intermediate buffers of limited capacity are located between the stages. If the next task for the product k cannot be executed due to loading of all the machines in the stage in which the task is to be executed, the product will stay in the buffer and wait. It is placed in the buffer preceding the stage in which the next task is to be executed.

For the described flow system, the operation scheduling task may be presented as follows:

For the given configuration of an unidirectional flow system, with the data which describe the stock of machines and the parameters related to the products which flow through the system at the same time (given in Table 1), the schedule of product flow of the lowest possible length has to be developed. Thus, the problem of optimization has to be solved which takes into account the time criterion in the form of the schedule length.

In distribution of tasks in space (assignment to machines) and in time (defining periods of machine loading with the assigned tasks), two types of production routes have to be considered:

- fixed routes: production routes in which each type of tasks (and, if necessary, the corresponding part feeder) is allocated to the machines of the same stage;
- alternative routes: production routes in which each type of tasks (and, if necessary, the corresponding part feeder) is allocated to at least one machine. The machines which were assigned this type of tasks may thus belong to different stages.

Solution of the problem thus requires not only distribution of the tasks among the machines but also allocation of the feeders.

In order to visualise the difference between fixed and alternative production routes, the following example may be used: for the given set of products $K = \{A, B\}$, the tasks have been assigned: $J_A = \{1, 2, 3\}$, $J_B = \{1,\ 3,\ 4\}$. Thus, there are 4 types of tasks. However, 5 tasks have to be executed, because a type 1 task will be executed twice: once for the product A and once for the product B. The set of stages is given: $V = \{v_1, v_2\}$, where every stage includes one machine. In case of fixed production routes, a type 1 task will be assigned to the stage v_1 or to the stage v_2 (to at least one machine assigned to the stage). It means that the type 1 task will be executed for both products in the stages v_1 or v_2. As compared with alternative routes, apart from allocation of the type of tasks to one stage, allocation of this type of tasks to both stages is also allowed. Then, type 1 tasks for individual products may be assigned to different stages. If, for example, a type 1 task consists in execution of a specific welded connection, welding machines may be located in 2 stages and both products will be welded at the same time. Each task is indivisible in time and in space, thus it is executed in one machine.

Taking into consideration alternative production routes is mostly related to the necessity of incurring relatively larger costs due to the use of auxiliary equipment, as compared with fixed routes. However, this type of routes helps build schedules of relatively shorter length.

The developed heuristic algorithm used to solve the described problem is based on the hierarchical approach. A block diagram of the developed two-level method is presented in Fig. 2. 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 tasks were assigned at the top level.

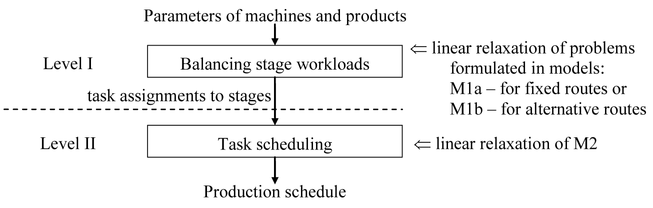
Parameters of machines and products
Level I
Balancing stage workloads
⇐ linear relaxation of problems formulated in models: M1a – for fixed routes or M1b – for alternative routes
task assignments to stages
Level II
Task scheduling
⇐ linear relaxation of M2
Production schedule

Fig. 2. Block diagram of the two-level method

Attention should be paid to the fact that at the level I the problem of balancing stage workload is solved, and not balancing machine workload. Thus, at the level II there are more possibilities of distribution of tasks, and they are assigned to the machines of the stage selected at the level I. The examples of the tasks of balancing machine workload may be found in: [3, 9, 20].

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

A production schedule with the shortest possible makespan is sought. The makespan has been divided into unitary time intervals l, where $l \in L^* = \{1, \ldots, H^*\}$. Taking into consideration too high a number of the time intervals may result in a major increase in the size of the problem, which may result in a relatively long calculation time or lack of the possibility of finding any solution to the problem due to limited possibilities of the discrete optimization packets. A low value of the time intervals may result in the inability to solve the problem when machines should be loaded for a longer time. In order to avoid these unfavourable cases, the built procedure for determination of the number of the time intervals H $(H \leq H^*)$, taken into account in the prepared mathematical models. The number of the analysed time intervals H has been verified in the calculation experiments described in Sec. 5.

The number of those time intervals H is determined according to the following procedure:

1. Following (1), determine δ_k – total processing time for product k.

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

2. On the basis of the relationship (2) determine the average time of machine loading (without transport times), rounded off to the nearest integral number.

$$\bar{\delta} = \text{round}\left(\frac{\sum\limits_{k\in K}\delta^k_{\min}}{M}\right) \tag{2}$$

3. For each machine i estimate its minimum loading ω_i, which takes into consideration the limited availability of the machines. For this purpose, assume $i := 1$ and execute the following:
   (a) Assume $\omega_i := 1$ and go to Step 3b.
   (b) If the condition (3) is met for machine i, go to Step 3d, otherwise go to Step 3e.

$$\sum_{\tau=1}^{\omega_i}\eta_{i\tau} = \bar{\delta} \tag{3}$$

   (c) If $\omega_i < H^*$ (H^* – the number of time intervals) assume $\omega_i := \omega_i + 1$ and return to Step 3b. If $\omega_i = H^*$ and the condition (3) is not met, increase the value H^* in order to allow the executing of all the assembly tasks or increase the availability of the machine in the following periods.
   (d) If $i < M$ (M – the number of the machines), go back to Step 3a; otherwise go to Step 4, which will take into account the determined values ω_i for $i = 1,\ldots,M$ constitute the estimate of the minimum loading for individual machines.
4. Estimate, according to (4), the value of loading for the most loaded machine.

$$C^{LB}_{\max} = \max_{i\in I}\omega_i \tag{4}$$

5. The number of the analysed time intervals H, where $H \le H^*$, meets the condition (5):

$$H = \text{round}\left(1.2\cdot C^{LB}_{\max}\right) \tag{5}$$

$L = \{1,\ldots,H\}$ – the set of time intervals, regarded in the algorithm, $L \subset L^*$.

3. The linear mathematical models

The three linear mathematical models are presented in this section. The mixed integer programming is used in the approach. The M1a and M1b models have been built for the level I of the method. They are used for load balancing of stages and distribution of tasks between stages. The M2 model for dedicated is for the level II. Tasks assigned to the stages (at the level I) are distributed among the machines. This model is dedicated for construction of schedules of product flow through production lines.

Models. M1a (for fixed routes) **and M1b** (for alternative routes). Balancing stage workloads.

Minimize:

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

subject to:

$$\sum_{j\in J}\sum_{k\in K}\frac{p_{jk}}{m_v}z_{vjk} + \sum_{l\in L,\, l\le\lambda}\ \sum_{i\in I,(i,v)\in D}(1-\eta_{il}) \le P_{\max};\quad v\in V \tag{7}$$

$$\sum_{v\in V} z_{vjk} = 1;\quad j\in J_k;\quad k\in K \tag{8}$$

$$\sum_{\varepsilon\in V:\varepsilon\ge v} x_{\varepsilon r} \ge x_{vj};\quad k\in K;\quad (j,r)\in R_k;\quad v\in V \tag{9a}$$

for the M1a model only

$$\sum_{v\in V:v\le\varepsilon} x_{vj} \ge x_{\varepsilon r};\quad k\in K;\quad (j,r)\in R_k;\quad \varepsilon\in V \tag{9b}$$

for the M1a model only

$$\sum_{\varepsilon\in V:\varepsilon\ge v} z_{\varepsilon rk} \ge z_{vjk};\quad k\in K;\quad (j,r)\in R_k;\quad v\in V \tag{9c}$$

for the M1b model only

$$\sum_{v\in V:v\le\varepsilon} z_{vjk} \ge z_{\varepsilon rk};\quad k\in K;\quad (j,r)\in R_k;\quad \varepsilon\in V \tag{9d}$$

for the M1b model only

$$z_{vjk} \le x_{vj};\quad v\in V;\quad j\in J_k;\quad k\in K \tag{10}$$

$$\sum_{v\in V_j} x_{vj} = 1;\quad j\in J \tag{11a}$$

for the M1a model only

$$\sum_{v\in V_j} x_{vj} \ge 1;\quad j\in J \tag{11b}$$

for the M1b model only

$$\sum_{j\in J_c} a_{vj}x_{vj} \le b_v m_v;\quad v\in V \tag{12}$$

$$x_{vj} = 0;\quad j\in J;\quad v\notin V_j \tag{13}$$

$$x_{vj}, z_{vjk} \in \{0,1\};\quad j\in J,\quad k\in K,\quad v\in V. \tag{14}$$

In the linear mathematical M1 model presented above, the load of the most loaded stage (6), determined according to (7), is minimized. The second component of the inequality (7) allows for a limited availability of machines in the makespan estimated according to (4). The remaining constraints guarantee: (8) – allocation of all tasks among the stages; (9) – maintenance of sequence limitations and unidirectional flow, where the constraints (9.a) and (9.b) refer to fixed production routes, and (9.b) and (9.c) to alternative routes: each type of the tasks which is executed as the next one is allocated to the same stage in which the previous task was executed or to a stage of a higher number; (10) – allocation of products to stages to which relevant tasks were allocated; (11.a) – allocation of each type of tasks to one stage only – constructed for the M1a model; (11.b) – allocation of each type of tasks to at least one stage – for the M1b model; (12) – maintenance of the limited machine working space; (13) – elimination of assignment of tasks to inappropriate stages; (14) – binary of decision-making variables.

The constraints (9), ensuring such allocation of the tasks to the stages which ensure maintaining of the given order limitations have been separately formulated for fixed routes (9.a and 9.b) and for alternative routes (9.c) and (9.d). The relationships (9.c) and (9.d) are also true in case of fixed routes. However, it is not necessary to use them in case of fixed routes, for which the given type of tasks is assigned only to one stage. Through this stage, all the products flow to which this type of tasks is assigned. Thus, while building the discussed constraints related to fixed routes, taking into account the x_{vj} variable will be sufficient. In case of alternative routes, it is not sufficient, thus the z_{vjk} variable has been taken into account, because the product k, for which the task j is executed, flows through one of the stages to which this type of tasks is assigned, and thus stage is marked with index v.

The solution of the problem formulated for M1 is the input for the task scheduling problem. The equality (15) describes the new parameter t_{vk} – production time for tasks of the product k for the stage v.

$$t_{vk} = \sum_{j\in J_k} p_{jk} z_{vjk}; \quad k\in K; \quad v\in V. \tag{15}$$

Model M2 constructed for task scheduling (for the level II):

Minimize:

$$\sum_{i\in I}\sum_{k\in K}\sum_{l\in L} l\, q_{ikl} \tag{16}$$

subject to:

$$\sum_{i\in I,\,(i,v)\in D}\ \sum_{l\in L} q_{ikl} = t_{vk}; \quad v\in V; \quad k\in K \tag{17}$$

$$\sum_{k\in K} q_{ikl} \le \eta_{il}; \quad i\in I; \quad l\in L \tag{18}$$

$$\begin{gathered} q_{ikl} + q_{\tau kf} \le 1; \quad k\in K; \\ (\tau,v),(i,v)\in D; \quad i\ne\tau; \quad l,f\in L \end{gathered} \tag{19}$$

$$\begin{gathered} l q_{ikl} - f q_{ikf} \le t_{vk} - 1 + \alpha\left(1-q_{ikf}\right); \quad (i,v)\in D; \\ l,f\in L; \quad l>f; \quad k\in K \end{gathered} \tag{20}$$

$$\begin{gathered} \frac{\sum\limits_{i\in I,\,(i,v)\in D}\ \sum\limits_{l\in L} l q_{ikl}}{t_{vk}} - \frac{\sum\limits_{\tau\in I,(\tau,\varepsilon)\in D}\ \sum\limits_{l\in L} l q_{\tau kl}}{t_{\varepsilon k}} \\ -\frac{t_{vk}+t_{\varepsilon k}}{2} \ge g_{vk}; \\ k\in K; \quad \varepsilon,v\in V; \quad \varepsilon<v; \quad t_{\varepsilon k},t_{vk}>0; \end{gathered} \tag{21}$$

$$\begin{gathered} \frac{\sum\limits_{i\in I,\,(i,v)\in D}\ \sum\limits_{l\in L} l q_{ikl}}{t_{vk}} - \frac{\sum\limits_{\tau\in I,(\tau,\varepsilon)\in D}\ \sum\limits_{l\in L} l q_{\tau kl}}{t_{\varepsilon k}} \\ -\frac{t_{vk}+t_{\varepsilon k}}{2} - g_{vk} = \sum_{l\in L} y_{vkl}; \\ k\in K; \quad v\in V-\{1\}; \quad \varepsilon\in V; \\ t_{\varepsilon k},t_{vk}>0; \quad \varepsilon<v; \quad \sum_{\psi=\varepsilon}^{v} t_{\psi k} = t_{vk}+t_{\varepsilon k} \end{gathered} \tag{22}$$

$$\begin{gathered} l y_{vkl} \ge \frac{\sum\limits_{f\in L}\ \sum\limits_{\tau\in I,\,(\tau,\varepsilon)\in D} f q_{\tau kf}}{t_{\varepsilon k}} \\ +\frac{t_{\varepsilon k}+1}{2} + g_{vk} - \alpha\left(1-y_{vkl}\right) \\ k\in K; \quad l\in L; \quad v\in V-\{1\}; \\ \varepsilon\in V; \quad t_{\varepsilon k},t_{vk}>0; \quad \varepsilon<v; \end{gathered} \tag{23}$$

$$\begin{gathered} \frac{\sum\limits_{f\in L}\ \sum\limits_{i\in I,\,(i,v)\in D} f q_{ikf}}{t_{vk}} - \frac{t_{vk}-1}{2} - l y_{vkl} \ge 1; \\ k\in K;\ l\in L;\ v\in V-\{1\};\ t_{vk}>0 \end{gathered} \tag{24}$$

$$\sum_{k\in K,\ t_{vk}>0} y_{vkl} \le d_v; \quad v\in V\backslash\{1\}; \quad l\in L \tag{25}$$

$$q_{ikl}, y_{vkl} \in \{0,1\}; \quad i\in I; \quad k\in K; \quad l\in L; \quad v\in V. \tag{26}$$

Parameter α, used in the notation of certain constraints of the presented models, is any integer which fulfils the following inequality: $\alpha > H$. The minimized sum (16) guarantees formation of shortest possible schedules. It also guarantees obtaining relatively short times of completing tasks for individual products. Further mathematical relationships guarantee: (17) – division of all tasks between machines; (18) – loading a machine during its availability in a given moment with a maximum of one task; (19) – product flow through a maximum of one machine in a given stage.

Another constraint (20) ensures integrity of the tasks in time and space: each task for the given product is executed in one machine on the continuous basis, that is it cannot be interrupted. It is assigned to the consecutive time intervals whose number is equal to t_{vk}: the time of loading the stage v by the product k. The task is assigned to one of the machines assigned to the stage v. If the tasks are executed for the product k assigned to the machine i ($(i,v)\in D$) and it begins in the time interval f, and ends in the time interval l, then $q_{ikl} = q_{ikf} = 1$, and also $q_{ikr} = 1$ for $r\in L$, where: $f<r<l$. Then, the relationship is maintained: $lq_{ikl} - fq_{ikf} \le t_{vk} - 1$, presented in the constraint (20). The component $\alpha(1-q_{ikf})$ included in the constraint (20) makes it condition to be met also when the product k is assigned to another machine, then $q_{ikf} = 0$, and the right side of the inequality takes a significant value.

In the constraints (21)–(24) the times have been taken into account for the beginning s_{vk} and the ending c_{vk} of the execution of the tasks in the machine for the product k in the stage v. They may be defined as follows:

$$\begin{gathered} s_{vk} = \frac{\sum\limits_{i\in I:(i,v)\in D}\ \sum\limits_{l\in L} l q_{ikl}}{t_{vk}} - \frac{t_{vk}}{2} + \frac{1}{2}; \\ c_{vk} = \frac{\sum\limits_{i\in I:(i,v)\in D}\ \sum\limits_{l\in L} l q_{ikl}}{t_{vk}} + \frac{t_{vk}}{2} - \frac{1}{2}; \\ v\in V; \quad k\in K; \quad t_{vk}>0. \end{gathered}$$

The constraint (21) ensures maintaining the order of execution of the tasks in an unidirectional flow system. The product flowing through the production system brings load on the machines placed in the stages with increasing values of stage indexes v. Let the product k brings load first in the stage ε, and then in the stage v ($\varepsilon < v$). The relationship is important here between s_{vk}, the time of beginning the execution of the task for the product k in the stage v and $c_{\varepsilon k}$, the time of ending the execution of the task in the stage ε. The transport time between the stages has to be taken into account in the developed relationships, marked g_{vk}.

The developed relationship comes in the form: $s_{vk} - c_{\varepsilon k} \geq 1 + g_{vk}$ for $k \in K$, ε, $v \in V$, where $\varepsilon < v$ (an unidirectional flow).

After entering the defined times for the beginning and the ending of the execution of the tasks, this relationship takes the form:

$$\frac{\sum\limits_{i \in I:(i,v) \in D} \sum\limits_{l \in L} lq_{ikl}}{t_{vk}} - \frac{t_{vk}}{2} + \frac{1}{2} - \left(\frac{\sum\limits_{\tau \in I:(\tau,\varepsilon) \in D} \sum\limits_{l \in L} lq_{\tau kl}}{t_{\varepsilon k}} + \frac{t_{\varepsilon k}}{2} - \frac{1}{2} \right) \geq 1 + g_{vk};$$
$$k \in K; \quad \varepsilon, v \in V; \quad \varepsilon < v; \quad t_{\varepsilon k}, t_{vk} > 0$$

and after reduction of similar phrases, the constraint results (21).

Another group of constraints (22)–(25) is related to using intermediate buffers. If the analysed constraint (21) is met in the form of the equality, that is if: $s_{vk} - c_{\varepsilon k} = 1 + g_{vk}$ for $k \in K$, ε, $v \in V$, where $\varepsilon < v$, then the tasks for the product k in the stage v are executed directly after the end of the tasks in the stage ε, and the break between the tasks is dedicated only for the transport of the product. However, if the constraint (21) is not met as the equation, then a certain sum may be added to the right side of the inequality, so as to obtain equality. This added sum represents a number of time intervals in which the buffer set before the stage v is loaded by the product k, awaiting the execution of the following tasks. In this way, based on the modification of the condition (21), the constraint (22) has been built. It is used solely for the determination of the number of time intervals in which the product k is waiting in the buffer before the stage v for the execution of further tasks. The constraints (23) and (24) make it certain that the buffers were loaded in the proper time intervals. The condition (23) guarantees that the product loads the buffer directly after the completion of the previous task and after transporting it to the buffer before the stage in which the next task is to be executed. For this reason, the right side of the inequality (23) takes into account the time of completion of the previous task and the transport time. At the same time, the condition (24) must be met, ensuring placement of the product in the buffer directly before the execution of the following task. Limited capacity of the intermediate buffers is not exceeded owing to the constraint (25). The last constraint of the model M2 (26) ensures binarity of decision variables.

4. The relaxation heuristic

The following is two-level heuristic of production scheduling for flowshop systems with intermediate buffers:

• Level I (stage load balancing)

Step 1. Assume iteration number $e := 1$ and solve the problem (6)–(13) – constructed for the M1 model (without constraint (14)) – it's the linear relaxation of M1. Let the used linear relaxation be marked: LP(M1a) for fixed production routes, LP(M1b) for alternative production routes. Determine heuristic solutions:

$\widetilde{x}^e_{vj} := x_{vj}$; $v \in V$; $j \in J$ – assignments types of tasks to stages;

$\widetilde{z}^e_{vjk} := z_{vjk}$; $v \in V$; $j \in J$; $k \in K$ – assignments tasks for products to stages.

Go to step 2.

Step 2.

a) Assume: $e := e + 1$. Determine allocations of tasks and products to stages in accordance with (27).

$$\tilde{x}^e_{vj} = \text{round}\left(\tilde{x}^{e-1}_{vj}\right), \quad z^e_{vjk} = \text{round}\left(z^{e-1}_{vjk}\right); \quad j \in J; \quad k \in K; \quad v \in V. \tag{27}$$

If the routes are alternative then go to step 2c. If the routes are fixed and if constraint (28) is fulfilled, go to step 2c, otherwise, go to step 2b.

$$\sum_{v \in V_j} \tilde{x}^e_{vj} = 1; \quad j \in J \tag{28}$$

b) Assume: $e := e + 1$. For the each of j types of tasks which do not fulfil constraint (28) select only one stage $v*$ in accordance with lexicographical order: 1 – stage of the largest value $\widetilde{x}^{e-1}_{vj}$ $(j \in J)$, 2 – stage of the smallest index v. Assume $\widetilde{x}^e_{v*j} = 1$ for j type of task, go to step 2c.

c) If constraint (12) for $x_{vj} := \widetilde{x}^e_{vj}$ $(j \in J)$ is fulfilled for each stage v, go to step 3; otherwise, for each stage which did not fulfil condition (12) determine coefficients of additional cut-off constraints. Those coefficients: $C_v = \{j : \xi_{vj} = 1\} \subset J$ meet condition (29) [6]. Return to step 1 and solve the task formulated there complemented with additional constraints (30), developed for stages v which did not meet constraint (12).

$$\sum_{j \in J} a_{vj}\xi_j \geq b_v + 1 \wedge \sum_{j \in J} \left(1 - x^e_{vj}\right) \xi_{vj} < 1; \quad \xi_{vj} \in \{0,1\} \tag{29}$$

$$\sum_{j \in C_v} x_{vj} \leq \overline{\overline{C}}_v - 1; \qquad v \in V \tag{30}$$

Step 3. Check whether all the tasks assigned to the products have been assigned to the stages: reviewing the allocations according to the increasing indexes. If any task for the product k has not been assigned, it should be assigned to the stage with the least load, to which the possibility of execution of the task of this type has been assigned, and this allocation will not prevent maintaining of order limitations.

Following (31), determine the total time of performing a task for product k in stage v, and go to step 4.

$$t_{vk} = \sum_{j \in J_k} p_{jk} \widetilde{z}^e_{vjk}; \quad v \in V; \quad k \in K \tag{31}$$

• Level II (distributing tasks in time and space) [21]:

Step 4. Solve the problem contained in mathematical model (16)–(25). It's the linear relaxation of M2 without constraint (26) about binary decision variables. Determine heuristic solutions:

$\widetilde{q}^e_{ikl} := q_{ikl}$; $i \in I$; $k \in K$; $l \in L$ – assignments products to machines in time intervals;

$\widetilde{y}^e_{vkl} := y_{vkl}$; $v \in V$; $k \in K$; $l \in L$ – assignments products to machines in time intervals.

Assume: $e := e + 1$. Using (32) and (33), determine initial solution: s^e_{ik}, c^e_{ik} – times of beginning and ending of task performance for the product k on the machine i.

$$\widetilde{q}^e_{ikl} = \text{round}\left(\widetilde{q}^{e-1}_{ikl}\right); \quad i \in I, \ k \in K, \ l \in L \tag{32}$$

$$\begin{gathered} s^e_{ik} = \min_{l \in L}\left(l \widetilde{q}^e_{ikl}\right) \\ \text{for } \widetilde{q}^e_{ikl} = 1, \ c^e_{ik} = s^e_{ik} + p_{jk} - 1; \ i \in I; \ k \in K. \end{gathered} \tag{33}$$

Having determined machine loads in accordance with (34), go to step 5.

$$\widetilde{q}^e_{ikl} = \begin{cases} 1, \text{if } s^e_{ik} \geq l \leq c^e_{ik} \\ 0, \text{otherwise} \end{cases}; \ i \in I; \ k \in K; \ l \in L. \tag{34}$$

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 (18) is fulfilled, go to step 6, if not – go to step 5d.
1. From among products which do not fulfil constraint (18) for $q_{ikl} = \widetilde{q}^e_{ikl}$ $(k \in K)$, select only one product k^* in accordance with lexicographical order: 1 – product which did not fulfil those constraints 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 $\widehat{j}$ mean an task performed in time interval l on machine i with respect to product k^*. Applying constraint (35), determine the number of time intervals 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 $\widetilde{K}$. Having modified the schedule in accordance with (36) and updated times $s^e_{\tau k}$, $c^e_{\tau k}$ $(\tau \in I, \ k \in K)$ in accordance with (33), go to step 6.

$$\begin{gathered} \beta = c^{e-1}_{ik^*} - l + 1; \quad \widetilde{K} : \widetilde{K} \\ = \left\{k \in K \backslash \{k^*\} : \widetilde{q}^{e-1}_{ikr} = 1 \, r \in \langle l, l + \beta - 1 \rangle; \ k \in K\right\}, \end{gathered} \tag{35}$$

$$\widetilde{q}^e_{\tau kr} = \begin{cases} \widetilde{q}^{e-1}_{\tau kr} \\ \text{for } \left((\tau \in I \backslash \{i\}, \ k \in K) \vee \left(\tau = i, \ k \in K \backslash \widetilde{K}\right)\right) \\ \widetilde{q}^{e-1}_{\tau kf} \text{ for } \tau = i, \ k \in \widetilde{K}, \\ \text{where: } r \geq l + \beta, \ f = r - \beta \end{cases}; \quad r \in L. \tag{36}$$

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 constraint (37) occurs. 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 analyzed product $\overline{k}$ and having modified the schedule in accordance with (38) and updated the times of task start and end on the basis of (37), go to step 7.

$$\begin{gathered} s^e_{ik} - c^e_{\tau k} \geq 1 + g_{vk} \text{ for } (i, v), (\tau, \varepsilon) \in D, \\ \text{when } \sum_{\psi = \varepsilon}^{v} t_{\psi k} = t_{vk} + t_{\varepsilon k}, \end{gathered} \tag{37}$$

$$q^e_{\tau kr} = \begin{cases} q^{e-1}_{\tau kr} \\ \text{for } \left((\tau \in I \backslash \{i\}, \ k \in K) \vee \left(\tau = i, \ k \in K \backslash \widetilde{K}\right)\right) \\ q^{e-1}_{\tau kf} \text{for} \tau = i, \ k = \overline{k}, \\ \text{when: } r \geq l + \beta, \ f = r - \beta \end{cases}; \quad r \in L. \tag{38}$$

Step 7. Halt condition for the previous phases of schedule modification is checked here. If $l < \max_{\tau \in I, k \in K} c^e_{\tau k}$, 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 machines and buffers in subsequent time intervals are reanalyzed. The following are checked in succession:

a) availability of a buffer in time interval l, which is located before stage v – relationship (25) is checked;
b) availability of machine i in time interval l – on the basis on the value of parameter η_{il}.

For each of the above points 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 (point a) or availability of a machine for loading by a given product, beginning from time interval l (point b). Before each schedule modification, $e := e + 1$ shall be assumed. Modification should be made analogously to relationship (36), (38). After each modification, the start and end times of loading individual machines with given products should be updated in accordance with (33).

Verification of the last time interval for a machine with the highest index shall end the algorithm. Start and end times of loading individual machines with products should and schedule length $C^h_{\max}$ are determined in accordance with relationship (39).

$$s^h_{ik} = s^e_{ik}; \quad c^h_{ik} = c^e_{ik} \quad \text{for} \quad i \in I; \quad k \in K; \quad l \in L; \\ C^h_{\max} = \max_{i \in I,\, k \in K} c^h_{ik}. \tag{39}$$

5. Calculation experiments

The presented two-level heuristic has been verified by means of calculation experiments. This section describes one of the experiments. The results of the other ones are presented in the final part of the section.

The following example is used not only to show the functioning of relaxation heuristics, but also to visualise the differences between fixed and alternative production routes.

A 3-stage unidirectional flow system is given with the configuration presented in Fig. 1 (Sec. 2). A set of stages has thus the form of: $V = \{v_1, v_2, v_3\}$. Machines $i \in I = \{m_1, m_2, m_3, m_4, m_5\}$ are spaced in the stages. Their belonging to the stages is known due to the set of lines D, defined in Table 1. This set comes in the form: $D = \{(m_1, v_1), (m_2, v_2), (m_3, v_2), (m_4, v_3), (m_5, v_3)\}$. The assumption has been made that all the machines are available within the analysed time intervals. There are intermediate buffers between the stages, with identical capacities: $d_2 = d_3 = 3$. Tasks have to be performed for 5 products $j \in K = \{k_1, k_2, k_3, k_4, k_5\}$. Tasks of the type $j \in J = \{o_1, o_2, o_3, o_4, o_5, o_6\}$ are assigned to the products. The assignment of the tasks to specific products and the sequences for the execution of these tasks (limitations of order) are known due to the following graphs:

for product k_1: $L \to$ o3 $\to$ o4 $\to$ o1 $\to$ o2 $\to$ o6 $\to U$
for product k_2: $L \to$ o4 $\to$ o5 $\to$ o1 $\to$ o2 $\to$ o6 $\to U$
for product k_3: $L \to$ o3 $\to$ o1 $\to$ o2 $\to U$
for product k_4: $L \to$ o3 $\to$ o4 $\to$ o1 $\to$ o5 $\to$ o2 $\to U$
for product k_5: $L \to$ o5 $\to$ o1 $\to$ o6 $\to$ o2 $\to U$,
where *L*/*U* denotes loading/unloading operations.

The times for the execution of individual tasks j for the given products k are entered in the matrix form:

$$[p_{jk}] = \begin{bmatrix} 2 & 2 & 3 & 3 & 4 \\ 3 & 3 & 2 & 2 & 2 \\ 1 & 0 & 2 & 2 & 0 \\ 1 & 2 & 0 & 2 & 0 \\ 0 & 3 & 0 & 3 & 4 \\ 3 & 3 & 0 & 0 & 2 \end{bmatrix}.$$

Different times for the execution of the tasks of the same type for different products result from the fact that the times are added here which are related to the preparation of the product for the execution of the task, e.g. the time necessary for the orientation of the product. $p_{jk} = 0$ means that no j type tasks are executed for the product k.

The execution of the tasks of each type requires assignment of the relevant part feeder ($J = J_c$). The parameters which describe the feeders and the available working space for the individual stages are as follows (pursuant to the markings presented in Table 1):

$$[a_{vj}] = \begin{bmatrix} 1 & 2 & 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 & 1 & 1 \\ 1 & 2 & 1 & 2 & 1 & 1 \end{bmatrix}, \quad [b_v] = \begin{bmatrix} 10 \\ 8 \\ 8 \end{bmatrix}.$$

The technical possibilities of the individual machines assigned to the given stages are known from the sets of the stages V_j, to which the machines capable of the execution of the j type tasks belong: $V_{o1} = \{v_2, v_3\}$; $V_{o2} = \{v_2, v_3\}$;$V_{o3} = \{v_1, v_2\}$; $V_{o4} = \{v_1, v_2, v_3\}$; $V_{o5} = \{v_1, v_2, v_3\}$; $V_{o6} = \{v_2, v_3\}$.

The adopted markings for the stages: $v_1 \ldots v_3$, of the machines: $m_1, \ldots, m_5$ are used to ensure legibility of the description of the task at hand. In the files of the data prepared for discrete optimisation packages, the machines, stages or time intervals are to be marked with consecutive natural numbers. This is due to the necessity of multiplying the variables by the indexes in some constraints, e.g. (9), (20)–(24).

For these data, relaxation heuristics has to be used for both types of production routes. The obtained solutions have to be compared with the results received based on the use of the presented linear mathematical models M1a, M1b and M2.

In order to solve the task, the linear mathematical dependencies have been encoded in the AMPL mathematical programming language [22]. The GNU Linear Programming Kit (GLPK) software has been used for the calculations.

• Level I:

The workload balancing for the machines allocated to the particular stages has been solved here. On the basis of the relationship (5) and the data, the number of the time intervals taken into account has been determined, which is $H = 13$ for fixed production routes, and $H = 14$ for alternative routes. The calculations for the level I heuristic method did not require increasing the number of the analysed time intervals.

The results of the linear relaxation in the M1a model (for fixed routes) are given in Table 2. Table 3 presents the results of the linear relaxation for the M1b model. This table gives the values of the variables which determine the layout of the feeder and of the variables used for assigning individual tasks for the products flowing through the system. For fixed production routes, an identical solution has been received as in the case of the solution of the problem formulated in the M1a model with binary decision variables. The allocation of the products to the machines is consistent with the layout of the part feeders: the allocation of the tasks types to the machines. Each task type has been assigned precisely to one stage. Table 3, on alternative routes, compares the solution determined with heuristics with the results obtained from using the M1b model with binary variables. The use of the M1b model has enabled such a solution in which the differences between the loads in the individual machines are smaller than in case of application of linear relaxation of this LP(M1a) model. Attention should be paid to the fact that the stage v_1 includes only one machine, because its load is smaller than the loads in the other stages.

Table 2
Comparison of the solutions – the level I – for the fixed routes

Using LP(M1a) – the linear relaxation of the M1a			Using the M1a model	
Stage	Task assignments	$round\,(\widetilde{x}_{vj})$	Task assignments	Workload
Stage v_1	o_3 o_4	$x_{13} = 1$ $x_{14} = 1$	o_3 o_4	10 (1 machine)
Stage v_2	o_1 o_5	$x_{21} = 0.92 \approx 1$ $x_{25} = 0.92 \approx 1$	o_1 o_5	24 (2 machines)
Stage v_3	o_2 o_6	$x_{32} = 1$ $x_{36} = 1$	o_2 o_6	20 (2 machines)

Table 3
Comparison of the solutions – the level I – for the alternative routes

	Using LP(M1a) – the linear relaxation of the M1b model					Using the M1b model		
Stage	Task assignments	$round\,(\widetilde{x}_{vj})$	Product assignments	$round\left(\widetilde{z}_{vjk}\right)$	Workload	Task assignments	Product assignments	Workload
Stage v_1	o_3 o_4	$x_{13} = 1$ $x_{14} = 0.88 \approx 1$	o_3: k_1, k_3, k_4 o_4: k_1, k_2, k_4	$z_{131} = z_{133} = z_{134} = 1$ $z_{141} = z_{142} = z_{144} = 0.881$	10	o_3 o_4 o_5	o_3: k_3 o_4: k_2 o_5: k_2, k_5	11
Stage v_2	o_1 o_2 o_5 o_6	$x_{21} = 0.88 \approx 1$ $x_{22} = 0.88 \approx 1$ $x_{25} = 0.88 \approx 1$ $x_{26} = 0.88 \approx 1$	o_1: k_2, k_4, k_5 o_2: k_2, k_4 o_5: k_2, k_4, k_5 o_6: k_5	$z_{212} = z_{214} = z_{215} = 0.88 \approx 1$ $z_{222} = z_{224} = 0.88 \approx 1$ $z_{254} = z_{255} = 0.88 \approx 1$, $z_{252} = 0.42$* $z_{265} = 0.88 \approx 1$	26	o_1 o_2 o_3 o_4 o_5 o_6	o_1: k_1, k_2, k_4, k_5 o_2: k_4 o_3: k_1, k_4 o_4: k_1, k_4 o_5: k_4	22
Stage v_3	o_1 o_2 o_6	$x_{31} = 1$ $x_{32} = 1$ $x_{36} = 1$	o_1: k_1, k_3 o_2: k_1, k_3, k_5 o_6: k_1, k_2	$z_{311} = z_{313} = 1$ $z_{321} = z_{323} = z_{325} = 1$ $z_{361} = z_{362} = 1$	18	o_1 o_2 o_5 o_6	o_1: k_3 o_2: k_1, k_2, k_3, k_5 o_6: k_1, k_2, k_5	21

* In case of the variables which specify the allocation of the task o_5 for the product k_2, all the resulting values of the variable z_{vjk} were lower than 0.5 and their values amounted to: $z_{152} = 0.46$, $z_{252} = 0.42$, $z_{352} = 0.12$. The task o_5 for the product k_2 is the only task which was not assigned to any stage on the basis of the rule of approximations. On the basis of the verification of allocation of products to the stages, described in step 3 of the heuristics, this task has been assigned to the stage v_2, the only stage which was assigned the possibility of execution of the o_5 type tasks ($x_{25} = 0.88$).

The comparison of the solutions determined with heuristic, after the application of linear relaxation M1a and M1b, shows that balancing machine workload in case of fixed production routes is more difficult than when the given type of the task may be assigned to different stages, as given in Table 3.

This verification covered also the constraint (12), which verifies working space of the machines placed in the individual stages. It has been met, thus there is no need to enter additional cutting off constraints and the problem assigned to the level II method may be solved.

• Level II:

The level II of the method employs the results of the task solved at the level I, given in Table 2 and in Table 3. On the basis of these results and the given times p_{jk} (processing time for the task j of the product k), the times may be determined for the loads in the individual stages by the products flowing through the system. The parameter t_{vk} is used for this purpose, determined on the basis of Eq. (15). Table 4 constitutes the summary of the results obtained at the level I. It is used for verification and not only for allocation of the tasks to the individual products, but also for the determination of load times in the individual stages by the products. These times, marked t_{vk}, are summarised in the bottom part of the table, for both types of production routes. Table 4 includes the information that tasks for the given product executed within the given stage are not separated with the tasks assigned to other products. Thus, the stage load times may be summed up by different tasks assigned to the same product. In Table 4, in the parentheses, the summed times p_{jk} are given. At the level II of the method, each set of these different tasks, assigned to the same product, is regarded as a one separate task.

The data given in Table 4 include also the parameters g_{vk}: the times for transporting the product k from the stage in which the previous task was completed to the stage v. $g_{vk} = 0$ means that there is no transport of the product to the stage v from the machines assigned to the other stages.

The received schedules of product flow through the system are given in Fig. 3. Figure 3a presents the schedule for fixed production routes built with heuristic. The same length of scheduling $C_{\max} = 21$ as in the case of the successive use of the models M1a and M2 has been determined with binary decision variables. All the conditions verified in the steps 4–8 have been met, thus the schedule did not have to be modified. The time of the calculations, obtained with a INTEL T1300 1.66 MHz processor computer, was: 2s with heuristics, and 2066s in case using mathematical models: M1a, M2. The calculations for the level II take much more time than in case of the time of solving the tasks formulated for the level I.

Table 4
Input data for the level II: the assignments of tasks for products to stages and the parameters: t_{vk}, g_{vk}

	For the fixed routes					For the alternative routes									
	Using M1a or LP(M1a)					Using LP(M1b) – heuristic					Using M1b (binary variables)				
Stages	k_1	k_2	k_3	k_4	k_5	k_1	k_2	k_3	k_4	k_5	k_1	k_2	k_3	k_4	k_5
Stage v_1	$o_3(1)$ $o_4(1)$ $t_{11}=2$	$o_4(2)$ $t_{12}=2$	$o_3(2)$ $t_{13}=2$	$o_3(2)$ $o_4(2)$ $t_{14}=4$		$o_3(1)$ $o_4(1)$ $t_{11}=2$	$o_4(2)$ $t_{12}=2$	$o_3(2)$ $t_{13}=2$	$o_3(2)$ $o_4(2)$ $t_{14}=4$			$o_4(2)$ $o_5(3)$ $t_{12}=5$	$o_3(2)$ $t_{13}=2$		$o_5(4)$ $t_{15}=4$
Stage v_2	$o_1(2)$ $t_{21}=2$	$o_1(2)$ $o_5(3)$ $t_{22}=5$	$o_1(3)$ $t_{23}=3$	$o_1(3)$ $o_5(3)$ $t_{24}=6$	$o_1(4)$ $o_5(4)$ $t_{25}=8$		$o_1(2)$ $o_2(3)$ $o_5(3)$ $t_{22}=8$		$o_1(3)$ $o_2(2)$ $o_5(3)$ $t_{24}=8$	$o_1(4)$ $o_5(4)$ $o_6(2)$ t_{25}=10	$o_1(2)$ $o_3(1)$ $o_4(1)$ $t_{21}=4$	$o_1(2)$ $t_{22}=2$		$o_1(3)$ $o_2(2)$ $o_3(2)$ $o_4(2)$ $o_5(3)$ t_{24}=12	$o_1(4)$ $t_{25}=4$
Stage v_3	$o_2(3)$ $o_6(3)$ $t_{31}=6$	$o_2(3)$ $o_6(3)$ $t_{32}=6$	$o_2(2)$ $t_{33}=2$	$o_2(2)$ $t_{34}=2$	$o_2(2)$ $o_6(2)$ $t_{35}=4$	$o_1(2)$ $o_2(3)$ $o_6(3)$ $t_{21}=8$	$o_6(3)$ $t_{32}=3$	$o_1(3)$ $o_2(2)$ $t_{33}=5$		$o_2(2)$ $t_{35}=2$	$o_2(3)$ $o_6(3)$ $t_{31}=6$	$o_2(3)$ $o_6(3)$ $t_{32}=6$	$o_1(3)$ $o_2(2)$ $t_{33}=5$		$o_2(2)$ $o_6(2)$ $t_{35}=4$
	$[t_{vk}]=\begin{bmatrix}2&2&2&4&0\\2&5&3&6&8\\6&6&2&2&4\end{bmatrix}$					$[t_{vk}]=\begin{bmatrix}2&2&2&4&0\\0&8&0&8&10\\8&3&5&0&2\end{bmatrix}$					$[t_{vk}]=\begin{bmatrix}0&5&2&0&4\\4&2&0&12&4\\6&6&5&0&4\end{bmatrix}$				
	$[g_{vk}]=\begin{bmatrix}0&0&0&0&0\\1&1&1&1&0\\1&1&1&1&1\end{bmatrix}$					$[g_{vk}]=\begin{bmatrix}0&0&0&0&0\\0&1&0&1&0\\2&1&2&0&1\end{bmatrix}$					$[g_{vk}]=\begin{bmatrix}0&0&0&0&0\\0&1&0&0&1\\1&1&2&0&1\end{bmatrix}$				

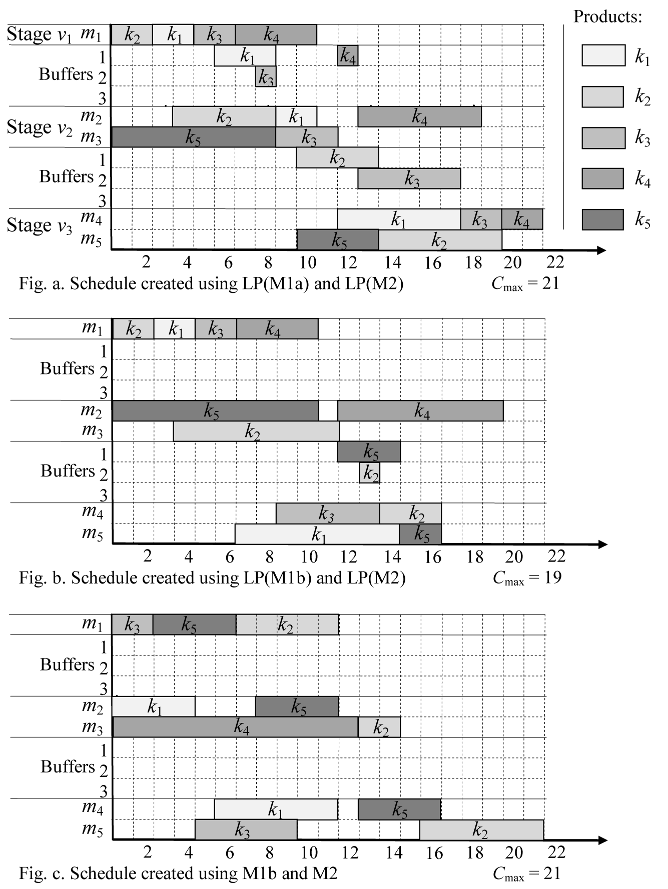

Fig. 3. Schedules for the fixed routes (Figure a) and for the alternative routes (Figure b and Figure c)

The allocation of products between the machines in the individual stages, with the alternative production routes taken into account, is visualised in Figs. 3b and 3c. Figure 3b refers to the developed relaxation heuristic. The development of the solution presented in Fig. 3b required modification of the preliminary solution, the one developed in the step 4 of the heuristic. In the primary solution, the allocation of the product k_4 to the machine m_2 (stage v_2) in the time intervals 1–18 has been determined. In the previous stage v_1, the tasks for this product end in the time interval 10. The value of the parameter g_{vk}, which defines the time for transport of this product (k_4) to the stage (v_2), in which the following tasks are executed, is 1. Thus the condition (37) has not been met, for which there are: $s_{ik} = 11$ (the beginning time of the execution of the task for the product k in the machine i) and $c_{\tau k} = 10$ (the ending time of the execution of the task for the product k in the machine τ) where the product k first loads the machine τ, and then the machine i. Thus the condition: $s_{ik} - c_{\tau k} \geq g_{vk} + 1$ (condition (37)) has been maintained, because the dependence: $11 - 10 \geq 1 + 1$ is not true. It would be met, had the right side of the inequality been lower by 1. It means that the allocations of the tasks to the machine m_2, beginning with the time interval 11, have to be "moved" by $\beta = 1$ units – according to the dependency (38). In accordance with the heuristic, the successive time intervals in the next machines are verified. One more modification of the preliminary solution had to be done in the discussed example. The product k_1 was assigned to the machine m_5 (the stage v_3) in the time intervals 6–13. The previous tasks for the product k_1 have been assigned to the machine m_1 (the stage v_1) in the time intervals 3 and 4. After completion of the tasks in the time interval 4, however, taking into consideration 2 time intervals (that is the time intervals 5 and 6) for transport of the product between the stages is required. According to the relationship (38) the schedule is modified ("moved" by $\beta = 1$ time interval), as a result of which the tasks for the product k_1 in the machine m_5 are executed in the time intervals 7–14, with the tasks for the product k_5 afterwards (Fig. 3b).

Figure 3c presents the obtained scheduling of the tasks for alternative routes. The schedule given in it has been built based on the models M1b and M2 (with binary variables). A longer schedule ($C_{\max} = 21$) has been obtained as compared with relaxation heuristics in which LP(M1b) and LP(M2) was used. This less beneficial solution may be justified with the fact that the solution was also obtained with the approximation method. The decomposition of the global problem into two tasks solved in succession does not guarantee obtaining an optimum solution, as it has been proven in the presented example. The same length of the schedule was determined as with fixed production routes. The advantage of the solution comes in the fact that intermediate buffers are not used. To achieve this, a larger number of the feeders should be used than with fixed production routes. Reduction of buffer loads may be also achieved with modification of the minimised objective function (16), taking into account buffer loads with particular products in it. In this case, the objective function takes the form (40):

$$\text{Minimize:} \quad \sum_{i\in I}\sum_{k\in K}\sum_{l\in L} l q_{ikl} + \sum_{v\in V}\sum_{k\in K}\sum_{l\in L} y_{vkl}. \tag{40}$$

In the comparisons of the schedules presented for fixed production routes (Fig. 3a) and alternative production routes (Fig. 3b), determined based on the presented heuristic, one has to take note of the fact that the difference between the lengths of these schedules depends obviously on the data, especially on the sequences of task execution. For example, if the task o_6 for the product k_1 was executed as the first one (and not the last one), all the types of the tasks for fixed routes would have been assigned to one stage (with two machines: in this example). With such data, the schedule for alternative routes would be considerably shorter from the solution for fixed production routes.

The compliment to the solution presented in Fig. 3 comes with Table 5, in which the assignments of the types of the tasks to the machines are given, that is the layout of the feeders. In the data presented at the beginning of the description of the example, it has been stated that each type of tasks requires a feeder. All the part feeders assigned to the machines have been used. The markings of the products are given in the parentheses to which these tasks have been assigned.

The other calculation experiments covered 4 groups of tasks. For each one of the groups, 30 examples were solved. Parameters of those groups and findings are presented in Table 6. Ten problems were run for each set of parameters. The findings include average values of two indexes, which were determined for each test task. The indexes serve the purpose of algorithm assessment, which is performed at two levels: the quality of findings (makespan) and the amount of time calculations require. The indexes, 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: model M1a or M1b, level II: model M2). Index h has been ascribed to he heuristic solution, and index O to the solution using models: M1 (M1a or M1b) and M2.

The indexes are as follows:

$$w_1 = \frac{C^h_{\max} - C^O_{\max}}{C^M_{\max}}; \qquad w_2 = \frac{C^O_{\max} - C^{LB}_{\max}}{C^{LB}_{\max}};$$

$$w_3 = \frac{C^H_{\max} - C^{LB}_{\max}}{C^{LB}_{\max}}; \qquad w_4 = \frac{CPU^h}{CPU^O},$$

where $C^h_{\max}$, $C^O_{\max}$ – makespans determined by means of the following algorithms: heuristic (in accordance with relationship (39)), using linear mathematical models: M1 (M1a or M1b) and M2; CPU^h, CPU^O computational times for methods: heuristic, using models: M1 (M1a or M1b) and M2; $C^{LB}_{\max}$ – lower bound on makespan determined by relationship (4).

Table 5
Assignments of the types of tasks and products to machines

		The fixed routes	The alternative routes	
Stages	Machines	Using the heuristics with LP(M1a) and LP(M2) or M1a and M2	Using the heuristics with LP(M1b) and LP(M2)	Using M1b and M2 (binary variables)
Stage v_1	m_1	o_3 (k_1, k_3, k_4), o_4 (k_1, k_2, k_4)	o_3 (k_1, k_3, k_4), o_4 (k_1, k_2, k_4)	$o_3(k_3)$, o_4 (k_2), o_5 (k_2, k_5)
Stage v_2	m_2	$o_1(k_1, k_2, k_4)$, o_5 (k_2, k_4)	$o_1(k_4, k_5)$, o_2 (k_4), o_5 (k_4, k_5), o_6 (k_5)	$o_1(k_1, k_5)$, o_3 (k_1), o_4 (k_1)
	m_3	$o_1(k_3, k_5)$, o_5 (k_5)	o_1 (k_2), o_2 (k_2), o_5 (k_2)	$o_1(k_2, k_4)$, o_2 (k_4), o_3 (k_4), o_4 (k_4), o_5 (k_4)
Stage v_3	m_4	$o_2(k_2, k_3)$, $o_6(k_2)$	o_1 (k_3), o_2 (k_3), o_6 (k_2)	$o_2(k_1, k_5)$, $o_6(k_1, k_5)$
	m_5	$o_2(k_1, k_5)$, $o_6(k_1, k_5)$	o_1 (k_1), o_2 (k_1, k_5), o_6 (k_1)	o_1 (k_3), o_2 (k_2, k_3), o_6 (k_2)

Table 6
Specification of parameters of groups of tasks and computational results

Gr.	Parameters of groups						For fixed routes [%]				For alternative routes [%]			
	M	ϑ	N	W	H	Ψ	w_1	w_2	w_3	w_4	w_1	w_2	w_3	w_4
1	4	2	6	3	18	8	6.4	4.4	5.8	0.60	5.1	4.0	4.6	0.67
2	5	2	8	4	20	10	6.7	4.3	5.7	0.56	5.8	4.2	4.8	0.64
3	5	3	10	6	25	12	7.8	6.2	7.1	0.48	8.2	5.1	6.1	0.60
4	6	3	12	7	30	12	9.9	6.5	7.7	0.45	9.0	5.9	7.6	0.59

Numbers of: M – machines, ϑ – stages, N – types of tasks, W – types of products, H – time intervals, Ψ – sum of capacity of intermediate buffers.

Findings of calculation experiments contained in Table 6 indicate significant reduction of calculation 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 0.5–0.7%, i.e. solutions were generated approximately 140–200 times faster than in the case of the two-level method with binary decision-making variables (see the index w_4). However, that advantage is achieved at the expense of a minor deviance from the optimum. This is indicated by the average values of index w_1, which did not exceed 10% for alternative routes and 9% for fixed routes. The maximum value of the index w_1 reached 12.7%. The average values of indexes w_2 and w_3 present differences between scheduling for fixed and alternative routes. The computational time for loading machines with alternative production routes is longer than in the case when we use fixed routes. The makespans was shorter about 7–12% for systems with alternative routes than for fixed routes. For the test problems, the CPU run time for the two-level approach with the presented heuristic was no longer than a several minutes (for the largest problems) on a PC 1.66 MHz.

The volume of test problem groups have been adjusted to the calculation capacity of the available discrete optimization packages which constitute an IT tool necessary to solve discrete optimization tasks (formulated in the models: M1a, M1b, M2).

6. Conclusions

The developed algorithm and the mathematical models built for it may be used for various types of simulations. Taking into consideration fixed and alternative production routes enables testing of the effect of route type on sequencing of tasks. It is significant especially in a case of assembly systems, where allocation of feeders to machines are not only related to the necessity of involving additional activities, but, sometimes, also costs related to feeder placement. Assignment and placement of feeders is also related to the necessity of considering limited working space in individual work stations. The conducted calculation experiments proved returning shorter schedules in a case of alternative routes (as compared with fixed routes), however, the analysis of costs takes it also into account, for example, costs related to operational use or the necessity of purchasing auxiliary equipment.

An application of the two-level approach allowed solutions of tasks of a relatively larger scale as compared with the monolithic approach. In each level of the method, a smaller number of parameters and variables in the formulated mathematical relationships have been taken into account. With the linear relaxation of the tasks of mathematical programming, problems are solved in a relatively short time. It is conducive for the possibility of re-scheduling. In case of breakdowns of any machine or when a new, urgent order has to be considered, the data are updated and the problem is solved again.

For the tasks of relatively smaller scales, it is recommended to solve the problem based on the built discrete optimisation tasks mathematical models. These models may be, obviously, modified and expanded. For example, some parameters may be taken into account in the models, which define priorities of executing tasks for individual products. Many formulated mathematical relationships may be used for building the task scheduling algorithm based on the monolithic approach.

Development of computer technology and software offers good perspectives for discrete optimisation, including solving problems of task scheduling formulated in the linear mathematical models. The currently applied task scheduling algorithms for production lines are properly modified and used in other fields, as well. For example, economic conditions related to, among others, the so-called globalisation have affected the necessity of task scheduling in support of supply chains. It requires building of the product flow schedule not only by production plants within the supply chain, but also between production plants, where transport tasks are sequenced. Development of optimisation methods and their application in new areas are reflected in, among others, [23] and [24], which presents continued major importance and new areas of application of heuristic algorithms.

REFERENCES

[1] J. Błażewicz, K. Ecker, E. Pesch, G. Schmidt, and J. Węglarz, *Handbook Scheduling*, Springer, Berlin, 2007.

[2] M. Pindeo, *Scheduling: Theory, Algorithms, and Systems*, Prentice Hall, Upper Saddle, 2002.

[3] T. Sawik, *Production Planning and Scheduling in Flexible Assembly Systems*, Springer-Verlag, Berlin, 1999.

[4] Y. Pochet and L.A. Wolsey, *Production Planning by Mixed Integer Programming, Series in Operations Research and Financcical Engineering*, Springer, New York, 2006.

[5] G. Schmidt, "Modelling production scheduling systems", *Int. J. Production Economics* 46–47, 109–118 (1996).

[6] T. Sawik, "Simultaneous loading, routing, and assembly plan selection in a flexible assembly system", *Math. Comput. Modelling* 28 (9), 19–29 (1998).

[7] T. Sawik, "Balancing and scheduling or surface mount technology lines", *Int. J. Production Research* 40 (9), 1973–1991 (2002).

[8] T. Sawik, "Loading and scheduling of a flexible assembly system by mixed integer programming", *Eur. J. Operational Research* 154 (1), 1–19 (2004).

[9] T.F. Gonzales, *Handbook of Approximation Algorithms and Metaheuristics*, Chaoman and Hall/CRC, New York, 2007.

[10] G. Schmidt, "Scheduling with limited machine availability", *Eur. J. Operational Research* 121, 1–15 (2000).

[11] T. Kis and E. Pesch, "A review of exact solution methods for the non-pre-emptive multiprocessor flowshop problem", *Eur. J. Operational Research* 164 (3), 592–608 (2005).

[12] I. Ribas, R. Leinstein, and J.M. Framinan, "Review and classification of hybrid flowshop scheduling problems from a production system and a solution procedure prespective", *Computers & Operations Resarch* 37 (8), 1439–1454 (2010).

[13] D. Quadt and H. Kuhn, "A taxonomy of flexible flow line scheduling procedures", *Eur. J. Operational Research* 178 (3), 686–698 (2007).

[14] R. Linn and W. Zhang, "Hybrid flow shop scheduling: a survey", *Computers & Industrial Engineering* 37, 57–61 (1999).

[15] R. Ruiz and J.A. Vázquez-Rodriguez, "The hybrid flow shop scheduling problem", *Eur. J. Operational Research* 205 (1), 1–18 (2010).

[16] M. Sterna, "Dominance relations for two-machine flow shop problem with late work criterion", *Bull. Pol. Ac.: Tech.* 55 (1), 59–69, (2007).

[17] M. Haouari, L. Hidri, and A. Gharbi, "Optimal scheduling of a two-stage hybrid flow shop", *Math. Meth. Operations Research* 64 (1), 107–124 (2006).

[18] H. Allaoui and A. Artiba, "Scheduling two-stage hybrid flow shop with availability constraints", *Computers & Operations Research* 33 (5), 1399–1419 (2006).

[19] Ch. Koulamas and G.J. Kyparisis, " The three-stage assembly flowshop scheduling problem", *Computers & Operations Research* 28 (7), 689–704 (2001).

[20] T. Sawik, "Hierarchical approach to production scheduling in make-to-order assembly", *Int. J. Production Research* 44 (4), 801–830 (2006).

[21] M. Magiera, "Two-level of production scheduling for flow-shop systems with intermediate storages", *Total Logistic Management* 1, 101–110 (2008).

[22] R. Fourer, D. Gay, and B. Kernighan, *AMPL, A Modelling Language for Mathematical Programming*, Duxbury Press, Pacific Grove, 2003.

[23] J. Kwiecień and B. Filipowicz, "Firefly algorithm in optimization of queuing systems", *Bull. Pol. Ac.: Tech.* 60 (2), 363–368 (2012).

[24] R. Nowotniak and J. Kucharski, "GPU-based tuning of quantum-inspired genetic algorithm for combinatorial optimization problem", *Bull. Pol. Ac.: Tech.* 60 (2), 323–330 (2012).