Decision Making in Manufacturing and Services
Vol. 16 • 2022 • pp. 47–65

Heuristic Algorithm for Lot Sizing and Scheduling on Identical Parallel Machines

Roger Książek*

Abstract. This paper presents a new heuristic algorithm for the task of lot sizing and scheduling for identical parallel machines. The new algorithm is based on the rolling-horizon approach and the fix-and-relax decomposition technique. Two variants of the algorithm are finally proposed for solving the problem of lot scheduling with parallel machines where the number of products and machines is greater than that of the machines. A computational experiment has been conducted for a group of 30 data sets. The results showed that the new algorithm efficiently provided good solutions for tasks with large numbers of machines and products.

Keywords: lot sizing, lot scheduling, identical parallel machines, heuristics, algorithm

Mathematics Subject Classification: 68M20, 90C11

JEL Classification: O14, D24

Submitted: March 10, 2022

Revised: December 12, 2022

1. INTRODUCTION

Scheduling the deliveries of raw materials, products, and semi-finished products as well as the receipts of finished products is crucial in production company management. Setting the dates for supply and distribution depends on a production plan that determines the dates and demands for particular raw materials and semi-finished products as well as the deadlines for the completions of particular production stages. This allows us to indicate the deadlines for the productions of semi-finished products and finished goods to be delivered to customers. Therefore, logistic flow management is related to the lot sizing and scheduling that enable us to define the volumes of production lots and plan production slots and changeovers.

Lot sizing and scheduling is a problem in the field of production engineering, which deals with the development of scheduling methods for different production systems (among other things). Over the last few decades, many methods of solving

*AGH University of Science and Technology, Faculty of Management, Krakow, Poland, e-mail: roger@agh.edu.pl

DOI: https://doi.org/10.7494/2022.16.4105

lot sizing and scheduling problems have been developed (Jans & Degraeve, 2008; Pochet & Wolsey, 2006).

This paper presents a new heuristic algorithm for the task of *lot sizing and scheduling* for identical parallel machines.

In such a task, a product lot is manufactured in order to meet a deterministic demand within a finite time frame that is divided into periods. The production runs on several identical parallel machines with finite production capacities. The production output of a given machine must not exceed its capacity throughout the entire period. If a given machine manufactures various products, a changeover is required between the lots of the different types. Each changeover generates costs and consumes time.

The aim of production scheduling is to develop a production plan that minimizes inventory and production costs. Compromise solutions that minimize total costs are desired. On one hand, large lots reduce the number of changeovers, but on the other hand, they require high inventory levels and result in increased costs. With smaller production lots and lower inventory levels, the costs are moderate; however, a large number of lots means that the costs of the changeovers are frequently incurred.

The aim of this study was to develop a heuristic algorithm for identical parallel machines. Haase (1994) presented a randomized algorithm that makes backward decisions about changeovers and lot sizing (backward add-method). Due to the random nature of the decision-making rules, this algorithm provides many solutions; the best ones have been selected. This paper shows that Haase's algorithm can be applied in tasks with many identical parallel machines.

2. MODELS OF TASKS WITH PARALLEL MACHINES

In a task of lot scheduling for parallel machines, the production runs on a group of identical parallel machines with finite capacity. The products are manufactured to meet a deterministic demand on a certain scheduling horizon divided by a given number of periods. The production output of a given machine must not exceed its capacity during any scheduled period. When switching production from one product onto another, a machine changeover is required. Each changeover generates costs and takes time; its time must be shorter than the duration of the whole period. The costs and times of changeovers do not depend on the orders in which the lots are produced.

The aim of production scheduling is to develop a production plan that minimizes inventory costs and production costs.

There are two ways to formulate the mixed-integer programming (MIP) models for the tasks of lot sizing and scheduling on identical parallel machines. In the first approach, the model defines the binary variables separately for each machine (Kimms & Drexl, 1998). The binary variables indicate whether a given machine meets certain conditions that are required in the task; e.g., the machine is ready to produce a given product in a given period of time or it is not.

Such a formulation of a task results in a situation where we get many practically identical symmetrical solutions that differ only in terms of the machine numbers. Analyses of symmetrical solutions only increase the calculation workload in the branch-and-bound method, as the same solution is evaluated multiple times.

In the second approach, the task formulation is based on the aggregation of machines. The binary variables are replaced by integer variables that indicate the number of machines that fulfill certain requirements in a task; e.g., the number of machines that have been changed over and are ready to produce a given lot in a specific period.

Binary variables y_{ijt} and z_{jt} can be replaced with aggregated variables where $y_{jt} = \sum_{i \in M} y_{ijt}$ and $z_{jt} = \sum_{i \in M} z_{ijt}$. Lasdon (1971) used such variables and formulated a DLSP model with identical parallel machines. Table 1 contains a list of the basic symbols that are used to describe the models.

Table 1. *List of basic symbols*

Data
$T = \{1, ..., T\}$ – set of periods, T – number of periods,
$N = \{1, ..., N\}$ – set of products, N – number of products,
$M = \{1, ..., \mu\}$ – set of machines, μ – number of machines.
Parameters
d_{jt} – demand for product j in period t,
C_t – length of period t,
p_j – unit time of manufacturing product j,
S_j^T – time of changeover to product j,
I_{j0} – initial inventory of product j,
S_j^C – cost of changeover to product j,
h_j – unit cost of product j inventory.
Continuous variables
a_{jkt} – relative part of total capacity of all machines whose productions of product j are stopped and productions of product k are started in period t, reserved for product k after changeover,
b_{jkt} – relative part of total capacity of all machines whose productions of product j are stopped and productions of product k are started in period t, reserved for product j before changeover,
I_{jt} – inventory of product j by end of period t,
$q_{(i)jt}$ – production lot of product j in period t (on machine i).
Binary and integer variables
f_{jkt} – number of machines changed over from product j to product k in period t,
f_{jjt} – number of machines producing product j in period $t-1$ that are ready to produce product j in period t.
$y_{jt} = 1$ – machine ready to produce product j in period t, 0 – otherwise,
$z_{jt} = 1$ – production of product j is started on machine in period t, 0 – otherwise.
$y_{ijt} = 1$ – machine i is ready to produce product j in period t, 0 – otherwise (y_{ij0} – initial state of machine),
$z_{ijt} = 1$ – production of product j is started on machine i in period t, 0 – otherwise.

The PLSP binary model cannot be directly transformed into the integer model with parallel machines; transforming the PLSP task in this way would be incorrect.

Kaczmarczyk (2011) presented an appropriate model called PLSP/F. The variables of flow f_{jkt} were the only integers in this model. These indicated the flow of the machine availability "units" between two products; this means that a certain number of machines that were prepared to manufacture a particular product in a previous period has been changed over and is now ready to manufacture another product. The variables of flow indicate the order of the manufacturing of particular products in a given period. The model can be formulated as follows:

$$\min \sum_{t\in T}\sum_{j\in N}\Big(h_{jt}I_{jt} + \sum_{k\in N:j\neq k} S^C_{jk}f_{jkt}\Big) \tag{1a}$$

$$I_{jt-1} + q_{jt} = d_{jt} + I_{jt}, \qquad t \in T, j \in N, \tag{1b}$$

$$\frac{p_j}{C_t q_{jt}} \leq f_{jjt} + \sum_{k\in N:j\neq k}(b_{kjt} + a_{jkt}), \quad t \in T, j \in N, \tag{1c}$$

$$b_{jkt} + a_{jkt} = f_{jkt}\Big(1 - \frac{S^T_{jk}}{C_t}\Big), \qquad t \in T, (j,k) \in N^2 : j \neq k, \tag{1d}$$

$$f_{jk0} = 0, \qquad t \in T, j \in N, j \neq k, \tag{1e}$$

$$f_{jj0} = y_{j0}, \qquad t \in N, \tag{1f}$$

$$\sum_{k\in N} f_{kjt-1} = \sum_{k\in N} f_{jkt}, \qquad t \in T, j \in N, \tag{1g}$$

$$\sum_{(j,k)\in N^2} f_{jk0} = m, \tag{1h}$$

$$q_{ijt}, I_{jt} \geq 0, \qquad t \in T, j \in N, \tag{1i}$$

$$a_{jkt}, b_{jkt} \in [0, m], \qquad t \in T, (j,k) \in N^2 : j \neq k, \tag{1j}$$

$$f_{jkt} \in \{0, \ldots, m\}, \qquad t \in T, (j,k) \in N^2. \tag{1k}$$

Limitations (1b)–(1d) ensure the correct values of the variable production and inventory volumes. Limitations (1e)–(1h) are accountable for the correct integer values of the flow variables.

In the case of a large number of products, this model is difficult to solve using standard MIP methods; however, the task becomes easier with an increased number of identical machines.

In tasks with binary variables, an increased number of machines results in the larger scope of a task. So far, no effective heuristic algorithm has been developed that solves this problem (Kaczmarczyk, 2011).

LSPIPM (lot sizing and scheduling problem with identical parallel machines), the model that was proposed by Beraldi et al. (2008), can only be used to effectively solve small tasks; therefore, a heuristic algorithm using a rolling horizon approach and fix-and-relax decomposition technique was formulated.

Mehdizadeh et al. (2015) presented a model of linear integer programming for the problem of lot sizing and scheduling on parallel machines provided that changeover times do not depend on the sequence of the lots that are allocated to the machine.

In this task, Mehdizadeh proposed a metaheuristic approach based on the vibration-damping optimization (VDO) algorithm.

To solve the problem of lot sizing and scheduling with identical parallel machines with fixed dates of the deliveries of finished goods, Mensendiek et al. (2015) formulated a mathematical model using the model for the task of allocation with binary variables; the model was aimed at minimizing the total delay. Due to the NP-hardness of the task, a Tabu-search algorithm and a hybrid genetic algorithm were developed to solve larger tasks.

The presented examples of models for the lot sizing and scheduling problem for one machine and parallel machines show that the formulated mathematical models are usually based on allocation tasks. This type of task is characterized by a large number of variables, and binary variables are predominant in the models; therefore, the mathematical models allow for optimal solutions in small tasks due to the NP-hardness.

The real problems regarding production scheduling are most often large tasks; solutions to these problems are expected to be found quickly. Therefore, appropriate heuristic and metaheuristic algorithms are being developed in order to solve these problems effectively – especially genetic and Tabu-search algorithms.

3. NEW HEURISTICS FOR TASKS WITH PARALLEL MACHINES

This algorithm is an adaptation of the BAPLSP algorithm that was developed by K. Haase for the problem of lot sizing and scheduling for one machine. There are two variants of the algorithm: one for a task where the number of products is greater than the number of machines, and another for a task where the number of products is smaller or equal to the number of available machines.

Similarly to the BAPLSP algorithm for a single machine task, let us assume (like Haase, 1994) that the production capacity C_t in all scheduling periods t is equal to 1 and the production times of particular products p_j are also equal to 1.

The solution is calculated backwards – from the last period of lot scheduling T to the first period. We always plan the maximum possible production volume of a given product, in a given period, and on a particular machine. The obtained solution may be unacceptable if some part of the demand for a certain product remains unscheduled.

In each period of solution search $\tau \in T$, the products with unsatisfied demands are allocated to consecutive machines. The product is allocated to a machine that offers spare capacity before or after the changeover.

The products are drawn from a set of currently available N'_τ products. The set is continuously updated for each machine and a given period τ. Each product in this set has a certain priority.

For a drawn product j and a given machine m in period τ, the production lot is scheduled backwards. The initial period of lot scheduling t is equal to period τ. The lot is scheduled backwards in a specific time slot $[t';\tau]$, where period t' indicates a number of lot-scheduling periods that is equal to $(\tau - t')$. The method of calculating the number of lot-scheduling periods is thoroughly described in the section of this paper that is dedicated to the new algorithm.

General description of algorithm's operation

```
1 Draw value of control parameters
2 τ = T
3 As long as some part of demand has not been scheduled and τ > 0
4     For each machine m
5        Draw product j, where j ∈ N'_τ
6        Sequentially for  t = τ; t = τ − 1; ...; t = t'
7             Schedule lots for product  j
8     τ = τ − 1
9 Show solution
```

The value of total unsatisfied demand TD (Haase, 1994) at the beginning of the algorithm's operation is equal to the sum of the total demand for the products. During the algorithm's operation, the value of total unsatisfied demand TD is reduced by the number of produced pieces $\widetilde{q}_{jt}$ after deciding on the quantity of produced pieces $\widetilde{q}_{jt}$ of a given product j in a given period t. If the value of the total unsatisfied TD demand is greater than zero ($TD \geq 0$) after the algorithm's operation is finished, this means that the given quantity of products will not be produced on time; hence, the received solution is unacceptable.

The value of the total unsatisfied demand for a given product j is equal to the value of unsatisfied demand $\widetilde{D}_{j0}$ (Haase, 1994) in period 0; i.e., the total number of products that should be planned for production in Periods 1 through T.

Example 1. Calculate the value of unsatisfied demand $\widetilde{D}_{j,t}$ and the value of the total unsatisfied demand TD for four scheduling periods $t = (27, \ldots, 30)$ for a task with two products (k and l).

t	...	27	28	29	30
d_{kt}	...	1	2	3	4
q_{kt}	...	-	4	-	-
d_{lt}	...	5	6	7	8
q_{lt}	...	-	10	-	-

Calculated values:

$\widetilde{D}_{k,30} = (4 - 0) = 4$
$\widetilde{D}_{k,29} = \widetilde{D}_{k,30} + (3 - 0) = 7$
$\widetilde{D}_{k,28} = \widetilde{D}_{k,29} + (2 - 4) = 5$
$\widetilde{D}_{k,27} = \widetilde{D}_{k,28} + (1 - 0) = 6$
$\widetilde{D}_{l,30} = (8 - 0) = 8$
$\widetilde{D}_{l,29} = \widetilde{D}_{l,30} + (7 - 0) = 15$
$\widetilde{D}_{l,28} = \widetilde{D}_{l,29} + (6 - 10) = 11$
$\widetilde{D}_{l,27} = \widetilde{D}_{l,28} + (5 - 0) = 16$
$TD = 6 + 16 = 22$

■

Let us assume that $\widehat{d}_{jt}$ is the unsatisfied demand for a given product j in a given period t. At the beginning of the algorithm's operation, its value is equal to demand d_{jt}. If a decision is made to produce a lot of product j in quantity $\widetilde{q}$ in period t, unsatisfied demand $\widehat{d}_{jt}$ is reduced by $\min(\widehat{d}_{jt}, \widetilde{q})$.

Thus, if a lot size $\widetilde{q}$ is greater than unsatisfied demand $\widehat{d}_{jt}$, the products that are manufactured in excess of the needs in period t will cover the demands of the subsequent periods. Due to the increasing warehousing costs, it was assumed that those products that were produced in period t would cover the demands in consecutive periods as much as possible. The procedure is continued until all of the surplus production from period t is used.

Therefore, the values of the unsatisfied demands to be planned for future periods $\widehat{d}_{jt+1}, \widehat{d}_{jt+2}, \ldots, \widehat{d}_{jT}$ also change. If lot size $\widetilde{q}$ is greater than unsatisfied demand $\widehat{d}_{jt}$ in the considered case, the unsatisfied demand $\widehat{d}_{jt}$ in period t will certainly be equal to 0, while it will decrease correspondingly to the volume of the planned lot $\widetilde{q}$ in periods $t+1$ and $t+2, t+3 \ldots$.

Lot size $\widetilde{q}$ deliberately does not have any subscript with a product or period because, during the algorithm's operation, we might decide to create a production lot of a given product more than once in a given period (in this situation, the planned production $q_{jt} = \widetilde{q}' + \widetilde{q}'' + \ldots$ for product j in period t).

The unsatisfied demand $\widehat{d}_{jt}$ to be scheduled in a time slot $n = [t, \ldots, T]$ and the decision to produce a certain lot size $\widetilde{q}$ of a given product j in period t can be described as a recurrence relationship:

$$\begin{cases} \widehat{d}_{jn} \leftarrow \max(0, \widehat{d}_{jn} - \widetilde{q}); & \widehat{q}_n \leftarrow \max(0, \widetilde{q} - \widehat{d}_{jn}); \\ \\ \widehat{d}_{j,n+1} \leftarrow \max(0, \widehat{d}_{j,n+1} - \widehat{q}_n); & \widehat{q}_{n+1} \leftarrow \max(0, \widehat{q}_n - \widehat{d}_{j,n+1}); \end{cases} \tag{2}$$

Example 2. Calculate the values of unsatisfied demand $\widehat{d}_{k,2}$ for a certain product k and $\widetilde{q} = 9$ in the 26^{th} period of production. The table presents the initial values of unsatisfied demand $\widehat{d}_{kt}$ to be scheduled in the particular periods.

t	...	25	26	27	28	29	30
$\widehat{d}_{kt}$	...	5	5	3	2	1	2

$n = 26$

$\widehat{d}_{k,26} \leftarrow \max(0; 5 - 9)$ $\qquad$ $\widehat{q}_{26} \leftarrow \max(0; 9 - 5)$

$\widehat{d}_{k,26} = 0$ $\qquad$ $\widehat{q}_{26} = 4$

$n = 27$

$\widehat{d}_{k,27} \leftarrow \max(0; 3 - 4)$ $\qquad$ $\widehat{q}_{27} \leftarrow \max(0; 4 - 3)$

$\widehat{d}_{k,27} = 0$ $\qquad$ $\widehat{q}_{27} = 1$

$n = 28$

$\widehat{d}_{k,28} \leftarrow \max(0; 2 - 1)$ $\qquad$ $\widehat{q}_{28} \leftarrow \max(0; 1 - 2)$

$\widehat{d}_{k,27} = 1$ $\qquad$ $\widehat{q}_{27} = 0$

$n = 29$

$\widehat{d}_{k,29} \leftarrow \max(0; 1-0)$ $\quad$ $\widehat{q}_{28} \leftarrow \max(0; 0-1)$

$\widehat{d}_{k,29} = 1$ $\quad$ $\widehat{q}_{29} = 0$

$n = 30$

$\widehat{d}_{k,30} \leftarrow \max(0; 2-0)$ $\quad$ $\widehat{q}_{28} \leftarrow \max(0; 0-2)$

$\widehat{d}_{k,30} = 2$ $\quad$ $\widehat{q}_{30} = 0$

■

For each period of backward scheduling τ, let $\widehat{D}_{jt}$ indicate the unsatisfied demand for product j in any given period t summed up in a time slot $W = [\tau - \lambda_j, \ldots, T]$:

$$\widehat{D}_{j,t} = \begin{cases} \sum_{s=t}^{T} \widehat{d}_{js}, & \text{if} \quad t \in W, \\ \widehat{d}_{jt}, & \text{else}, \end{cases} \quad t = 1, \ldots, T. \tag{3}$$

A random parameter λ_j for a given product j determines the maximum duration of the backward lot scheduling that results in a build-up of inventory. Therefore, when planning lots for a certain backward scheduling period τ based on unsatisfied demand $\widehat{D}_{j,t}$, the costs of warehousing the produced goods will occur only between periods $\tau - \lambda_j$ and τ.

Example 3. Calculate the value of the unsatisfied demand $\widehat{D}_{k,t}$ for a given product k in the 29^{th} period of the solution search and with the value of a random parameter $\lambda_k = 2$. The table presents the initial values of unsatisfied demand $\widehat{d}_{kt}$ to be scheduled in the particular periods.

t	$\ldots$	25	26	27	28	29	30
$\widehat{d}_{kt}$	$\ldots$	5	5	3	2	1	2

$W = [29-2, \ldots, 30] = [27, \ldots, 30]$

$\ldots$

$\widehat{D}_{k,25} = 5$ $\quad$ $(t = 25) \notin W$

$\widehat{D}_{k,26} = 5$ $\quad$ $(t = 26) \notin W$

$\widehat{D}_{k,27} = 3 + 2 + 1 + 2 = 8$ $\quad$ $(t = 27) \in W$

$\widehat{D}_{k,28} = 2 + 1 + 2 = 5$ $\quad$ $(t = 28) \in W$

$\widehat{D}_{k,29} = 1 + 2 = 3$ $\quad$ $(t = 29) \in W$

$\widehat{D}_{k,30} = 2$ $\quad$ $(t = 30) \in W$

■

For each period of backward scheduling τ, let $\breve{R}_{j\tau}$ indicate a certain expected reserve that is saved for a given product j in time slot $W = [\tau - \psi, \ldots, T]$:

$$\breve{R}_{j\tau} = \sum_{s \in W} \widehat{d}_{js}, \quad t \in W. \tag{4}$$

If a random parameter $\psi = 0$, then an expected reserve $\breve{R}_{j\tau}$ for a given product j in scheduling period τ equals total unsatisfied demand $\breve{R}_{jt} = \sum_{s\in[t,\ldots,T]} \widehat{d}_{js}$.

Example 4. Calculate the values of the expected saved inventory $\breve{R}_{k29}$ of a given product k in the 29^{th} scheduling period for random values of parameter $\psi = (0, 2)$. The table presents the initial values of unsatisfied demand $\widehat{d}_{kt}$ in the particular periods.

t	...	25	26	27	28	29	30
$\widehat{d}_{kt}$	...	5	5	3	2	1	2

$\psi = 0$
$\breve{R}_{k,29} = 1 + 2 = 3$ $\qquad W = [29 - 0, 30] = [29, 30]$
$\psi = 2$
$\breve{R}_{k,29} = 3 + 2 + 1 + 2 = 8$ $\qquad W = [29 - 2, 30] = [27, \ldots, 30]$

■

Let $r_{j\tau}(i)$ indicate the value of the lost profit if product j is not selected (assuming that a given machine is available to produce product i in period in period τ and the random parameter is σ):

$$r_{j\tau}(i) = \begin{cases} (1-\gamma)\breve{R}_{j\tau}h - \gamma S_j^C & \text{for } j \neq i; \widehat{D}_{j,\tau} \geq \sigma - 1 \\ (1-\gamma)\breve{R}_{j\tau}h & \text{for } j = i; \widehat{D}_{j,\tau} \geq \sigma - 1 \\ -\infty & \text{otherwise} \end{cases} \tag{5}$$

The lost profit value is defined in the same way as in Haase's (1994) work on the problem of lot sizing and scheduling for production on a single machine.

For each machine m and product j in period τ, the value of the lost profit is calculated as the sum of the saved (positive) costs of inventory $\breve{R}_{j\tau}h_j$ and the incurred (negative) costs of launching a new production lot S_j^C.

It is possible to calculate the value of the lost profit and allow for the possibility of drawing a product to be scheduled for production if the unsatisfied demand $\widehat{D}_{j\tau}$ for a given product j in a given period τ is higher than minimum-required unsatisfied demand $\sigma - 1$.

Random parameter σ indicates the minimum required unsatisfied demand $\widehat{D}_{j,\tau}$ for a given product j in a given period τ. A parameter that is equal to 1 does not limit the possibility of product selection. A parameter that is equal to 2 means that the demand for a given product is sufficient to schedule its production for at least one period. Respectively, a parameter that is equal to 3 – not less than 2 periods, etc.

Parameter γ determines the weights of the incurred costs. Depending on its value, this parameter allows us to control the lot size (parameter γ has a value within a range of $[0; 1]$). If the value of parameter γ is close to 1, relatively large production lots should be expected. As the changeover cost has a significant influence on the value of lost profit $r_{j\tau}$, those products with changeover costs that are lower than S_j^C are more likely to be selected.

However, if parameter γ is close to 0, we can expect a greater number of relatively small production lots to be created. The savings on inventory costs $\breve{R}_{j\tau}h_j$ are considered to be more important.

Let $\rho_{j\tau}(i)$ be the criterion for selecting a product to be manufactured. Similarly to the case with one machine that was presented by Haase (1994):

$$\rho_{j\tau}(i) = \begin{cases} 0; & \text{for } r_{j\tau} = -\infty, \\ (r_{j\tau} - \min\{r_{k\tau} | k \in N \wedge r_{k\tau} > -\infty\} + \varepsilon)^{\delta}; & \text{else,} \end{cases} \tag{6}$$

where $\delta \geq 0, \varepsilon > 0$.

Those products that are admitted to the draw have values of decision criterion $\rho_{j\tau}$ that are greater than 0 and are "compared" with the worse options (those products with lower $\rho_{j\tau}$ values). If the value of decision criterion $\rho_{jt}(i)$ is equal to 0, the product is not taken into account in the draw. A high value of the decision criterion means that the product is more likely to be selected. Therefore, the value of parameter $\delta > 1$ increases the chances of products with higher lost profit values $r_{j\tau}(i)$; conversely, the chances decrease when $\delta < 1$. Furthermore, parameter $\epsilon > 0$ makes it probable for a product with the worst lost profit value $\min\{r_{k\tau}(i) | k \in N \wedge r_{k\tau}(i) > -\infty\}$ to be drawn.

The products are drawn based on the calculated $\rho_{j\tau}(i)$ values.

Example 5. Calculate the criterion values for three products (j, k, l) in the 27^{th} period of a solution search and a given machine m that is available to produce product k in this period. The inventory costs and changeover costs in this period are equal for all of the products and are 2 and 10, respectively.

t	...	25	26	27	28	29	30
$\widehat{d}_{jt}$	...	0	0	0	0	0	1
$\widehat{d}_{kt}$	...	5	5	3	2	1	2
$\widehat{d}_{lt}$	...	2	7	0	0	0	0

Parameter	Value
ψ	2
γ	0.3
δ	2
ϵ	1
σ	2

$\widehat{D}_{j,27} = \widetilde{D}_{j,27} = 0 + 0 + 0 + 1 = 1$
$\widehat{D}_{k,27} = \widetilde{D}_{k,27} = 3 + 2 + 1 + 2 = 8$
$\widehat{D}_{l,27} = \widetilde{D}_{l,27} = 0 + 0 + 0 + 0 = 0$

$\breve{R}_{j,27} = 0 + 0 + \widehat{D}_{j,27} = 1$
$\breve{R}_{k,27} = 5 + 5 + \widehat{D}_{k,27} = 18$
$\breve{R}_{j,27} = 2 + 7 + \widehat{D}_{l,27} = 9$

$r_{j,27}(k) = 0.7 \cdot 1 \cdot 2 - 0.3 \cdot 10 = -1.6$
$r_{k,27}(k) = 0.7 \cdot 18 \cdot 2 = 25.2$
$r_{l,27}(k) = -\infty$

$\rho_{j,27}(k) = (-1.6 + 1.6 + 1)^2 = 1$
$\rho_{k,27}(k) = (25.2 + 1.6 + 1)^2 = 772.84$
$\rho_{l,27}(k) = 0$

■

The BAPLSP/F$^{N>\mu}$ algorithm for lot sizing and scheduling for parallel machines has been developed for those cases where the numbers of products are greater than the numbers of machines.

BAPLSP/F$^{N\leq\mu}$ is an algorithm for those cases where the numbers of products are smaller or equal to the numbers of machines.

The formal description of the stochastic heuristic $BAPLSP/F$ algorithm is as follows for the cases of BAPLSP/F$^{N>\mu}$ and BAPLSP/F$^{N\leq\mu}$:

BAPLSP/F$^{N>\mu}$, BAPLSP/F$^{N\leq\mu}$:

Initialization of variables:

$\gamma,\ \delta,\ \epsilon,\ \psi,\ \omega,\ \alpha$ {parameter drawing}

$\widetilde{y}_{m\tau} := \varnothing,\ c_{m\tau} := 1,\ m \in M, \tau \in T$ {[no availability to manufacture products, initial production capacity}

$i_m := \varnothing,\ t_m := T,\ m \in M$ {lack of product, we are considering period for machine m}

$q_{j\tau} := 0,\ \widehat{D}_{j\tau},\ TD, \widehat{d}_{j\tau},\ j \in N, \tau \in T$ {initialization of variables}

Algorithm's operation:

$\tau := T$ {starting from last period}

`while` $TD > 0\ \wedge\ \tau > 0$: {start of Loop 1}

  `for` $m \in M$ `if` $\tau \leq t_m$: {one by one for each available machine in period}

    $t_m := \tau$

    `if` $\widetilde{y}_{m,t_m} = \varnothing\ \wedge\ t_m < T$

      $\widetilde{y}_{m,t_m} := i_m$ {transfer of machine availability}

    `random` $\sigma_j;\ j \in N$

    `if random uniform` $(0,1) \leq \alpha$:

      $\lambda_j :=$`random`$(\sigma_j, max(\sigma_j + 1, t_m);\ j \in N$

    `else:` $\lambda_j := t_m$

    `determine` $r_{j,t_m}(i_m);\ \rho_{j,t_m}(i_m);\ j \in N$

    `if` $\sum_{j\in N} \rho_{j,t_m}(i_m) = 0$:

      $t_m := t_m - 1;$ `continue for next` m

    $\widehat{i} :=$`random proportional to` $\rho_{j,t_m}(i_m)$

    `determine` $\widehat{D}_{\widehat{i},t_m}$

    `while` $\widehat{D}_{\widehat{i},t_m} > 0\ \wedge\ t_m > 0$: {start of Loop 2, schedule for selected product}

      $\widetilde{q} := 0,\ \widetilde{c} := 0$ {initial production volume, demand for production capacity}

      `if` $\widetilde{y}_{m,t_m} = \varnothing\ \vee\ \widetilde{y}_{m,t_m} = \widehat{i}$: {cases for changeovers}

        $\widetilde{y}_{m,t_m} := \widehat{i}$

        $\widetilde{q} := \widetilde{c} := \min(c_{m,t_m}, \widehat{D}_{\widehat{i},t_m})$

      `elseIf` $\widetilde{y}_{m,t_m} \neq \widehat{i} \wedge c_{m,t_m} \geq S^T_{i_m}$:

        $\widetilde{y}_{m,(t_m-1)} := \widehat{i}$

        $\widetilde{q} := max(0, min(c_{t_m,m} - S^T_{i_m}, \widehat{D}_{i_m,t_m})$

        $\widetilde{c} := \widetilde{q} + S^T_{i_m}$

      `elseIf` $\widetilde{y}_{m,t_m} \neq \widehat{i} \wedge c_{mt_m} < S^T_{i_m}$:

        $\widetilde{y}_{m,t_m-1} := \widehat{i},\ t_m := t_m - 1;$ `continue while`

      $q_{\widehat{i},t_m} := q_{\widehat{i},t_m} + \widetilde{q},\ c_{m,t_m} := c_{m,t_m} - \widetilde{c}$

      `determine` $\widehat{d}_{\widehat{i},t_m},\ \widehat{D}_{\widehat{i},t_m},\ TD$

33 **if** $c_{m,t_m} \leq \omega$ $\boxed{\vee\ t_m \geq \tau - \lambda_{\widehat{i}}^*}$: {Usage of production capacity, *–extension for BAPLSP/F$^{N>\mu}$ }

34 $t_m \coloneqq t_m - 1$; **else: break**

35 $i_m \coloneqq \widehat{i}$

36 **if all** $t_m < \tau$: $\{\tau \coloneqq \tau - 1$ **and** $\widehat{D}_{j\tau} = \widehat{D}_{j,(\tau+1)} + \widehat{d}_{j\tau};\ j \in N\}$

In Line 1, all of the values of the fixed parameters are drawn for the entire run of the solution-search algorithm.

In Line 2, a given machine m is not ready ($\widetilde{y}_{mt}$) to produce any product in a given period t at the beginning of the algorithm's operation. The remaining production capacity c_{mt} of a given machine m in a given period t is equal to the total available production capacity in period t, as we are not producing anything yet.

In Line 3, the last product i_m that is produced on machine m is undefined. The period of machine availability t_m in which we can start planning products is equal to the last planning period T.

In Line 4, the initial production volume of a given product j in a given scheduling period τ is equal to zero. The volume of the unsatisfied demand $\widetilde{D}_{j,t}$ for particular products and periods is calculated, as is total unsatisfied demand TD. The unsatisfied demand $\widehat{D}_{jt}$ for particular products and periods is calculated for time slot $W = [\tau, \ldots, T]$. The volume of the unscheduled $\widehat{d}_{j\tau}$ products is initially equal to demand d_{jt}.

In Line 6, the scheduling starts from the last period $t = T$. The solution search continues until the total TD demand is met.

In Line 7, the production lots of given products are successively launched on all machines m. The schedule for a given machine m is created (provided that it is available in the given period τ – it is not producing another product), and the available production capacity allows us to plan the production in Periods 1 through τ.

In Lines 9 and 10, if a given machine m does not produce any product for a certain number of periods subsequent to τ and the machine is to produce a certain product in a period after the non-production period, then the machine is available ($\widetilde{y}_{m,t_m}$) to produce the last product i_m in period τ.

In Lines 11–14, two parameters are drawn (σ_j, and λ_j) for all products $j \in N$.

If the unsatisfied demand for product j in period τ is greater than or equal to σ_j, a new production lot will be created (the lot will not be shorter than σ_j) – compare this to 5.

The λ_j parameter determines the length of the backward scheduling of a given product that results in inventory build-up. The value of parameter λ_j is drawn with a certain probability of α; it is a random input parameter for a given run of the new algorithm. This determines the probability of a situation where we draw the value of parameter λ_j; otherwise, parameter λ_j is equal to the maximum possible length of the backward scheduling of a given product. Random values of parameter λ_j for product j belong to a range that is limited by the minimum length of created production lot σ_j and the maximum possible length of the backward scheduling of a given product j.

In Line 15, the value of lost profit $r_{jt}(i_m)$ and the value of the criterion for product-selection decision $\rho_{jt}(i_m)$ are calculated for all products $j \in N$ for a given

machine m and the availability of this machine for the production of product i_m in period τ ($t_m = \tau$) – compare Formulae (5) and (6).

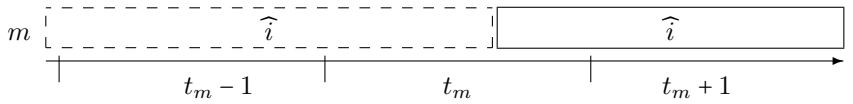
In Lines 16–18, product $\widehat{i}$ is drawn for lot scheduling. If there is no product $\sum_{j \in N} \rho_{jt} = 0$ that is available to be drawn for a given machine m in a given period τ, the solution for the next machine is created.

In Line 19, we calculate the total unsatisfied demand $\widehat{D}_{\widehat{i},t_m}$ for a drawn product $\widehat{i}$ on machine m in a period of lot scheduling for a given machine t_m (at the beginning of the backward lot sizing of product i, this period is equal to period τ).

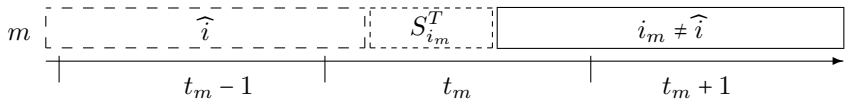
In Line 20, we schedule the production lots backwards until the unsatisfied demand $\widehat{D}_{\widehat{i},t_m} > 0$ for product $\widehat{i}$ in a period of lot scheduling t_m for a given machine m is fulfilled. Lot-sizing period t_m must be within planning horizon $t_m \geq 0$; the first period is the last period in the backward scheduling.

In Line 21, we define the size of production lot $\widetilde{q}$ and production capacity demand $\widetilde{c}$ in period t_m; the values are calculated accordingly:

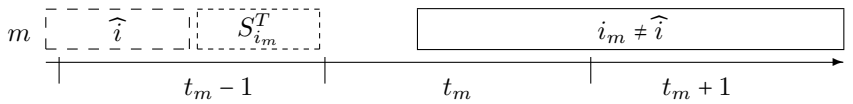
1) When machine m is available to produce the same selected product $\widehat{i}$ (or is not ready to produce any product – this situation occurs only once for each machine; i.e., when the production on a given machine starts for the first time [an illustration of this case is not provided]) (Lines 22–24):



2) When machine m is available to produce a different product than the selected product $\widehat{i}$, and the remaining production capacity c_{m,t_m} of machine m in lot-scheduling period t_m allows for a changeover (Lines 25–28):



3) When machine m is available to produce a different product than the selected product $\widehat{i}$, and the remaining production capacity c_{m,t_m} of machine m in lot-scheduling period t_m does not allow for a changeover (Lines 29–30):



In the first case, the production lot size $\widetilde{q}$ and capacity demand $\widetilde{c}$ in the lot-scheduling period t_m of the selected product $\widehat{i}$ and for a given machine m are calculated on the basis of available production capacity c_{m,t_m} and total demand $\widehat{D}_{\widehat{i},t_m}$.

In this case, we do not use available production capacity c_{m,t_m} to make a changeover. We use all of the capacity to manufacture the product (or to continue the previously started lot of product $\widehat{i}$). The selected machine m is available to produce product $\widehat{i}$ in lot-scheduling period t_m.

In the second and third cases, it is necessary to take the time that is required to make a changeover $S^T_{i_m}$ into account when calculating the value of production capacity demand $\widetilde{c}$ in order to start producing the lots of product i_m later on. In both cases, machine m is available to produce product i_m in period t_m due to a change of products on a given machine m and the resulting changeover $S^T_{i_m}$; in the earlier period $t_m - 1$, it is available to produce a selected product $\widehat{i}$.

In the second case, we still have some spare production capacity $c_{t_m,m} - S^T_{i_m}$ in period t_m after the changeover; we use this to start the production of product $\widehat{i}$ in this period.

In the third case, the available production capacity $c_{t_m,m}$ does not allow for the required changeover S_{i_m} in period t_m; so, the launch of a new lot is only possible in the previous lot-scheduling period $t_m - 1$.

In Line 32, the number of unscheduled products $\widehat{d}_{\widehat{i},t_m}$ and total unsatisfied demand $\widehat{D}_{\widehat{i},t_m}$ for a given product $\widehat{i}$ in lot-scheduling period t_m are calculated anew in case the production of product $\widehat{i}$ in quantity $\widetilde{q}$ is possible in period t_m. The size of unsatisfied TD demand is reduced by the volume of the scheduled production $\widetilde{q}$ of the given product $\widehat{i}$.

In Lines 33 and 34, a decision is made regarding whether or not to continue lot scheduling for product $\widehat{i}$ and given machine m. If the spare production capacity c_{m,t_m} of a given machine m in lot-scheduling period t_m is less than or equal to the assumed permissible level of unused available production capacity ω, the scheduling for a given product $\widehat{i}$ is continued. Otherwise, the scheduling for a given product $\widehat{i}$ in scheduling period t_m is completed.

For those tasks where the numbers of products are less than or equal to the numbers of machines, the production lot is always continued if the lot-scheduling period t_m for machine m fits in a time slot $[\tau - \lambda_{\widehat{i}}, \ldots, \tau]$.

In Line 35, we set the availability of machine i_m (the last product that is produced on the machine) for a given product $\widehat{i}$after completing the (backward) lot scheduling for a given product $\widehat{i}$.

In Line 36, the scheduling returns to the previous period $\tau - 1$ if all of the possible machines $m \in M$ are used in a given period τ. Additionally, the value of total unsatisfied demand $\widehat{D}_{j,\tau-1}$ is updated for each product.

4. COMPUTATIONAL EXPERIMENTS

In order to evaluate the new BAPLSP/F algorithm, computational experiments were conducted for a group of different data sets.

Optimal solutions were sought with GUROBI software (Version 6.0.4); the GUROBI solver is recognized as the state-of-the-art solver for mathematical programming (GUROBI, 2022). The GUROBI solver was designed from scratch using a modern multi-core processor architecture; it allows us to solve tasks that are formulated as LP (linear programming) models, mixed-integer linear programming (MILP) models, quadratic programming (QP) models, etc. The GUROBI solver can solve models with

millions of variables and limitations on standard mobile and desktop computers; it also provides interfaces for the majority of the popular programming languages.

The new heuristic algorithm has been implemented in the Python 2.7 programming language. It allows us to code a clear and synthetic program; thus, it is often used in software prototyping. In order to accelerate the process, the calculations were performed with a PyPy compiler, which is a just-in-time compiler (JIT). The first time a function code fragment is called, it compiles it into a machine code (PyPy, 2022). The machine code consists of a sequence of binary numbers that are direct commands and arguments for the processor.

4.1. Data sets

A group of 30 data sets was developed for the purpose of our experiments. The sets were randomly generated based on experiments and industrial data. They were developed around two actual examples of scheduling productions for four weeks. The first example (with three products) came from the electronics industry (Kaczmarczyk, 2006), and the second one (with two products) – from the automotive industry (Miodońska, 2006).

All of the data sets had a number of periods T that was equal to 30 and a fixed period length C that was equal to 100. The tasks were diversified in terms of the numbers of products ($N = 5, 10, 15$) and machines ($\mu = 5, 10$). All of the data was randomly generated as integers by means of a uniform distribution U. The random parameters for each data set were as follows:

- The demand d_{jt} for product j in period t, was drawn from uniform distribution $U[1, 100]$ and reduced to 0 with a 0.4 probability; in addition, the demand in six initial scheduling periods was always 0 in order to ensure that there were acceptable solutions.
- Unit time p_j of manufacturing product j was randomly generated by means of a uniform distribution $U[1, 5]$.
- The unit cost h_j of product j's inventory was randomly generated by means of a uniform distribution $U[1, 5]$.
- The changeover time S_j^T for product j was randomly generated by means of a uniform distribution $U[0.2C, 0.8C]$ for equal production times of the machines in periods C.
- The changeover cost S_j^C of product $j$was randomly generated by means of a uniform distribution $U[10H_j, 150H_j]$, where $H_j = h_j C/p_j$ is equal to the total inventory cost of product j in a quantity that is equal to the daily production volume for the duration of one period.

The demand was drawn so that the level of the machine utilization was within a range of 75–90%. We assumed three changeovers for each product. Table 2 presents the average machine utilization levels without changeover times for the data sets that were used in the computational experiments.

Table 2. *Average machine-utilization level without changeover times for data sets that were used in computational experiments*

N/μ	5 [%]	10 [%]
5	79.8	82.2
10	86.0	75.0
15	80.0	87.8

N – number of products, μ – number of machines

4.2. Results of computational experiments

All of the calculations were made using a computer with an Intel COREi7 processor, 4710HQ, and 16 GB of RAM.

Solutions were sought with GUROBI 6.0.4 software with standard settings using the PLSP/F (1) model for the task of lot sizing and scheduling for identical parallel machines.

Heuristic solutions were found for all of the data sets by means of the heuristic BAPLSP/F algorithm for the same ranges of random parameters that were generated using a U-distribution or an integer distribution U^{INT}. The ranges of the random parameter values are shown in Table 3.

Table 3. *Value ranges for random parameters*

Parameter	Distribution	Value
γ	U	$[0.05, 0.95]$
δ	U	$[0.00, 30.00]$
ϵ	U	$[0.00, 1.00]$
ψ	U^{INT}	$[0, 3]$
σ	U^{INT}	$[1, 3]$
ω	U	$[0.00, 1.00]$
α	U	$[0.20, 1.00]$

U – uniform distribution, U^{INT} – integer uniform distribution

For each case, ten thousand runs of the BAPLSP/F heuristic algorithm were performed. From the acceptable solutions for the given task, the task with the best value of the target function was chosen as the solution.

The calculation time of ten thousand algorithm runs (Table 4) ranged from about 2 to 5 seconds depending on the size of the task. For all of the data sets, the search for optimal solutions was carried out with the GUROBI software. The calculation times were limited to 600 seconds each.

Let GAP determine the distance between the previously found acceptable integer solution and the best-known estimate for the task:

$$GAP = \left(\frac{|ObjBound - ObjVal|}{|ObjVal|} \right) 100\%, \tag{7}$$

where:

$ObjBound$ – best-known estimate for task,

$ObjVal$ – best-known solution for task.

Let G determine the relative distance between the obtained BAPLSP/F solution and the target function value of the solution that was computed with GUROBI:

$$G = \left(\frac{FC(BAPLSPF)}{FC(GUROBI)} - 1 \right) 100\%, \tag{8}$$

where:
$FC(BAPLSPF)$ – value of target function computed with new algorithm,
$FC(GUROBI)$ – value of target function computed with GUROBI software.

The average quality G of the solutions that were obtained using the BAPLSP/F algorithm for the particular data sets is shown in Table 4.

Table 4. *Results of computational experiments*

μ	N	G [%]	Time of calculations [s]		GAP [%]
			Heuristics	MIP	
5	5	4.75	1.89	3.96	0.00
	10	3.50	2.21	600.00	27.60
	15	–	3.31	600.00	–
10	5	0.99	2.41	1.23	0.00
	10	9.63	3.89	380.96	3.35
	15	–	4.78	600.00	–

μ – number of machines,
N – number of products,
MIP – by means of GUROBI solver.

For those sets with 15 products, it was not possible to calculate acceptable solutions for the PLSP/F task with the GUROBI solver. For those sets with five products (be it for five or ten machines), the GUROBI solver found the optimal solutions ($GAP = 0$). The values of the target functions in the solutions that were calculated using the BAPLSP/F algorithm were 4.71% worse on average than the solutions that were calculated using the GUROBI software in a maximum time of 600 seconds. The actual average calculation time for the new algorithm and the GUROBI solver showed that those sets with the smallest numbers of machines and with the smallest numbers of products had similar solution search times. As far as the remaining sets, the GUROBI software needed much more time to find a solution with a quality that was comparable to that of a heuristic solution (Table 4).

The heuristic algorithm computed the worst solution for the task with five products and five machines. The value of the target function was about 11% worse than could be found in the optimal solution. In another task with five products and five machines, a solution that was close to the optimal solution was found (the difference in the target function value being about 0.3%).

Figure 1 shows the average quality of the solutions G that were obtained with the BAPLSP/F algorithm as compared to the best-known solutions that were obtained

with GUROBI (for the PLSP/F model). GUROBI needed 5 s, 10 s, 50 s, 100 s, and 600 s, respectively, to find a solution.

Those solutions that were computed with GUROBI in times that were comparable to the new BAPLSP/F algorithm were 24% worse on average than those that were obtained with the new heuristics. A doubled solution search time resulted in a less than 9% average improvement in the obtained solutions. The solutions that were found after 50 seconds were only about 4% worse than the heuristic solutions.

For the prepared data sets, the GUROBI software needed about 100 seconds to provide solutions that were similar in quality to those that were obtained by means of the new heuristic algorithm in less than 5 seconds. Extending the time of the solution search with the GUROBI software by more than 100 seconds allowed us to improve the quality of the obtained solutions by only a few percentage points on average.

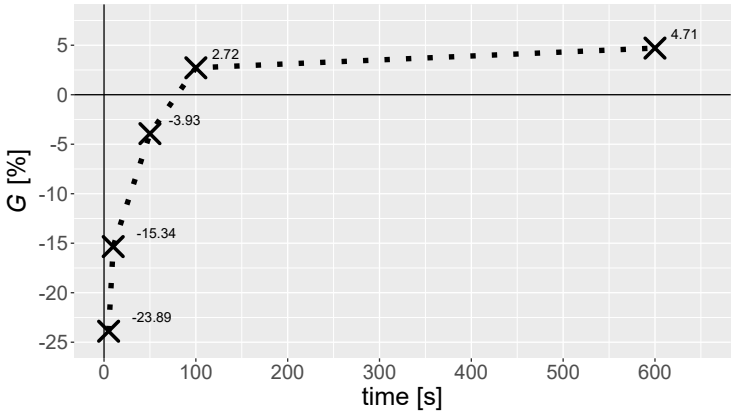


Fig. 1. *Average quality of solutions G found by GUROBI: 5 s, 10 s, 50 s, 100 s, and 600 s*

5. CONCLUSIONS

Lot sizing and scheduling plays an important role in the management of a production company. It is not always possible to use PLCM methods. As the PLCM models of lot sizing and scheduling are NP-hard, a small increase in a number of products significantly increases the time that is required to find a solution. In certain practical situations, waiting a long time for a good solution is not acceptable.

Therefore, new heuristic algorithms are constantly being developed for various lot-sizing and scheduling problems.

This paper describes a new algorithm for the task of lot-sizing and production scheduling with parallel machines. This algorithm is an adaptation of the heuristic BAPLSP algorithm that was proposed in 1994 by Hasse for a task with one machine.

The reduced time that is required to obtain a solution is an advantage of the new heuristic algorithm (BAPLSP/F). As a result, it is possible to run the BAPLSP/F algorithm to search for a solution many times. In such a case, the best solution is selected from a set of acceptable solutions. Each attempt to find a solution means a separate run of an algorithm; the algorithm does not "learn" during its operation.

As the computational experiments have shown, the new algorithm efficiently provides good solutions for tasks with large numbers of products and machines. Increased numbers of products only slightly extends the times that are required to find solutions for the tasks of lot sizing and scheduling with parallel machines. Therefore, the practical application of the new BAPLSP/F algorithm that is proposed in this paper has become possible.

REFERENCES

Beraldi P., Ghiani G., Grieco A. & Guerriero E. (2008). Rolling-horizon and fix-and-relax heuristics for the parallel machine lot-sizing and scheduling problem with sequence-dependent set-up costs. *Computers and Operations Research*, **35**(11), pp. 3644–3656. DOI: 10.1016/j.cor.2007.04.003.

GUROBI (2022). http://www.gurobi.com/products/gurobi-optimizer [11.12.2022].

Haase K. (1994). Lotsizing and Scheduling for Production Planning. *Lecture Notes in Economics and Mathematical Systems,* vol. 408, Springer-Verlag.

Jans R.F. & Degraeve Z. (2008). Modeling industrial lot sizing problems: A review. *International Journal of Production Research*, **46**(6), pp. 1619–1643. DOI: 10.1080/00207540600902262.

Kaczmarczyk W. (2006). Modele PLC planowania wielkości i szeregowania partii z identycznymi liniami równoległymi. *Zeszyty Naukowe. Automatyka / Politechnika Śląska*, **144**, pp. 23–32.

Kaczmarczyk W. (2011). Proportional lot-sizing and scheduling problem with identical parallel machines. *International Journal of Production Research*, **49**(9), pp. 2605–2623. DOI: 10.1080/00207543.2010.532929.

Kimms A. & Drexl A. (1998). Some insights into proportional lot sizing and scheduling. *Journal of the Operational Research Society*, **49**(11), pp. 1196–1205. DOI: 10.2307/3010100.

Lasdon L. & Terjung R. (1971). An efficient algorithm for multi-item scheduling. *Operations Research*, **19**, pp. 946–969. DOI: 10.1287/opre.19.4.946.

Mehdizadeh E., Tavakkoli-Moghaddam R. & Yazdani M. (2015). A vibration damping optimization algorithm for a parallel machines scheduling problem with sequence-independent family setup times. *Applied Mathematical Modelling*, **39**(22), pp. 6845–6859. DOI: 10.1016/j.apm.2015.02.027.

Mensendiek A., Gupta J.N.D. & Herrmann J. (2015). Scheduling identical parallel machines with fixed delivery dates to minimize total tardiness. *European Journal of Operational Research*, **243**(2), pp. 514–522. DOI: 10.1016/j.ejor.2014.12.002.

Miodońska B. (2006). *Koordynacja w łańcuchach dostaw* [MSc Thesis]. AGH University of Science and Technology, Krakow.

Pochet Y. & Wolsey L. (2006). Production Planning by Mixed Integer Programming. *Springer Series in Operations Research and Financial Engineering,* Springer, New York. DOI: 10.1007/0-387-33477-7.

PyPy (2022). https://pypy.org/ [12.11.2022].