STUDIA INFORMATICA 2015
Volume 36 Number 3 (121)

Jerzy RASZKA, Lech JAMROŻ
Cracow University of Technology, Institute of Computer Science

HEURISTIC ALGORITHM FOR SCHEDULING MULTIPLE-OPERATION JOBS OF CYCLIC MANUFACTURING PROCESSES

Summary. The paper deals with the problem of evaluating the performance of job-shop systems under a cyclic manufacturing process. This process for each job is specified uniquely. Modelling based on timed event graphs terminology allows the performance of a system of repetitive production processes to be evaluated. The objective of the schedule production is to minimize the cycle time and the number of jobs that are actually required in the process. The considered problem is specified as a linear programming problem and is solved by a heuristic algorithm.

Keywords: Job-shop, timed event graph, cycle time, performance evaluation

HEURYSTYCZNY ALGORYTM PLANOWANIA WIELOOPERACYJNYCH ZADAŃ CYKLICZNYCH PROCESÓW PRODUKCYJNYCH

Streszczenie. Artykuł dotyczy oszacowania wydajności systemów typu job-shop, w których występują cykliczne procesy produkcyjne. Proces produkcyjny jest jednoznacznie określony dla każdego zadania. Modelowanie oparte na terminologii czasowych grafów znakowanych pozwala oszacować wydajność systemu cyklicznych procesów produkcyjnych. Celem planowania produkcji jest minimalizacja czasu cyklu i liczby zadań wymaganych w procesie. Rozważany problem przedstawiony jest jako problem programowania liniowego i rozwiązywany za pomocą algorytmu heurystycznego.

Słowa kluczowe: job-shop, czasowy graf znakowany, czas cyklu, oszacowanie wydajności

1. Introduction

The application of effective methods and models in manufacturing field are determined by automation of production control. Scheduling is one of the most fundamental problems to be solved within the manufacturing industry. It relates to the sequence of operations on machines and is directly linked with productivity. There is a great need for improving scheduling within the manufacturing environment. Scheduling problems, generally considered within large scale systems, are difficult to solve in most practical situations. The logical strategy is to pursue scheduling methods which consistently and efficiently generate good schedules [8, 13].

We have considered a cyclic manufacturing process in which each job is specified as a sequence of machines to be visited. Every sequence is defined uniquely for each job. We are assuming that the time spent by jobs on the machines is fixed and deterministic. The input sequence of loading jobs into the system is also given.

In this paper, we are interested in evaluating the performance of such job-shop systems under a cyclic manufacturing process. A performance evaluation method allows us to determine the limits of using a system, i.e. the throughput of the system or the waiting times of machines. Such operation sequences on machines should be found that would minimise the makespan, that is the total time required to complete all jobs. This problem has been studied by H.P. Hillion and J.M. Proth [10]. It has been shown that the solution to this problem is the solution to an integer linear programming problem derived from the event graph model Laftit S., Proth J.M., Xie X. L. [14].

Several efficient heuristic approaches have been proposed [7, 12, 16]. Recently, multiobjective heuristic algorithms have been applied to job-shop scheduling problems [20, 21]. Furthermore, immune algorithms for production job scheduling can be found in [18]. Usually, heuristic procedures [14] and simulation methods [11] are more dedicated to solving such problems. What is more, problems in the field of scheduling belong to the category of NP-complete problems [9].

Models of asynchronous systems such as timed Petri nets (TPN) [2, 19] can be directly used to study the performance of systems, in particular by using a special class of TPNs called timed event graphs (TEG). By using known results about the TEG [11], the model makes it possible to evaluate this performance measured as the production rate in a steady state [6]. It is possible to fully utilize some machines, which are then referred to as a bottleneck, during a steady state with a minimal number of jobs in process. This means that the maximum production rate is obtained when those machines are fully utilized. The minimum distribution of jobs in a process leads to the full utilization of the bottleneck machines [4, 10].

2. Petri Net modelling

2.1. Mathematical structure of Petri nets

We use Petri nets to model the dynamic behaviour of job-shop systems. They are a special type of graph which consists mainly of two types of elements: places and transitions. We can assume that a set of places represents system states and the set of transitions represents events which modify the state of the system.

Definition 1. A Petri net structure is specified as a four-tuple N=(P,T,F,W) where:

$P=\{p_1, p_2,..., p_r\}$ is a finite set of places;

$T=\{t_1, t_2, ..., t_n\}$ is a finite set of transitions;

$(P\cup T\neq 0, P\cap T=0)$ i.e. places and transitions are disjoint sets;

$F\subseteq(PxT)\cup(TxP)$ is a flow relation (set of directed arcs);

$W:F\rightarrow\{1,2,...\}$ is a weight function (assigns a weight to each arc).

A Petri net structure is weighted-bipartite, directed graph consisting of two kinds of nodes: places and transitions. Since this graph is bipartite, there are no arcs from place to place or from transition to transition. A net structure represents the static part of the modelling stystem. In order to determine the formal definition of Petri net an initial marking is introduced.

Definition 2. A Petri net is the couple PN=(N, M_0) where:

N=(P,T,F,W) is a Petri net structure;

M_0 is an initial marking function which assigns a non-negative number of so called tokens to each place the net, M_0: $P\rightarrow\{0,1,2,...\}$.

Let any function M: $P\rightarrow\{0,1,2,...\}$ be called a marking in a net N.

A Petri net PN is said to be ordinary if all of its arc weights are equal to one. The use of asynchronous models for performance evaluation requires the introduction of a time parameter.

Definition 3. A timed Petri net is specified as a triplet TPN=(PN, M_0, T), where $T:T\rightarrow R^+$ is a function assigning a non-negative rational number to each transition $t\in T$.

We assume that each transition $t\in T$ has a duration $\tau_t = T(t)$ measured in time units.

Definition 4. A timed event graph (TEG) is such a ordinary timed Petri net in which $|{\bullet p}|=|p{\bullet}|=1$ for all $p\in P$, where:

$\bullet p=\{t:(t,p)\in F\}$ - the set of input transitions of p;

$p\bullet=\{t:(p,t)\in F\}$ - the set of output transitions of p.

In a TEG ever arc has the weight equal to one, each place has exactly one upstream and downstream transition. Situations such as confluent places, 'or' places and multiplying tokens are forbidden. A token in a TEG may be in one of the following two states: available or

unavailable. In the literature event graphs are sometimes also referred to as marked graphs or as decision free Petri nets [1, 4].

2.2. Timed event graph modelling

In order to simulate of cyclic manufacturing processes a marking in a TEG is changed according to the transition enabling and firing rule. The firing time of a transition is the time that elapses between the starting and the completion of the firing of the transition. It is assumed that a firing is initiated as soon as the transition is enabled. The firing of transition $t \in T$ at time τ_t induces two actions on the marking at the time when tokens are removed from the input place of t and added to the output place of t. The tokens are consumed by a transition remain in the preceding places during the firing time. Durations associated with firing times will be used to represent operations times in manufacturing processes, where transitions represent machines.

The holding time of a place is the time a token must spend in the place before contributing to the enabling of the downstream transitions. The holding times can be the minimal time tokens have to spend in places, while firing times represent the actual time it takes to fire a transition.

The marking M_n is said to be reachable from the marking M_0 if a sequence of firings $\partial = t_1 t_2 ... t_n$ that transforms M_0 to M_n exists. The set of all possible markings reachable from M_0 is denoted by $R(M_0)$. For a connected TEG, with initial marking M_0, a firing sequence can lead back to M_0 if it fires every transition an equal number of times [1].

Let us consider a job-shop system. A job-shop system consists of different types of jobs $J_1, J_2, ..., J_n$ and a certain set of machines $m_1, m_2, ..., m_r$. The manufacturing process of each job is supposed to be uniquely defined as a manufacturing routing through the system (i.e., a sequence of machines to visit) with a given time spent on each machine. The routing corresponding to job type J_v is determined by:

$$\langle J_v \rangle = \langle m_1(v), m_2(v), ..., m_r(v) \rangle, \quad v = 1,2,...,n$$

while, the fixed seqencing of the jobs on machine m_u is represented by

$$\langle m_u \rangle = \langle J_1(u), J_2(u), ..., J_s(u) \rangle, \quad u = 1,2,...,r.$$

The jobs are running through various manufacturing processes. Not all jobs go through the same path of manufacturing. The same job type may appear several times in the input sequence of jobs into the system.

Let us now consider the TEG model of the job-shop cyclic manufacturing. The way to model was extensively explaned in [4, 10].

In TEG model each transition firing corresponds to executing the operation determined by the order specified in the manufacturing routing. The place corresponds to a storage buffer

or resource. The presence of a token in a place is interpreted as the condition associated with the place being true. In the other interpretation, a token is put in a place to indicate that the resource is available.

The example shown in Fig.1 illustrates the model of a manufacturing routing of the $(s+1)^{th}$ operation job $<J_k>=<m_0(k),..,m_{i-1}(k), m_i(k), m_{i+1}(k),.., m_s(k)>$, where transition t_i^k corresponds to the execution of operation $m_i(k)$ and p^b_i represents storage buffer. Places are represented by circles and transitions by bold bars. A token on a line represents the job as it flows through the system.

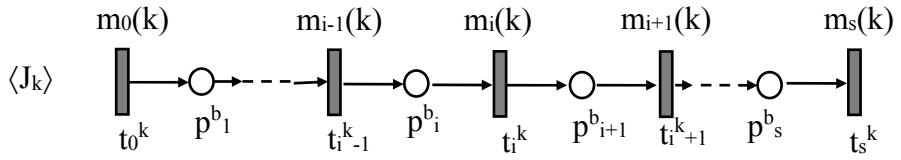

Fig. 1. The model of routing of J_k job as a sequence of transition
Rys. 1. Model marszruty zadania J_k jako sekwencja tranzycji

We are assuming that the job recirculates as soon as it has been completed. The closed loop model is shown in Fig. 2. The repetitive operation of the manufacturing process is modelled by a loop closing in the place p^r_k representing resources. A token in such a place can model a free transportation resources for starting a new job of the same type. The total number of tokens in a cycle remains constant, and shows how many jobs can be performed in parallel in the same cycle.

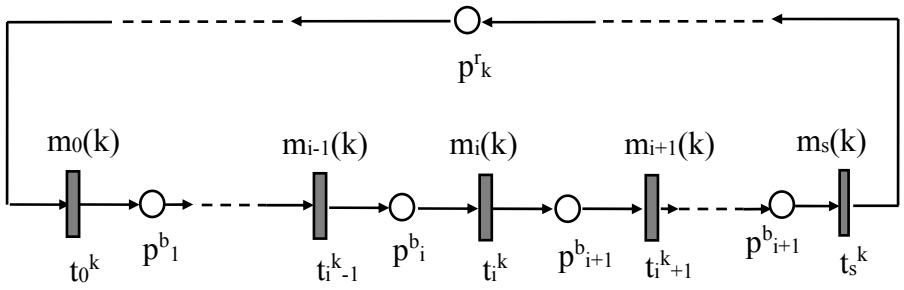

Fig. 2. The model of cyclic manufacturing process related to J_k job
Rys. 2. Model cyklicznego procesu produkcyjnego dla zadania J_k

In the papers [10] authors introduced three types of elementary circuits: process circuits, command circuits and mixed circuits.

The circuit $(t_0^k, p^b_1, \ldots, t_i^k, p^b_{i+1}, \ldots, t_s^k, p^r_k, t_0^k)$ of Fig. 2 is referred to as a process circuit, which model the cyclic manufacturing process. The command circuits model the control of the system. One command circuit is associated to each machine. For instance $(t^{k}_{i}{}^{-1}, p^1_i, t^{k}_{i}{}^{+1}, p^2_i, t^k_i, p^3_i, t^{k}_{i}{}^{-1})$ is a command circuit corresponding to m_i machine, which is given in Fig. 3.

In each command circuit there is one token, which prevents transitions corresponding to the same machine to be fired simultaneously

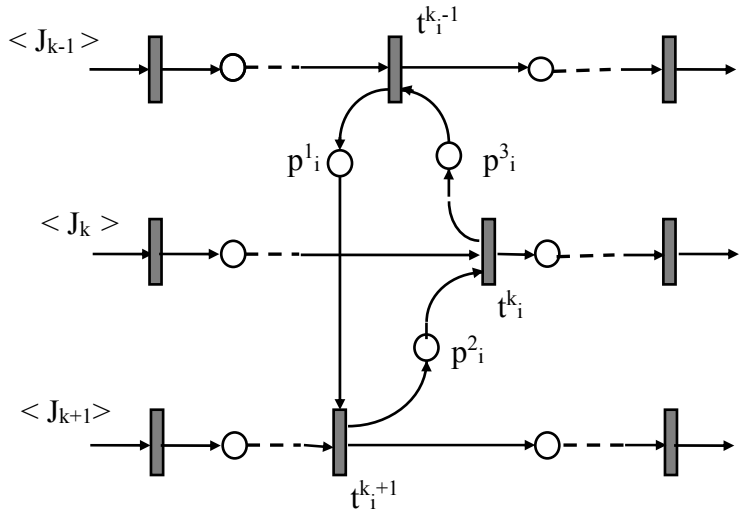

Fig. 3. The model of the sequencing of jobs on m_i machine
Rys. 3. Model sekwencji zadań wykonywanych na maszynie m_i

According to Hillion and Proth [10] the model related to cyclic manufacturing processes is given in Fig. 4.

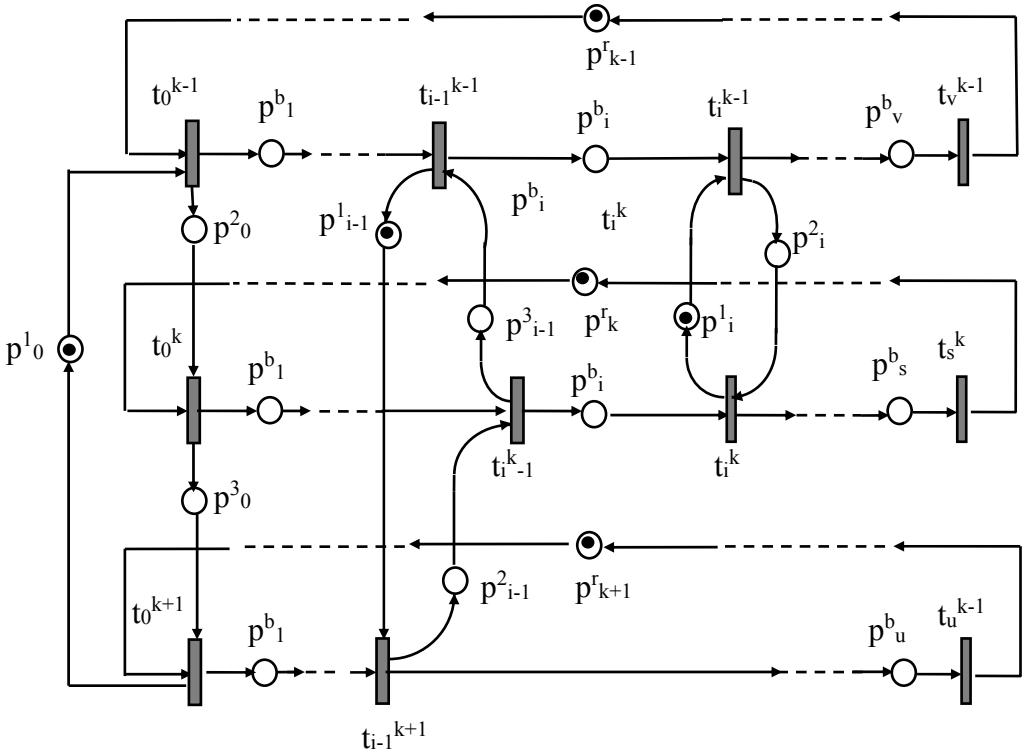

Fig. 4. TEG model of cyclic manufacturing processes
Rys. 4. Model TEG cyklicznych procesów produkcyjnych

The mixed circuits is partially composed of parts of the command circuits, and partially of parts of the process circuits. For instance (t_{i-1}^{k+1}, p^2_{i-1}, t_i^k-1, p^b_i, t_i^k, p^b_s, t_s^k , p^r_k, t_0^k, p^3_0, t_0^{k+1}, p^b_1, t_{i-1}^{k+1}) is a mixed circuit.

The loop-closing places p^r_k k=1,2, … , in Fig. 5 will denote by r_k, k=1,2, … , and, likewise p^b_k, will denote by b_k, k=1,2,… , olso the places in command circuits will denote by c_k, k=1,2, …, .

3. Cycle time in a performance evaluation

We are interested in how long it will take each transition to regularly initiate firing in a TEG. In terms of Petri nets, this time, called cycle time, is determined by the time necessary to complete a firing sequence and reload back to the start after firing each transition at least once. We have studied the properties of schedules associated with periodic sequences. To obtain the maximum performence within the system, the transition must fire as soon as it is enabled. The times at which those transitons fire are the earliest possible times [3].

In order to compute the cycle time, we need to determine specific circuits in the TEG, called elementary circuits. An elementary circuit is a directed path that goes from one node i.e., a place or a transition, back to this node in such a manner that no other nodes are repeated. Let Γ be the set of elementary circuits of a TEG. For each $\gamma \in \Gamma$, the cycle time of γ is defined by [10]:

$$C(\gamma) = (\tau(\gamma)/M(\gamma)), \tag{1}$$

where:

$\tau(\gamma)=\Sigma_{t\in\gamma}\,\tau_t$ - is the sum of the transition firing times in circuit γ.

$M(\gamma)=\Sigma_{p\in P}\,M(p)$ - is the number of tokens circulating in circuit γ at marking M.

We will refer to the elementary circuits for which the cycle time is the greatest. The maximum cycle time taken over all elementary circuits is given

$$C(\gamma^*) = \max_{\gamma\in\Gamma} C(\gamma). \tag{2}$$

Then γ^* is a circuit called critical. The known $C(\gamma^*)$ allows us to evaluate a performance measure, namely the production rate specified by

$$\lambda = \min_{\gamma\in\Gamma} (M(\gamma)/\tau(\gamma)) = 1/C(\gamma^*) \tag{3}$$

The critical circuit γ^* for which $C(\gamma^*)$ is cycle time bounds the throughput of the system. The machines determining the value of $C(\gamma^*)$ are referred to as bottleneck machines. In order to discuss the maximum performance, we must consider the minimum distribution of jobs in the

process which leads to the full utilization of bottleneck machines. In such a situation, the system works at its maximum rate and its productivity is the greatest.

The maximum performance is achieved when transitions fire as soon as they are enabled. The times at which transitions fire are the earliest possible times [3]. In a TEG, for given cycle time *C,* the earliest instants of initiating the firing of transition $t_i \in T$ are, by definition, determined as follows:

$$s_i(k) = s_i(1) + (k-1)C \tag{4}$$

where:

$s_i(k)$ - is the instant of the k^{th} firing initiation of transition t_i.

Let us consider the fragment of the TEG in which transition t_i constitutes the input transition and t_j - the output transition of place $p \in P$.

Hence, for cycle time *C,* the instant of the k^{th} firing completion of transition $t_i \in T$ must be less then the instant of the $(k+M_0(p))^{th}$ firing initiation of transition $t_j \in T$ and the following relations must be true:

$$s_i(k) + \tau(t_i) \le s_j(k+M_0(p) \tag{5}$$

$$s_i(1) + (k-1)C + \tau(t_i) \le s_j + (k-1+M_0(p))C\,, \quad \forall p \in P, \tag{6}$$

$$\tau(t_i) \le s_j(1) - s_i(1) + M_0(p)C. \tag{7}$$

Obtaining the minimum cycle time can be expressed as a linear programming problem, and a polynomial algorithm exists to solve it [14]. When the circuits and the transition firing times are known, the production rate depends on the initial marking M_0 in the net.

4. Problem formulation

We are considering how to schedule multiple-operation jobs on different types of machines, where the completion of a job may require a few operations to be performed in a particular sequence. Each job is completed in a multi-machining operation and is associated with a period of processing time.

The objective of the schedule is to minimize the cycle time *C* and the number of jobs that are actually required in the process. This is important because the number of pallets or carts circulating in the system should be as low as possible.

Let us now consider the elementary circuit $\gamma \in \Gamma$ of a TEG. The total transition firing times belonging to circuit γ is given by $\tau(\gamma) = \Sigma_{t \in \gamma} \tau\,(t)$. According to [14], it is assumed that every place holds no more than two tokens. We can always extend the TEG so that the optimal marking exists in which evere place contains no more than one token. For this purpose, for each place of the model, one additional place and additional transition whose firing time is equal to zero are introduced.

From now on, we will represent

$$M(\gamma) = \sum_{p\in\gamma} M(p), \tag{8}$$

In particular, the inequality (7) for all places belonging to elementary circuit $\gamma\in\Gamma$ can be expressed as:

$$\tau(\gamma)/M(\gamma) = C(\gamma) \le C. \tag{9}$$

For any elementary circuit $\gamma\in\Gamma$ and the minimum number of tokens $M_{min}(\gamma)$ such that $C(\gamma)\le C$, the following relation is true

$$M_0(\gamma) \ge M_{min}(\gamma), \tag{10}$$

where $M_0(\gamma)$ is the initial marking determining the number of tokens in circuit γ at a steady state. Let us now turn to the problem of minimizing the number of jobs in the process. The objective function should be to minimize the total number of tokens that are not changed by any transition firing and are uniquely determined by the initial marking M_0. Now we represents the criterion to minimize as follows [14]:

$$\Phi(M_0^*) = \min_{Mo} \sum_{p\in P} M_0(p), \tag{11}$$

subject to

$$\tau(\gamma)/M_0(\gamma) \le C, \quad \forall\gamma\in\Gamma, \tag{12}$$

$$\tau((\bullet p) \le s_{p\bullet}(1) - s_{\bullet p}(1) + M_0(p)C, \qquad \forall p\in P. \tag{13}$$

The problem (11) - (13) is specified as a linear programming problem.

5. Solution methodology

5.1. Construction of a feasible schedule

This section presents numerical results of applying a heuristic algorithm based on the idea of Laftit S. at el. [14] to obtain a solution to problem (11) - (13).

We can evaluate the value of C from inequalities (8), (10). Let β be a positive real value. The value of β may represent the longest cycle time of bottleneck machines.

The marking M_0 is a feasible marking for β if cycle time C fulfils inequality (13) and $C\ge\beta$. The objective of function $\Phi(M_0^*)$ is to minimize the total number of tokens with the set of instances of first firing initiations of transitions.

The main idea behind obtaining a feasible solution of problem (11)-(13) is based on the criterion associated with relation (12). This can be rewritten as:

$$\nabla_M^{\beta}(\gamma) = M(\gamma) - \tau(\gamma)/\beta. \tag{14}$$

We consider the value of $\nabla_M^{\beta}(\gamma)$ to be related to any $p \in P$. Let $\gamma \in \Gamma_p$ be an elementary circuit containing the fixed place p. We introduce a criterion in which the set of places is taken into account

$$\nabla_M^{\beta}(p) = \min_{\gamma \in \Gamma p.} \nabla_M^{\beta}(\gamma) , \quad \forall p \in P. \tag{15}$$

For any β, marking M belongs to a set of feasible markings if $\nabla_M^{\beta}(p) \geq 0$, $\forall p \in P$. For $p \in P$ with $\nabla_M^{\beta}(p) \geq 1$, we can obtain a new marking belonging to the feasible marking by removing one token from p. Let us represent this marking by M' so that M'(p')=M(p')-1, M'(p)=M(p), $\forall p \in P$, $p' \neq p$. Thus

$$0 \leq \nabla_{M'}^{\beta}(p) < \nabla_M^{\beta}(p) , \quad \forall p \in P. \tag{16}$$

The removal of a token from $p \in P$ modifies the value of $\nabla_M^{\beta}(p')$ which is associated with places belonging to circuits containing p. This minimisation criterion can be expressed as:

$$\nabla^*(p) = \sum_{p' \in P} (\nabla_M^{\beta}(p') - \nabla_{M'}^{\beta}(p')) . \tag{17}$$

The choice of the place from which the token will be removed is determined by the minimal value of $\nabla^*(p)$ i.e., $p=p^*$ such that

$$\nabla^*(p^*) = \min_{p \in G} \nabla^*(p) , \tag{18}$$

where G is the set of places for which $\nabla_M^{\beta}(p) \geq 1$.

5.2. The heuristic algorithm

We start by choosing the places to be marked so that each place will contain only one token. At each iteration, the number of tokens will change in the selected places for which $\nabla_M^{\beta}(p) \geq 1$, $p \in P$, and if $M(p) \geq 1$, we can remove one token from p. In this case, the marking remains feasible under the constraint (16). Unfortunately, if M(p)=0, then we must compute marking M' which is reachable from marking M by a transition firing so that M'(p)=1. The removal of a token from some place may modify $\nabla_M^{\beta}(p')$, where the value for the other place p' belongs to elementary circuits containing p. In this case, the marking is also feasible. The algorithm is as follows:

Step 1. Determine a starting marking M_0, and evaluate β.

Step 2. Compute the value of criterion $\nabla_M^{\beta}(p)$, $p \in P$. If $\nabla_M^{\beta}(p) < 1$ for any $p \in P$, stop the computation.

Step 3. Compute the value of criterion $\nabla^*(p)$, for $p \in G \subset P$, so that $\nabla_M^{\beta}(p) \geq 1$.

Step 4. Choose place p^* from subset G, so that $\nabla^*(p^*) = \min \nabla^*(p)$, $p \in P$.

Step 5. If $M(p^*) \geq 1$, then remove one token from p^*, otherwise compute marking M' reachable from M so that $M'(p^*) \geq 1$, remove one

token from p^* and rename marking from M' to M.

Step 6. Go back to step 2.

6. Applications

Here we will present an application of the previous algorithm to determine a heuristic schedule of jobs on machines. We find an initial marking associated with the distribution of jobs. The algorithm was coded in C++ and a numerical result applying the algorithm is presented.

6.1. An example

In this example, four job types with fifteen operations are to be scheduled on four non-identical machines. A number of these jobs comprise three or four operation and each operation may be scheduled on up to four different machines. A TEG model structure of the cyclic processes is given in Fig. 5. The places correspond to: storage buffers (b), resources (r) and command places (c). The transitions correspond to the execution of an operation. The firing times of transitions represent the operation processing time. The range of their values is placed in square brackets. The firing time of transitions are given in Table 1.

Table 1

Transition firing times

Machine	Jobs			
m_i	J_1	J_2	J_3	J_4
m_1	1	3	2	2
m_2	4	2	-	1
m_3	3	1	1	3
m_4	3	1	1	2

The routing jobs through the system and the processing times for each machine (in parenthesis):

$<J_1>$: $< m_1(1),\ m_2(4),\ m_3(3),\ m_4(3)$
$<J_2>$: $< m_4(1),\ m_2(2),\ m_1(3),\ m_3(1)$
$<J_3>$: $< m_1(2),\ m_3(1),\ m_4(1)$
$<J_4>$: $< m_3(3),\ m_1(2),\ m_2(1),\ m_4(2)$

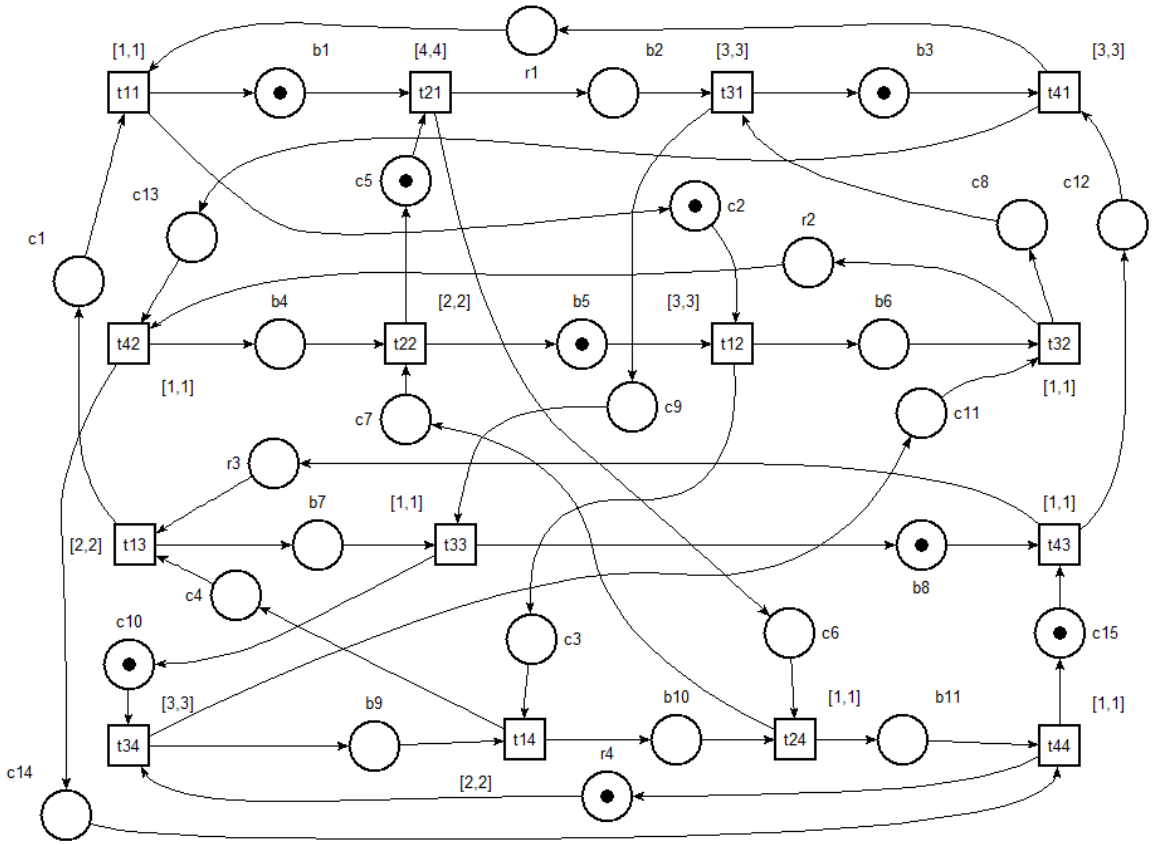

Fig. 5. Example of TEG model of cyclic processes
Rys. 5. Przykład modelu TEG cyklicznych procesów

Table 2

Solved initial marking M_0*

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}	b_{11}	r_1	r_2	r_3	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_{10}	c_{11}	c_{12}	c_{13}	c_{14}	c_{15}
1	0	1	0	1	0	1	0	1	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	0	0	0	0	1

Assuming that we start from an initial state where each place has one token. The machine m_1 and m_3 is a bottleneck machine with cycle time equal 8 units. The total number of tokens for marking M_0* is 10.

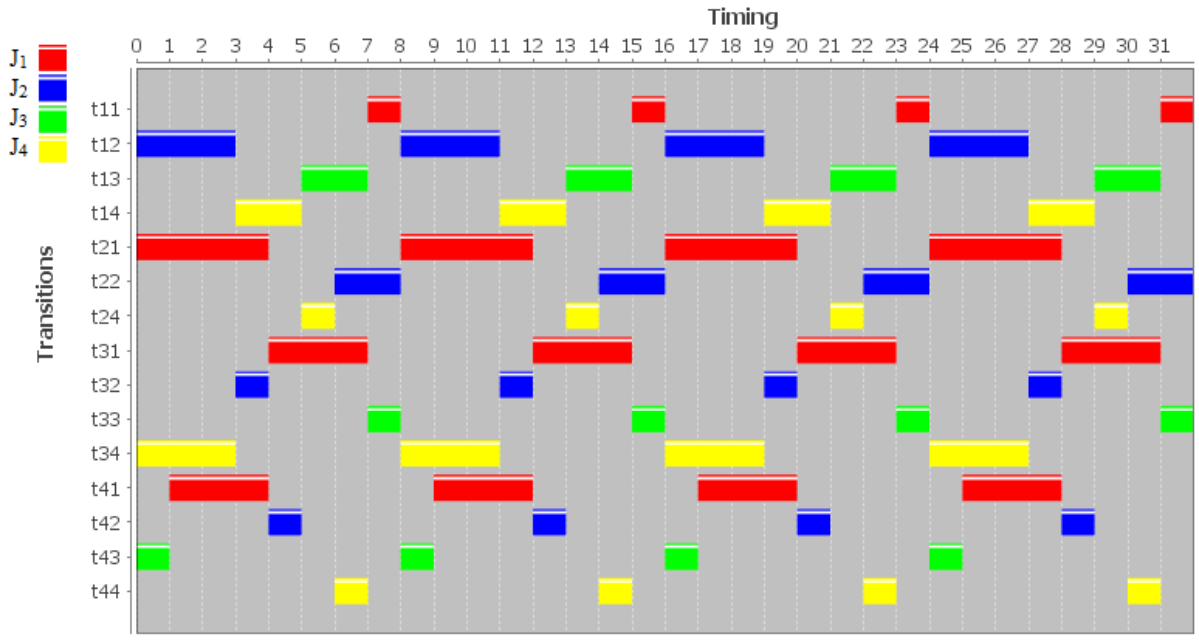

Fig. 6. Timed diagram of firing of transitions
Rys. 6. Diagram czasowy realizacji przejść

7. Concluding remarks

In this paper, the manufacturing scheduling problem has been formulated and numerically solved. We have considered a type of job-shop system, assuming a deterministic and repetitive operation of cyclic manufacturing process. We used the timed even graph to study the system.

The objective in scheduling the machines was to find the maximum performance of the system while keeping the number of jobs in the process as low as possible. The production rate of this system can be determined from the initial state with enough jobs in the process so that the bottleneck machines are fully utilized. This minimum number of jobs in the process allows the system to operate optimally.

The problem of the minimum distribution of jobs was specified as a linear programming problem and solved by a heuristic algorithm. We provided a numerical solution which allows the performance of the job-shop system to be evaluated. The tools of the Petri net are well suited to modelling cyclic manufacturing processes. Their advantages include their graphic nature and the ability to using them in simulation models.

BIBLIOGRAPHY

1. Baccelli F.L., et al.: Synchronization and Linearity. An Algebra for Discrete Event Systems. John Wiley & Sons, 1992.
2. Banaszak Z. (ed.): Modelling and control of FMS. Petri net approach. Wroclaw Technical University Press, 1991.
3. Carier J., Chretienne P.: Timed Petri net schedules. Advances in Petri nets, Rozenberg G. (ed.), LNCS, Springer Verlag, 1988.
4. DiCesare F. et al.: Practice of Petri Nets in Manufacturing. Chapman & Hall, 1993.
5. Dubois D., Stecke K. E.: Dynamic analysis of repetitive decision-free discrete event processes. The algebra of timed marked graphs and algorithmic issues. Annals of Operation Research. Volume on ”Production Planning and Scheduling”, 1990.
6. Gaubert S.: An algebraic method for optimizing resources in timed event graphs. Lecture Notes in Computer and Information Science, Springer Verlag, Vol. 144, 1990, p. 957÷966.
7. Goldin G., A.: Problem Solving Heuristics - Affect and Discrete Mathematic. European Mathematical Information Service, The Electronic Library of Mathematics – ZDM 2004, Vol. 36 (2), p. 56÷60.
8. Graves S. C.: A review of production scheduling. Operations Research, vol. 18, 1981, p. 841÷852.
9. Hartung S., Nichterlein A.: NP-Hardness and Fixed-Parameter Tractability of Realizing Degree Sequences with Directed Acyclic Graphs. Lecture Notes in Computer Science Volume 7318, 2012, p. 283÷292.
10. Hillion H. P., Proth J. M.: Performance evaluation of job-shop systems using timed event graphs. IEEE Transactions on Automatic Control, Vol. 34, No.1, 1989, p. 1÷9.
11. Jamroż L., Raszka J.: Simulation method for the performance evaluation of system of discrete cyclic processes. Proceedings of the 16th IASTED International Conference on Modelling, Identification and Control, Innsbruck, Austria Feb. 17–19th 1997, 190÷193.
12. Kolahan F., Kayvanfar V.: A Heuristic Algorithm Approach for Scheduling of Multi-criteria Unrelated Parallel Machine. World, Academy of Science, Engineering and Technology 59, 2009, p. 102÷105.
13. Kusiak A. (ed.): Modelling and design of flexible manufacturing systems. Elsevier Science Publishers, Amsterdam 1986.
14. Laftit S., Proth J.M., Xie X. L.: Optimization of invariant criteria for event graphs. IEEE Trasnsactions on Automatic Control, Vol. 37, No. 5, 1992, p. 547÷555.

15. Magott J.: Performance evaluation of computer systems using Petri nets. Scientific Papers of the Institute of Technical Cybernetics of Wroclaw Technical University, No. 80, s. Monographs 15, 1989, (in Polish).
16. Melnikov B.: Discrete Optimization Problems – Some New Heuristic Approaches. Eighth International Conference on High-Performance Computing in Asia-Pacific Region (HPC Asia 2005), 30 November-3 December 2005 Beijing, China.
17. Murata T.: Petri nets: proprties, analysis and applications. Proceedings of the IEEE, Vol. 77, No. 4, 1989, p.541÷580.
18. Skołud B., Wosik I.: Algorytmy ewolucyjne w szeregowaniu zadań produkcyjnych Zarządzanie Przedsiębiorstwem nr 1, 2008.
19. Szpyrka M.: Sieci Petriego w modelowaniu i analizie systemów współbieżnych. WNT, Warszawa 2008.
20. Tang L., Wang H., Li G., Xu F.: Adaptive heuristic search algorithm for discrete variables based multi-objective optimization. Springer-Verlag, Berlin Heidelberg 2013 October 2013, Volume 48, Issue 4, p. 821÷836.
21. Varthanana P. A., Muruganb N., Mohan Kumarc G.: A simulation based heuristic discrete particle swarm algorithm for generating integrated production–distribution plan. Applied Soft Computing Journal, Vol. 12, No. 9, p. 3034÷3050, Sept. 2012.

Omówienie

W artykule przedstawiono propozycję algorytmu heurystycznego do optymalizacji procesu produkcyjnego. Problem optymalizacyjny został sprowadzony do postaci problemu programowania liniowego [10, 14]. Rozważany jest system typu job-shop, w którym wielooperacyjne zadania wykonywane są na równolegle pracujących maszynach. Opisywany system produkcyjny jest przykładem systemu cyklicznych procesów [2, 13, 11].

Do modelowania systemu wykorzystano czasowe grafy znakowane jako narzędzie wspomagające jego dynamiczną analizę [1, 3, 5]. Badania koncentrowały się na najniższym poziomie operacyjnym systemu. Za miarę wydajności działania systemu wybrano czas cyklu odtwarzania początkowego znakowania w sieciowym modelu. Minimalny czas cyklu zapewnia jak najszybsze wykonywanie zadań, najkrótszy czas przebywania zadań w systemie oraz najlepsze zrównoważenie obciążeń maszyn.

Realizacja współbieżności procesów w warunkach ograniczonych zasobów wymagała określenia mechanizmów synchronizacji, zapewniających bezblokadowy ich przebieg. Rozważany system był analizowany z jakościowego i ilościowego punktu widzenia. Analiza jakościowa pozwoliła wykryć zakleszczenia w przypadku korzystania ze współdzielonych

zasobów i wiąże się z implementacją zasady wzajemnego wykluczania. Jej ostatecznym celem jest dowodzenie poprawności modelowanego systemu. Z kolei analiza ilościowa dotyczy wydajności pracy systemu określonej czasem cyklu i związana jest z oceną skuteczności zastosowanego modelu. Proponowane podejście do modelowania i szeregowania zadań stanowi praktyczne narzędzie do sterowania cyklicznymi procesami w systemach produkcyjnych.

Addresses

Jerzy RASZKA: Warszawska 24, 31-155 Kraków (Poland) Cracow University of Technology, Institute of Computer Science, jraszka@pk.edu.pl.

Lech JAMROŻ: Warszawska 24, 31-155 Kraków (Poland) Cracow University of Technology, Institute of Computer Science, ljamroz@pk.edu.pl.