AN OPTIMAL NON-BLOCKING DISPATCHING IN FREE-CHOICE MANUFACTURING FLOWLINES BY USING MACHINE-JOB INCIDENCE MATRIX

Ivica Sindičić [(a)], Stjepan Bogdan [(b)], Tamara Petrović [(b)]

[(a)] ABB, Bani 72, 10000 Zagreb, Croatia
[(b)] LARICS - Laboratory for Robotics and Intelligent Control Systems, Department of Control and Computer Engineering, Faculty of Electrical Engineering and Computing, University of Zagreb, Zagreb, Croatia

[(a)]ivica.sindicic@gmail.com , [(b)]stjepan.bogdan@fer.hr

ABSTRACT

In this paper we extend a method for resource allocation in particular class of flexible manufacturing system, namely, free-choice multiple reentrant flowlines (FMRF), which is based on MJI matrix. The proposed method is a solution to the problem on how to allocate jobs to resources and how to allocate resources to jobs. A solution is in a form of repeatable resource sequence over the set of resources available for particular choice job. As addition proposed in this paper, the solution is enhanced by the procedure that provides an optimal utilization of resources based on operation price and balanced use of all resources. Although efficiency of the proposed methods have been demonstrated on examples involving manufacturing workcells, the method can be used for other discrete event systems as well, as long as the system under study belongs to free-choice multiple reentrant flowlines class.

Keywords: dispatching, manufacturing systems, optimal control

1. INTRODUCTION

The first step in the supervisory controller design is modeling of the system and investigation of its structural properties. There are many approaches to modeling and analysis of manufacturing systems, including automata [1], Petri nets [2, 10], alphabet-based approaches, perturbation methods [3], control theoretic techniques, expert systems design, and so on.

One way to model relations between tasks in an FMS is in form of Steward sequencing matrix [4], also referred to as design structure matrix (DSM). DSM is a square matrix containing a list of tasks in rows and columns. The order of tasks in rows or columns indicates the execution sequence. Although very useful in production planning, DSM lacks of information related to the resources required for execution of tasks. This aspect of an FMS is captured by the resource requirements matrix [5], also know as the machine-part incidence matrix (MPI). Each column of MPI represents one resource, while rows represent part types processed by the system. The most common usage of MPI is in the field of manufacturing cells design by implementation of various clustering methods [6]. In [17] we proposed construction of machine-job incidence matrix (MJI) which can be obtained from MPI and DSM matrices.

Efficient procedures for determination of simple circular waits (CWs) [7, 8] as well as other important structural properties (which are responsible for stability in the sense of absence of deadlock), such as critical siphons and critical subsystems [9, 11], based on MJI, are presented in [13]. It should be noted that MJI matrix can be straightforwardly transformed in matrix model described in [9].

Generally, scheduling requires a) allocation (dispatching) of available resources to predetermined operations (tasks), and b) definition of sequences in order to provide stable behavior of the system. Usually, supervisory controller not only stabilizes the system (in a sense of deadlock and bounded buffers) but in the same time optimizes some performance criteria.

Herein we extend a method for resource allocation in particular class of FMS, namely, free-choice multiple reentrant manufacturing systems (FMRF), which is based on MJI matrix, described in details in [17]. A solution is in a form of repeatable resource sequence over the set of resources available for particular choice job. As addition, proposed in this paper, the solution is enhanced by the procedure that provides an optimal usage of resources based on price and balanced use of all resources.

2. SYSTEM DESCRIPTION AND PROBLEM FORMULATION

We make the following assumptions that define the sort of discrete-part manufacturing systems: No pre-emption – once assigned, a resource cannot be removed from a job until it is completed, Mutual exclusion – a single resource can be used for only one job at a time, Hold while waiting – a process holds the resources already allocated to it until it has all resources required to perform a job. Furthermore, we assume that there are no machine failures. Multiple reentrant flowlines (MRF) class of systems, investigated herein, has the following properties: a) each part type has a strictly defined sequence of operations, b) each operation in the system requires one and only one resource with no two consecutive jobs using the same resource, c) there are no choice jobs, d) there are no assembly jobs, e) there are shared resources in the system.

2.1. System description

Let Π be the set of distinct types of parts produced (or customers served) by an FMS. Then each part type $P_k \in \Pi$ is characterized by a predetermined sequence of job operations $J^k = \left\{J_1^k, J_2^k, J_3^k, ..., J_{L_k}^k\right\}$ with each operation employing at least one resource. (Note that some of these job operations may be similar, *e.g.* J_i^k and J_j^k with $i \neq j$ may both be drilling operations.) We uniquely associate with each job sequence J^k the operations of raw part-in, J_in^k, and completed product-out, J_out^k.

Denote the system resources with $R = \{r_i\}_{i=1}^n$, where $r_i \in R$ can represent a pool of multiple resources each capable of performing the same type of job operation. In this notation $R^k \subset R$ represents the set of resources utilized by job sequence J^k. Note that $R = \bigcup_{k\in\Pi} R^k$ and $J = \bigcup_{k\in\Pi} J^k$ represent all resources and jobs in a particular FMS. Since the system could be re-entrant, a given resource $r \in R^k$ may be utilized for more than one operation $J_i^k \in J^k$ (*sequential sharing*). Also, certain resources may be used in the processing of more than one part-type so that for some $\{l, k\} \in \Pi$, $l \neq k$, $R^l \cap R^k \neq \varnothing$ (*parallel sharing*). Resources that are utilized by more than one operation in either of these two ways are called *shared resources*, while the remaining are called *non-shared resources*. Thus, one can partition the set of system resources as $R = R_\mathrm{s} \cup R_\mathrm{ns}$, with R_s and R_ns indicating the sets of shared and non-shared resources, respectively. For any $r \in R$ we define the *resource job set* $J(r)$. Obviously, $|J(r)| = 1$ (> 1) if $r \in R_\mathrm{ns}$ ($r \in R_\mathrm{s}$). Resource r, with its job set $J(r)$, comprise *resource loop* $L(r)$, $L(r) = r \cup J(r)$.

We define a *job vector* $\mathbf{v} : J \rightarrow \aleph$ and a *resource vector* $\mathbf{r} : R \rightarrow \aleph$ that represent the set of jobs and the set of resources corresponding to their nonzero elements. The set of jobs (resources) represented by $\mathbf{v}$ ($\mathbf{r}$) is called the *support* of $\mathbf{v}$ ($\mathbf{r}$), denoted *sup*($\mathbf{v}$) (*sup*($\mathbf{r}$)); *i.e.* given $\mathbf{v} = [v_1\ v_2\ \ldots\ v_\mathrm{q}]^\mathrm{T}$, vector element $v_i > 0$ if and only if job $v_i \in$ *sup*($\mathbf{v}$). In the same manner, given $\mathbf{r} = [r_1\ r_2\ \ldots\ r_\mathrm{p}]^\mathrm{T}$, vector element $r_i > 0$ if and only if resource $r_i \in$ *sup*($\mathbf{r}$). Usually, index i is replaced with job (resource) notation, for example, r_MA stands for the component of resource vector r that corresponds to resource MA. The definitions of job and resource vectors imply that the job and resource sets should be ordered.

MRF class is a special case of FMRF - systems with jobs that do not have predetermined resources assigned. That is, several resources might be capable and available to perform a specific job (MRF property *c* is not valid, *i.e.* there *are* jobs with choice). We define $R(J_i^k)$ as a set of resources that could be allocated to choice job $J_i^k \in J^k$.

An example of FMRF workcell is given in Figure 1. with J = {RP1, BP, MP, RP2} and R = {M1, M2, B, R}. From buffer (job BP), part proceeds to machine M1 *or* machine M2 (choice job MP). Hence, vector representation of resources that could be allocated to choice job MP is $\mathbf{r}_\mathrm{MP} = [1\ \ 1\ \ 0\ \ 0\ ]^\mathrm{T}$ and R(MP)=*sup*($\mathbf{r}_\mathrm{MP}$)={M1, M2}.

2.2. Problem formulation

Since the system contains shared resources and choice jobs, the scheduling problem discussed in the paper is twofold: i) for a given choice job $J_i^k \in J^k$ define *allocation sequence* of resources in $R(J_i^k)$, and ii) for a given shared resource $r_\mathrm{s} \in R_\mathrm{s}$ with resource job set $J(r_\mathrm{s})$, define *dispatching policy*. Both solutions, allocation sequence and dispatching policy, should be such that overall system is stable in a sense of deadlock.

A solution of problem i) offers an answer on how to *allocate resources to jobs*. For that purpose we propose a result in a form of *repeatable resource sequence* over the set of resources available for particular choice job.

On the other hand problem ii) is related to the number of active jobs in a particular parts of FMRF systems, called *critical subsystems*. In the chapters that follow we show why critical subsystems are important, how they can be determined from MJI matrix, and how their content (number of active jobs) can be controlled. In fact, solution to problem ii) describes how to *allocate jobs to resources*.

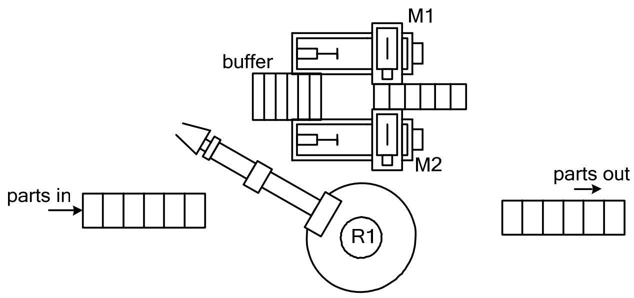

Figure 1: An example of FMRF class of FMS

3. MACHINE-JOB INCIDENCE MATRIX (MJI)

As we stated in Introduction, DSM is a square matrix containing a list of tasks in rows and columns with matrix elements indicating an execution sequence. The second matrix used for the system description is MPI. It captures relations between resources and parts processed by the system. Since the sequence $J^k = \left\{J_1^k, J_2^k, J_3^k, ..., J_{L_k}^k\right\}$ represents part P_k processing order, by combining DSM and MPI matrices, we get a general form of machine-job incidence matrix $\mathbf{\Lambda}$ for an FMRF system [17]. In case job i is performed by resource j, matrix element (i, j) is equal to '1', otherwise is '0'. For an MRF system, each operation in the system requires only one resource (there are no choice jobs), hence, exactly one element '1' would appear in each row of MJI matrix. On the other hand, column representing shared resource comprises multiple entries of '1'.

$$\mathbf{\Lambda} = \begin{array}{c|cccc|} & R_1 & R_2 & \cdots & R_q \\ J_1^1 & \mathbf{0/1} & \mathbf{0/1} & \mathbf{...} & \mathbf{0/1} \\ J_2^1 & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \\ \vdots & \vdots & \vdots & & \vdots \\ J_{L^1}^1 & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \\ J_1^2 & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \\ \vdots & \vdots & \vdots & & \vdots \\ J_{L^2}^2 & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \\ \vdots & \vdots & \vdots & & \vdots \\ J_1^m & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \\ \vdots & \vdots & \vdots & & \vdots \\ J_{L^m}^m & \mathbf{0/1} & \mathbf{0/1} & \cdots & \mathbf{0/1} \end{array}$$

Machine-job incidence matrix can be defined separately for each part type in an FMS.In that case overall MJI matrix can be written as:

$$\mathbf{\Lambda} = \begin{bmatrix} {}^1\mathbf{\Lambda}^\mathrm{T} & {}^2\mathbf{\Lambda}^\mathrm{T} & ... & {}^m\mathbf{\Lambda}^\mathrm{T} \end{bmatrix}^\mathrm{T} \qquad \textbf{(1)}$$

For the system given in Figure 2. MJI attains the following form:

$$\mathbf{\Lambda} = \begin{array}{c|cccc|} & \mathrm{M1} & \mathrm{M2} & \mathrm{B} & \mathrm{R} \\ \mathrm{RP1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \\ \mathrm{BP} & \mathbf{0} & \mathbf{0} & \mathbf{1} & \mathbf{0} \\ \mathrm{MP} & \mathbf{1} & \mathbf{1} & \mathbf{0} & \mathbf{0} \\ \mathrm{RP2} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{1} \end{array}$$

It can be seen that robot R is shared resource as the corresponding column has two elements equal to '1'.

As we demonstrate in [17] one of the benefits provided by MJI matrix is reduction of computational complexity in FMRF system analysis and simulation.

4. MJI AND RESOURCE SEQUENCING

As already mentioned, in addition to the assumptions made at the beginning of Chapter II, a general class of FMRF systems has the following nonrestrictive capabilities: i) some jobs have the option of being machined in a resource from a set of resources (allocation of jobs), ii) job/part routings are NOT deterministic, iii) for each job there exists a material handling buffer (routing resource) that routes parts.

In this Chapter we are interested to determine repeatable resource sequence over the set of resources available for execution of each choice job in the system. The sequence should prevent conflicts and deadlocks simultaneously. In the analysis that follows we consider one part-type regular FMRF with each buffer capable of holding one part at a time.

Let MJI matrix of the system is given as

$$\mathbf{\Lambda} = \begin{array}{c}\left[\begin{array}{ccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \\ \begin{array}{ccccccccc} M1 & M2 & M3 & M4 & M5 & B1 & B2 & B3 & B4 \end{array}\end{array} \begin{array}{c} J1 \\ JB1 \\ J2 \\ JB2 \\ J3 \\ JB3 \\ J4 \\ JB4 \\ J5 \\ \end{array} \qquad \textbf{(2)}$$

From the matrix we see that a part production requires sequence of 5 jobs, and system is composed of 5 machines and 4 buffers. For choice jobs J2, J3 and J4, we have R(J2)={M2, M3}, R(J3)={M3, M4, M5}, and R(J4)={M2, M4}. All machines, except M5, are shared resources. There are many part routes that complete the required job sequence - to mention just few of them: σ_1={M1→M2→M3→M2→M1}, σ_2={M1→M3→M5→M4→M1}, σ_3={M1→M3→M3→M4→M1}, and so on. We partition set of part routes, denoted $\boldsymbol{\Sigma}$, into two disjoint sets, $\boldsymbol{\Sigma} = \boldsymbol{\Sigma}_\mathrm{R} \cup \boldsymbol{\Sigma}_\mathrm{NR}$, with $\boldsymbol{\Sigma}_\mathrm{R}$ comprising all part routs with multiple use of resources allocated to choice jobs, and $\boldsymbol{\Sigma}_\mathrm{NR}$ containing all part routes without multiple use of resources. In our example σ_1 and σ_3 belong to $\boldsymbol{\Sigma}_\mathrm{R}$, while σ_2 is an element of $\boldsymbol{\Sigma}_\mathrm{NR}$.

It is obvious that conflicts might occur in $\sigma_i \in \boldsymbol{\Sigma}_\mathrm{R}$ since same resource is used for execution of more than one job. In σ_1 resource M2 is used for jobs J2 and J4. On the other hand part routes in $\boldsymbol{\Sigma}_\mathrm{NR}$ are inherently conflict free. We define resource sequences as combination of several routes from $\boldsymbol{\Sigma}_\mathrm{NR}$. As a result, the structure of repeatable resource sequences over the set of resources available for execution of choice jobs would, by itself, provide not only conflict free but also deadlock free behavior of the system.

The number of all possible part routs, $N = |\,\boldsymbol{\Sigma}|$, is defined as $N = \prod_{i=1}^{n}(\sum_{j=1}^{m}\mathbf{\Lambda}_{i,j})$ where n and m correspond to the number of rows and columns of $\mathbf{\Lambda}$, respectively. In our example one has N = 1·1·2·1·3·1·2·1·1=12. For each part route σ_i we define *MJI sub-matrix* $\mathbf{\Lambda}^i$ in a way that in case of multiple entries '1' in row (choice job), sub-matrix comprises only the one that corresponds with resource belonging to the part route. For σ_2={M1→M3→M5→M4→M1} MJI sub-matrix attains a form

$$\mathbf{\Lambda}^2 = \begin{array}{c} \left[\begin{array}{ccccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array}\right] \begin{array}{l} J1 \\ JB1 \\ J2 \\ JB2 \\ J3 \\ JB3 \\ J4 \\ JB4 \\ J5 \end{array} \\ \begin{array}{ccccccccc} M1 & M2 & M3 & M4 & M5 & B1 & B2 & B3 & B4 \end{array} \end{array}.$$

Having defined sub-matrices, search for sequences in $\boldsymbol{\Sigma}$ is in fact search for all MJI sub-matrices that are characterized by single entry '1' in each row and multiple or no entry '1' in each column. It is easy to show that there exists a procedure of complexity O(N) for calculation of such matrices.

Resource sequences are related to choice jobs, for this reason, we introduce *reduced form* of MJI sub-matrices, denoted $\mathbf{\Lambda}^{*i}$, such that encompass only rows corresponding to those jobs, and without columns related to the buffers. Reduced form of $\mathbf{\Lambda}^2$ from previous example is given as

$$\mathbf{\Lambda}^{*2} = \begin{array}{c} \left[\begin{array}{ccccc} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{array}\right] \begin{array}{l} J2 \\ J3 \\ J4 \end{array} \\ \begin{array}{ccccc} M1 & M2 & M3 & M4 & M5 \end{array} \end{array}.$$

Now, let us suppose that resource allocation policy requires that resources M3 and M2 should be used repetitively, one after the other, for execution of job J2. This repeatable resource allocation sequence can be written in a form of matrix

$$\mathbf{S}_{\mathrm{J2}} = \begin{array}{c} \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix} \\ \begin{array}{ccccc} M1 & M2 & M3 & M4 & M5 \end{array} \end{array},$$

or in general form

$$\mathbf{S}_{\mathrm{j}} = \begin{bmatrix} s_j^1 \\ s_j^2 \\ \vdots \\ s_j^{\omega} \end{bmatrix}. \qquad \textbf{(3)}$$

Resource allocation sequence matrix should be defined for each choice job in the system. Our goal is to find set $\mathfrak{I} = \{\mathbf{S}_j\}$ (where $w = |\mathfrak{I}|$ equals to the number of choice jobs in the system) such that system has no conflicts and it is deadlock free. In [17] it has been proven that elements of such set, *i.e.* sequence matrices, are formed of rows of MJI sub-matrices.

As an example, let us examine 3-step sequence for system (2), defined by the following resource allocation sequence matrices,

$$\mathbf{S}_{\mathrm{J2}} = \begin{bmatrix} s_{J2}^1 \\ s_{J2}^2 \\ s_{J2}^3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

$$\mathbf{S}_{\mathrm{J3}} = \begin{bmatrix} s_{J3}^1 \\ s_{J3}^2 \\ s_{J3}^3 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$$

$$\mathbf{S}_{\mathrm{J4}} = \begin{bmatrix} s_{J4}^1 \\ s_{J4}^2 \\ s_{J4}^3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

In order to get a better insight in the system behavior, Figure 2. presents how parts pass through the line. It should be noted that sequences are executed in a way that new allocation of resources (new step) is performed after all jobs are finished and parts reside in the buffers. First part enters the system and it is processed by M3 (J2M3-'1' in the first row of $\mathbf{S}_{\mathrm{J2}}$), than, it proceeds to job J3 on M4 (J3M4-'1' in the first row of $\mathbf{S}_{\mathrm{J3}}$), and than to job J4 on M2 (J4M2--'1' in the first row of $\mathbf{S}_{\mathrm{J4}}$). The second part enters the system and it is processed by M3 (J2M3-'1' in the second row of $\mathbf{S}_{\mathrm{J2}}$), than, it proceeds to job J3 on M5 (J3M5-'1' in the second row of $\mathbf{S}_{\mathrm{J3}}$), and so on. Conflict occurs for the first time at k+2 as J4 on part 1 is planned for M2 while, in the same time, the third part, that just entered the system, requires the same machine (J2 on M2).

This example clearly demonstrates that repeatable resource sequences can lead the system in conflict. In [17] it has been shown how to determine conflict-free set $\mathfrak{I} = \{\mathbf{S}_j\}$. Furthermore, we proved that usage of conflict-free set of resource allocation sequences not only resolves possible conflicts in the system, but also has direct consequence on the system stability, *i.e.* it provides deadlock-free behavior of the system.

part	*k*	*k*+1	*k*+2	*k*+3	*k*+4	*k*+5	...
1	J2M3	J3M4	J4M2				
2		J2M3	J3M5	J4M2			
3			J2M2	J3M3	J4M4		
4				J2M3	J3M4	J4M2	
...					...	...	

Figure 2: Presentation of parts passing through the line.

4.1. Sequence optimization

In order to determine an optimal sequence we introduce cost matrix $\mathbf{\Lambda}_{\mathrm{W}}$. The cost matrix, with the form identical to MJI matrix $\mathbf{\Lambda}$, captures cost of execution of ith job by using jth resource. By using cost matrix we can extract cost of each MJI sub matrix as

$$C^p = \sum_{i=1}^{n}\sum_{j=1}^{n}(\mathbf{\Lambda}_{\mathrm{W}}^p)_{ij}\,, \qquad \textbf{(4)}$$

where $\mathbf{\Lambda}_{\mathbf{W}}^{p} = \mathbf{\Lambda}_{\mathbf{W}} \cdot (\mathbf{\Lambda}^{p})^{\mathrm{T}}, \mathbf{\Lambda}^{p} \in \Sigma_{NR}$, $p = 1..k$, is an MJI submatrix. If one assumes that sequence matrices are comprised of ω rows (sequence of ω repeatable steps), where each MJI submatrix $\mathbf{\Lambda}^{p}$ will be used ω_p times, *i.e.* $\omega = \sum_{p=1}^{k} \omega_p$, $\omega_p \in \mathbb{N}$, $0 \le \omega_p \le \omega$, then the total cost generated by those sequences is

$$C^{tot} = \sum_{p=1}^{k} \omega_p C^{p} . \qquad \textbf{(5)}$$

It is clear that minimization of the total cost, defined as (5), is trivial problem – one should use only $\mathbf{\Lambda}^{p}$ with the smallest C^{p} in order to achieve minimal cost. However, in that case it might happen that utilization of the system resources would be highly unbalanced or even some resources would not be used at all. Hence, the cost function should be extended with a relation that captures resources utilizations.

For each MJI sub-matrix one can define a *resource utilization vector* as

$$\mathbf{u}^{p} = \mathbf{1}^{\mathrm{T}} \cdot \mathbf{\Lambda}^{p}, \qquad \textbf{(6)}$$

where $\mathbf{1}_{\mathrm{mx1}}$ is vector with all elements equal to 1. As a result resource utilization vector $\mathbf{u}^{i}$ is a binary vector with j^{th} element equal to 1 if corresponding resource participates in execution of the sequence containing rows of $\mathbf{\Lambda}^{i}$ sub-matrix. Finally, an integer vector that represents overall usage of the system resources is determined as

$$\mathbf{u} = \sum_{p=1}^{k} \omega_p \mathbf{u}^{p} . \qquad \textbf{(7)}$$

Now, the second objective, balanced usage of the system resources, can be defined in the following form

$$\frac{(1-\varepsilon)}{(1+\varepsilon)} \le \frac{u_i}{u_j} \le \frac{(1+\varepsilon)}{(1-\varepsilon)}, \ \forall i, j = 1..q, \qquad \textbf{(8)}$$

where u_i and u_j are i^{th} and j^{th} component of $\mathbf{u}$, ε is design parameter such that $0 \le \varepsilon < 1$, and q is the number of resources that should be balanced (for q = m all resources in the system shall be included in optimization). Fully balanced utilization of resources is achieved if $\frac{u_i}{u_j} = 1, \ \forall i, j = 1..q$. However, this goal might be very difficult (in some cases even impossible) to obtain, which depends on the system structure and executed sequence. Hence, by introducing parameter ε one is able to relax rigorous balancing requirement – for $\varepsilon \approx 1$ the system could become unbalanced, while for ε = 0 one requires full balance of the system resources exploitation.

Minimization of (5) by varying ω_p, $p = 1,\dots,k$ under conditions (8) with predefined ε and ω is a mixed integer linear programming problem which can be solved using standard algorithms.

4.2. Case study

The proposed method has been tested on the system presented in [15] and [16]. Although, this example has only two choice jobs and it is comprised of MRF and FMRF sub-systems, it has been chosen so that the proposed method can be compared with various control techniques already implemented on this particular manufacturing system. The system's PN model is shown in Figure 3. The system has 3 part types, P1, P2 and P3, 4 machines M1-M4, and 3 robots R1-R3. Part routes for P1 and P2 are predetermined (MRF), while P3 has choice jobs (FMRF). All resources, except for M1, are shared (only utilization of shared resources will be optimized).

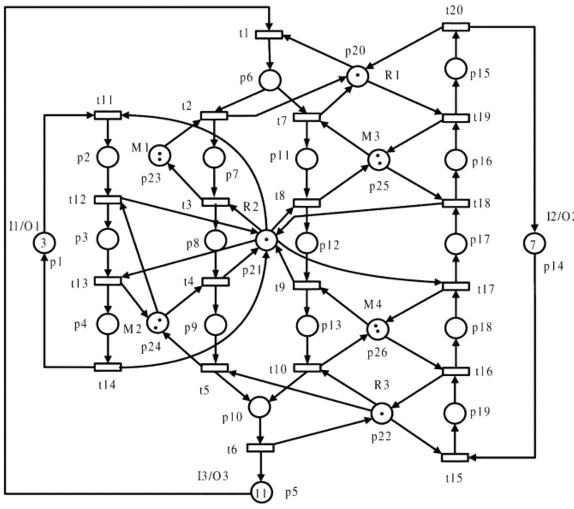

Figure 3: PN model of the system used for the case study [15].

The goal is to find the optimal sequence that includes all shared resources in the system for $\omega = 6$ and $\varepsilon = 0.2$. Such value of ε gives $0.8 \le \frac{u_i}{u_j} \le 1.2$, *i.e.* it is required that usage of resources is balanced. The following costs have been used in optimization:

c_{M11}=6, c_{M31}=4, c_{M22}=5, c_{M42}=9.

Three MJI sub-matrices have been used for construction of sequences such that $\mathbf{u}^{1}$= [1 1 0]$^{\mathrm{T}}$, $\mathbf{u}^{2}$= [0 0 1]$^{\mathrm{T}}$ and $\mathbf{u}^{3}$= [0 1 1]$^{\mathrm{T}}$, where components correspond with resources M2, M3 and M4. This gives overall usage of shared resources as $\mathbf{u} = \omega_1 \mathbf{u}^{1} + \omega_2 \mathbf{u}^{2} + \omega_3 \mathbf{u}^{3} = [\omega_1 \quad \omega_1 + \omega_3 \quad \omega_2 + \omega_3]^{\mathrm{T}}$. Optimization yields to the following values: $\omega_1 = 3$, $\omega_2 = 2$, $\omega_3 = 1$ with total usage of resource within the sequence equal to $\mathbf{u}$ = [3 4 3]$^{\mathrm{T}}$ as it is shown in Figure 5. Results obtained by simulation with MJIWorkshop, software tool presented in [12], are shown in Figures 5 and 6. It can be seen from Figure 6 that system is deadlock free, *i.e.* flow of the parts is uninterrupted.

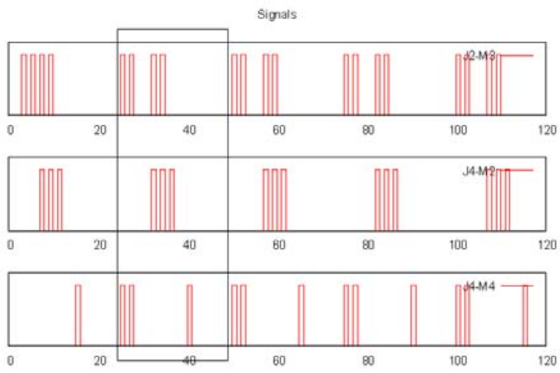

Figure 5: Utilization of M2, M3 and M4 within the sequence.

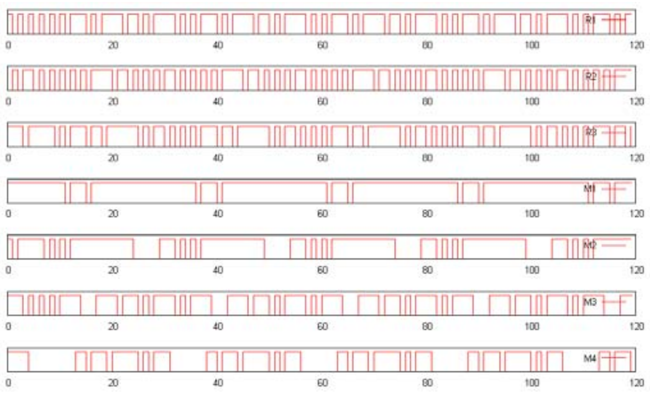

Figure 6: Utilization of all resource in the system.

5. CONCLUSION

In a finite-buffer flexible manufacturing systems, any dispatching policy for interrupted part flow has to essentially take into account the composition of the interconnection between jobs and resources. The proposed optimal non-blocking dispatching policy is based on machine-job incidence matrix (MJI), obtained from Steward sequencing matrix and Kusiak machine-part incidence matrix, and explained in details in [17].

Since FMRF systems contain shared resources and choice jobs, a solution to allocation of resources to jobs is determined in a form of repeatable resource sequence over the set of resources available for particular choice job. Obtained sequences not only stabilize the system but provide an optimal utilization of resources based on price and balanced use of all resources.

Efficiency of the proposed method has been demonstrated on an example involving multi-part type manufacturing system.

REFERENCES

[1] W.M. Wonham, *Supervisory Control of Discrete Event Systems*, Lecture notes, 2005.

[2] T. Murata, Petri nets: properties, analysis and applications, *Proc. IEEE*, 77, 4, 1989, pp. 541–580.

[3] Y.C. Ho, X.R. Cao, *Perturbation Analysis of Discrete Event Dynamic Systems*, Kluwer Academic Publishers, 1991.

[4] D.V. Steward, The Design Structure System: A Method for Managing the Design of Complex Systems, *IEEE Transactions on Engineering Management*, 28, 1981, pp. 71-74.

[5] A.Kusiak, J. Ahn, Intelligent Scheduling of Automated Machining Systems, *Computer-Integrated Manufacturing Systems*, 5, 1, 1992, pp. 3-14.

[6] T.R. Browning, Applying the Design Structure Matrix to System Decomposition and Integration Problems: A Review and New Directions, *IEEE Transactions on Engineering Management*, 48, 3, 2001, pp. 292-306.

[7] M.D. Jeng, F. DiCesare, Synthesis Using Resource Control Nets for Modeling Shared-Resource Systems, *IEEE Trans. Rob. Autom.* RA-11, 1995, pp. 317–327.

[8] R.A. Wysk, N.S. Yang, S. Joshi, Detection of Deadlocks in Flexible Manufacturing Cells, *IEEE Trans. Rob. Autom,* 7, 6, 1991, pp. 853−859.

[9] S. Bogdan, F.L. Lewis, J. Mireles, Z. Kovacic, *Manufacturing Systems Control Design: a matrix based approach*, Springer, 2006.

[10] M.V. Iordache, P.J. Antsaklis, *Supervisory Control of Concurrent Systems: A Petri Net Structural Approach*, Birkhauser, Boston, USA – 2006.

[11] M.C. Zhou, M.P. Fanti, *Deadlock resolution in computer-integrated systems*, Marcel Dekker/CRC Press, New York 2005.

[12] I. Sindičić, T. Petrović, S. Bogdan, Modeling and Simulation of Manufacturing Systems based on Machine-Job Incidence Matrix, *Proc. Int. Conference on Mathematical Modelling*, Vienna, 2009.

[13] T. Petrović, S. Bogdan, I. Sindičić, Determination of Circular Waits in Multiple-Reentrant Flowlines based on Machine-job Incidence Matrix, *Proc. European Control Conference,* Budapest, 2009.

[14] S. Lee, D.M. Tilbury, Deadlock-Free Resource Allocation Control for a Reconfigurable Manufacturing System With Serial and Parallel Configuration, *IEEE Trans. on SMC-part C*, 37, 6, 2007, pp.1373-1381.

[15] ZhiWu Li, MengChu Zhou; Two-Stage Method for Synthesizing Liveness Enforcing Supervisors for Flexible Manufacturing Systems Using Petri Nets, *IEEE Transactions on industial informatics, Vol. 2, No. 4,* November 2006, pp. 313-325

[16] I.Sindičić, S.Bogdan, T.Petrović; Dispatching in Free-choice Multiple Reentrant Manufacturing Flowlines by using machine-Job Incidence Matrix; 6th IEEE Conference on Automation Science and Engineering – CASE 2010, p617-623, Toronto, Canada, 22-24. August 2010

[17] I.Sindičić, S.Bogdan, T.Petrović; Resource Allocation in Free-choice Multiple Reentrant Manufacturing Systems Based on Machine-job Incidence Matrix;, *IEEE Transactions on Industrial Informatics*, Vol. 7, No. 1, 2011, pp. 105-114.