Raport Badawczy
Research Report

RB/18/2004

Modelling of distributed control for repetitive production flow prototyping

B. Skołud, Z.A. Banaszak

Instytut Badań Systemowych
Polska Akademia Nauk

Systems Research Institute
Polish Academy of Sciences

POLSKA AKADEMIA NAUK

Instytut Badań Systemowych

ul. Newelska 6

01-447 Warszawa

tel.: (+48) (22) 8373578

fax: (+48) (22) 8372772

Kierownik Pracowni zgłaszający pracę:
Prof. dr hab. inż. Zdzisław Bubnicki

Warszawa 2004

Modelling of distributed control for repetitive production flow prototyping

Bozena Skolud[1] and Zbigniew A. Banaszak[2]

[1]Department of Engineering Processes Automation and Integrated Manufacturing Systems, Silesian University of Technology, 18a Konarskiego Str., 44-100 Gliwice, Poland, phone +48 322371601, fax +48 322371624, e-mail: skolud@polsl.gliwice.pl

[2] Laboratory of Knowledge and Artificial Intelligence, Systems Research Institute of Polish Academy of Sciences, 75 Podwale, Str., 50-449 Wroclaw, Poland, +48 683282540, e-mail: z.banaszak@iizp.uz.zgora.pl

Abstract. The scope of this paper is to present the distributed control modelling of multi-assortment repetitive production. The paper focuses on production flow allocation both in the steady-state and transient phases. The method aimed at dispatching rules designing, and guaranteeing a deadlock-free and starvation-free system functioning while maximizing a rate of system resources is proposed. The concept of the production constraint-based flow coordination and the system's performance evaluation are discussed in detail.

1. Introduction

The globalisation of economic activity and the diversification of customer demands have significantly increased industrial competition. The decision making for a system manufacturing a variety of products and changing their characteristics is one of the most important and difficult tasks of a manager. The problem is observed mainly in companies characterised by multi-task machinery, production planning based on the receipt of customer order and the fact that both product lead time and price vital for customer satisfaction are subjects of negotiation (Hendry, Kingsman 1989). The ability of quick validation of the market demands and the corresponding reaction decides about the competitive advantage. The manufacturer should take the decision about the order acceptance the moment the production order is placed. Being competitive compels the organizational method of production flow, and, first and foremost, the time at which the method is chosen and applied. The speed of decision making in the area of production flow, and the creation of control procedures are of crucial importance.

A number of analytical and computer simulation models dealing with the problem of predicting and verifying system performance fail when addressing the issue of a whole system performance evaluation, where the performance depends on the synchronization of their interactions. From the control perspective, it is more appropriate to define a specification of some new desired behaviour, and to determine whether the specification can be met through a set of controllable events (Hendry and Kingsman 1989, Kim *et al* 2001, Sihn *et al* 2000).

Therefore, an appropriate tool supporting the decision making in the course of production flow selection is necessary. Moreover, the considered computational complexity of tasks motivates the search for more effective approaches.

This research summarises our previous work (Banaszak and Zaremba 2001, Skołud 2000, Zaremba and Banaszak 1995, Zaremba *et al* 1999, Skołud and Krenczyk 2001, Banaszak *et al* 2003) and presents a modelling framework providing constraints-based methodology for the production flow management support. Moreover, the concept of distributed control based on the dispatching rules allocation and the relevant reachability problem are discussed. Consequently, the distributed control procedure and the computer supported production flows management are presented.

2. Modelling of the production flow

Let us consider a model of multiple assortment manufacturing providing a framework for couping with the problem of the production flows management, i.e. an examination of the consistency between production order requirements (client's demands) and enterprise capacity (producer's requirements). Thus, the model is composed of two parts (Skolud 2000):

- the model of the manufacturing system (the system at the producer's requirements),
- the model of production order (the client's demand).

Both models provide a framework enabling to analyse a set of conditions in the context of the method aimed at production order prototyping for a small or medium enterprise.

Let us consider a manufacturing system as a triple $S = (\{M_i, i= 1,...,m\}, \{P_j, j = 1,...,n\}, \{CS^j_{ik}, i=1,...m, k=1,...,m, j=1,...,n\})$, determining a set of machine tools, manufacturing, and storage's capacity, where:

M_i – the i-th machine,

P_j – the j-th process already realised in the system,

CS^{j}_{ik} –the buffer located between the i-th and the k-th machine for the j-th process (C^{j}_{ik} -the capacity of the buffer).

The production order represents the client's demands, and is described by P_j= (I_j, tz_j, M_{Aj}, B_j, T') where:

P_j – the i-th manufacturing process,

I_j – the production volume of the i-th manufacturing process,

tz_j – the i-th process completion data (time),

A_j – the matrix of alternative routes of the i-th manufacturing process,

B_j, – the batch size of the i-th manufacturing process,

T'– the period of a batch delivery.

The main problem to be addressed is stated in the following way: Given is a production order, a set of machine tools and their availability within the presumed time horizon. Is it possible to complete a given production order in the system characterised by the above constraints imposed by manufacturing system S?

3. Production flow

The processes running in the system share machines and compete for an access to some commonly shared resources. It is important to synchronise these processes in the system, assuring the expected level of the system operation. The idea underlying the Optimal Production Technology (OPT) (Goldratt and Cox 1987, Lepore and Cohen 1999) is to provide the maximum number of bottlenecks in order to maximise machine utilisation and to enhance the throughput. The consideration of the system quantity parameters makes sense under the condition that the system can achieve its functionality by following the assumed qualitative requirements. However, one should be aware that the obtained solution may not be the best one, because it is very probable that the set of solutions obtained in this way does not contain the optimal one.

From the dispatcher's point of view it is important to ensure that the allocation of the dispatching rules and the allocation of buffers and their capacities guarantee deadlock-free and starvation-free realization of the processes already running in the systems as well as the processes to be introduced later on.

3.1. Simultaneous processes allocation (steady-state)

A system of repetitive processes is treated as a distributed system composed of autonomous and repetitive process flows that compete for access to the shared machine tools. A procedure of distributed flow control consists of the dispatching rules assigned to the machines shared by concurrently flowing processes.

The access of real-time processes to the shared machines may result in a conflict, which may be solved by proper allocation of the buffers capacity and design of the dispatching (priority) rules, handling the required synchronisation of the processes execution.

The dispatching rule σ_i controlling the access of the processes to the i-th shared machine is the sequence: $\sigma_i = (Pa_1, Pa_2, ..., Pa_j, ..., Pa_n)$ that determines the number of the processes executed on the i-th machine (e.g. machine tool), where: Pa_j – the-a_I-th process, $i \in \{1,2,...,m\}$, m – the number of machines, $a_j \in \{1,...,n\}$, n – the number of processes. For illustration, let us consider processes P_1, P_2, and P_3 that compete for the access to machine M_1. The dispatching rule: $\sigma_1 = (P_1, P_1, P_3, P_2)$ guarantees the access to the machine tool M_1 twice for process P_1, once for process P_3, and once for process P_2.

As it was already mentioned above, the rhythmic character of the production flow is desired. Repetitive production means that for every period T, the same sequence of operations is repeated on a machine. Period T is determined by a sequence of operations qualified by the dispatching rules.

The dispatching rules can be determined either from the master schedule, or deduced from a computer-supported procedure. The relevant procedure consists of the following six steps:

Step.1. Assign each process once for each dispatching rule allocated to machines on the process route,

Step 2. Check the balance condition.

The system balance is accomplished in the case when the following equations are satisfied for the shared machine (Skolud 2000):

$$\chi_1 n_{1,1} = \chi_2 n_{2,1} = \ldots = \chi_i n_{i,1} = \ldots = \chi_m n_{m,1},$$

$$\chi_1 n_{1,2} = \chi_2 n_{2,2} = \ldots = \chi_i n_{i,2} = \ldots = \chi_m n_{m,2},$$

$$\ldots$$

$$\chi_1 n_{1,j} = \chi_2 n_{2,j} = \ldots = \chi_i n_{i,j} = \ldots = \chi_m n_{m,j}, \quad (1)$$

$$\ldots$$

$$\chi_1 n_{1,n} = \chi_2 n_{2,n} = \ldots = \chi_i n_{i,n} = \ldots = \chi_m n_{m,n},$$

where:

χ_i – the repetitiveness of the dispatching rule allocated on the i-th machine,

$n_{i,j}$ – the repetitiveness of the j-th process in the dispatching rule allocated on the i-th machine.

Step 3. Check the buffer capacity condition.

Simultaneously with condition (1) the condition of the buffers space allocation in the system should be satisfied. It means that the capacity of buffer $CS^j_{i,k}$ allocated between machines M_i, and M_k for the j-th process should satisfy the following condition (2):

$$CS^j_{i,k} \geq n_{i,j} \cdot \chi_i, \tag{2}$$

If conditions (step 2) and (step 3) hold, then the dispatching rules allocated to the machines guarantee that a system composed of repetitive processes is in the cyclical steady-state, i.e., is deadlock-free and starvation-free. In that context the conditions guaranteeing the transition from the given steady-state to another one can be considered. Accordingly, if conditions (2) and (3) hold, the next step of the procedure may be considered.

Step 4. Determine cycle time T (system repetitiveness).

Step 5. Check the sufficient condition of the due time order completion.

Step 6. If non process is executed in due time, enlarge the number of the processes occurrence in the dispatching rules allocated to the machines shared by that process and go to step 3.

If conditions (corresponding to the steps 1-6) are satisfied or the disposal time searching is over, then *STOP.*

However, it should be noted that the above-mentioned procedure relates only to the steady-state behaviour of the system, i.e. to cases that are free of transient periods.

Additionally, the execution of the assigned dispatching rules depends on the starting conditions specifying the necessity of the initial intermediate buffers occupation by the elements at different production phases.

For the purpose of illustration, let us consider the system composed of 3 machines M_1, M_2, M_3. The matrices (3) describe processes P_1, P_2 and P_3, waiting for realisation in the system. Dispatching rules $\sigma_1 = (P_3, P_1)$, $\sigma_2 = (P_1, P_2)$, $\sigma_3 = (P_2, P_3)$ are allocated to machines M_1, M_2, and M_3, respectively.

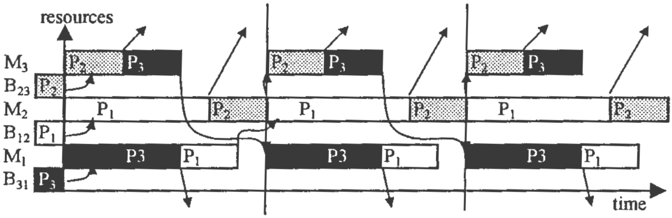

Figure 1. Gantt's chart of the production flow for the initial occupation of the buffers.

The first row of each matrix (3) describes the machine number (followed along the production route), and the second row represents operations times.

$$P_1 = \begin{bmatrix} 1 & 2 \\ 2 & 5 \end{bmatrix},\ P_2 = \begin{bmatrix} 2 & 3 \\ 2 & 2 \end{bmatrix},\ P_3 = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}. \tag{3}$$

Let us assume that intermediate buffers are located between each pair of subsequent machines along each production route. Consider a situation where elements are allocated to intermediate buffers at different production phases. In this case, the system is synchronised according to its bottleneck, i.e. machine tool M_2, see Figure 1.

3.2. Introduction of the process to the already functioning system (steady-state)

Let us assume that some processes are performed in the system while some of its capability still remains free. In such case the first constraint that should be taken into account is the system productivity. This raises the following question: Is there any batch production of the processes waiting for execution that does not disturb the processes already running in the system? This question is answered in two phases: through the batch sizing and buffers capacity verification.

Batch sizing

Theorem 1. System $S=(\{M_i, i=1,...,m.\}, \{P_j, j=1,...n\}, \{CS^j_{ik}, i=1,...m, k=1,...,m, j=1,...,n\})$ is given, where M_i is the i-th machine and P_j is the j-th process executed in the system, CS^j_{ik} is the buffer located between the i-th and the k-th machine for the j-th process (C^j_{ik}-buffers capacity). The system is in the steady-state at time period T. The state of the system is described by the list of elements $V=\{V_1, V_2,...,V_i,..., V_m\}$. Each element of the list is a vector describing the occupancy of the i-th machine during period T. The vector is the following: $V_i=[v_1^i, v_2^i, ..., v_t^i, ..., v_T^i]$. Each element v_k^i of vector V_i corresponds to a time unit. The values of the vector elements are:

$$V_i = \begin{cases} 0, \text{ if the machine is not occupied in the k-th time unit} \\ 1, \text{ if the machine is occupied in the k-th time unit} \end{cases}$$

Matrix P_{n+1} describes the process of new production order Z_{n+1}. So, each process waiting to enter the system can be described by the relevant matrix: $P_{n+1(2 \times R)}$,

where:

R - the number of machines sharing the process,

1st row of the matrix containing the number of machines (route of the process),

2nd row of the matrix containing the time of the operations realized on a given machine.

If conditions (4) and (5) hold, we are dealing with the case of batch production that does not disturb the processes already running in the system.

$$\forall\, i \in (P_{n+1})^2,\ \exists\, t \in \{1,\ldots,T\},\ v_t^{\,i} = 0, \tag{4}$$

$$\exists\, t \in \{1,\ldots,T\},\ \forall r \in \langle t, .., t+p_{2,r}-1 \rangle,\ v_t^{\,i} = 0, \tag{5}$$

where:

$(P_{n+1})^1$ – the number of machines (first row of matrix P_{n+1}),

v_t^i – the t-th element of vector V_i,

T – the cycle time,

P_{n+1} – the matrix of process Z_{n+1},

$(P_{n+1})^2$ – the realization time (the second row of matrix P_{n+1}),

$p_{2,r}$ – the r-th element of $(P_{n+1})^2$.

For the proof see (Skolud 2000).

Condition (3) of the theorem 1 assures that the machines required for the execution of process P_{n+1} are not critical. It means that the machine is not used at some moments (during the period T). The assumption of condition (4) guarantees that at least one-unit batch production can be executed in the system.

Buffers allocation

What allocation of the buffers capacity makes it possible to realize a given production batch for production order Z_{n+1} to assure its due time completion?

The elements executed in one cycle should wait for realization in the next period. Thus, the buffers capacity should be bigger than the number of the elements in a given production batch.

Theorem 2. System $S=(\{M_i, i=1,...,m.\}, \{P_j, j=1,...n\}, \{C^j_{ik}, i=1,...m, k=1,...,m, j=1,...,n\})$ is given. If condition (5) holds, then the realization of production batch size K is acceptable.

$$C^j_{ik} \geq 2*K^j-1 \tag{6}$$

where:

C^j_{ik} - the capacity of the buffer allocated between the i-th and k-th machine for the j-th process

K^j – the batch size of the j-th process.

For the proof - see (Skolud,2000)

Condition (6) guarantees a smooth realisation of production batch K in the system, i.e., with no influence on the production already realized in the system. If the realisation of two operations takes place in the same period T, then the relevant buffer capacity allocated for this process between two neighbouring machines is equal to K. If the operation of the j-th process is processed during a certain period of time, and the second operation in the next period, then the buffer with the capacity at least equal to $2*K^j-1$ assures a smooth production flow without disturbances, see an illustrative example below.

Illustrative example. Let us consider the process P_1 running on M_1 and M_2 with operation times $t_1=1$ and $t_2=4$, respectively. The batch consists of p=3 workpieces. Figure 2 presents the running processes and the capacity of the buffer required for the system to function without disturbances.

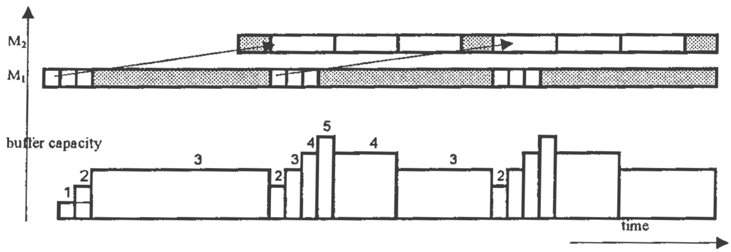

Figure 2. The buffers capacity allocation in the system

4. Control procedure designing

Let us consider distributed control (Kim *et al* 2001), which means that machines must not communicate with each other, and decisions are controlled locally by a machine operator.

The problem of the qualitative functioning of the system is discussed, when the system passes between two arbitrary given steady-states (see Figure 3).

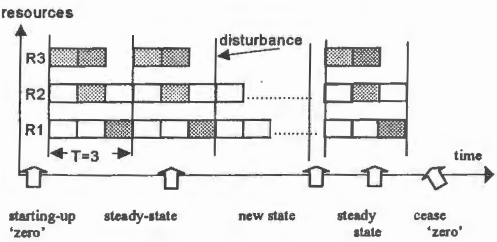

Figure 3.The illustration of possible applications of the reachability problem

However, a general solution still remains open. It involves many factors, including the structural (e.g. buffers capacity), and functional (e.g. dispatching rules) constraints. That is a well known fact that the buffer capacity assignment conditions, as well as the conditions imposed on the dispatching rules allocation to guarantee cyclic steady-state execution of the processes, play a primary role. The detailed discussion of the relationships among an initial state, the buffers capacity allocation, and the dispatching rules assignment in a class of repetitive manufacturing systems can be found in (Skolud and Krenczyk 2001).

The reachability problem is observed in the case of manufacturing starting up and its termination, particularly, in the case of transition from 'zero' state, when none of machines is busy, to the expected steady-state, known from the derived dispatching rules. The termination of the production creates the opposite situation, i.e. the transition from the known state to 'zero' state. In both cases, the transition procedure depends on the local dispatching rule allocated to machines and manufacturing route. The procedure of the manufacturing cease is created depending on the number of the elements executed during the starting-up rule realisation. The cease rule should assure the production termination without the occurrence of deadlocks and starvation.

The presented problem can be solved by direct application of starting-up and cease rules. The sequence of the transition between two steady-states T1 and T2 can be divided into two stages. Firstly, all production realized in period T1 is terminated and the system is reduced to state S_0. Secondly, the starting-up rules are made and the system transits from state S_0 (the system is empty) to T2. Such solution guarantees the deadlock-free transition from state T1 to T2. However, the method needs a relatively long time. Let us observe that in the cease phase that moves the system to state S0 the elements introduced during the starting-up phase to fill the inter-operational buffers were removed from the system. Subsequently, the transition from S_0 up to new state T2 is obtained by the starting-up processes that are realized in both steady-state phases to fill the inter-operational buffers again.

It is easy to observe that for all processes, the first emptying inter-operational buffers (i.e., in the cease phase), the subsequent filling-up buffers (during the starting-up phase) is performed by the same number of the same type of work-pieces. Thus, for the transition between two known steady-states, when the same processes are being started or terminated, the cease and starting-up rules are designed in such a way as to follow the demands imposed by the starting and/or terminating processes.

Particularly, this may be observed in the case of the transition from 'zero' state, when none of the machines is busy, to the expected steady-state known from the derived dispatching rules. The case of production termination creates the opposite situation, i.e. the transition from the known state to the 'zero' state. In both cases, the transition procedure depends on the local dispatching rule allocated to the machines and to the production route. The procedure of the production cease depends on the number of the elements executed during the starting-up rule realization. The cease rule should assure the production termination without deadlock and starvation.

The above mentioned problem inspires and motivates the construction of dispatching rules allocated locally to the system machines, so-called *'macro-rules'* (Skolud and Krenczyk 20001), which are a combination of dispatching rule σ_i, starting-up rule σ_i^R and the cease rule σ_i^W:

$$\sigma_i = \{\sigma_i^R, \sigma_i, \sigma_i^W\}, \tag{7}$$

The starting-up and the cease rule are executed uniquely, whereas the dispatching rule is executed repetitively, securing the steady-state behaviour of the system. Let us notice that the dispatching rule provides the self-synchronisation capability of the system, and brings the whole production to the end without deadlocks. The procedure of the starting-up and cease rules construction is given in (Krenczyk and Skolud 2001).

Procedure of starting-up rule construction

1. Given is the local dispatching rule $R_i(p_{i1}, p_{i2}, \ldots, p_{io_i})$, i=(1,...,m), allocated to the i-th machine.

2. Rank of processes $P_{p_{i1}}, P_{p_{i2}}, \ldots, P_{p_{io_i}}$ according to their increasing numbers N_{iw}, where:

N_{iw} - successive number of the operation in process $P_{p_{iw}}$ executed on the i-th machine,

$P_{p_{iw}}$ – successive occurrence of the process on the i-th machine , where w=1,2,...,o_i.

3. For settled process succession, the repetitiveness of each process in the starting-up rule allocated to the i-th machine is $K_{i1}^R, K_{i2}^R, \ldots, K_{io_i}^R$ according to:

$$K_{iw}^R = (O_{iw} - N_{iw}) \cdot \chi_i, \tag{8}$$

where:

K_{iw}^R – the product of process $P_{p_{iw_i}}$ operation numbers left to be performed after the i-th machine and the rule repetitiveness allocated to the i-th machine,

O_{iw} – the number of process $P_{P_{lw_i}}$ operation executed in the system,

χ_i – the repetitiveness of the dispatching rule allocated to the i-th machine.

Procedure of the cease rule construction

In the case when and the system starting-up procedures are applied and the completion time assumed for each process than the cease procedure is applied. It differ from the starting procedure only in step 3, which is the following one:

1. For the settled succession, the repetitiveness of each process in the cease rule $K_{i1}^{W}, K_{i2}^{W}, \ldots, K_{io_i}^{W}$ is obtained according to:

$$K_{iw}^{W} = (N_{iw} - 1) \cdot \chi_i, \tag{9}$$

The problem of a unique process introduction and/or termination may be extended to greater number of processes and may be combined together.

Illustrative example.

Let us consider a manufacturing system composed of 4 machines M_1, M_2, M_3, and M_4. The following production orders Z_1, Z_2, and Z_3 are processed in the system. The production orders correspond to processes P_1, P_2, and P_3 described by matrices MP_1, MP_2 and MP_3. The first row of each matrix contains the number of the machines, the second row contains operation times, and the third pre-set times.

$$MP_1 = \begin{bmatrix} 1 & 3 & 4 \\ 4 & 2 & 3 \\ 0 & 0 & 0 \end{bmatrix}, \quad MP_2 = \begin{bmatrix} 4 & 3 & 1 & 2 \\ 5 & 7 & 5 & 4 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \quad MP_3 = \begin{bmatrix} 2 & 1 \\ 4 & 3 \\ 0 & 0 \end{bmatrix}$$

Dispatching rules σ_1=(3,2,2,1), σ_2=(2,2,3), σ_3=(1,2,2), σ_4=(2,2,1) assuring deadlock-free functioning of the system in the cyclical steady-state are allocated to machines M_1, M_2, M_3, and M_4, respectively.

Process P_4 should be introduced into the system. Matrix MP_4 describes this process.

$$MP_4 = \begin{bmatrix} 2 & 4 \\ 3 & 5 \\ 0 & 0 \end{bmatrix}.$$

The local dispatching rules for a new steady-state are the following ones: σ_1=(3,2,2,1), σ_2=(2,2,3,4), σ_3=(1,2,2), σ_4=(2,2,1,4). In order to transit from a given cyclic steady-state to another one, only P_4 is taken into account in the starting-up rule. Thus, other processes are realized without any changes.

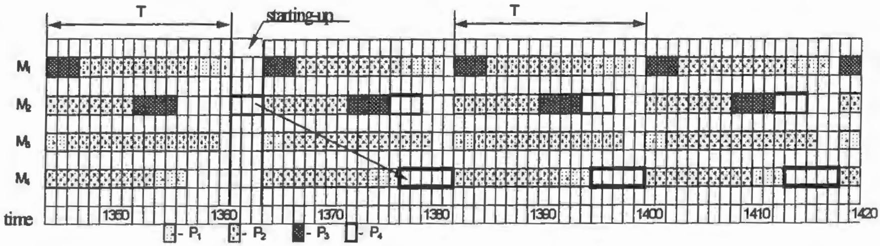

Fig.4. Gantt's chart. The Starting-up of process P_4

σ_1 = {(1,1,2,2), (3,2,2,1), (2,2,2,2,3)},

σ_2 = {(3,4), (2,2,3,4), (2,2,2,2,2,2)},

σ_3 = {(1,2,2,2,2), (1,2,2), (1,2,2)},

σ_4 = {(2,2,2,2,2,2), (2,2,1,4), (1,1,4)}.

For system machines, the starting-up rules are: σ_1^R=(), σ_2^R=(4), σ_3^R=(), σ_4^R=(). Each rule is executed once (Figure 4).

Application example

A method presented has been implemented in te software system (SWZ [ ]). Let us consider its application to the production flow planning in the machine working factory, which manufacture gear-boxes. The assortment and its description is given in the table 1. The production system structure and processes flow are illustrated in the figure 5.

Table 1. Assortment

No.	Name of element	No. of operation	Series size	No. of indivisible units (5 pcs. each)
1.	KZS33/2,5/A	3	4050	810
2.	KZS33/2,5/D	5	2025	405
3.	KZS16/4/D	4	4050	810
4.	KZS33/4/A	5	2025	405
5.	3M01403	4	2025	405
6.	3M01002	7	2025	405
7.	2M00704	5	4050	810
8.	5M01302	7	2025	405
9.	3M00802	6	2025	405
10.	6M00802	6	2025	405

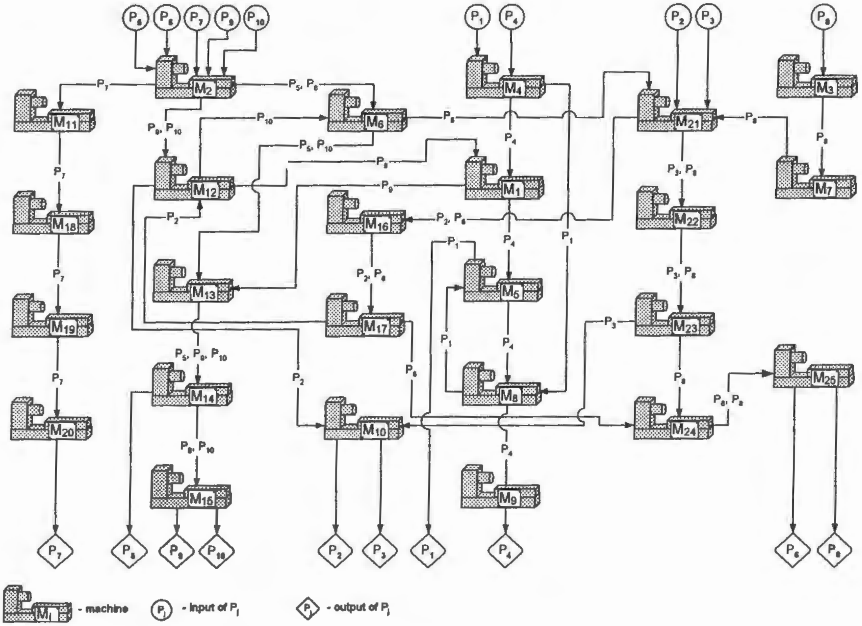

Fig.5. The structure of the manufacturing system (M1-M25 – machines, P1 input of P1, P1 output of P1)

Application of the SWZ system allows for quick answer what dispatching rule should be allocated to the machines and as well as answer the question how long the production will be realised. Below the list of dispatching rules is given.

R_1={(4,4,4,9,9,9);(4,9);(4,9,9)}

R_2={(5,5,5,6,6,6,6,6,6,7,7,7,7,7,7,7,7,9,9,9,9,9,10,10,10,10,10);(5,6,7,7,9,10);()}

R_3={(8,8,8,8,8,8);(8);()}

R_4={(1,1,1,1,4,4,4,4);(1,1,4);()}

R_5={(4,4);(1,1,4);(4,4,1,1,1,1)}

R_6={(6,6,6,6,6,5,5,10,10,10);(5,6,10);(6,5,10,10)}

R_7={(8,8,8,8,8);(8);(8)}

R_8={(1,1,4);(1,1,4);(1,1,4,4,4)}

R_9={();(4);(4,4,4,4)}

R_{10}={();(2,3,3);(2,2,2,2,3,3,3,3,3,3)}

R_{11}={(7,7,7,7,7,7);(7,7);(7,7)}

R_{12}={(10,10,10,10,2,9,9,9,9);(2,9,10);(10,2,2,2,9)}

R_{13}={(5,9,9,10,10);(5,9,10);(5,5,9,9,9,10,10,10)}

R_{14}={(9,10);(5,9,10);(5,5,5,9,9,9,9,10,10,10,10)}

R_{15}={();(9,10);(9,9,9,9,9,10,10,10,10,10)}

R_{16}={(2,2,2,6,6,6);(2,6);(2,6,6,6)}

R_{17}={(2,2,6,6);(2,6);(2,2,6,6,6,6)}

R_{18}={(7,7,7,7);(7,7);(7,7,7,7)}

R_{19}={(7,7);(7,7);(7,7,7,7,7,7)}

R_{20}={();(7,7);(7,7,7,7,7,7,7,7)}

R_{21}={(2,2,2,2,6,6,6,6,3,3,3,3,3,3,8,8,8,8);(2,3,3,6,8);(6,6,8,8)}

R_{22}={(3,3,3,3,8,8,8);(3,3,8);(3,3,8,8,8)}

R_{23}={(3,3,8,8);(3,3,8);(3,3,3,3,8,8,8,8,8)}

R_{24}={(6,8);(6,8);(6,6,6,6,6,8,8,8,8,8)}

R_{25}={();(6,8);(6,6,6,6,6,6,8,8,8,8,8,8)}

Assigned dispatching rules can be allocated respectively to machines. This allocation guarantees deadlock-free and starvation-free system behaviour. Additionally using data of operations time the series realisation time can be assigned.

5. Computer-supported management

In order to verify whether a given work order can be processed in an FMS that has some unused production capability, a constraint-propagation-based approach has been implemented. The underlying idea assumes the examination of conditions encompassing a relationship between particular constraint and production order and/or production system parameters. In the case the conditions hold, the production order may be accepted for processing; in the opposite case, however, some constraints have to be changed.

The concept of the constraint propagation has been implemented in the computer-aided production planning software package: the System of Production Order Validation (SWZ v.3) that is currently available in the Internet (http://swz.of.pl). The system supports a shop floor dispatcher in the course of the decision-making concerning the dispatching rules allocation, i.e. in the case of production flow planning and a control course. So, an integration of production planning (e.g. batch sizing) and control of its flow (the dispatching rules construction) can be simultaneously considered. The SWZ functions in an interactive mode. Basing on the given input data, the package helps to determine the parameters of the system operation, production and delivery batch sizes, delivery periods, Gantt's chart, the rate of machine utilization, the set of dispatching rules, the realization time, the system efficiency, etc.

Moreover, the package generates the control procedure (i.e., macro-rules) and integrates the production planning and control of its flow.

So, the obtained production flow prototype provides both a master schedule and a set of dispatching rules standing.

6. Concluding remarks

In this paper the problems involved in production flow control have been discussed. One of them, the so called "reachability" problem plays an important role in such cases as a new production start-up, production termination, as well as prediction of unforeseen disturbances of the production flow.

It has been shown that system performance depends not only on the effectiveness of its components, but also on the synchronisation of their interactions. Consequently, apart from the system capability and customer requirements balancing issues, the production flow control problems were discussed.

The methodology based on the theoretical results provided here has been implemented in a software package that allows users to investigate the effects of a new work order execution. The software system permits the rapid production prototyping and serves as a computer–aided production management tool, enabling the production planning as well as the distributed control of concurrent production flows.

Beside the above-presented approach, we believe that another essential element of the production flow planning is the issue of the integration of financial constraints. This topic will be developed in our further work.

7. References

Banaszak Z.A., Tang X.Q., Wang S.C., Zaremba M.B., 2000, Logistics models in flexible manufacturing. *Computers in Industry*, vol. 43 No.3, 237-248.

Banaszak Z., Skołud, Zaremba M., 2003, Computer-aided prototyping of production flows for a virtual enterprise. *Journal of Intelligent Manufacturing* Vol.14 Nr 1, 81-106.

Goldratt E:, Cox J.F., 1987, *The goal*, Coton on Hudson, NY; North River Press

Hendry L.C., Kingsman B.G., 1989, Production planning systems and their applicability to make to order companies. *European Journal of Operational Research*, no.40, 1989, 1-15.

Kim Y., Kang S., Lee S., Yoo S.B., 2001, *A* distributed, open, intelligent product data management system. *International Journal of Computer Integrated Manufacturing*, Vol14., No.2., 224-235

Krenczyk D., Skołud B., 2001, Self-synchronisation of distributed systems: application of dynamical dispatching rules (in Polish), Zeszyty naukowe Politechniki Śląskiej, Series Automatyka, Z.129, 2001, 243-252.

Lepore D., Cohen O., 1999, *Deming and Goldratt – The theory of constraints and the system of profound knowledge*. The North River Press.

Sihn W., von Briel R., Rost T., 2000, Interactive factory planning. *Preprints of the IFAC Symposium on Manufacturing, Modeling, Management and Control*, July 12-14, Patras, Greece, 411-417.

Skołud B.: *Planning of the multiassortment repetitive production* (in Polish), 2000, Z. Naukowe Politechniki Śląskiej, Mechanik, No.136, Gliwice, 2000

Skołud B., Krenczyk D., 2001, Flow synchronisation of the production systems: the distributed control approach. *The 6th IFAC Workshop on Intelligent Manufacturing Systems*, Poznań 24-26, 127- 132.

Zaremba M.B., Banaszak Z.A., Majdzik P., Jędrzejek K.J., 1999, Distributed flow control design for repetitive manufacturing processes. *Proc. of the 14th Triennial Worlds Congress of IFAC*, July 5-9, Beijing, P.R. China, Vol. A,1-6.

Zaremba M.B., Banaszak Z.,A., 1995, Performance evaluation for concurrent processing in cyclic systems. *Concurrent Engineering: Research and Application*. Vol.3, No.2, 123-130.

http://swz.of.pl