Assigning Jobs to Agents by Means of Petri Net-Based Models

František Čapkovič, Member, IEEE
Institute of Informatics
Slovak Academy of Sciences
845 07 Bratislava, Slovakia
Email: Frantisek.Capkovic@savba.sk

Abstract—Many times agents have to realize several jobs. However, only one job can be processed by one agent in the same time. The problem when and how to assign the jobs to agents is solved in this paper. Both agents and jobs are modelled by place/transition Petri nets (P/T PN). The task of assigning the jobs to an agent is understood to be a task of the jobs cooperation. To coordinate the jobs a supervisor is synthesized. The supervisor forces a desired job cooperation strategy to utilize the agent manufacturing capacity as exhaustively as possible. The problem when and how to assign the jobs to an agent is understood to be a task of the jobs cooperation. To coordinate the jobs a supervisor is synthesized. The supervisor forces a desired job cooperation strategy to utilize the agent manufacturing capacity as exhaustively as possible. The supervisor synthesis is based on discrete-event systems (DES) control theory and performed by means of P/T PN. In case of assigning several jobs to the agent(s) it is necessary to force the job priorities. The P/T PN firing count vector (Parikh’s vector) is utilized here to ensure the desired priorities. Then, the time parameter is added to the final P/T PN structure. In such a way timed PN (TPN) arises. It allows to analyze the time circumstances. Namely, processing of each job by the agent takes some time.

I. INTRODUCTION

In [1] the problem of cooperation and negotiation of agents by means of Petri net-based models was solved. In contrast to that, the problem of assigning several jobs to one or several agents will be examined here. We will start from [1]–[4] and basic information about P/T PN given in [5] will be used in the process of modelling the agents and jobs to be assigned as well as at synthesizing the supervisor ensuring the job cooperation. At the supervisor synthesis the theory of supervision developed in [6], [7] will be utilized and little modified.

First of all let us introduce P/T PN. As to the structure P/T PN are bipartite directed graphs < P, T, F, G > with P, T, F, G being, respectively, the set of places, the set of transitions, the set of edges from places to transitions and the set of edges from transitions to places. Here, the relations \( P \cap T = \emptyset \), \( F \cap G = \emptyset \) stand. Moreover, P/T PN have their dynamics - a movement of the PN marks in consequence of firing the enabled PN transitions. Namely, the number of marks in PN places (i.e. the state of PN places) is dynamically changed during this process. Dynamics can be formally expressed as the quadruplet < \( X, U, \delta, x_0 \) >, with \( X, U, \delta, x_0 \) being, respectively, the set of states (marking the places), the set of discrete events (states of transitions), the transition function and the initial state vector. Here, the relations \( X \cap U = \emptyset \), \( \delta : X \times U \to X \) hold. To model dynamics mathematically, the formal expression of \( \delta \) can be rewritten [1]–[4] into the system form \( x_{k+1} = x_k + B.u_k \), \( k = 0, \ldots, N \). Here, the matrix of parameters \( B = G^T - F \) where \( F, G \) are the incidence matrices of the bipartite graph edges. They correspond to the sets of edges \( F, G \). The system parameters (entries of \( B \)) and variables (entries of \( x_k \), \( u_k \), including \( x_0 \)) are positive integers. To avoid the occurrence of negative integers among entries of the vector variables \( x_k, u_k \), in any step \( k \) of the system dynamics development the condition \( F.u_k \leq x_k \) has to be satisfied. Let \( \sigma_{p_i} \in \{0,1,\ldots,c_{p_i}\} \) be the state (marking) of \( p_i \) with \( c_{p_i} \) being the capacity. Let \( \gamma_{t_j} \in \{0,1\} \) is the state of \( t_j \) (disabled or enabled). Then, \( x_k = (\sigma_{p_1}^k, \ldots, \sigma_{p_n}^k)^T \) is the state vector of the system \( x_k = (\gamma_{t_1}^k, \ldots, \gamma_{t_m}^k)^T \) is its \((m \times 1)\) control vector. \((.)^T\) symbolizes transpose of matrices and vectors. Just such an exact mathematical expression of P/T PN, in contrast to high-level PN, yields the possibility to deal with the PN models in analytical terms. However, P/T PN will be used here not only for modelling the jobs and agents but also for the supervisor synthesis.

A. P/T PN-based supervision

Principally, there exist two basic kinds of P/T PN-based supervision [6], [7], namely: (i) the supervision based on P-invariants of P/T PN utilizing the state vector; (ii) the extended supervision utilizing not only the state vector but also the control vector and Parikh’s vector.

1) The supervision based on P-invariants: P-invariant was defined in [5] as the vector \( w \) satisfying the equation \( w^T.B = 0 \). Hence, in case of more invariants \( w^T.B = 0 \), where \( w \) is \((n \times n_s)\) matrix and \( n_s \) is number of invariants. More precisely, P-invariant is the vector \( w \) satisfying the condition \( w^T.x = w^T.x_0 \) for each of the state vectors \( x \) reachable from \( x_0 \). Let us impose the restrictive condition

\[
L_p.x \leq b
\]  
(1)
on linear combinations of state vector entries. It prescribes a maximal number of marks kept together by selected entries. \( L_p \) is \((n_s \times n)\) integer matrix and \( b \) is \((n_s \times 1)\) integer vector. Then, the parameters of the supervisor corresponding to (1) can be acquired [3], [4] from the equation \((L_p.I_s).(B^T.B_s^T)^T = 0 \). This equation is the purposeful extension of the P-invariants definition \( w^T.B = 0 \). \( I_s \) is \((s \times s)\) identity matrix, \((L_p.I_s)\)
corresponds to $W^T$ and $(B^T B_s^T)^T$ corresponds to $B$. Hence, 
$L_pB + B_s = 0$ and subsequently 
\[ B_s = -L_pB \]  
Thus, the supervisor structure is given by $F_s$, $G^T_s$. The initial 
state of the supervisor follows [1], [3], [4] from $L_p x_0 + x_0 = b$. Namely, the inequality (1) can be transformed to the 
equation $L_p x + x_s = b$ by means of introducing the auxiliary 
variables (slacks) being entries of the supervisor state vector $x_s$. Hence, $^* x_0 = b - L_p x_0$.

2) The extended supervision: This method [6], [7] extends 
the conditions concerning the linear combination of the control vector entries and Parikh’s vector entries. The modified form ([3], [4]) of the inequality is 
\[ L_{p1}x + L_{p1}u + L_{v_p}v_p \leq b \]  
where $L_{p1}$, $L_{v_p}$ are $(n_s \times m)$ matrices of integers. The definition of the Parikh’s vector $v_p$ results from the evolution of the system describing P/T PN. Namely, starting from $x_0$, 
\[ x_1 = x_0 + B_1 u_0, \quad x_2 = x_1 + B_1 u_1 = x_0 + B_1 (u_0 + u_1), \]  
i.e. in general, $x_k = x_0 + B_1 (u_0 + u_1 + \ldots + u_{k-1})$. Just 
$v_p = u_0 + u_1 + \ldots + u_{k-1}$ is the Parikh’s vector. Its entries give us important information about how many times the particular transitions are fired during the development of the P/T PN dynamics from the initial state $x_0$ to a final (terminal) state $x_k$. It was proved in [6], [7] (and modified in [3], [4]) that when $b - L_p x \geq 0$ holds, the supervisor structure is as follows 
\[ F_s = \max (0, (L_p B + L_{v_p}), L_{p1}) \]  
\[ G^T_s = \max (0, (L_{p1} - \max (0, (L_p B + L_{v_p})) - \min (0, (L_p B + L_{v_p}))) \]  
and its initial state is $^* x_0 = b - L_p x_0 - L_{v_p} v_p^0$. The 
max(), min() are, respectively, the operators of maximum and 
minimum for matrices. They are executed element by element.

B. Timed PN

To deal with the time relations at assigning the jobs, timed Petri nets (TPN) will be used - see e.g. [8], [9] and so on - namely, T-timed TPN, where time delays will be added to the output transitions of places. The delays represent the times which are necessary for processing the tasks represented by the places. To see courses of processing the jobs by the agent(s) in time graphically, the TPN tool [10] will be used in Matlab.

II. PROBLEM FORMULATION

Consider a non-software agent of substantial character - e.g. a single machine of a manufacturing system like robot or a machine tool. Suppose that several jobs must be processed by this single agent. Naturally, the agent is able to process only one of the jobs in the same time. Consequently, utilizing of its manufacturing capacity is limited. Thus, in order to ensure processing all of the jobs, the jobs have to be assigned to the agent successively according to a prescribed order. Namely, usually the sequence of jobs cannot be arbitrary. It often depends on a technology. Therefore, processing the jobs by the agent have to be performed in the prescribed order.

A. Petri Net Based Model of the Agent and Jobs

To illustrate the situation, let us introduce Fig. 1. The P/T PN-based models of three autonomous jobs are introduced there. Reserve the PN place $p_1$ (missing in Fig. 1) for modelling the agent $A$. The jobs $J_1$, $J_2$, $J_3$ are modelled, respectively, by the subnets $\{t_2, t_1, p_3, t_2\}$, $\{t_4, t_3, p_5, t_4\}$, $\{p_6, t_5, p_7, t_6\}$. Here, $p_2$, $p_4$, $p_6$ express, respectively, the states when $J_1$, $J_2$, $J_3$ are idle, while $p_3$, $p_5$, $p_7$ express, respectively, the states when $J_1$, $J_2$, $J_3$ are processed by the agent $A$. The PN transitions $t_1$, $t_3$, $t_5$ expresses, respectively, the events starting the jobs $J_1$, $J_2$, $J_3$. Analogically, $t_2$, $t_4$, $t_6$ expresses, respectively, the events ending the jobs $J_1$, $J_2$, $J_3$.

In Fig. 2 the model of jobs processed by the agent $A$ (represented by $p_1$) is introduced. Here, the agent $A$ represents a source (a processing capacity) shared by the three jobs. When the transition $t_1$ is fired, processing the job $J_1$ starts and simultaneously, other two jobs $J_2$, $J_3$ are out of the possibility to be simultaneously processed, i.e. they are idle. The same is valid when the transition $t_3$ is fired (then $J_2$ is processed, $J_1$ and $J_3$ are idle) or when the transition $t_5$ is fired (then $J_3$ is processed, $J_1$ and $J_2$ are idle). Processing jobs takes a time. It means that after the time the job being processed should finish its activity and give a chance another job to be processed. However, because the jobs are autonomous, any job wants to retain the whole agent capacity only for itself, even for repeated processing. Consequently, it is necessary to resolve this problem in order to satisfy a global goal of the whole system (e.g. consisting of a group of agents).

Now let us introduce the approach making possible to obtain the structure given in Fig. 2 automatically. Having autonomous jobs and the verbal condition that only one job
can be processed by \( A \) in the same time we can express the condition in analytical form as follows

\[
\sigma_{p_3} + \sigma_{p_5} + \sigma_{p_7} \leq 1 \tag{6}
\]

In other words, only one of the places \( p_3, p_5, p_7 \) can be active in the same time. Let us utilize the supervision based on P-invariants for mutual exclusion of the jobs.

As we can see in Fig. 1 the P/T PN-based models of the elementary autonomous jobs have the following parameters

\[
F_{J_i} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}, \quad G_{J_i}^T = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad i = 1, 2, 3
\]

It can be said that the whole structure of the autonomous jobs is represented by the diagonal matrices

\[
F_J = \begin{pmatrix} F_{J_1} & 0 & 0 \\ 0 & F_{J_2} & 0 \\ 0 & 0 & F_{J_3} \end{pmatrix}, \quad G_J^T = \begin{pmatrix} G_{J_1}^T & 0 & 0 \\ 0 & G_{J_2}^T & 0 \\ 0 & 0 & G_{J_3}^T \end{pmatrix}
\]

The formal synthesis of the agent \( A \) which will process the three jobs as a fictive supervisor can be found as follows. Using the supervision method based on P-invariants, the condition (6) can be expressed in the following matrix form

\[
L_p \cdot x \leq b, \quad L_p = (0 1 0 1 0 1), \quad b = (1)
\]

Hence, the parameters of the supervisor are as follows

\[
B_s = -L_p \cdot B = (-1, 1, -1, 1)
\]

\[
F_s = (1010101), \quad G_s = (01010101)
\]

Let \( x_0 = (10101010)^T \) is the initial state of the group of the autonomous jobs. Thus, the supervisor initial state is

\[
0^0 x_s = b - L_p \cdot x_0 = (1) - (0) = (1)
\]

Complement these results into the matrices \( F_J, G_J \). Then the model of the final structure (agent plus jobs) is the following

\[
F_{A,J} = \begin{pmatrix} 1 & 0 & 1 & 0 & - & - & - & - \\ - & - & - & - & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}, \quad G_{A,J} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}
\]

\[
0^0 x_{A,J} = (0^0 x_s^T x_s^T)^T = (1|10101010)^T
\]

The final P/T PN structure of processing the three jobs by the single agent is in Fig. 2 where the place \( p_1 \) together with its interconnections with the jobs represents the agent \( A \).

However, it is necessary to say that even such a structure is not able to ensure correct assigning the jobs for the agent. Namely, when a job is being processed (i.e. when it keeps the capacity of the agent), it is practically impossible to stop this job. Namely, autonomous jobs are self-interest, i.e. egoistic. An interference which is able to change the jobs has to arrive from a control device (a supervisor). Therefore, we have to find a suitable approach which will guarantee a prescribed sharing the agent capacity by the particular jobs. There is only one possibility how to do, namely, by means of prescribing the effective policy how to share the agent capacity and strictly realize the policy. The supervisor synthesized by virtue of the policy is able to ensure the strict abidance of the policy.

### III. The Supervisor Synthesis

Theory of DES supervision is known from DES control theory. Principally, there exist two methods for the supervisor synthesis based on P/T PN. First of them is the method based on the P/T PN place invariants (P-invariants) [3], [4], [6] imposing constraints on the state vector \( x \) while the second of them [7] imposes constraints not only on the state vector \( x \) but also on the control vector \( u \) and on the Parikh’s vector \( v_p \) (its meaning and definition will be introduced below).

Here, we will use the latter methods. Although the P-invariant based method for the supervisor synthesis is suitable for the enough wide class of applications, it is limited yet as to forming the sufficiently general constraints determining the policy of the supervisor. For example, it is impossible to express priorities among the agents at the access to common sources. Fortunately, we can use the extended supervision method.

Just the Parikh’s vector gives us possibility to express priorities at the supervisor synthesis in our case of assigning jobs to the agent. Let us utilize this fact and apply the entries \( v_j \) of the Parikh’s vector \( v_p \) at the supervisor synthesis. Let us synthesize the supervisor by means of the extended method. Use only one matrix, namely \( L_{u_p} \), in the condition (3) and keep the matrices \( L_p, L_t \) to be zero matrices. Start from the structure given in Fig. 2. The priorities are usually given e.g. by the technology of a manufacturing system where the agent is included. Suppose the job priorities in the form \( \pi_{J_1} > \pi_{J_2} > \pi_{J_3} \) at processing the jobs \( J_1, J_2, J_3 \) by the agent \( A \). Here, the term \( (> \pi_{J_3}) \) ensures the working cycle continuation. The priorities can be expressed by means of the relations among the entries of the Parikh’s vector as follows

\[
v_3 > v_1, \quad v_1 > v_5, \quad v_5 > v_3 \tag{9}
\]

Here, the indices of the Parikh’s vector entries correspond to the indices of the transitions. Simultaneously, the inequalities (9) in the presented form define the working cycle of the agent. The priorities are determined by the sequence of firing the transitions \( \{t_1, t_5, t_3\} \). The working cycles are realized by means of repeating the firing sequence. The conditions (9) can be expressed in the matrix form

\[
L_{u_p} \cdot v_p \leq b, \quad L_{u_p} = \begin{pmatrix} 1 & 0 & -1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & -1 & 0 \end{pmatrix}, \quad b = (0)
\]

Considering \( L_p = 0, L_t = 0 \) in (4), (5) we obtain the following structure of the supervisor

\[
F_s = \max(0, L_{u_p}) = \begin{pmatrix} 100000 \\ 000100 \\ 001000 \end{pmatrix}
\]
time delays corresponding to the transitions \( t_1, \ldots, t_{12} \) are represented by the entries of the row vector

\[
\tau = (1, 20, 1, 20, 1, 40, 1, 40, 1, 60, 1, 60)
\]

Using TPN simulator HYPENS we obtain assigning three jobs \( J_1, J_2, J_3 \) to two agents \( A_1, A_2 \) in time. Start from the state when \( \sigma_{p_{19}} = 1 \) (i.e. foremost \( J_1 \) will be processed by \( A_1 \)) and \( \sigma_{p_{21}} = 1 \) (i.e. foremost \( J_2 \) will be processed by \( A_2 \)). It is possible to obtain the courses of markings the places \( p_i, i = 1, \ldots, 23 \), with respect to time. However, the most important and the most interesting courses are the courses of marking the places \( p_2, p_7, p_{12} \) representing jobs assigned to
Fig. 4. The behaviour (marking) of the places $p_1$, $p_3$, $p_5$, $p_7$ representing, respectively, the agent idleness and processing the jobs $J_1$, $J_2$, $J_3$, with respect to time.

Fig. 5. The case of assigning 3 jobs to 2 agents.

VI. CONCLUSION

The problem of assigning several jobs to an agent in order to process them effectively was solved by means of P/T PN-based model. To avoid selfish behaviour of jobs being processed (as to the consumption the agent manufacturing capacity) a supervisor was synthesized. The P/T PN were also used at the supervisor synthesis. The extended method utilizing the state vector $x$, control vector $u$ and Parikh’s vector $v_P$ was partly used here (only $v_P$). Namely, relations among the entries of $v_P$ are very important at assigning the jobs priorities during their processing by the agent. In general, assigning $m$ jobs to $n$ agents can be investigated and tested. The method was illustrated on processing $m = 3$ jobs by $n = 2$ agents.

After dealing with the structural matters by means of P/T PN, time was introduced into the final P/T PN. TPN arisen in such a way makes possible to study the effective processing jobs with respect to their duration in time. The agent idle time should be as short as possible. Its length can be changed during the simulation process. Although the optimal results can be obtained only at the classical scheduling computed by means of optimization methods (e.g. by mathematical programming), the approach combining the P/T PN-based supervision and the consecutive timing (by means of arising TPN) proposed here seems to be sound and satisfying for technical applications.

ACKNOWLEDGMENT

The work was partially supported by the Slovak Grant Agency for Science VEGA under grant # 2/0039/13 and by the Slovak Grant Agency APVV under grant APVV-0261-10. The author thanks both agencies for the support.

REFERENCES


Fig. 6. The courses of jobs $J_1$, $J_2$, $J_3$ represented, respectively, by $p_2$, $p_7$, $p_{12}$ processed by $A_1$ with respect to time as well as the course of the idleness of $A_1$ represented by $p_{16}$ with respect to time.


Fig. 7. The courses of jobs $J_1$, $J_2$, $J_3$ represented, respectively, by $p_4$, $p_9$, $p_{14}$ processed by $A_2$ with respect to time as well as the course of the idleness of $A_2$ represented by $p_{17}$ with respect to time.

