Annales Societatis Mathematicae Polonae
Series IV: Fundamenta Informaticae VI.2 (1983)

CONTROL OF PETRI NETS BY FINITE AUTOMATA

Hans-Dieter B u r k h a r d

Humboldt University of Berlin

Received June 15, 1982

AMS Categories: 68C30, 68D90

A b s t r a c t

Petri nets are considered where the firings are controlled by finite automata. The control may be distributed to different automata working over disjoint sets of transitions. To avoid deadlocks and conflicts for the whole system the distribution of control must be organized in an appropriate manner. The existence of deadlocks and conflicts is shown to be undecidable in general, but conflict resolving and deadlock free controls can be constructed for given nets.

Key words: Petri nets, automata, control of concurrent systems, conflict, deadlock.

Introduction

The paper considers control for concurrent processes, studied within the Petri net model. Control may be needed with respect to synchronization and conflict resolution. For Petri nets, control means control of transition firings and it is in some sense a restriction of the usual firing rule or a special firing strategy like firing in priority nets /5/, firing under occurrences of external events /7/, firing by maximum strategy /1/, /8/ or using queue regimes /1/.

In this paper a model of Petri nets under the control of finite deterministic automata is studied, where the automata determine the firing regime by giving permissions to the transitions. The control may be distributed to different control automata independently working over different parts of the net. Each automaton performs conflict resolution over its control area, but nevertheless there may be conflicts in the system resulting from the concurrent work for different parts of the system.

The behaviour of such systems is studied, especially the problems of conflicts and deadlocks. Conflicts are considered as conflicts with respect to concurrency. Deadlocks denote situations where no continuation of work is possible. The existence of deadlocks and conflicts in a system is shown to be undecidable, even for special classes of control. But, on the other hand, individual controls are constructable for the nets such that the resulting systems are conflict-free and deadlock-free, respectively.

The paper is divided into four sections:

1. Systems of controlled Petri nets and their behaviour,
2. Conflicts and deadlocks,
3. Undecidability results,
4. Construction of controls.

Throughout the paper $\mathbb{N}$ denotes the set of natural numbers, A^* is the set of finite sequences over a set A, whereby e denotes the empty word, P(A) is the power set of A and $P^+(A) := P(A) \setminus \{\emptyset\}$.

1. Systems of controlled Petri nets and their behaviour

In this section we define the system of Petri nets under the control of finite deterministic automata. The behaviour of such systems is defined in a way that allows the comparison with uncontrolled Petri nets.

A Petri net is given by

(1) $N = (P, T, \{t^- / t \in T\}, \{t^+ / t \in T\}, m_o)$

with finite disjoint sets P and T of places and transitions, respectively, and with the initial marking $m_o \in \mathbb{N}^P$. The vectors t^- and t^+ from $\mathbb{N}^P$ determine the change of markings by firings of transitions : A transition $t \in T$ is firable at the marking $m \in \mathbb{N}^P$ iff $t^- \leq m$, after its firing the new marking is given by $m + \Delta t$ with $\Delta t := t^+ - t^-$. Operations and relations over vectors are understood componentwise.

Alternatively a Petri net can be denoted by

(2) $N = (P, T, F, \mu, m_o)$

with the flow relation $F \subseteq (P \times T) \cup (T \times P)$ and the

multiplicity function $\mu : F \longrightarrow \mathbb{N}$, whereby

$$F = \{(p,t) \mid t^-(p) > 0\} \cup \{(t,p) \mid t^+(p) > 0\} ,$$

$$\mu(p,t) = t^-(p) \quad , \quad \mu(t,p) = t^+(p) \quad .$$

As usual, a firing sequence is a sequence of subsequently firable transitions. L_N is the set of all sequences firable in N starting from m_o. The reachability set is the set of all markings reachable from m_o :

$$(3) \qquad R_N := \{m_o + \Delta u \mid u \in L_N\} \qquad \text{whereby } \Delta t_1 \ldots t_n := \sum_{i=1}^{n} \Delta t_i)$$

Let U be a subset of T and let m be a marking. Then U is concurrently firable at m iff $U^- := \sum_{t \in U} t^- \leq m$, after the firing of U we have the new marking $m + \Delta U$ with $\Delta U := \sum_{t \in U} \Delta t$. We denote by

$$(4) \qquad U(m) := \{ t \mid t \in U \wedge t^- \leq m \}$$

the set of transitions from U firable at m and by

$$(5) \qquad \hat{U}(m) := \{ V \mid V \subseteq U \wedge V^- \leq m \} \setminus \{\emptyset\}$$

the set of all nonempty subsets of U that are concurrently firable at m.

We denote a control by C. It receives information from the net about the actually firable transitions and decides which transitions have to fire and which do not have to. To allow concurrent firings it must have information about concurrently firable transition sets, and hence it needs inputs like $\hat{T}(m)$ (cf. (5)), while T(m) would not be sufficient in general.

The control should be able to regard some history of firings (to realize alternating firings, for instance), hence we use the concept of automata.

The control may be distributed to distinguished units independently working over different parts of the net (local control). Hence we consider partitions $\mathcal{U} = \{U_1,\ldots,U_n\}$ of the transition set T, such that an individual control automaton A_i is working over each "control area" U_i.

(6) Definition

Let $N = (P, T, F, \underset{\sim}{\alpha}, m_o)$ be a Petri net.
A control for N (by finite deterministic automata) is given by

$$C := (\{U_1,\ldots,U_n\}, A_1,\ldots,A_n) ,$$

where $\{U_1,\ldots,U_n\}$ is a partition of T and each $A_i = (P^+(P^+(U_i)), P^+(U_i), Z_i, \sigma_i, \lambda_i, z_i^o)$, i=1,...,n, is a finite deterministic automaton with the input set $P^+(P^+(U_i))$ (to receive information $\hat{U}_i(m)$), the output set $P^+(U_i)$ (to allow several transitions to fire) the finite state set Z_i with the initial state z_i^o, the next state function $\sigma_i : Z_i \times P^+(P^+(U_i)) \longrightarrow Z_i$ and the output function $\lambda_i : Z_i \times P^+(P^+(U_i)) \longrightarrow P^+(U_i)$ satisfying the condition

(+) $\lambda_i(z,\mathcal{U}) \in \mathcal{U}$ for all $z \in Z_i$ and $\mathcal{U} \subseteq P^+(U_i)$

We call C a global control if $n = 1$ and a local control if $n > 1$.

Now a system of a Petri net under control by finite deterministic automata is denoted by $\gamma := (N, C)$, where C is a control for N as in definition (6). It works as an interactive system where an automaton A_i receives the information $U_i(m)$ about the possibilities for (concurrent) firings in its control area. Then it decides by $\lambda_i(z, \hat{U}_i(m))$ which transitions have to fire. Some time later, when these transitions have fired, the new information $\hat{U}_i(\bar{m})$ is given to the automaton, which makes a new decision. Thereby the new marking $\bar{m}$ depends on the actions in other parts of the net,

too.

We suppose that the computations of an automaton A_i and the firings in its control area U_i exclude each other, but the actions for different parts of the system may be concurrent. Nevertheless we can use firing sequences to express several aspects of the behaviour, as in the case of usual Petri nets they are useful as a simple description tool.

The following notation of a situation of a system permits us to define the behaviour by concatenations of simple actions (computations and firings). A situation s of a system $\mathcal{S} = (N, C)$ is given by

$$(7) \qquad s := (m, z_1, \ldots, z_n, V_1, \ldots, V_n) \quad \text{with } m \in \mathbb{N}^P,\ z_i \in Z_i,\ V_i \subseteq U_i \quad (i=1,\ldots,n).$$

It describes a snapshot in a moment of rest where m is the actual marking of the net N, the z_i are the actual states of the automata A_i, and the sets V_i denote those transitions which still have to fire according to the last decisions of the automata A_i. The initial situation is given by
$s_o := (m_o, z_1^o, \ldots, z_n^o, \emptyset, \ldots, \emptyset)$.
$S := \mathbb{N}^P \times Z_1 \times \ldots \times Z_n \times P(U_1) \times \ldots \times P(U_n)$ denotes the set of all situations.

Now we use an automata-like formalism (like for nets in /9/) to define changes of situations. We suppose that the sets $T, Z_1, \ldots, Z_n$ are pairwise disjoint and define $X := T \cup Z_1 \cup \ldots \cup Z_n$. Then we consider a partially defined function $\sigma : S \times X \longrightarrow S$, where we have for $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n) \in S$, $t \in T$, $z \in \bigcup_{i=1}^{n} Z_i$:

$$(8) \qquad \sigma(s,t) := \begin{cases} (m + \Delta t, z_1, \ldots, z_n, V_1, \ldots, V_{i-1}, V_i \setminus \{t\}, \\ \quad V_{i+1}, \ldots, V_n), \text{ if } t \in V_i \text{ and } t^- \leq m, \\ \text{not defined, otherwise} \end{cases}$$

(change by firing)

$$\sigma(s,z) := \begin{cases} (m, z_1, \ldots, z_{i-1}, \delta_i(z_i, \hat{U}_i(m)), z_{i+1}, \ldots, z_n, \\ V_1, \ldots, V_{i-1}, \lambda_i(z_i, \hat{U}_i(m)), V_{i+1}, \ldots, V_n) \\ \text{if } z = z_i,\ V_i = \emptyset \text{ and } \hat{U}_i(m) \neq \emptyset, \\ \text{not defined, otherwise} \end{cases}$$

(change by computation in A_i).

Thus, a change by firing may occur, if the transition has got permission by the last decision of its control device (and it has not fired after this time point) and if it is firable at the actual marking. For the moment we simply ignore not firable transitions. But in fact, there must have been a conflict if a transition from a set V_i is not firable (by (+) in definition (6)). A well-formed system should not cause such conflicts by generating accurate conflict resolutions (cf. definition (19)). - A change by computation may occur if all transitions from the last decision for this **area have fired and if there are firable transitions in this area at the actual marking.**

As defined above, the set X denotes the actions (firings and computations) of a system. Now we can describe the result of action sequences starting in the initial situation s_o by a function $\alpha: X^* \longrightarrow S$ as follows:

(9) $\alpha(e) := s_o$

$$\alpha(rx) := \begin{cases} \sigma(\alpha(r), x) \text{ for } r \in X^*, x \in X, \\ \text{if } \alpha(r),\ \sigma(\alpha(r), x) \text{ are defined,} \\ \text{not defined, otherwise.} \end{cases}$$

With the help of this function α we have:

(10) Definition

(1) $LA_{\gamma} := \text{dom}\,\alpha$ is the language of action sequences (over the alphabet X) of γ

(2) $L_{\gamma} := h(LA_{\gamma})$, where $h: X \longrightarrow T \cup \{e\}$ is a homomorphism with $h(t) := t$ for $t \in T$ and $h(z) := e$ for $z \in \bigcup_{i=1}^{n} Z_i$, is the language of firing sequences (over the alphabet T) of γ.

(3) $RS_{\gamma} := \text{cod}\,\alpha$ is the set of reachable situation of γ.

(4) $R_{\gamma} := \pi_1(RS)$, where π_1 denotes the projections to the first coordinates (the marking) of the situations. is the set of reachable markings of γ.

(11) Conclusions

For a system $\gamma = (N, C)$ we have:

(1) $R_{\gamma} = \{m_o + \Delta u \;/\; u \in L_{\gamma}\}$.

(2) $L_{\gamma} \subseteq L_N$, $R_{\gamma} \subseteq R_N$
(since control is a restriction of firing in N).

(3) If the partition $\tilde{r}_o = \{\{t\} \;/\; t \in T\}$ is used in C, then it holds:
$L_{\gamma} = L_N$, $R_{\gamma} = R_N$.

It is known for Petri nets that the languge L_N of firing sequences do not faithfully represent parallel work: Addition of a run place, for instance, suppresses all parallelism but does not change the language L_N. Special structures including partial orderings have been developed to describe concurrent firings, while certain aspects of parallelism can be represented by sequences over the alphabet $P^+(T)$. Hence it might be reasonable to consider the behaviour of systems under parallel work in such a way, too, while the use of process-like structures with partial orderings would too much extend the formalizm without a real need for this paper as we think. This is founded in some sense by proposition (16).

Parallelism of actions in our systems is reflected by the following definitions, which are developed in an analogous way as for firing sequences before. Now we consider instead of X the alphabet $\hat{X} := P(T) \times (Z_1 \cup \{0\}) \times \ldots \times (Z_n \cup \{0\}) \setminus \{(\emptyset, 0, \ldots, 0)\}$ $(0 \notin \bigcup_{i=1}^{n} Z_i)$ of parallel actions.

The partially defined function $\hat{\sigma}: S \times \hat{X} \longrightarrow S$ is given for $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n)$, $x = (V, \bar{z}_1, \ldots, \bar{z}_n)$ by

$$(12)\quad \hat{\sigma}(s,x) := (m + \Delta V,\ z'_1, \ldots, z'_n,\ V'_1, \ldots, V'_n)$$

with

$$z'_i := \begin{cases} \delta_i(z_i, \hat{U}_i(m)), & \text{if } \bar{z}_i = z_i,\ V_i = \emptyset,\ \hat{U}_i(m) \neq \emptyset \\ z_i, & \text{if } \bar{z}_i = 0 \end{cases}$$

$$V'_i := \begin{cases} \lambda_i(z_i, \hat{U}_i(m)), & \text{if } z_i = \bar{z}_i,\ V_i = \emptyset,\ \hat{U}_i(m) \neq \emptyset \\ V_i \setminus V, & \text{otherwise} \end{cases}$$

if $V \subseteq \bigcup_{i=1}^{n} V_i$, $V^- \subseteq m$, $\bar{z}_i \in \{0, z_i\}$, otherwise $\hat{\sigma}(s,x)$ is not defined.

Furthermore we define (cf. (9), (10)):

$$(13)\quad \hat{\alpha}(e) := s_0$$

$$\hat{\alpha}(rx) := \begin{cases} \hat{\sigma}(\hat{\alpha}(r), x) \text{ for } r \in X^*,\ x \in X, \\ \text{if } \hat{\alpha}(r),\ \hat{\sigma}(\hat{\alpha}(r), x) \text{ are defined,} \\ \text{not defined, otherwise.} \end{cases}$$

$$(14)\quad (1)\quad \hat{LA}_\pi := \operatorname{dom} \hat{\alpha}$$

$$(2)\quad \hat{L}_\pi := h(\hat{LA}_\pi), \text{ where } h: \hat{X} \longrightarrow P^+(T) \cup \{e\}$$

is a homomorphism with

$$h((V, z_1, \ldots, z_n)) := \begin{cases} V, & \text{if } V \neq \emptyset \\ e, & \text{otherwise} \end{cases}$$

$$(3)\quad \hat{RS}_\pi := \operatorname{cod} \hat{\alpha}$$

(4) $\hat{R}_\gamma := \overline{\iota\iota}_1(\hat{RS}_\gamma)$

(15) Conclusions

(1) $\hat{RS}_\gamma = RS_\gamma \qquad \hat{R}_\gamma = R_\gamma$

(2) $\{t_1\}\{t_2\}\ldots\{t_k\} \in \hat{L}_\gamma$ iff $t_1 t_2 \ldots t_k \in L_\gamma$.

The following proposition shows that the mentioned differences with respect to parallelism are caused by the nets and not by control:

(16) Proposition

Let N be a Petri net and let C, C' be controls for N. Then we have

$L_{(N,C)} = L_{(N,C')}$ iff $\hat{L}_{(N,C)} = \hat{L}_{(N,C')}$.

Proof: The if-part follows by (15.2). The only-if-part is shown by induction for words from $(P^+(T))^*$. For the induction step we suppose $W_1 \ldots W_k W_{k+1} \in \hat{L}_{(N,C)}$. Then there exists a sequence $u \in L_{(N,C)}$ where u is a "sequentialization" of $W_1 \ldots W_k$ (i.e., $u = u_1 \ldots u_k$ and each u_i is a word built up from the transitions from W_i). Furthermore for all $t \in W_{k+1}$ we have $ut \in L_{(N,C)}$ and hence $ut \in L_{(N,C')}$ by $L_{(N,C)} = L_{(N,C')}$.

Now $W_1 \ldots W_k$ is in $\hat{L}_{(N,C')}$ by induction and in both systems we reach the marking $m_o + \Delta u$ after the firing of $W_1 \ldots W_k$. Then all transitions $t \in W_{k+1}$ may be allowed to fire by control in the system (N,C') since all ut are in $L_{(N,C')}$. We have $W_{k+1}^- \leq m_o + \Delta u$, since W_{k+1} is firable at $m_o + \Delta u$ in (N,C) and hence it is also firable in (N,C') and thus we get $W_1 \ldots W_{k+1} \in \hat{L}_{(N,C')}$.

The results (15) and (16) justify the use of firing sequences also with respect to concurrent firings in this paper. Thus the equivalence of controls is defined as follows:

(17) Definition

Let N be a Petri net and let C, C' be controls for N.

C and C' are equivalent with respect to N ($C \underset{N}{\sim} C'$) iff $L_{(N,C)} = L_{(N,C')}$.

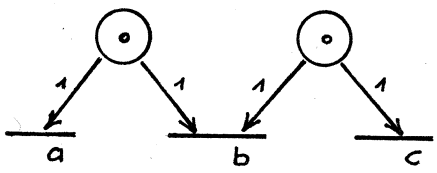
(18) Conclusion

If $C \underset{N}{\sim} C'$, then we have $R_{(N,C)} = R_{(N,C')}$. (By (11.1))

2. Conflicts and deadlocks

By (+) in definition (6) we have ensured that only firable transitions are chosen by the control. But such chosen transitions may become not firable by firings of other transitions not belonging to the same control area, as in the following example:

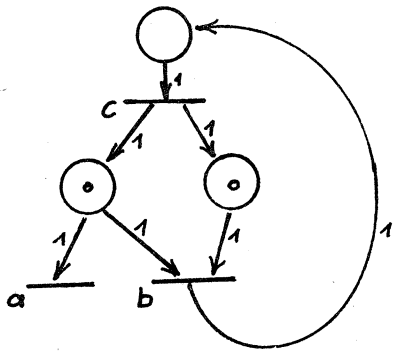
Example 1



If we use the partition $\{\{a,b\},\{c\}\}$ and A_1 decides for b while A_2 decides for c then the order of A_2 can not be satisfied after the firing of b. The reason is a conflict between the transitions b and c for which that control has given no resolution.

As in usual Petri nets, it os possible that no transition is firable at a certain marking, i.e $T(m) = \emptyset$. But it may also happen that there are firable transitions which can not be directed for firing since the control is blocked by other not firable transitions, as in the following example:

Example 2



We use the partition $\{\{a,c\},\{b\}\}$ and suppose that decisions are made for a by A_1 and for b by A_2. After the firing of b, transition a is not firable any more, but transition c becomes firable. Nevertheless, c is not allowed to fire under the control because we still have $V_1 = \{a\} \neq \emptyset$ and thus A_1 is not in the position for a new computation in favour of c. Such effects are reflected by the following definitions.

(19) Definition

Let $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n)$ be a situation of a system $\gamma = ((P, T, F, \ell, m_o), (\{U_1, \ldots, U_n\}, A_1, \ldots, A_n))$.

(1) γ has a conflict in s iff $(\bigcup_{i=1}^{n} V_i)^- \not\leq m$.

(2) γ has a deadlock in s iff $T(m) \cap \bigcup_{i=1}^{n} V_i = \emptyset$ and $T(m) \subseteq \bigcup_{V_i \neq \emptyset} U_i$.

(3) γ is called conflict-free (deadlock-free) iff there are no conflicts (deadlocks) in the situations $s \in RS_\gamma$.

Note that this definition of conflict-free systems is related to concurrency (again $(\bigcup_{i=1}^{n} V_i)^- \leqq m$ is stronger than $t^- \leqq m$ for all $t \in \bigcup_{i=1}^{n} V_i$).

(20) Proposition

Let $\mathfrak{N}$ be a conflict-free system and let $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n) \in RS\,\mathfrak{N}$.

Then $\mathfrak{N}$ has a deadlock in s iff $T(m) = \emptyset$.

Proof: If there is a deadlock in s, where we have $T(m) \neq \emptyset$, then we have a nonempty set V_i by $T(m) \subseteq \bigcup_{V_i \neq \emptyset} U_i$. Now, by $\bigcup_{i=1}^{n} V_i \cap T(m) = \emptyset$ it follows that $(\bigcup_{i=1}^{n} V_i)^- \not\leqq m$ and thus we have a conflict in s, which contradicts our presumption. -The other part of the proof is trivial.

(21) Proposition

Systems of Petri nets under global control are conflict-free.

Proof: Global control means n = 1, thus we have only the control automaton A_1 and U_1 coincides with the whole transition set T. For the decisions of A_1 we have by (5) and (+) in (6):

$$W = \lambda_1(z, \hat{U}_1(m)) \in \hat{U}_1(m) = \{V \mid V \subseteq U_1 \wedge V^- \leqq m\} \setminus \{\emptyset\},$$

hence $W^- \leqq m$. Up to the next decision of A_1, subsequent firings of transitions have to concern subsets $W' \subseteq W$. Since it follows from $W^- \leqq m$ and $W' \subseteq W$ that $(W \setminus W')^- \leqq m + \Delta W'$, conflicts are impossible.

Proposition (21) shows that conflicts are caused by unsuitable distributions of control. We shall study the possibilities of suitably distributed controls in section 4.

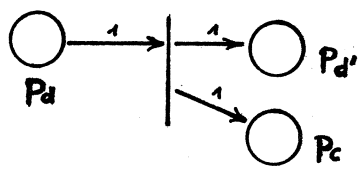
3. Undecidability results

The automata A_i given in defintion (6) may have useless states, inputs and outputs. For instance, the necessary states are those states which occur in the action sequences of LA For simplicity we shall sometimes use simplified automata in the following.

By control we can realize priority firings in Petri nets using automata with $\lambda_i(z, \hat{U}_i(m)) = \{t\}$, where t has highest priority in $U_i(m)$. As it was shown by HACK /5/, priority nets can simulate deterministic counter machines. Following his ideas, we can simulate such machines by controlled Petri nets, too.

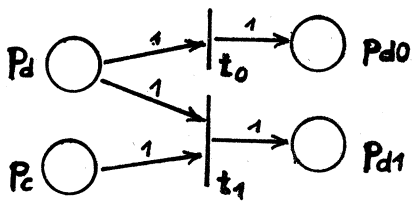
The states d of a deterministic counter machine $\mathcal{M}$ are simulated by places p_d, where p_d is marked by one token iff $\mathcal{M}$ is in the state d. The counters c of $\mathcal{M}$ are simulated by places p_c, where the contents of c are given by the number of tokens in p_c.

An addition assignment "d: c:= c+1; go to d';" is simulated by



and a simple control automaton A_i with only one state z and $\lambda_i(z, \{\{t\}\}) = \{t\}$.

A testassignment "d: if c = 0 then goto d0
else c:= c-1; goto d1;"
is simulated by



and a control automaton A_i with one state z and

$\lambda_i(z, \{\{t_o\}\}) = \{t_o\}$, $\lambda_i(z, \{\{t_o, t_1\}, \{t_o\}, \{t_1\}\}) = \{t_1\}$.

Note that the control tasks can be performed by different automata for each simulation of an assignment (local control) and also by only one common automaton with the state z (global control).

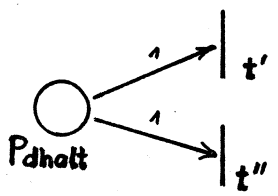
It is not decidable whether a deterministic counter machine can reach the stop state d_{halt} from its start state d_{start} (whereby the counter contents are zero in the beginning). Hence, by the given simulation it is undecidable whether a token may reach the place p_{dhalt} in our system from the initial marking m_o with $m_o(p_{dstart}) = 1$ and $m_o(p) = 0$ for all $p \neq p_{dstart}$.

Since d_{halt} is the stop state of $\mathcal{M}$, no transition takes a token from p_{dhalt} in the net of our constructed system. We can propose that there are no other stop states in $\mathcal{M}$, and thus we have a deadlock in our system iff the machine $\mathcal{M}$ halts. We get:

(22) Proposition

It is undecidable whether a system $\mathcal{S}$ is deadlock-free.

Now we can add two new transition to the constructed net at the place p_{dhalt}:



If we use different control automata for t' and t'', then we may reach a conflict in the new system iff the deterministic counter machine $\mathcal{M}$ halts. Thus we have

(23) Proposition

It is undecidable whether a system $\mathcal{S}$ is conflict-free.

Using appropriate additional constructions at the place p_{dhalt} it can also be shown:

(24) Corollary

The following problems are undecidable:

(1) $C \tilde{N} C'$?

(2) $s \in RS_{\mathcal{S}}$?

(3) $m \in R_{\mathcal{S}}$? (Reachability problem)

(4) $R_{\mathcal{S}}$ finite ? (Boundedness problem)

Several other problems which can be formulated for systems following the definitions for Petri nets like the boundedness problem for special places or the liveness problem for transitions, are undecidable, too. The proofs are carried out by additional net constructions, again.

Since, for instance, the boundedness problem is decidable for uncontrolled nets, we have a more complicated behaviour of the systems by their restrictions of firings. By the same methods as in /1/ one can prove that there are systems $\mathcal{S}$ such that $L_{\mathcal{S}} \neq L_N$ for all Petri nets N. Thus we have:

(25) Corollary

The behaviour of systems can in general not faithfully be simulated by uncontrolled Petri nets.

As mentioned above, the simulation of deterministic counter machines can even be performed with the help of only global controls. Therefore as long as the additional constructions are valid under global control, we have the undecidability results even for only globally controlled systems. This is the case with respect to (22) and (24), but not for (23) (cf. (21)).

The undecidability results for systems are not surprising since

by control we can easily simulate priority-firings as well as several other firings strategies for which it is known that they permit simulation of deterministic counter machines by Petri nets. (cf. /1/, /5/, /8/).

Hence we have to ask for special firing strategies and special classes of controls, respectively, which do not allow such simulations. Thereby, a class of controls has to permit the control of arbitrary Petri nets, i.e., for each Petri net we need at least one control in this class corresponding to the transition set of the net.

A first -but trivial- answer is given by (11.3.). The next proposition shows that all reasonable (i.e. conflict-free) control strategies extend the computational power of Petri nets in the sense of a simulation of deterministic counter machines. Thereby the notion of simulation is not so strong as before, but it is faithful enough to use the undecidability of the halting problem again. It will become clear during the corresponding construction.

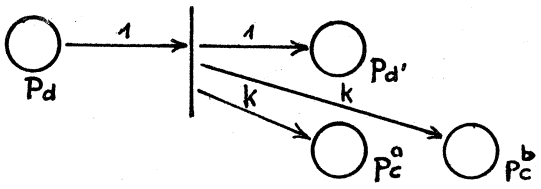
By a Petri net structure we denote a Petri net without the multiplicity function μ. The definition (6) of a control C does not depend on μ, hence we can say that C is a control for a Petri net strucure $\bar{N} = (P,T,F,.,m_o)$ if it is a control for the net $N = (P,T,F,\mu,m_o)$ with an arbitrary multiplicity function μ.

(26) Proposition

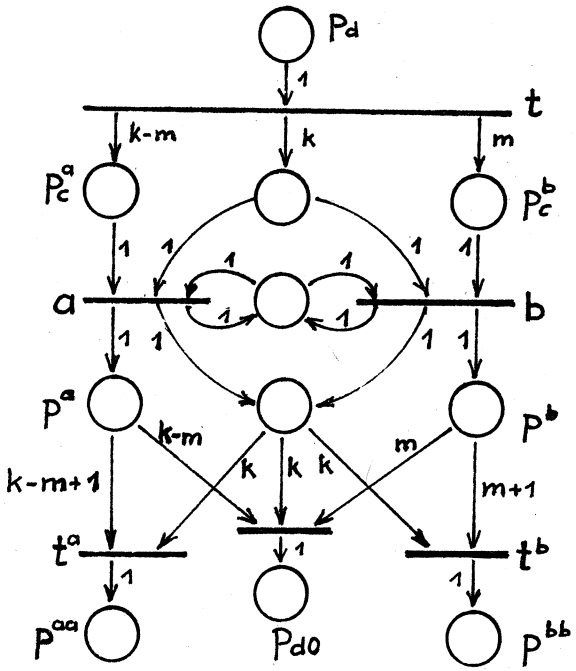
For each deterministic counter machine $\mathcal{M}$
there exists a Petri net structure $\bar{N} = (P,T,F,.,m_o)$
such that for each control C for $\bar{N}$
there exists a multiplicity function μ
such that the following holds for $N = (P,T,F,\mu,m_o)$:
If (N,C) is conflict-free, then (N,C) simulates $\mathcal{M}$.

Proof: The difficult part of simulation is the simulation of the test assignment. The idea is to use the fact that automata output becomes cyclic if the inputs are equal over a long time. The counters are now simulated by two places p_c^a and p_c^b for each

counter c, and the contents x in c are simulated by k•x tokens in both places, where k is a number to be specified depending on the control. Therefore the addition assigment is simultated by



In the center of test simulation we have the following net (structure):



k, m $(k > m)$ are to be specified depending on the control. The place p_{d0} will be marked iff transition a fires exactly k-m times and d fires exactly m times (the sum of their firings during one test run is always k). This has to be the case if p_c^a and p_c^b were clean (i. e., the contents of c were zero) before starting the test run by firing t.

We have to ensure that p_{d0} cannot be marked otherwise, i.e., if the contents of c were greater than zero. In this case we have at least k tokens in each of the two places p_c^a and p_c^b. Since we propose the constructed system to be conflict-free the transitions a and b have to be controlled by a common automaten A_i which in each computation may allow only one of the two transitions to fire. By the construction, the automaton A_i receives the input $\{\{a\}\{b\}\}$ for k times thereby producing an output $\lambda_i(z, \{\{a\},\{b\}\}^k) = W_1 \ldots W_k$ whereby $z \in Z_i$ (no assumption can be made about z) and $W_l \in \{\{a\},\{b\}\}$ for $l = 1, \ldots, k$.

Now, if k is large enough, the sequnece $W_1 \ldots W_k$ becomes periodical (since the state changes have to become cyclic because A_i is finite). Hence for $m = \text{card}(Z_i)$ and $k = m^2 + 2m$ we have two possibilities:

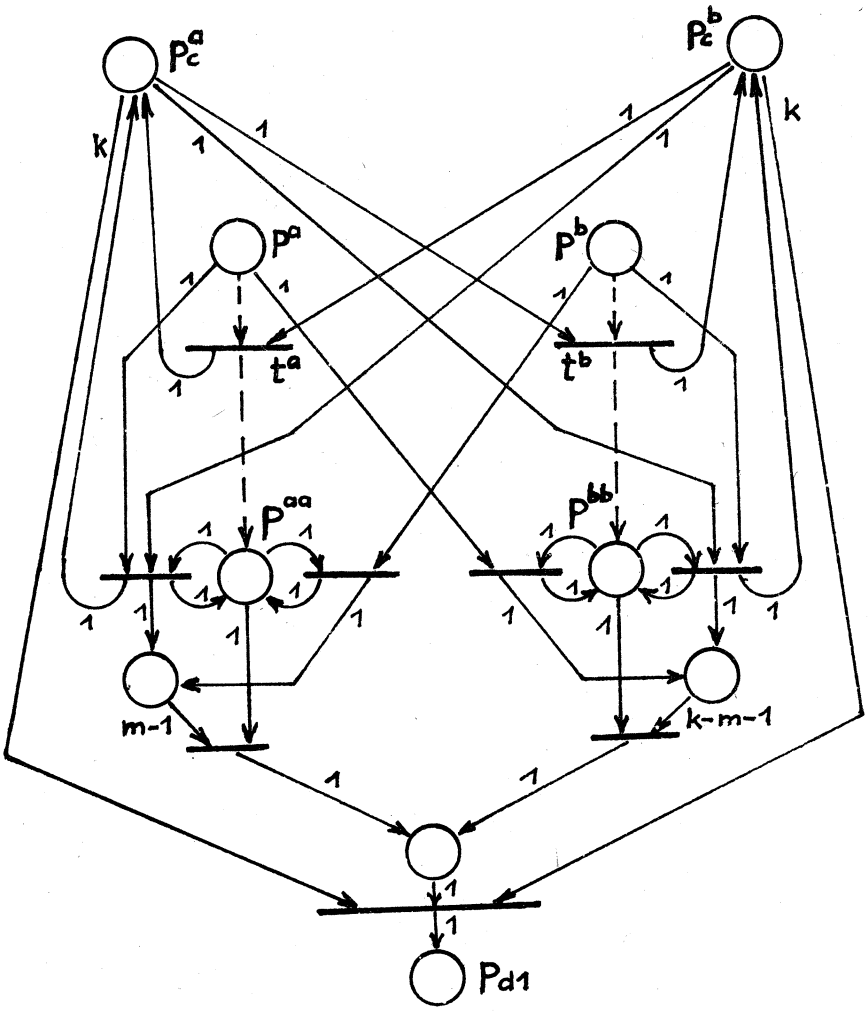
1. $\{b\}$ occurs more than m times in $W_1 \ldots W_k$ (if it occurs in the period which can not be longer than m).

2. $\{b\}$ occurs less than m times in $W_1 \ldots W_k$ (if it does not occur in the period but perhaps in the initially not periodical sequence $W_1 \ldots W_l$ of $W_1 \ldots W_k$, where we have to have $l < m$).

Regarding that the sum of firings of a and b is k, we obtain: In the first case the place p^{bb} is marked at the end, and in the second case p^{aa} is marked at the end, while it is impossible to mark p_{db}.

Thus we have an accurate distinction for testing.

It is necessary now to initialize the test construction for further tests and to realize that the markings on the places p_c^a and p_c^b are both decreased by exactly k tokens with respect to their marking before the test was started. For this correction our construction is completed as follows (note that no correction is needed if the counter contents were zero):



The verification is left to the reader.

By the given constructions we can now simulate a deterministic counter machine $\mathcal{M}$ in such a way that $\mathcal{M}$ reaches its state d_{halt} from its state d_{start} (with counter contents initially zero) if and only if a token can reach the place p_{dhalt} in our system where the initial marking is given by $m_o(p_{dstart}) = 1$ and $m_o(p) = 0$ for all places $p \neq p_{dstart}$. Thereby the net structure depends only on $\mathcal{M}$, while the multiplicity function of the net depends on the numbers of states in the controlling automata. The only presumptions about the control concern the ability to control the net (It is a demand about the inputs and outputs of the automata) and the avoidance of conflicts (which is only needed to ensure that for each test-construction the transitions a and b are controlled by common automata)

By the given simulation we can now transfer the undecidability of the halting problem to the undecidability of reaching a token at the place p_{dhalt} for conflict free systems built up from arbitrary Petri nets and controls from only special classes. The consequences for the undecidability of system properties follow as discussed for the general case. For instance if we have such a class of controls which permits to build conflict-free and not conflict-free systems we can prove the undecidability whether systems are conflict-free or not in the following way: We construct conflict-free systems simulating deterministic counter machines as described above and add the construction given to prove proposition (23). The further details are left to the reader.

The result (26) shows that the firing rule of Petri nets is very sensitive with respect to (regular) restrictions. They may immediately increase the computational power of the nets and lead to undecidability results. This should be taken into account for solutions of the reachability problem, where a stop criterion might sometimes be interpreted as a restriction of the firing rule. It must not lead to the simulation of deterministic counter machines and therewith make a solution impossible.

4. Construction of controls

By the results (22) - (26) it is not possible to decide whether systems are conflict-free and deadlock-free, respectively. This may be the case even if we restrict ourselves to special classes of controls (special firing strategies) as it was shown by (26).

But, as we shall see, it is possible to construct individual controls for single nets such that the resulting systems are conflict-free and deadlock-free, respectively. Thereby the controls to be constructed will lead to a deadlock-free (conflict-free) system together with the considered net, while they may permit simulation of deterministic counter machines together with other nets. Hence it will not contradict the result (26).

(27) Proposition

Let N be a Petri net with an infinite firing language L_N (otherwise a deadlock-free system with N does not exist) Then it is possible to construct a global control C such that (N,C) is deadlock-free

Proof: We use the pumping lemma for firing languages /2/:
There are numbers k, l for each language L_N such that the following holds: If the length of a sequence $u \in L_N$ is greater than k, then there exists a decomposition $u = u_1u_2u_3$ such that $1 \leqq$ length of $u_2 \leqq l$ and

$u_1u_2^{n+1}u_3 \in L_N$ for all $n \in \mathbb{N}$.

Hence it is possible to fire u_1 and then to iterate the firings u_2. Such sequences u_1, u_2 can be found (that follows from the proof in /2/). Now the global control has to realize at first the firing of u_1 and than the iterated firing of u_2. An automaton for such a control can easily be constructed.

By (21) we know that global controls are conflict-free by definition. The question arises if such solutions can be applied to local controls. The answer has be a negative one for the general case. Example 1 has shown that local control may give rise to conflicts, while global control does not. In the same sense a globally controlled system iterating the firing of bc in example 2 would be deadlock-free while a locally controlled system as described there is not deadlock-free. The reason is in some sense the extension of nondeterminism by refinements of the partitions of T into control areas We have:

(28) Proposition

Let N be a Petri net and let $C = (\{U_1, \ldots, U_n\}, A_1, \ldots, A_n)$ and $C' = (\{U_1 \cup U_2, U_3, \ldots, U_n\}, A_{1/2}, A_3, \ldots, A_n)$ be controls for N, whereby $A_i = (P^+(P^+(U_i)), P^+(U_i), Z_i, \mathcal{S}_i, \lambda_i, z_i^o)$, $i = 1, \ldots, n$, and
$A_{1/2} = (P^+(P^+(U_1 \cup U_2)), P^+(U_1 \cup U_2), Z_1 \times Z_2, \mathcal{S}_{1/2}, \lambda_{1/2}, (z_1^o, z_2^o))$
with $\mathcal{S}_{1/2}((z_1, z_2), \mathcal{U}) = (\mathcal{S}_1(z_1, \mathcal{U} \cap P^+(U_1)), \mathcal{S}_2(z_2, \mathcal{U} \cap P^+(U_2)))$
and $\lambda_{1/2}((z_1, z_2), \mathcal{U}) = \lambda_1(z_1, \mathcal{U} \cap P^+(U_1)) \cup \lambda_2(z_2, \mathcal{U} \cap P^+(U_2)))$.

Then we have $L_{(N,C)} \supseteq L_{(N,C')}$ and $R_{(N,C)} \supseteq R_{(N,C')}$.

The idea of the proof is to show that all situations
$s = (m, (z_1, z_2), z_3, \ldots, z_n, V_{1/2}, V_3, \ldots, V_n) \in RS_{(N,C')}$ and their changes according to $\hat{\mathfrak{G}}$ can be simulated by a situation
$s = (m, z_1, z_2, z_3, \ldots, z_n, V_{1/2} \cap U_1, V_{1/2} \cap U_2, V_3, \ldots, V_n) \in RS_{(N,C)}$

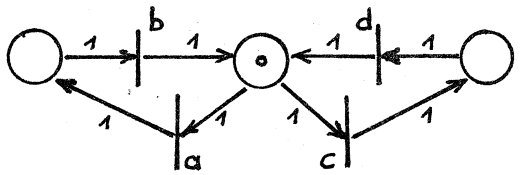
and related changes whereby the firings are the same. The loss of possibilities in (N,C') is caused by the synchronization of the actions over U_1 and U_2. This may lead to the avoidance of conflicts and deadlocks. Furthermore, by the simulation of situations as above, it can be shown:

(29) Corollary

(1) If (N,C) is conflict-free, then (N,C') is conflict-free.
(2) If both systems are conflict-free and (N,C) is deadlock-free, then (N,C') is deadlock-free.

If the systems are not conflict-free, then (N,C) might be deadlock-free while (N,C') is not deadlock-free as in the following example:

Example 3:



With $U_1 = \{a,b\}$, $U_2 = \{b,d\}$ we have deadlock-free systems (N,C) (but not conflict-free), while for systems (N,C') constructed from (N,C) as in proposition (28) with $U_{1/2} = \{a,b,c,d\}$ we have to have deadlocks: The first decision has to be $\{a,c\}$ (according to the decisions in (N,C)), and than we have a blocking of new decisions by waiting for c after firing of a.

By defintion (6) the control automata A_i have to perform conflict resolutions over their areas U_i since (+) ensures $(\lambda_i(z,\hat{U}_i(m)))^- \leq m$, while we might have $(U_i(m))^- \not\leq m$.

Nevertheless conflicts in locally controlled systems may arise from decisions of different control devices As it was pointed out for propositon (21), we need an appropriate distri-

bution of controls to have conflict-free systems. Therefore we use the concept of conflict-partitions developed in /4/ for purposes of control with local conflict resolutions and global synchronization under limitations of parallelity.

(30) Definition

Let $N = (P,T,F,\mu,m_0)$ be a Petri net, $m \in \mathbb{N}^P$ and $\mathcal{T}$ a partition of T.

(1) The set of basic conflicts at m is given by
$BC(m) := \{ U/U \subseteq T(m) \wedge U^- \not\leq m \wedge \forall V \subseteq U: V^- \leq m \}$.

(2) The set of conflict-free transition sets at m is given by
$CF(m) := \{ U/U \subseteq T(m) \wedge U^- \leq m \}$.

(3) The set of local conflict resolution functions for $\mathcal{T}$ at m is given by
$LCR(\mathcal{T},m) := \{ \varphi / \varphi: \mathcal{T} \longrightarrow P(T(m)) \wedge \wedge \forall U \in \mathcal{T}: (\varphi(U) \subseteq U \wedge \varphi(U) \in CF(m)) \}$.

(4) The set of conflict partitions at m is given by
$\Pi(m) := \{ \mathcal{T} / \mathcal{T}$ a partition of $T \wedge \wedge \forall \varphi \in LCR(\mathcal{T},m): \bigcup_{U \in \mathcal{T}} \varphi(U) \in CF(m) \}$

Thereby a conflict resolution function φ represents the possibilities for firings in a situation $s = (m,z_1,\ldots,z_n,V_1,\ldots,V_n)$ (cf. (7)) of a system where for each control area U_i we have a conflict free transition set $V_i = \varphi(U_i) \in CF(m)$. Now a conflict partition $\mathcal{T} = \{U_1,\ldots,U_n\} \in \Pi(m)$ ensures by $\bigcup_{i=1}^{n} \varphi(U_i) = \bigcup_{i=1}^{n} V_i \in CF(m)$ that we have no conflict at s.

For $\mathcal{U}_1, \mathcal{U}_2 \subseteq P^+(T)$ we define that $\mathcal{U}_2$ is a covering of $\mathcal{U}_1$ as usual:

(31) $\mathcal{U}_1 \leq \mathcal{U}_2$ iff $\forall U_1 \in \mathcal{U}_1 \exists U_2 \in \mathcal{U}_2: U_1 \subseteq U_2$.

We shall use the following notations:

(32) For $\mathcal{U} \subseteq P^+(T)$ we denote by $\nu_{\mathcal{U}} \subseteq T \times T$ the binary relation

with $(t,t') \in v_{\mathcal{U}}$ iff $t=t' \vee \exists U \in \mathcal{U}: t,t' \in U$.
By $v^*_{\mathcal{U}}$ we denote the transitive closure of $v_{\mathcal{U}}$ and by $\mathcal{T}_{\mathcal{U}} := T/_{v^*_{\mathcal{U}}}$ the partition of T by the equivalence relation $v^*_{\mathcal{U}}$

The following lemma is well-known:

(33) For $\mathcal{U} \subseteq P^+(T)$ and partitons $\mathcal{T}$ of T we have:
$$\mathcal{T} \geq \mathcal{U} \text{ iff } \mathcal{T} \geq \mathcal{T}_{\mathcal{U}}$$

Now we can characterize the conflict partitons in $\Pi(m)$:

(34) Proposition

(1) $\mathcal{T} \in \Pi(m)$ iff $\mathcal{T} \geq BC(m)$.

(2) $\Pi(m) = \{\mathcal{T} / \mathcal{T} \geq \mathcal{T}_{BC(m)}\}$

Proof: We have $\mathcal{T} \in \Pi(m)$ iff there exists a function $\mathcal{S} \in LCR(\mathcal{T}, m)$ with $\bigcup_{U \in \mathcal{T}} \mathcal{S}(U) \notin CF(m)$. This is the case iff we have $V \subseteq \bigcup_{U \in \mathcal{T}} \mathcal{S}(m)$ with $V \in BC(m)$ $V \nsubseteq \mathcal{S}(U)$ and $V \not\subseteq U$ for all $U \in \mathcal{T}$, i.e., iff $\mathcal{T} \not\geq BC(m)$. - (34.2) follows from (34.1) and lemma (33).

Thus, $\mathcal{T}_{BC(m)}$ is the minimum in $\Pi(m)$ and each enlargement of $\mathcal{T}_{BC(m)}$ is a conflict partition. Note that $v_{BC(m)}$ is in general not transitive (cf. example 1).

Since conflicts must be avoided for all reachable situations of a system we need partitions of T which are conflict partitions for different markings.

(35) Defintion

Let $\mathcal{T}$ be a partition of T and $M \subseteq \mathbb{N}^P$
The set of conflict partitons for M is given by
$\Pi(M) := \{\mathcal{T} / \forall m \in M: \mathcal{T} \in \Pi(m)\}$

(36) Conclusion

If $M \subseteq M'$, then $\overline{\pi}(M') \leq \overline{\pi}(M)$.

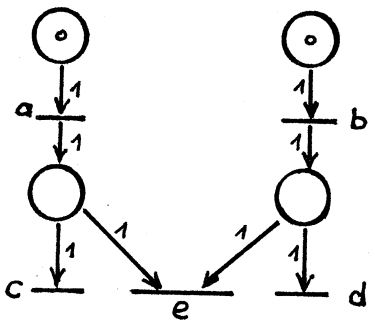
Using (34.1) we find that $\mathcal{T} \in \overline{\pi}(M)$ iff $\mathcal{T} \geq \bigcup_{m \in M} BC(m)$ and then we have by (33) and $BC(M) := \bigcup_{m \in M} BC(m)$:

(37) Proposition

$$\overline{\pi}(M) = \{ \mathcal{T} / \mathcal{T} \geq \mathcal{T}_{BC(M)} \} .$$

Now the problem is to determine the set M to be used. To obtain finer partitions, **M should be as small** as possible (36). Starting with a conflict-free system $\mathcal{S}$ one could think of the set $R_{\mathcal{S}}$ to get a finer partition than in $\mathcal{S}$. But in a new system $\mathcal{S}'$ over a partion from $\overline{\pi}(R_{\mathcal{S}})$ we may have a larger set $R_{\mathcal{S}'}$ (by allowing more nondeterminism, cf. (28)), and thus $\mathcal{S}'$ might not be conflict-free as in example 4:

Example 4



If we use a global control realizing the firing of acbd we get a set $R_{\mathcal{S}}$ with $\mathcal{T}_0 = \{\{a\},\{b\},\{c\},\{d\},\{e\}\} \in \overline{\pi}(R_{\mathcal{S}})$. But a locally controlled system $\mathcal{S}'$ over $\mathcal{T}_0$ is not conflict-free since we have $R_{\mathcal{S}'} = R_N$ by (11.3). - What we can show is the following:

(38) Proposition

Let $\gamma = (N,C)$ be a system where $\mathcal{T}$ is the partition used in C. Then it holds:
If $\mathcal{T} \in \overline{\Pi}(R_\gamma)$, then γ is conflict-free.

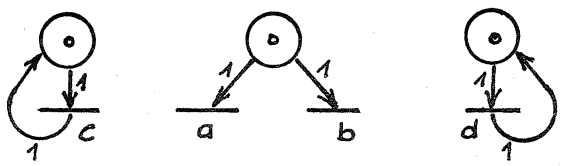
Prof: We have to show $(\bigcup_{i=1}^{n} V_i)^- \leqq m$ for all situations $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n) \in RS_\gamma$. The proof is by induction on the sequences in LA_γ, whereby the basic step is trivial. For the induction step we have to consider changes by firings and by computations, respectively.

We suppose $(\bigcup_{i=1}^{n} V_i)^- \leqq m$ for $s = (m, z_1, \ldots, z_n, V_1, \ldots, V_n) \in RS_\gamma$
After the firing of $t \in \bigcup_{i=1}^{n} V_i$ we have
$(\bigcup_{i=1}^{n} V_i \setminus \{t\})^- \leqq m + \Delta t$, i.e., no conflict appears after change by firing.

A change by computation leads from a to a new situtation $s' = (m, z_1, \ldots, z_{j-1}, z_j', z_{j+1}, \ldots, z_n, V_1, \ldots, V_{j-1}, V_j', V_{j+1}, \ldots, V_n)$.
We have $V_i \leqq m$ $(i=1 \ldots n)$ by $(\bigcup_{i=1}^{n} V_i)^- \leqq m$ and $(V_j')^- = (\lambda_j(z_j, \hat{U}_j(m)))^- \leqq m$ by (+) in definition (6) i.e., $V_1, \ldots, V_n, V_j' \in CF(m)$ and thus by definition (30) we have $(\bigcup_{i=1}^{n} V_i \setminus V_j) \cup V_j' \in CF(m)$ since $\mathcal{T} \in \Pi(R_\gamma) \leqq \Pi(m)$. Hence we have no conflict at s'.

Note that $\mathcal{T} \in \Pi(R_\gamma)$ is sufficient but not necessary to have a conflict-free system since the conflicting transitions may be ignored by control as in the following example:

Example 5



For $U_1 = \{a,c\}$ and $U_2 = \{b,d\}$ we use the automata A_1, A_2 giving permission for firing only to c and d, respectively. Then the system $\mathcal{Y}$ is conflict-free but $\{U_1,U_2\} \notin \Pi(R_{\mathcal{Y}})$.

Now, $\Pi(R_{\mathcal{Y}})$ depends on $\mathcal{Y} = (N,C)$ and especially on the partition $\mathcal{Z}$ used in C. Thus, to verify $\mathcal{Z}$ we had to construct and to analize $\mathcal{Y}$ before. Moreover, the set $R_{\mathcal{Y}}$ may be not recursive by (24). But by (37) all partitions $\mathcal{Z} \geq \mathcal{Z}_{BC(R_{\mathcal{Y}})}$ are suitable and by (36) all partions $\mathcal{Z} \in \Pi(M)$ for $M \supseteq R_{\mathcal{Y}}$. Thus, for instance, it is possible to use $\Pi(R_N)$ but then we are in the neighbourhood of the reachability problem and it can be proved that the problem to decide $v_{BC(R_N)}$ is equivalent to the reachability problem.

The partitions from $\Pi(\mathbb{N}^P)$ always have to be useful. They are in general not the finest useful partitions but they can easily be constructed from the net structure since it holds:

(39) Proposition

Let $N = (P,T,F,\mu,m_o)$ be a Petri net and let $v_p^* \subseteq T \times T$ be the transitive closure of the relation $v_p \subseteq T \times T$ with $(t,t') \in v_p$ iff $t = t' \vee Ft \cap Ft' \neq \emptyset$. Then it holds:

$$\mathcal{Z}_{BC(\mathbb{N}^P)} = T/_{v_p^*}$$

The proof is by showing $v_p \subseteq v_{BC(\mathbb{N}^P)}$ and $v_{BC(\mathbb{N}^P)} \subseteq v_p^*$ whereby only transitions t_1,t_2 with $t_1 \neq t_2$ have to be regarded. For $(t_1,t_2) \in v_p$ we have $(t_1,t_2) \in v_{BC(m)} \subseteq v_{BC(\mathbb{N}^P)}$ for the markings m with $m(p) = \text{Max}\{t_1^-(p), t_2^-(p)\}$, $p \in P$.

For the case $(t_1,t_2) \in v_{BC(\mathbb{N}^P)}$ there have to exist $m \in \mathbb{N}^P$, $U \subsetneq T$ such that $t_1,t_2 \in U \in BC(m)$. If there existed transitions $t,t' \in U$ with $(t,t') \notin v_p^*$ we would have $U = U_1 \cup U_2$, $U_1,U_2 \neq \emptyset$, $FU_1 \cap FU_2 = \emptyset$. Then $U_1,U_2 \notin BC(m)$ would hold by definition (30) and hence $U_1^-, U_2^- \not\leq m$. This would imply $U^- = U_1^- + U_2^- \not\leq m$ (since $FU_1 \cap FU_2 = \emptyset$) in contradiction to $U \in BC(m)$. Thus, for all $t,t' \in U$ (and especially for t_1,t_2) we have $(t,t') \in v_p^*$.

The partition $T/_{v_p}^*$ is always useful for the construction of conflict-free systems and thus we have justified the intuitevely given proposals in /3/, /10/ by the following corollary:

(40) Corollary

If $\mathcal{T} \geq T/_{v_p}^*$ then a system $\mathcal{N} = (N, C)$, where $\mathcal{T}$ is used in C, is conflict-free.

Conclusions

There are different starting points to define systems of Petri nets under the control of automata. In this paper we have tried to find an approach with not too may technical difficulties, and it is the author's believe that related specifications of systems would lead to related results. For instance, the undecidability results could be applied to such systems by (26). In the consequence (cf. (25)), uncontrolled Petri nets are not sufficient to replace such systems (as far as the sets $R_{\mathcal{N}}$ are unbounded). The concept of conflict partitions is valid for other forms of distributed control too. Related solutions for the deadlock avoidance problem in our systems constitude an open problem.

References

/1/ Burkhard, H.D., Ordered firing in Petri nets. Elektron. Informationsverarbeitung und Kybernetik 17(1981) 2/3, 71-86.

/2/ Burkhard, H.D., Two pumping lemmata for Petri nets. Elektron. Informationsverarbeitung und Kybernetik 17(1981) 7, 349-362.

/3/ Burkhard, H.D., On local conflict resolutions for Petri nets. Manuscript, 1981.

/4/ Burkhard, H.D., Local control and conflict resolution in Petri nets. Conference "Logics of Programs", Bialowieza, Sept. 1981.

/5/ Hack, M., Petri net languages. CSG Memo 124, Project MAC, M.I.T. Press, 1975.

/6/ Minsky, M., Computation: Finite and infinite machines. Prentice Hall, 1967.

/7/ Moalla, M., J.Pulou, J.Sifakis, Synchronized Petri nets: A model for the description of non-autonomous systems. MFCS 1978, LNCS 64, 374-384.

/8/ Salwicki, A., T.Müldner, On algorithmic properties of concurrent programs. Manuscript. Warsaw 1979.

/9/ Starke, P.H., Petri-Netze. Berlin 1980.

/10/ Starke, P.H., A note on conflicts in Petri nets. Bull. of the EATCS 1981, No. 14, 26-33.