We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

5,300
Open access books available

130,000
International authors and editors

155M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

13

Reachability Analysis of Time-Critical Systems

Štefan Hudák, Štefan Korečko and Slavomír Šimoňák
Department of Computers and Informatics
Faculty of Electrical Engineering and Informatics
The Technical University of Košice
Slovak Republic

1. Introduction

The systems whose functionalities (i.e. semantics) are defined with respect to time and whose correctness can only be assessed by taking time into consideration are called time - critical systems. This class of systems includes embedded real - time systems such as process control systems, digital signal processing systems, patient monitoring systems, flight control systems and weapon systems. The definition of time - critical systems covers a broader class then that of real - time systems. As an example can serve a typical computer system with a mouse input device where making two clicks on the mouse (the first and the second click at the beginning and at the end of the time interval duration 2 seconds respectively), has a quite different meaning than a double clicking within the time interval duration an half of second.

There is much expectation from the application of formal description techniques (FDT) and verification techniques to that field, since often time - critical systems have severe reliability and safety requirements. The theory and techniques developed for designing and understanding sequential and concurrent systems are not immediately suitable for time - critical systems: new formalisms and new methods are needed to specify their properties and prove correctness of implementations.

Several extensions to mathematical theories, such as logic, to cope with concurrent and real-time systems have been already proposed (Olderog, 1991; Ostroff, 1989), or algebra (Baeten & Bergstra, 1991). Petri Nets (PN) have deserved great attention in the role of FDT for concurrent, and in the last past as an FDT for *time - critical systems*. Two kinds of extensions to PN formalism are relevant with respect to time-critical systems:

- extensions that add time modelling capabilities (Time PNs),
- extensions that add functional modelling capabilities.

Different PN formalisms have been introduced in the literature; for example see (Genrich, 1986; Ghezzi et al., 1991; Jensen & Kristensen, 2009). Stochastic PNs represent important extension - that serves mainly to deal with performance evaluation and not in specification and verification of time-critical issues. Function modelling capabilities have been added to PNs at so-called high-level PNs. Among them we mention Colored Petri Nets (Jensen & Kristensen, 2009), Predicate Transition Nets (Genrich, 1986; Genrich & Lautenbach, 1981),

Numerical Petri Nets (Billington et al., 1988), PROT Nets (Bruno & Marchetto, 1986), Evaluative Petri Nets (Hudák, 1980) and TER Nets (Ghezzi et al., 1991).

2. Reachability Problem in Petri Nets

Petri Nets are very well known FDT, and many sources in the literature treat the subject in the depth (Peterson, 1981; Reisig, 1985; Murata, 1989; Hudák, 1999).
We define PN to be a 4-tuple $N = (P, T, pre, post)$, where:

- P is a finite set of places
- T is a finite set of transitions
- $pre : P \times T \to \{0,1\}$ - preset function, $post : P \times T \to \{0,1\}$ - postset function

The functions *pre, post* define a structure on the set $P \cup T$. It is very common to represent the PN by the bipartite oriented graph (Fig. 1a). The following useful notations can be defined:

${}^\bullet t = \{p \mid pre(p,t) \neq 0\}$ the set of preconditions of t	${}^\bullet p = \{t \mid post(p,t) \neq 0\}$
$t^\bullet = \{p \mid post(p,t) \neq 0\}$ the set of postconditions of t	$p^\bullet = \{t \mid pre(p,t) \neq 0\}$

By the marking of PN $N = (P, T, pre, post)$ we mean a totally defined function

$$m: P \to \mathbf{N} \tag{1}$$

We use **N** for the set of natural numbers , i.e, **N**={0,1,2,…} and m to describe the situation or configuration in PN N. Namely we say the condition represented by the place p in PN N holds iff $m(p) \neq 0$. Without loss of generality we assume that P and T have k and s elements respectively, i.e. $P = \{p_1, p_2, ..., p_k\}$, $T = \{t_1, t_2, ..., t_s\}$ and we fix some ordering of both, places and transitions from now on. Using the ordering of places we can consider m to be the k-dimensional nonnegative integer vector, i.e. $\vec{m} \in \mathbf{N}^k$. More formally

$$\vec{m} = (m(p_1), m(p_2), ..., m(p_k))$$

And $m(p_i)$ is the value of m in p_i , i = 1,2,..., k, according to (1). In our example (Fig. 1a) $m(p_i) = 1$ iff i = 1, or alternatively $\vec{m} = (1,0,0,0,0)$. For the simplicity we will use the denotation m for both interpretations of the marking m. We say t is enabled in m, and denote it $m \xrightarrow{t}$ iff for every $p \in t$, $m(p) \geq pre(p,t)$. In Fig. 1a t_2 is enabled in $m = (1,0,0,0,0)$ because ${}^\bullet t_2 = \{p_1\}$ and $m(p_1)$ = 1, and $pre(p_1, t_2)$ = 1. Once the transition t is enabled it can fire. The effect of the firing t in m is the creation of a new marking m' that depends on m and t. We use a denotation

$$m \xrightarrow{t} m'$$

and m' is defined in the following way:

$$m'(p)=\begin{cases} m(p)-pre(p,t) & if \quad p\in {}^{\bullet}t \wedge p\notin t^{\bullet} \\ m(p)+post(p,t) & if \quad p\in t^{\bullet} \wedge p\notin {}^{\bullet}t \\ m(p)-pre(p,t)+post(p,t) & if \quad p\in {}^{\bullet}t \cap t^{\bullet} \\ m(p) & otherwise \end{cases}$$

In PN N of Fig. 1a we can write $m=(1,0,0,0,0)\xrightarrow{t_2} m'=(0,1,1,0,1)$. Notice that transition t_3 will be enabled in m'. We say the sequence of transitions $\sigma = t_1t_2...t_r$ is *admissible firing sequence* in PN N, provided a sequence of markings $m_0, m_1,..., m_r$ exists and such that $m_{i-1}\xrightarrow{t_i} m_i$, i = 1, 2,..., r. In that case we write $m_0\xrightarrow{\sigma} m$ or $m_0\xrightarrow{*} m$, when σ is immaterial. The marking m is to be called *the reachable marking* in N from m_0 (via σ). We fix the marking m_0 to be *the initial marking* of PN $N=(P,T,pre,post)$ and we denote it N_0 = (N, m_0) or $N_0=(P,T,pre,post,m_0)$. Given PN $N_0=(P,T,pre,post,m_0)$ we define the set of reachable markings

$$R(N_0)=\{m \mid m_0\xrightarrow{\sigma} m\}$$

Reachability Problem (RP). Given PN $N_0=(P,T,pre,post,m_0)$ and a k-dimensional non-negative integer vector q, the problem whether $q\in R(N_0)$ is called *(the instance of) the reachability problem* of PN N_0 (for the state q).
The reachability problem can be treated in the terms of so called *vector addition systems* (VAS).

Definition 1*: A (k-dimensional) vector addition system (VAS) W_k is a couple W_k = (q_0 ,W), where $q_0\in$ $\mathbf{N}^k$ is the initial state of W_k , W is a finite set of (k- dimensional) integer vectors, i.e*

$$W=\{w_i \mid \forall i(1\le i\le s): w_i=(a_{i_1},a_{i_2},...,a_{i_k}), a_{i_j}\in \mathrm{Z} \ \ for\ all\ j=1,2,...,k\}$$

By a reachable state (vector) of W_k we call each $q\in$ $\mathbf{N}^k$ such that

1. $q = q_0 + w_{i1} + ...+ w_{in}$ for some integer $n \ge 0$, $w_{ij}\in$W, j = 1,..., n and
2. for $\forall j(1 \le j \le n)$: $q_j = q_0 + w_{i1} + ...+ w_{ij} \in \mathbf{N}^k$

We call the set of all such vectors the set of reachable state vectors, or simply the reachability set of VAS W_k, and denote it as R(W_k). More formally:

$$R(W_k)=\{q \mid q\ \in \mathrm{N}^k, q=q_0+w_{i_1}+w_{i_2}+...+w_{i_n}, n\in \mathrm{N},$$
$$\forall j(1\le j\le n): w_{i_j}\in W, q_j=q_0+w_{i_1}+w_{i_2}+...+w_{i_j}\in \mathrm{N}^k\}$$

In what follows we use the words: *reachable state vector, state vector*, or *state* as synonyms. If it is clear what dimensionality W_k has, we omit the index k, and also we say just a vector addition system instead a k-dimensional VAS, and the vector w, rather than the k-dimensional vector w.

Definition 2*: Given any VAS W_k = (q_0,W), where $q_0 \in \mathbf{N^k}$ and $W \subseteq Z^k$ for some k > 0, and let R(W_k) be its reachability set. For any $q \in \mathbf{N^k}$ a problem whether $q \in R(W_k)$ is called the reachability problem of VAS (with respect to q). We will occasionally use the abbreviation $RP(q,W_k)$) for it.*

With a vector addition system $W_k = (q_0, W)$ we can associate a tree structure, which we call the vector state tree, and define it as follows:

Definition 3: *Let $W_k = (q_0, W)$ be a vector addition system with q_0 as its initial state vector, and W the finite set of k-dimensional integer vectors. Then by the vector state tree of VAS W_k, denoted by VST_w, we mean a double labelled oriented tree*

$$VST_w = (T_w, lab_1, lab_2, q_0),\ T_w = (V, E, r_0)$$

is an oriented rooted tree, V - a set of vertices, $E \subseteq V \times V$ - a set of edges, $r_0 \in V$ - the root of T_w and the two labelling mappings

$$lab_1 : V \to \mathrm{N}^k,\ lab_2 : E \to W$$

The VST is defined such that $lab_1(r_0) = q_0$ and any vertex v of T_w, with $lab_1(v)$ = q has a son u $\in V$ with $lab_1(u)$ = q′ and $lab_2(v,u)$ = a iff q′ = q + a for some a $\in W$ and q,q′ $\in \mathbf{N^k}$.

Let $N_0 = (P, T, pre, post, m_0)$ be a PN with the initial marking m_0. Recall m_0 can be represented as a *k*-dimensional nonnegative integer vector, i.e. $m_0 \in \mathbf{N^k}$ and m_0 = (m_0 (p_1),..., m_0 (p_k)). The latter assumes an ordering of places in *P* and transitions in *T*, i.e. *P* = { p_1,..., p_k} and *T* = { t_1,..., t_s }. From the analytical properties of PNs (Reisig, 1985) we have that

$$m_0 \xrightarrow{t} m \Leftrightarrow \vec{m} = \vec{m}_0 + (\Psi(t) \times c^T)^T$$

and Ψ(*t*) is the Parikh mapping vector of the "string" *t* over the (ordered) alphabet *T*. Recall again that any transition $t \in T$ can be represented as a *k*-dimensional integer vector

$$\vec{t} = \overrightarrow{post}(t) - \overrightarrow{pre}(t)$$

and

$$\overrightarrow{post}(t) = (post(p_1,t), post(p_2,t),..., post(p_k,t)),\ \overrightarrow{pre}(t) = (pre(p_1,t), pre(p_2,t),..., pre(p_k,t))$$

It can be easily seen that

$$m_0 \xrightarrow{t} m \Leftrightarrow \vec{m} = \vec{m}_0 + \vec{t}$$

and we can construct for PN $N_0 = (P, T, pre, post, m_0)$ the vector addition system $W_k = (q_0, W)$ and such that $q_0 = \vec{m}_0$, $W = \{\vec{t}_i \mid t_i \in T, i = 1,2,...,s\}$, and *k* = *card P*. The following result holds.

Theorem 1. *For any PN $N_0 = (P, T, pre, post, m_0)$ there is an vector addition system $W_k = (q_0, W)$ and such that R(W_k)= R(N_0), and k = card P.*
Proof: follows from the above construction.

In what follows we make use of the special symbol - ω, which we add to **N**, creating by that virtue the new set $\mathbf{N}_\omega$. The ω element has the following properties: for any $a \in$ **N**: $a \leq \omega$ and

ω ± *a* = ω. Having a (*k*-dimensional) vector $q = (q_1, q_2, ..., q_k)$ and a set $A \subseteq \{1,2,...,k\} = \mathrm{K}$ we denote by $\omega_A q$ a vector, that we will call ω *vector* and will be defined such that

$$\omega_A q = (a_1, a_2, ..., a_k) \text{ where } a_i = \begin{cases} \omega & \text{if } i \in A \\ q_i & \text{otherwise} \end{cases}$$

In (Hudák, 1999) a new RP algorithm was proposed, in which a core structure is a finite state automaton (fsa) of the type M_w. Here we can only in a very short way summarize the RP algorithm.

RP algorithm: Given: VAS W_k = (q_0 ,W), $q \in \mathbf{N^k}$ - a state to be decided reachable or not;

Step 1 : Create fsa M_w=(Q, W, δ, ρ_0) ;

Step 2 : Construct MILP$_w$ (A, X_0, B(q), r);

Step 3 : if MILP$_w$ (A, X_0, B(q), r) = true then go to Step 4 else go to Step 5;

Step 4 : $q \in$R(W_k). Stop.

Step 5 : $q \notin$ R(W_k). Stop.

Here, $Q \subseteq \mathbf{N^k}$ is the set of states, δ: $Q \times W \to Q$ is the next-state function, $\rho_0 = \omega_A q_0$, $A \subseteq \boldsymbol{K}$ is the initial ω (macro) state. In many cases M_w's state diagram contains strongly connected components (scc), i.e it has the loop structure (see Fig. 8). MILP$_w$ (A, X_0, B(q), r) stands for *the modified integer linear programming problem* (Hudák, 1999), and denotes a predicate that is true iff X_0 is the solution to the problem, A is the matrix of ρ-*simple loops* (Hudák, 1999) of fsa M_w (taken as integer vectors), r is the number of strongly connected components of M_w's state diagram on the path leading from the macrostate $\omega_B q_0$ to a macrostate $\omega_{B'} q$, provided q is the state whose reachability is solved by the instance of MILP, and $B(q)$ is some linear expression free of any ρ-*simple loops* and depending on the state q.

The complexity of the RP algorithm was assessed (Hudák, 1999) and the worst-case upper bound of time complexity was settled. It turns out to be $O(2^{b^{2^{k+2} \times k^2}})$, provided k=*card* P, P to be the set of places of PN $N_0 = (P, T, pre, post, m_0)$ and $b \geq 0$ is some constant. Fortunately, in some cases the worst-case upper bound time complexity can be lowered (Hudák, 1999). There the reader can find also the proof of lower bound on the space complexity of RP in spirit of R.J.Lipton, which had been shown to be $O(c^k)$ for some constant c>0.

Fsa M_w is a finite state automaton with some interpretation of its states and input alphabet. That means that fsa can be thought of as M_w = (M,I) where M = (K, Σ, δ, r_0) we call the *basic fsa* for M_w, and I = (f, g, r_0) - be an *interpretation*:

$f : K \to \mathrm{N}_\omega^k$ - is a state labelling mapping

$g : \Sigma \to \mathrm{Z}^k$ - is an input labelling mapping

Properties of the interpretations were studied, that was inspired by V. N. Redko (Hudák, 1999).

Problem: *Let M = (K,Σ, δ, r_0) be a fsa; is there any interpretation I = (f,g,r_0) for <M,I> to be the finite state automaton of the type M_w for the PN $N_0 = (P, T, pre, post, m_0)$?*

for <M,I> to be the finite state automaton of the type M_w for the PN N_0 = (P,T, pre , $post$,m_0)?

Here follow results of the study and solution to the problem settled and also some remarks, that reflect some interesting issues connected with the problem (Hudák, 1999).

1. Assume we are given a fsa M =(K, Σ, δ, q_0); for any such M and a constant k > 1 there is a tripple of mappings I_k = <f, g, r_0>, and such that it is the interpretation for the fsa M, and <M, I_k> is M_w-type fsa for the VAS W_k = <r_0, {$g(a)$ | $a \in \Sigma$}>. I_k and VAS W_k can be effectively constructed on the basis of M and of the constant k.

2. Assume we are given a fsa of the type M_w - $M' = (Q, W, \delta_w, r_0)$, $Q \subseteq \mathbf{N}^k$, $W \subseteq \mathbf{Z}^k$, $r_0 \in \mathbf{N}^k$; then on the basis of M' the fsa $M = (K, \Sigma, \delta, q_0)$ and the couple of mappings $<f, g>$ can be constructed effectively; the latter is the isomorphism of fsa's M and M', and $<f, g, r_0>$ represents the interpretation for the fsa M.

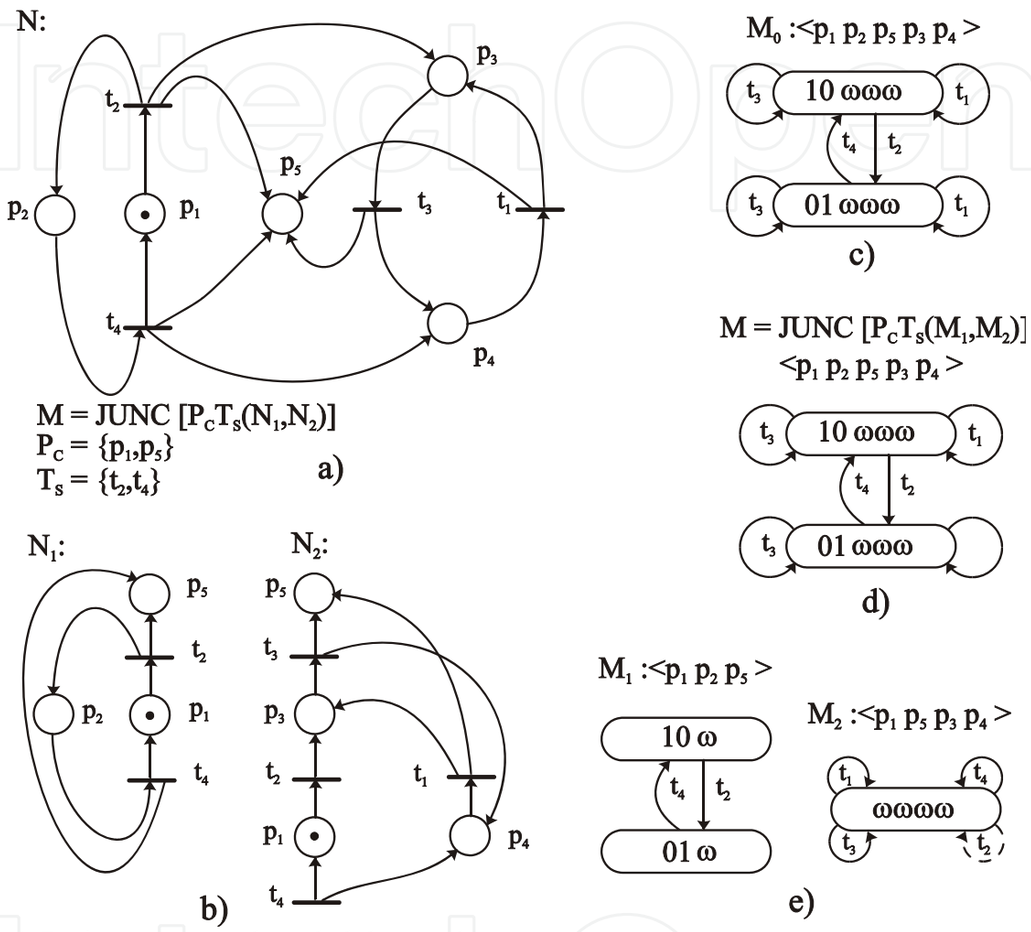


Fig. 1. De/compositional reachability analysis

2.1 De/compositional reachability analysis of PNs

We have pointed out that the complexity of RP is tremendous and so RP is an intractable problem. On the other hand there are some (quite practical and vital) problems for to be dealt with and for that a solving the reachability problem is an inevitable task. An example of such a problem can serve time analysis of time-critical systems (Ghezzi et al., 1994). An outcome from that situation, and that seems to be the only way, is to apply the divide and conquer approach to the RP. In (Hudák, 1994) we have proposed a new de/composition method of PNs that is suitable for the reachability analysis of Petri Nets in general case. The method is based on the RP algorithm by the first author (Hudák, 1999).

The de/composition can be accomplished in three ways: T-(P-, PT-) de/compositions, that we call T-(P-, PT-) JUNCTION respectively (Hudák, 1994).

Fig. 1 illustrates the method and results achieved. In Fig. 1a) and 1b) there are PN N and the subnets N_1 and N_2, the product of PT decomposition. Fig. 1c) and e) show the fsa of type M_w for the PNs N and N_1, N_2 respectively. They are fsa's with ω states that says the state space is infinite. Fig. 1d) shows the result of composition based on M_1 and M_2. Comparison of the state diagrams in Fig.1c) and d) shows the two automata are isomorphic.

3. Time-Critical Systems

A large class of systems can be represented and understood by abstracting away from the time aspect. That is the case of sequential systems, i.e. systems whose provided functionalities - their semantics - do not depend on the speed of execution. It is also the case of properly designed concurrent systems, such as time - sharing operating systems, whose overall correctness does not depend on the speed of execution of the component processes. In these two classes of systems time affects performance, not functional correctness.

In other cases, however, time issue becomes essential. In real - time systems, correctness depends not only on the results produced by computations, but also on the time at which such results are produced. The systems may enter incorrect state if the right result is produced too early or too late with respect to certain time bounds (Ghezzi et al., 1991). For example, an aircraft should not only modify its course once a mountain appears to be on the route, but it should also do it before crashing, i.e. within a given time.

The systems whose functionalities (i.e. semantics) are defined with respect to time and whose correctness can only be assessed by taking time into consideration are called time - critical systems. This class of systems includes embedded real - time systems such as process control systems, digital signal processing systems, patient monitoring systems, flight control systems and weapon systems. The definition of time - critical systems covers a broader class then that of real - time systems. As an example can serve a typical computer system with a mouse input device where making two clicks on the mouse (the first and the second click at the beginning and at the end of the time interval duration 2 seconds respectively), has a quite different meaning than a double clicking within the time interval duration an half of second.

3.1 Environment Relationship Nets

Environment-Relationship Nets (ER nets) (Ghezzi et al., 1991) represent a very strong extension to ordinary Petri Nets, that provides means to incorporate the notion of the time into the concept. We start with some notions. ER net is a net which can be characterized as follows:

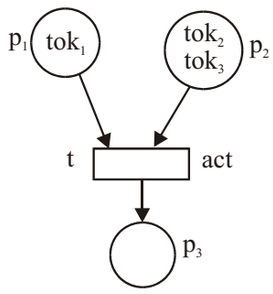
1. tokens are environments on ID (the set of identifiers) and V (the set of all values identifiers can take upon). The universal set of environments ENV = V^{ID}.
2. each transition is associated with an action. An action is a relationship

$$\alpha(t) \subseteq ENV^{k(t)} \times ENV^{h(t)}, \ and \ k(t) = |{}^{\bullet}t|, h(t) = |t^{\bullet}|$$

   By $\pi(t)$ we denote the projection of $\alpha(t)$ on $ENV^{k(t)}$ only, and call it *the predicate of t.*
3. a *marking* is an assignment of multisets of environments *(envs)* to places.
4. transition t is enabled in a marking m iff $\forall p_i \in {}^{\bullet}t$ there is at least one token env_i and such that the tuple $<env_1, ..., env_{k(t)}> \in \pi(t)$. The tuple $<env_1, ..., env_{k(t)}>$ is called *the*

enabling tuple for t. Notice there can be more than one enabling tuple for t in the marking m. The same token can belong to several enabling tuples.

5. *A firing* (of t in m) is a triple $x =< enab, t, prod >$ such that $< enab, prod > \in \pi(t)$.
6. The occurrence of a firing $x =< enab, t, prod >$ in a marking m causes a producing a new marking m' for the net, obtained from the marking m by removing the enabling tuple *enab* from the preset places of t.
7. in a natural way we define a *firing sequence* and *the sequence of admissible markings.* We may use the notation $m \xrightarrow{x} m'$ provided that $x =< enab, t, prod >$ is a firing that produces m' from m. We may also define the set of reachable markings, boundedness, liveness and other notions that can be defined for ordinary Petri Nets.



$\mathbf{tok_1}$ = {<a,0>,<b,1>}

$\mathbf{tok_2}$ = {<a,0>,<b,2>}

$\mathbf{tok_3}$ = {<a,0>,<b,1>}

$\mathbf{act} = \{<<p_1,p_2>,p_3> |\ p_1.a = p_2.a,\ p_1.b < p_2.b,\ p_3.a = p_1.a,\ p_3.b \in \{x |\ p_1.b \le x \le p_1.b + p_2.b\}\}$

Fig. 2. ER Net

3.2 Time ER Nets

When we assume that each environment contains a special variable called *chronos*, whose values are of numerical type representing *the timestamp* of the environment (token), by that the time can be incorporated into the net and we obtain Time ER net. All notions introduced for ER nets remain valid, for TER net, except the firing rule; the latter has to be modified. The timestamp of the token expresses the time of the environment's creation. We denote by $env_i.chronos$ the value of the chronos of the token env_i. We assume the firing x to be used in what follows. The firing x has the structure: $x =< enab, t, prod >$, $enab =< env_1, env_2, ..., env_{k(t)} >$ $prod =< env'_1, env'_2, ..., env'_{h(t)} >$, $k(t) = |^\bullet t|, h(t) = |t^\bullet|$. The following are the axioms that have to be satisfied by any action (Ghezzi et al., 1991).

Axiom 1: *(Local monotonicity) For any firing x* $env'_j.chronos \ge env_i.chronos$, *for all j,i,* $1 \le j \le h(t)$, $1 \le i \le k(t)$.

Axiom 2: *(Constraint on timestamps) For any firing x there is the value denoted time(x) and such that* $env'_j.chronos = time(x)$, *for all* $1 \le j \le h(t)$. *The value time(x) is called the time of the firing.*

Axiom 3*: (Firing sequence monotonicity) For any firing sequence* $s = x_1\, x_2 ...\, x_{|s|}$, $time(x_i) \le time(x_j)$, *if* $i < j$, $1 \le i,j \le |s|$.

For an ER net satisfying Axiom 2 we will write *prod.chronos* to denote the value *time(x)* the all chronos have in *prod*. An ER net where all environments contain chronos and that satisfies Axioms 1 and 2 is called *Time ER (TER) net*. Given ER net that satisfies Axiom 2, we will call the firing sequence $s = x_1\, x_2 ...\, x_{|s|}$, to be *the time ordered firing sequence* iff for each *i,j,*

$i < j$, $time(x_i) \leq time(x_j)$. We will call two firing sequences s, s' to be equivalent iff s is a permutation of s'. The following result can be proven (Ghezzi et al., 1991)

Theorem 2. *Let E be an ER net satisfying Axioms 1 and 2; for each firing sequence s with the initial marking m_0 there exists a firing sequence s′ equivalent to s that is time ordered.*

Given a transition t and the enabling tuple *enab* of the TER net, we can define *the set of possible firing times*, denoted *f-time*:

$$f-time(enab) = \{t \mid < enab, prod > \in \alpha(t), t = prod.chronos\}$$

Let $s = x_1 x_2 ... x_{|s|}$, be a firing sequence of a TER net with the initial marking m_0, and $m_0 \xrightarrow{s} m$ and $m_0 \xrightarrow{x_1 x_2 ... x_i} m_i$; the firing sequence s is *strong* iff it is time ordered, and for each $t' \in T$ and for each m_i, $(1 \leq i \leq |s|-1)$ there exists no tuple $enab_i^{'}$ for t' in m_i and such that $time(x_{i+1}) > \sup(f-time(enab_i^{'}))$. Another axiom can be stated to hold in TER nets.

Axiom 3′: *All firing sequences are strong.*

TER net that satisfies Axioms 1,2 and 3′ is called *the strong* TER net (STER net).

Finally, when we assume that the only type of tokens is chronos then we obtain *Time Basic (TB) nets*. In the class of TB nets we can distinguish two significant classes:

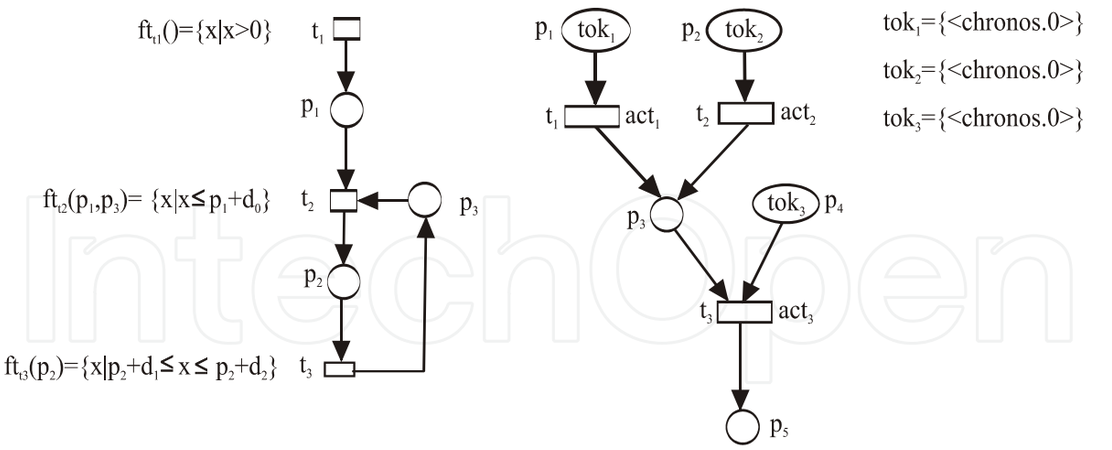
1. *weak-time semantics* WTS TB nets are those TB nets that satisfy Axiom 1,2, and 3,
2. *strong-time semantics* STS TB nets are those TB nets that satisfy Axiom 1,2 and 3′.

Fig. 3 shows an example of a TB net and a TER net. Another example of a TB net can be found in Fig. 7, where the net specifies a time-critical system, called 'the voice station', whose operation is governed by a simple protocol. The latter is formally described altogether by the tf_t predicates and by the structure of the TB net itself.

The modelling power of TER nets is such that it covers all known extensions of Petri Nets including those which incorporate the concept of the time (e.g. MF nets)(Ghezzi et al., 1991). The tradeoff between modelling power and the tractability of problems that of our concern should be taken into account. The greater is the modelling power, less is a chance to solve interesting problems: reachability, liveness, boundedness and others.

4. Time Reachability Analysis of TB Nets

There was pointed out that time-critical systems represent an important class of discrete systems for which the reachability analysis with respect to time is an inevitable task to be done during design and analysis of such the systems. We call the problem in that case *time reachability problem* or *time reachability analysis* (TRA) *problem*. In practice (Ghezzi et al., 1994), to cope with the problem mostly simulation techniques are used.



a) TB net b) TER net

Fig. 3. TER Nets

We have started the development of a new methodology of the reachability analysis, based on the new algorithm to solve the reachability problem by the first author (Hudák, 1981; Hudák, 1999), and the de/compositional approach to the problem (Hudák, 1994). Because of the specification of time critical systems by TB nets and that the task of TRA is inevitable in the design of such the systems, the new methodology presupposes an extension of the original results on de/compositional reachability analysis to TB nets.

The new methodology of the time reachability analysis consists of following steps.

1. developing a new semantics of TB nets based on time intervals (TI) rather than on time points (Ghezzi et al., 1994; Hudák, 1996),
2. extending new RP algorithm to TB (TER) nets in a way to be able to use the properties of fsa's of the type M_w,
3. exploring properties of Time Interval Profiles (TIP) as a suitable generalization of enabling tuples in TI semantics,
4. creating a new methodology for TRA of time-critical systems specified by TB nets that would have to be based on "the loop-spectral analysis" of TB nets (Hudák & Teliopoulos, 1998b).

4.1 Time Interval Semantics of TB Nets

In this section we deal with the time interval semantics of TB nets. We want to extend the solution of the reachability problem (RP) (for Petri Nets) to Time Environment Relationships Nets (TER), particularly to TB nets. Basic property of TB nets is that their tokens (timestamps) have the individuality (Ghezzi et al., 1991).

For the purpose to deal with the Time Reachability Problem (TRP) we will make use of several interpretations of tokens (Fig. 4)

- the case a) is the usual interpretation of the token as being the chronos (Ghezzi et al., 1991),
- the cases b) and c) will be used when we deal with the reachability issue itself, especially in a constructing of the finite state automaton (fsa) of type M_w (Hudák, 1996) for TRP, which we will denote as $M_{\tau w}$.

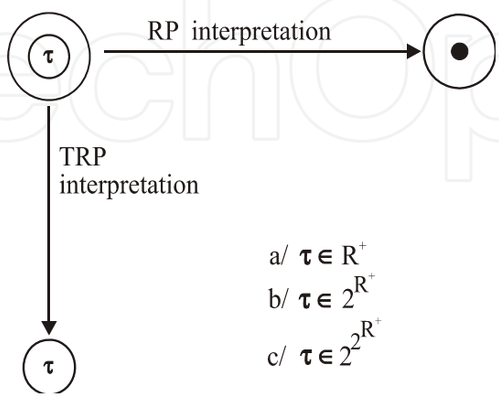


Fig. 4. Token Interpretation

In (Hudák, 1996) we deal with TB's semantics based on TIs. Any token (chronos) τ in TI semantics is considered to be a TI $\tau = [\tau i, \tau a] \subseteq \mathbf{R}^+$. Besides the set operations $\cap$, $\cup$, $()^C$ a new operations "+" and " ·" have been defined (Hudák, 1996). Given $\tau = [\tau i, \tau a] \subseteq R^+$ we define the operations:

1. $0 + \tau = \tau + 0 = \tau$ where 0 stands for empty TI.
2. $c + \tau = [\tau i + c, \tau a + c]$, $c \cdot \tau = [\tau i \cdot c, \tau a \cdot c]$ for any suitable constant $c \in \mathbf{R}$
3. $\tau' + \tau'' = \tau \Leftrightarrow_{df} \tau = \{\theta \in R^+ \mid \theta = \theta' + \theta'', \theta' \in \tau', \theta'' \in \tau''\}$
4. following are true statements: $\tau' + \tau'' = \tau'' + \tau'$, $(\theta + \tau) + \tau' = \theta + (\tau + \tau')$

In TI semantics we introduce the concept of TI relations that can hold between TIs (Fig. 5). Given any transition $t \in T$ of a TB net with the enabling predicate tf_t (Ghezzi et al., 1994), in TI semantics we replace any enabling tuple $en = (m(p_1), m(p_2), ..., m(p_{|\bullet t|}))$ with a corresponding collection of TIs that is called Time Interval Profile (TIP). In (Hudák, 1996) properties of TI operations were studied. For a reason that will be clear soon, a crucial issue to be dealt with was the property of TI expression

$$(\tau_1 + A) \cap (\tau_2 + B) \tag{2}$$

The study yielded several important results. Given $en = TIP(\tau_1, \tau_2, ..., \tau_{|\bullet t|})$ and tf_t-enabling predicate, we were able to prove the canonical representation exists for any *en* and tf_t (*en*).

Theorem 3. *(canonical representation of tf_t) Given TB net $\tau N_0 = (P, T, \theta, pre, post, tf, q_0)$ and let tf_t (en) be in the form $(\tau_1 + A) \cap (\tau_2 + B)$, then tf_t (en) has for given en the unique representation*

$$tf_t(en) = \tau en + tf_t(0)$$

where τen is a TI that depends on en , and $tf_t(0)$ is a TI that depends only on the structure of TB net and does not depend on en.

To put it in another way, any *t*-generated TI τ_t can be represented as a sum of two TIs: *τen*-the determinate TI that depends on TIP *en* in question and on *t* (or tf_t) and a constant TI $tf_t(0)$, which depends only on the structure of the TB net in question.

The 'anatomy' of *τen* was studied in (Hudák & Teliopoulos, 1998b). There was discovered that any *τen* can be classified as belonging exclusively to one of the two possible classes: mono-generated *τens* (i.e. $\tau en = [\tau_t i, \tau_t a]$ and $\tau_t \in en$) and non mono-generated *τens* (i.e. $\tau en = [\tau_{t_1} i, \tau_{t_2} a]$ and $\tau_{t_1}, \tau_{t_2} \in en$).

i	Rel_i representation		Formal characterization
	Graphical	Symbolic	
1	{ [}]	$\tau <_{sq} \tau'$	$(\tau i < \tau' i) \wedge (\tau' i < \tau a < \tau' a)$
2	[ ] { }	$\tau' \sqsubseteq_a \tau$	$(\tau i < \tau' i) \wedge (\tau a = \tau' a)$
3	{ [ ] }	$\tau' \sqsubset \tau$	$(\tau i < \tau' i) \wedge (\tau' a < \tau a)$
4	[ ] { }	$\tau \sqsubseteq_i \tau'$	$(\tau i = \tau' i) \wedge (\tau a < \tau' a)$
5	[ ] { }	$\tau' \sqsubseteq_i \tau$	$(\tau i = \tau' i) \wedge (\tau' a < \tau a)$
6	[{ }]	$\tau \sqsubset \tau'$	$(\tau' i < \tau i) \wedge (\tau a < \tau' a)$
7	[ ] { }	$\tau \sqsubseteq_a \tau'$	$(\tau' i < \tau i) \wedge (\tau a = \tau' a)$
8	[{] }	$\tau' <_{sq} \tau$	$(\tau' i < \tau i) \wedge (\tau' a < \tau a)$
9	{ } [ ]	$\tau \leq \tau'$	$(\tau a < \tau' i)$

{ } $\tau = \{\tau i, \tau a\}$ interval [ ] $\tau' = \{\tau' i, \tau' a\}$ interval

Fig. 5. Relations defined on TI variables

Given TB net $\tau N_0 = (P, T, \theta, pre, post, tf, q_0)$ we call Petri net N_0 = (*P*, *T*, *pre*, *post*, m_0) *basic* PN for TB net τN_0 . In extending the RP algorithm to TB nets we have chosen an approach to create the fsa M_w (in the case of TB nets we denote it $M_{\tau w}$), in such a way that $M_{\tau w}$ will be isomorphic with the corresponding fsa M_w for the basic PN in question. As we already mentioned, fsa's $M_{\tau w}$ and M_w will contain scc's. That implies that TIP's properties on ρ-loops are of very importance. The properties were studied in (Hudák & Teliopoulos, 1998b) and can be characterized as follows.

There are few assumptions taken for loops:

(a_1) the initial marking $\mu_0 = (\rho_0, \tau_{\rho_0})$ to be such that $\rho_0(p) \leq 1$,

(a_2) $post(p,t) \leq 1$ for any *p, t* of τN_0 , i.e. upon *t*-firing exactly one TB token can be generated,

(a_3) μ_0 is a live marking,

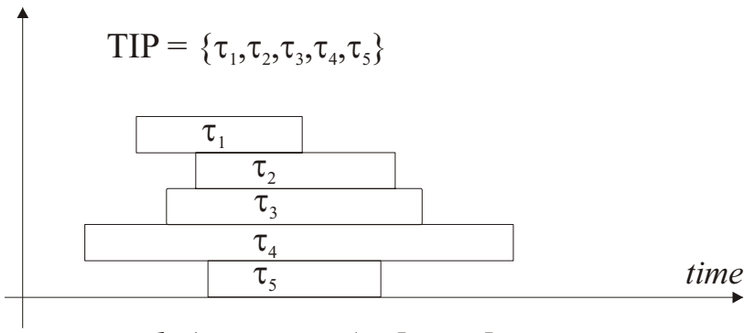
(a_4) the existence of $en_t^{(0)}$ - t's enabling tuple upon its first firing, as the first transition of σ_t, (a_5) in the loop σ, the only generators of TI tokens in any $p \in {}^{\bullet}t_i$, where t_i is from σ_t (i.e. $t_i \in T(\sigma_t)$), are again the transitions of σ_t.

Theorem 4. *(characterization of $tf_t(en)$)*

Let $\sigma_t = t_1 \cdots t_r$ be a ρ - simple loop satisfying the assumptions a_1-a_5 For any $i \geq 0$, $\forall \tau \in T(\sigma_t)$

$$tf_t(en^{(i+1)}) = tf_t(en^{(i)}) + tf_t(0)$$
$$tf_\ell(0) = tf_{t_1}(0) + \ldots + tf_{t_r}(0)$$

provided $T(\sigma_t) = \{t_i \mid \sigma_t(t_i) \geq 1\}$ and $\sigma_t(t_i)$ is the number of occurrences of t_i in σ_t.



$\tau = dti(\tau_1,\tau_2,\tau_3,\tau_4,\tau_5) = [ei, ea]$
$ei = \tau_5 i$, $ea = \tau_4 a$
$\tau = [\tau_5 i, \tau_4 a]$

Fig. 6. TIP and determinate time interval

Corollary 1. *For any $n>0$, $t \in \sigma_t = t\, t_1 \cdots t_r$*

$$tf_t(en^{(n)}) = tf_t(en^{(0)}) + n \cdot tf_\ell(0),$$
$$tf_t(en^{(n)}) = \tau en^{(0)} + n \cdot tf_\ell(0) \quad and$$
$$tf_t(en^{(\omega)}) = \tau en^{(0)} + \omega \cdot tf_\ell(0)$$

Lemma 1. *Let $p_1,p_2,\ldots,p_n$ be a sequence of places of TB net $\tau N = (P,T,\theta,pre,post,m_0,tf)$, $t_1,t_2,\ldots,t_{n-1}$ be a sequence of transitions from T and such that $p_i \in {}^{\bullet}t_i, p_{i+1} \in t_i^{\bullet}$ σ, q, q'- be a firing sequence and two markings respectively and such that $\sigma = x_1 x_2 \ldots x_n$, $q \xrightarrow{\sigma_t} q'$, $x_i = < en_i, t_i, prod_i >$, $i = 1,2,\ldots,n$. Let further τ_i be a chronos with the time interval (TI) interpretation (we propose to call it TI token, TI chronos, or simply TI) in p_i in the state q and such that $\tau_i = en(p_i)$ (τ_i is dwelling in p_i in the state q) and τ_{t_i} be a t_i - generated TI upon t_i firing, i=1,...,n-1 and $en_j(p_{j+1}) = \tau_{t_j}$, j=1,...,n-1.*

Then it is true that $\tau_1 \prec \tau_{t_1}, \tau_{t_{j-1}} \prec \tau_{t_j}$, $i = 1,2,\ldots,n-1 \prec \in \{<_{sq}, \leq\}$ and $\tau_{t_j} \in \tau_{q_j}(p_{j+1})$

Provided that

$q_j = [q\sigma_j]$, $\sigma_j = x_1 x_2 \ldots x_j$, *and* $[q\sigma_j] = \vec{q} + \overrightarrow{t(x_1)} + \ldots + \overrightarrow{t(x_j)}\overrightarrow{t(x_j)} = \vec{t_j} \leftrightarrow x_j = < en, t_j, prod >$

Corollary 2. *Under conditions of Lemma 1 the following holds: for all i>0:* $\tau_t^{(i)} \prec \tau_{t_r}^{(i)}$.

Lemma 2. *Given a* ρ_0 *-loop in* $M_{\tau w}$ *for the TB net* $\tau N_0 = (N_0, tf, \theta)$, $\sigma_t = t_1 \cdots t_r$, $t_i \in T$, *i=1,...,r, and two TI tokens* τ_1', τ_2' *belonging to* $en_t^{(0)} = TIP(\tau_1, \tau_2, ...\tau_t)$, *and* τ_j' *is* t'_j*-generated, TI chronos, i.e.* $\tau_j' = \tau_{t_j}$, $t_j' \in T(\sigma_t)$, *j=1,2,…,r.*

Suppose now that $\tau_1' \prec \tau_2'$ *then* $\tau_{t_1}^{(i+1)} \prec \tau_{t_2}^{(i+1)}$ *for any i > 0 and* $\prec \in \{<_{sq}, \geq\}$.

What the results in the above assertions say is that:

- given the loop structure of particular scc in M_w we are able to compute and thus predict the TIs which will be created upon passing the (stabile and selffeeded) loops (Theorem 4 and Corollary 1).
- relation $\prec \in \{<_{sq}, \leq\}$ between TIs will be preserved on the (stabile and selffeeded) loops of any scc of the fsa M_w (Lemma 1, Lemma 2, and Corollary 2).

4.2 On the Way to Loop-Spectral Time Reachability Analysis

In the state diagram of fsa M_w ρ - simple loops play profound role in the reachability analysis of the ordinary PN as the results of this work demonstrate. The role of ρ - simple loops grows even more in the issue of TRA of TB nets.

We distinguish two subclasses of loops in $M_{\tau w}$: *selffeeded* and *stabile* loops. The loop is selffeeded one if in a *t* - firing (*t* belongs to the loop) *t* "consumes" only tokens that was created solely by firings of loop's transitions. A loop can be called stabile if at any *t*-firing (*t* belongs to the loop) all tokens at precondition places are uniformly generated, i.e. at any *t*-firing at each repetition *t* consumes tokens from the same generators, i.e. transitions that generated tokens consumed by *t*. There is a strong relation between the two types of loops (Hudák & Teliopoulos, 1998b). Each loop becomes stabile after some initialization, after that some TIP (we call it initial) is reached which starts stable part of computation.

After defining initial TIP for any loop on a path we can define the TIs of any chronos (tokens) in which it can exist in the future. The structure of those TIs reminds very much spectral image of time sequences (Hudák & Teliopoulos, 1998b). Results achieved, however show that once $M_{\tau w}$ has been constructed we can predict precisely future of any token and discover perhaps a moment when it disappears because of the emptiness or dummy feature of its TI. For any TI τ_t (*t* - generated TI) we can construct a formula for the TI to be calculated. The RP algorithm works almost in the same way in the case of TB nets as it does in the case of ordinary Petri Nets.

5. TRA by example

We are now going to demonstrate TRA based on our approach. As an example we have chosen the TB net for the voice station (Ghezzi et al., 1994) which is depicted in Fig. 7. An interested reader can find a full description of the voice station in (Ghezzi et al., 1994).

5.1 The voice station

The voice station consists of three parts: coder, decoder and medium to transmit voice signals formatted into packets. Here we abstract from the structure of the packets, the nature of the medium, and the decoder. We only concentrate on the signal transmission, i.e. the voice coder and the part of the interface that transmits the packets.

The voice coder produces packets of constant length at a constant rate -one packet each 10 milliseconds. Once a new packet is produced and the station interface is ready to accept a new packet, the packet becomes ready to be transmitted. The reaction time of the interface, which takes ready packet and puts it in the transmission buffer, waiting to be transmitted on the medium, is assumed to be 0.2 milliseconds. The station interface waits for a transmission token (i.e. the interface is based on a token transmission protocol, e.g. a token ring) which is available for transmitting the packet already prepared in the buffer, 0.1 milliseconds after its arrival in the interface.

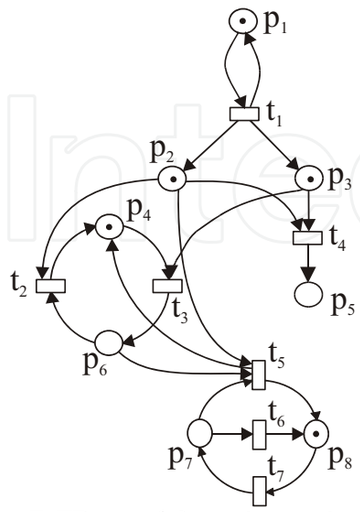
A packet is transmitted if it is in the buffer and the transmission token is available. Otherwise, after waiting for ready packet for 0.15 milliseconds, the transmission token goes to the next station of the ring. If a packet is not transmitted by the time a new packet is produced 10 milliseconds after the not transmitted packet was produced, the latter is discarded and the new packet is considered for transmission.

5.2 TB net of the voice station

In Fig. 7 the voice station is modelled by TB net and in that strong time semantics is assumed. The voice coder is modelled by transition t_1 and places p_1, p_2 and p_3. The token (chronos) in p1 records the time at which the last packet has been created by the coder. A token in p_3 represents the packet produced by the coder. A token in place p_2 records the packet creation time. Its timestamp is used by transition t_2 to determine whether the packet can still be processed or must be discarded. Transition t_1 fires 10 milliseconds after the production of a token in place p_1, i.e. 10 milliseconds after the production of the last packet.

The station interface is modelled by transitions t_2, t_3 and t_4 and places p_4, p_5 and p_6. A token in place p_4 represents the interface ready to accept a new packet from the voice coder; a token in place p_6 represents the interface ready to transmit a packet on the transmission medium. The firing of transition t_3 represents the station interface accepting (place p_4) a new packet produced by the coder (place p_3) and putting the newly produced packet in the transmission buffer (place p_6), ready to be transmitted. Transition t_3 can fire 0.2 milliseconds after both the packet has been produced and the interface is ready to accept a new packet. This can occur only within 10 milliseconds after the packet has been produced (10 milliseconds after place the packet has been produced); otherwise the packet is no longer considered for transmission and is discarded (transition t_4). The tokens stored in place p_5 represent all the messages that could not be transmitted. Transition t_2 represents the station interface becoming ready to accept a new packet from the coder if the current packet to be transmitted cannot be transmitted because its validity time elapsed. The transition token flowing through the station is modelled by transitions t_5 and t_6 and places p_7 and p_8. A token in p_7 represents the availability of a transmission token at the station interface. If a packet is ready for transmission (p_6 is marked) 0.1 millisecond after the arrival of the transmission token in the interface, t_5 fires, i.e. the packet is transmitted and the interface becomes ready for accepting a new packet. If no packet is ready for transmission, transmission token exits

the interface 0.15 milliseconds after its arrival, which is represented by t_6 and its tf_{t6} function. Transition t_7 models the return of the transmission token from the ring.



Time-Functions:

$tf_{t1}(p_1) = (\tau = p_1+10)$
$tf_{t2}(p_2,p_6) = (\tau = p_2+10 \text{ and } \tau \geq p_6)$
$tf_{t3}(p_3,p_4) = (\tau = \max(p_3,p_4)+0.2 \text{ and } \tau<p_3+10)$
$tf_{t4}(p_2,p_3) = (\tau = p_3+10 \text{ and } p_2=p_3)$
$tf_{t5}(p_2,p_6,p_7) = (\tau = p_7+0.1 \text{ and } \tau \geq p_2 \text{ and } \tau \geq p_6)$
$tf_{t6}(p_7) = (\tau = p_7+0.15)$
$tf_{t7}(p_8) = (p_8 \leq \tau \leq p_8+8)$

Initial marking:

$m_0(p_1) = \{18\}$ $m_0(p_4) = \{10\}$
$m_0(p_2) = \{18\}$ $m_0(p_8) = \{10\}$
$m_0(p_3) = \{18\}$

Fig. 7. TB net of the voice station

The fsa of the type M_w for the basic PN of the TB net of the voice station is depicted in Fig. 8. In (Hudák & Teliopoulos, 1997) it was demonstrated how *dti* τ*en* can be created and that it can be either mono-, or non mono-generated TI.

In the rest of this section we will use the notation $\vec{x}$ for an interval from *0* to *x* , $T(\sigma)$ for a set of transitions of a sequence σ and $P(\sigma)$ for a set of places connected with σ . More precisely

$$T(\sigma) = \{t \mid t \in T \wedge \sigma = t_1 t_2 \ldots t_{|\sigma|} \wedge \exists t_i (t_i = t \wedge i \in 1\ldots \mid \sigma \mid)\} ,$$

$$P(\sigma) = \{p \mid p \in P \wedge \exists t (t \in T(\sigma) \wedge p \in {}^\bullet t \cup t^\bullet)\} .$$

Let us consider now the situation in the voice station and look at how τ*ens* are generated. In the Fig. 8 we can see *selffeeded* loops σ_t there, e.g. $\sigma_t = t_7 t_1 t_3 t_5 t_7 t_1 t_3 t_5 \ldots$. Several remarks are in the order now as far as ρ- simple loops are concerned:

1. σ_t is selffeeded loop; that simply means that for any $p \in P(\sigma_t)$ and such that $p \in t_j^\bullet$ there exists a transition $t_i \in T(\sigma_t)$ and $p \in {}^\bullet t_i$, and also it holds that for any $p \in P(\sigma_t)$ and such that $p \in {}^\bullet t_i$, there exists a transition $t_j \in T(\sigma_t)$ such that $p \in t_j^\bullet$.
2. let $en_t^{(0)} = TIP(\ldots,\tau_1,\ldots,\tau_2,\ldots)$ denotes that τ_1, τ_2 belong to $en_t^{(0)}$ and for any $\tau \in TIP(\ldots,\tau,\ldots)$ it is true that τ is t'- generated for some $t' \in T(\sigma_t)$;
3. we say $\sigma_t = tt_1 \ldots t_r$ forms *a chain*, if there exists a sequence of places $p_1, p_2, \ldots, p_r$ such that $p_{i-1} \in {}^\bullet t_i \cap t^\bullet{}_{i-1}$; in that case

$$\tau^{(n)} = \tau en^{(0)} + tf_{\sigma_t}(0) \quad \text{and}$$
$$tf_{\sigma_t}(0) = tf_t(0) + tf_{t_1}(0) + \ldots + tf_{t_r}(0)$$

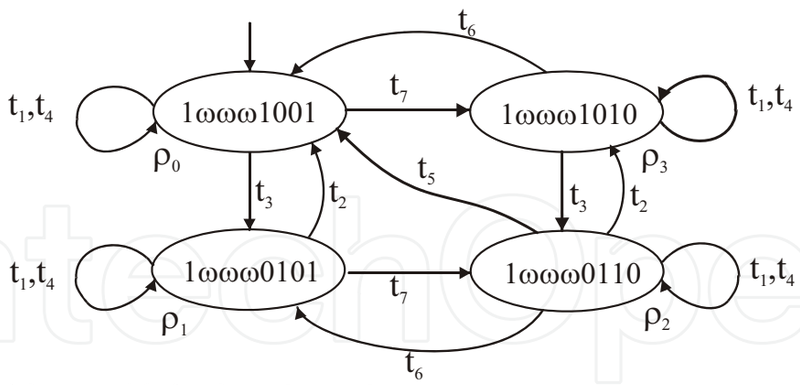


Fig. 8. FSA of the type M_w for the voice station

We assume first, that σ_t is not a chain. However, according to the remark 1, we can create a chain $\sigma_t^{(1)} = t' t_1' t_2' ... t_{r1}'$ to be a part of σ_t. Let further τ_1, τ_2 be t_1' - and t_1''-generated *dti*s respectively. Then two different properties the dtis may have:

- **i.** $t_1', t_1'' \in T(\sigma_t^{(1)})$ or
- **ii.** t_1', t_1'' belong to different chains.

In the *i*. case if t_1' precedes t_1'' then $\tau_1^{(1)} \prec \tau_2^{(1)}$ and also $\tau_1^{(i)} \prec \tau_2^{(i)}$ for any $i > 1$. The latter follows from Lemma 1, and Lemma 2 guarantees that TIP's relations are preserved on loops. So in any chain containing t'_i

$$\tau_{t_i'}^{(n)} = \tau en_{t_i'}^{(n-1)} + tf_{t_i'}^{(0)}$$

In Fig. 9 we demonstrate how the limits of *dti* τen can be created. Any τ generated in a chain preserves the relation $\prec \in \{<_{sq}, \leq\}$ between τ and the *TIs* generated in the chain in the previous times. We can see that TI limits' generation schema can be quite complicated (see Fig. 9). As we go through the chain, the situation may get even more complicated (Fig. 10). In spite of some irregularity of the situation from the start, the process of calculation of TIs in the TB net of voice station, as Fig. 10 shows, $\tau en_{t_i'}$ changes after t_{i-1}' firing and what will determine the $\tau_{t_i'}$ TI to be generated upon next t_i' firing. It seems reasonably to count separately the changes of τeni and τena. We have to realize that the case *d*) in Fig. 9 shows the typical situation in evaluating $\tau en^{(n-1)}$ for the n-th firing.

The same phenomenon can be discovered in ii) case either; for any TIP on the loop σ_t (selffeedable) every TI τ "arriving" at *p* will preserve $\prec \in \{<_{sq}, \leq\}$ with its predecessor. So

$$\tau_j^{(i)} \prec \tau_j^{(i+1)} \quad j = 1,2$$

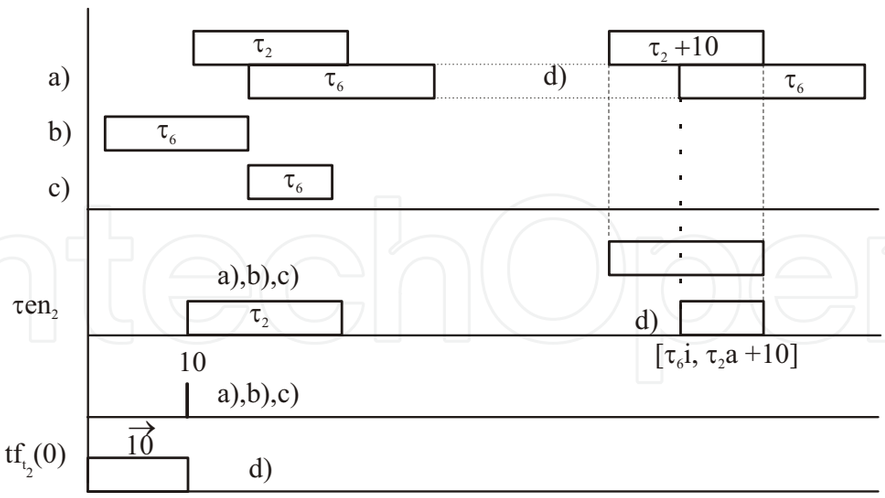


Fig. 9. TIP based calculation of dti τen for t_2

Now what we have to cope with is the problem what kind of relations we can expect between $\tau_1^{(i)}$ and $\tau_2^{(i)}$. We have to explore the following:

1. that chain approach to estimate interval limits is well founded, and
2. that selffeedable loop σ_t guaranties the stabile regime i.e. if $\sigma_t = tt_1...t_r$ is selffeedable loop, and

$$\tau_t^{(n)} = [\tau en_\alpha i + n \cdot \tau eni(tf_{\sigma_t}(0)), \tau en_\alpha a + n \cdot \tau ena(tf_{\sigma_t}(0))]$$

where $\tau eni(tf_{\sigma_t}(0))$ - expresses the contribution to the left limit of τen_t by σ_t-loop, and the same meaning wrt the right limit of τen_t is attached to the $\tau ena(tf_{\sigma_t}(0))$ (see below) .

Let us compute TI τ_4 that can be t_2-or t_5 -generated (Fig.7). We choose first for τ_4 to be t_5-generated. According to Theorem 3 (Hudák, 1996), and Fig.7 we may write

$$\tau_4 = \tau en(\tau_2, \tau_6, \tau_7) + tf_{t_5}(0)$$

so

$$\tau en(\tau_2, \tau_6, \tau_7) = [\tau_6 i, \tau_7 a] , \quad tf_{t_5}(0) = \overrightarrow{0.1}$$

TI τen will be determined (in the case of τ_4 as being t_5-generated) by $\tau_6 i$ and $\tau_7 a$ as far as the lower and upper bound of τen is concerned respectively. We are able to create a chain of transitions leading to creation of τ_6 (and thus $\tau_6 i$). Then chain will be $\{t_1 t_3\}$. To generate τ_7 (and thus $\tau_7 a$) t_7 has to be fired so the chain for τ_7 is $\{t_7\}$.

To make creation of τ_6 and τ_7 repeatedly the loop $\sigma_t = t_1 t_3 t_7 t_5$ has to be executed repeatedly in the stable regime. From σ_t only the part $t_1 t_3 t_5$ "works" to create the lower bound of τen of, and on the other hand the part $t_7 t_5$ is the only chain of transitions participating to create the upper bound of τen of t_5. We prefer to denote the parts of the loop σ_t. We denote by $iL(t_5)$ = $t_1 t_3 t_5$ the part *which determines the* $\tau en_{t5} i$ and by $aL(t_5)$ = $t_7 t_5$ the part *which determines the* $\tau en_{t5} a$. Let τen_{ti} be the initial value of the determinate TI of the transition t_i prior to its first firing as the member of the stable loop σ, i.e.

$$\tau_{ti}^{(1)} = \tau en_{ti} + tf_{ti}(0)$$

We denote be $\tau_{ti}^{(1)}$ t_i - generated TI after j-th firing of t_i in the stable loop σ and

$$\tau_{ti}^{(j)} = \tau en_{ti}^{(j-1)} + tf_{ti}(0)$$

According to Lemma 1, Lemma 2, i.e. due to properties of TB nets and the stabile selffeded loops we have that

$$\tau_{ti}^{(j)} \prec \tau_{ti}^{(j+1)}$$

and also

$$\tau en_i^{(j)} \prec \tau en_i^{(j+1)} \quad \prec \in \{<_{sq}, \leq\}$$

Assume $\tau en_t = [\tau_{t'} i, \tau_{t''} a]$. To calculate $\tau en_t^{(j+1)}$ in the case of the stabile loop, we have to consider the contribution of σ to the lower and upper bound of τen. The contribution is denoted by

$$tf_{iL(t)} \text{ and } tf_{aL(t)}$$

to be the contribution to the lower and upper bound respectively. In our example for t_5-generated τ_4 we get

$$\tau_4^{(n+1)} = \tau_{t_5}^{(n+1)} = \tau en_{t_5}^{(n+1)} + tf_{t_5}(0)$$
$$\tau en_{t_5}^{(n+1)} = [\tau_6^{(n)} i, \tau_7^{(n)} a]$$

To compute $\tau_6^{(n)} i$ and $\tau_7^{(n)} a$ we first compute $\tau_6^{(1)}$, $\tau_7^{(1)}$. Notice

τ_6 is t_3 - generated i.e. $\tau_6 = \tau_{t3}$
τ_7 is t_7 - generated i.e. $\tau_7 = \tau_{t7}$

We compute $\tau_t^{(1)}$ for all $t \in T(\sigma)$; in this particular case. We start with the calculation of initial values for σ′= uσσ ... provided σ = $t_7 t_1 t_3 t_5$ and $u = t_3 t_7 t_2 t_6 t_7 t_6$:

$$\tau_{t_3} = \tau_6 = tf_3(0) + \tau en_3 = 0.2 + dti_2(10,18) = 0.2 + 18 = 18.2$$
$$\tau'_{t_7} = tf_7(0) + \tau en_7 = \vec{8} + 10 = [10,18]$$
$$\tau_{t_6} = tf_6(0) + \tau en_2 = 0.5 + \tau_{t_7} = 0.15 + [10,18] = [10.15,18.15]$$
$$\tau''_{t_7} = tf_7(0) + \tau en_7 = \vec{8} + \tau_{t_6} = \vec{8} + [10.15,18.15] = [10.15,26.15]$$
$$\tau_{t_5} = tf_5(0) + \tau en_5 = 0.5 + [\max(\tau_6 i, \tau_2 i, \tau_7 i + 0.1), \tau_7 a]$$
$$= \overrightarrow{0.1} + [\max(18.2,18,10.25), 26.15] = [18.2,26.25]$$

We are going now to construct i- and a- determinate loops for the TB net of the voice station (Fig. 7). We are choosing first σ = $t_7 t_1 t_3 t_5$. For the loop chosen we have $T(\sigma) = \{t_1, t_3, t_5, t_7\}$ and $P(\sigma) = \{p_1, p_2, p_3, p_4, p_6, p_7, p_8\}$.

We can see (Fig. 7) that in P(σ) there will be both:

- *mono-generated* TIs (τ_1, τ_2, τ_3, by t_1, while τ_7 by t_7) and also
- *non mono-generated* TIs (τ_4 either by t_5 or t_2, while τ_8 either by t_5 or t_6)

In the following calculations based on the loop σ we choose for τ_4 and τ_8 to be t_5-generated. In the table below you can see the values of $tf_{ti}(0)$ for i = 1,2,3,5,6,7.

	i	a
$tf_{t1}(0) = 10$	10	10
$tf_{t2}(0) = 10$	10	10
$tf_{t3}(0) = \overrightarrow{0.2}$	0	0.2
$tf_{t5}(0) = \overrightarrow{0.1}$	0	0.1
$tf_{t6}(0) = 0.15$	0.15	0.15
$tf_{t7}(0) = \vec{8}$	0	8

Table 1. Values of $tf_{ti}(0)$

Now we can calculate

$$tf_{iL(t_5)}(0) = (tf_{t_1}(0))_i + (tf_{t_3}(0))_i + (tf_{t_5}(0))_i = 10 + 0 + 0 = 10$$
$$tf_{aL(t_5)}(0) = (tf_{t_7}(0))_a + (tf_{t_5}(0))_a = 8 + 0.1 = 8.1$$

Now we write the formula to calculate the value of τ_4

$$\tau_4 = [18.2 + n \cdot tf_{iL(t_5)}(0), 26.25 + n \cdot tf_{aL(t_5)}(0)]$$
$$\tau_4 = 18 + [0.2 + n \cdot 10, \quad 8.25 + n \cdot 8.1]$$

The expression for τ_4 has a peculiar feature. Notice that after each iteration of $\sigma_t = t_1t_3t_7t_5$ the lower bound of TI τ_4 will be increased by 10, while the upper bound by 8.1. So after some number of iteration τ_4 become a dummy interval with lower bound greater then the upper bound!!!

Fig. 10 illustrates the method used to predict TIs of TB net for the voice station. Notice the tendency of TIs in places p_4, p_7 and p_8. They tend to become shorter, when we go through the loop chosen.

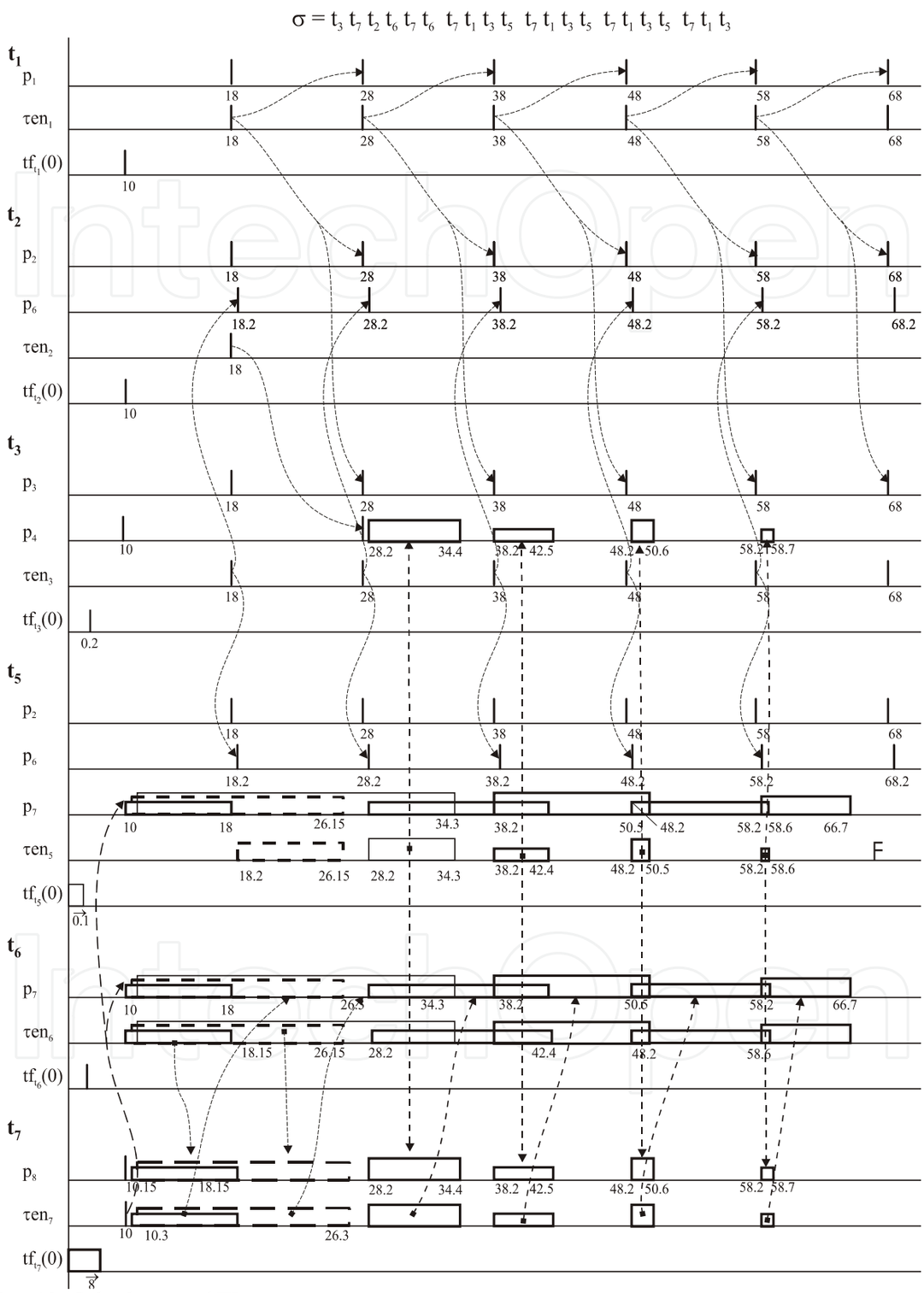


Fig. 10. Calculation of TIs

6. mFDT Environment

6.1 Motivation and formal methods involved

One of the assertions widely accepted in formal methods community states, that no single notation will ever address all aspects of a complex system (Bowen & Hinchey, 2006). This is also the case of Petri nets, which provides means to express non-determinism and concurrency, an easy-to-understand graphical notation and valuable analytical properties, but lacks other features, such as a verified development process and a formally sound and effective de/composition techniques. To cope with such an "incompleteness" of formal methods there have been many attempts to their integration. One of them was a proposal (Hudák & Grofčík, 2001) to develop a toolset called *multi Formal Description Technique Environment* (*mFDT Environment*, *mFDTE*), which will integrate Petri nets (PN) with two methods with complementary features – the *B-Method* and process algebras *ACP* and *APC*.

The ***B-Method*** (Abrial, 1996), with its B-Abstract Machine Notation (*B-AMN*) specification language, is a state-based model-oriented formal method. It offers a well-defined development process, which allows to specify a software system as a collection of so-called B-machines and to refine such an abstract specification to a concrete one. A consistency of the abstract specification and correctness of refinement are verified by means of proof obligations (*PObs*). There is an industrial tool, called Atelier B (Atelier B, 2009), which supports the whole development process and includes prover for PObs. The B-Method can be used for an additional analysis and implementation of PN models. On the other hand, we can confront invariants, listed in B-AMN specification, with invariants derived from corresponding PN.

Process algebras view systems as processes, described in an algebraic way. In Process algebras we can deal with de/composition of systems very elegantly, because they support compositionality by definition. We picked out the *Algebra of Communicating Processes* (*ACP*) (Baeten & Weijland, 1990) and developed a new *Algebra of Processes Components* (*APC*) (Šimoňák, 2003; Šimoňák et al., 2008). APC is a modification of ACP, which allows a comfortable description of PN processes. Analytical apparatus of PN can be used for verification of process algebraic specification of a system and process algebras can be used for a de/composition of PN.

6.2 mFTDE structure and tools

The mFDTE will consist of tools for integrated formal methods and interfaces between languages of these methods (Fig. 11). Tools will allow designer to gain from advantages of individual methods and interfaces will provide correct and formally proved translation from a specification in one method to the equivalent specification in another one.

The tools are in an implementation and testing phase now and can be obtained by request from the authors. Current versions of the tools are described below and translation processes of existing interfaces in the following subsections.

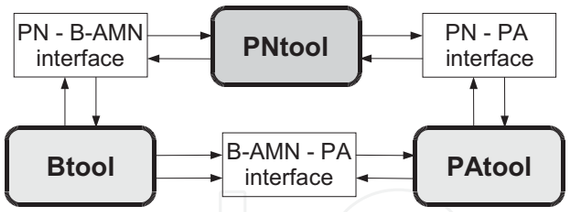


Fig. 11. mFDTE structure

The ***PNtool*** (Fig.12) is, quite naturally, a hearth of mFDTE and provides the richest functionality. The tool supports Generalized PN (*GPN*, also known as Place/Transition nets), TB nets, Evaluative PN (*EvPN*) and a limited subset of Coloured PN. EvPN (Hudák, 1980) is a Turing-powerful extension of GPN, which allows negative markings ($m(p)<0$) and place capacities defined with respect to individual arcs. In EvPN it is also possible that a change of a net marking, caused by a firing of some transition t, depends also on the marking of places, which are not adjacent to *t*.

PNtool provides a graphical editor and simulator for all supported PN types. The Petri Net Markup Language (PNML), an XML-based interchange file format for Petri-nets, is used to store GPN models. This allows using GPN created in another software tools for Petri nets, such as Petri Net Kernel, Renew, PEP and TINA. An extended version of PNML is used also for EvPN and we plan another extension for TB nets.

Computation of S- and T-invariants and the reachability analysis is supported for GPN. The current version of PNtool implements the first step of RP algorithm – creation of fsa M_w. For educational purposes the tool includes a step-by-step visualization of M_w creation. The automaton created can be saved in the form of Petri net, using another modification of PNML. PNtool also contains a part of PN - B-AMN interface, allowing a translation of GPN and EvPN to computationally equivalent B-machine.

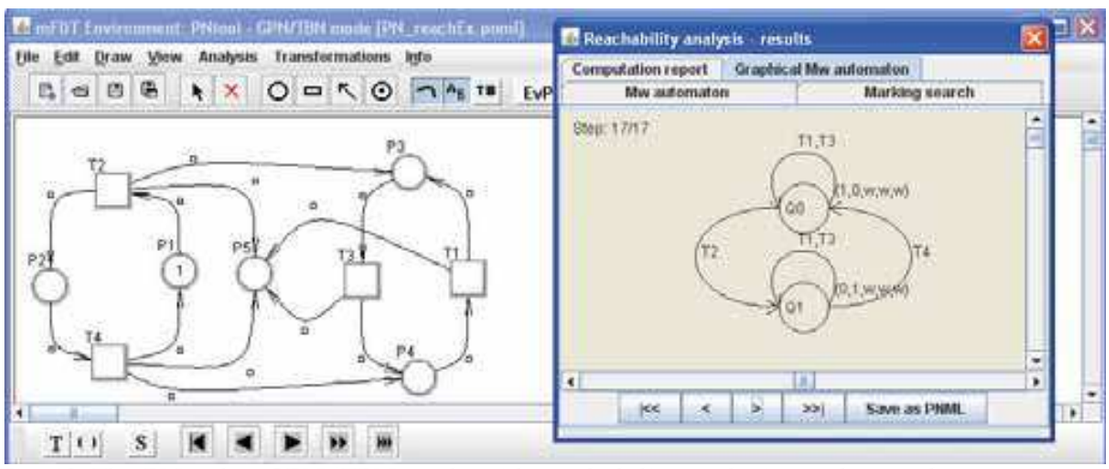


Fig. 12. The PNtool in GPN/TBN mode with reachability analysis results window open

The ***Btool*** focuses on a translation from B-AMN to JAVA. The translation process is inspired by that of jBTools (Voisinet, J.C. et al., 2002), but differs in various aspects, such as machines import mechanism and handling of output parameters.

Finally, the ***PAtool*** includes an editor for ACP and APC specifications and the PN – PA interface to translate the specifications from and to PN. To store the specifications a newly developed XML –based Process Algebra Markup Language is used.

6.3. Petri nets – B-AMN Interface

A theory of translations between B-AMN and PN, introduced in (Korečko, 2006), makes it possible to transform any GPN or EvPN into the computationally equivalent B-machine and almost any B-machine into the equivalent Coloured PN.

The *B-machine* is an abstract specification component of B-Method In general, a B-machine consists of a set of state variables (clause VARIABLES), an invariant to restrict the variables (clause INVARIANT), an initial operation to establish an initial state (INITIALISATION) and a set of operations to modify the variables (OPERATIONS). There are also other clauses intended for additional assertions and data components (parameters, sets and constants).

A basic idea of the translations is to link together similar behavioural concepts of both methods. Therefore places of PN are transformed to state variables of B-machine, initial marking to initialisation operation, transitions and adjacent arcs to operations and vice versa.

By translation of some GPN or EvPN N we get a computationally equivalent B-machine $\pi(N)$ (π is a mapping from PN to B-AMN). The two specifications, N and $\pi(N)$, are in fact bisimilar.

```
MACHINE mchPiN
VARIABLES sv_1, sv_2, sv_3, sv_4, sv_5
INVARIANT sv_1∈ℕ ∧ sv_2∈ℕ ∧ sv_3∈ℕ ∧ sv_4∈ℕ ∧ sv_5∈ℕ
INITIALISATION sv_1:=1 || sv_2:=0 || sv_5:=0 || sv_3:=0 || sv_4:=0
OPERATIONS
  op_t1= SELECT sv_4>=1 THEN sv_5:=sv_5 + 1 || sv_3:=sv_3 + 1 || sv_4:=sv_4 - 1 END;
  op_t2= SELECT sv_1>=1 THEN sv_2:=sv_2 + 1 || sv_1:=sv_1 - 1 || sv_5:=sv_5 + 1 || sv_3:=sv_3 +
1 END;
  op_t3= SELECT sv_3>=1 THEN sv_5:=sv_5 + 1 || sv_3:=sv_3 - 1 || sv_4:=sv_4 + 1 END;
  op_t4= SELECT sv_2>=1 THEN sv_2:=sv_2 - 1 || sv_1:=sv_1 + 1 || sv_5:=sv_5 + 1 || sv_4:=sv_4 + 1
END
END
```

Fig. 13. B-machine obtained from the Petri net N from Fig.1

A B-machine *mchPiN*, obtained from the net N from Fig.1, can be seen in Fig.13. Values of machine variables are naturals ($\mathbb{N}$) and correspond to markings of N (sv_i to $m(p_i)$). Similarly, operations correspond to transitions of N. The operations consists of a guarded command "SELECT P THEN S END", which means "do S, if P holds". If P doesn't hold, then the command is not feasible. Operator "||" stands for parallel composition, so "S_1|| S_2" means "do S_1 and S_2 simultaneously". As it was said, a B-machine obtained by the translation can be used for an additional analysis of PN specification. For example, to check a deadlock freedom of N, we add a predicate saying "there must be at least one feasible operation in each state of $\pi(N)$" and prove PObs of $\pi(N)$. The extended invariant for *mchPiN* has the form (3).

$$sv\_1\in\mathbb{N} \wedge \ldots \wedge\ sv\_5\in\mathbb{N} \wedge (sv\_4>=1 \vee sv\_1>=1 \vee sv\_3>=1 \vee sv\_2>=1) \qquad (3)$$

To allow a refinement of B-machine obtained, we have to use a slightly modified always feasible form of operations with "IF *P* THEN *S* ELSE SKIP END" instead of "SELECT *P* THEN *S* END". A theory of PN to B-AMN translation, including an example of EvPN translation can be found in (Korečko, 2006; Korečko, 2009). The translation can be further extended to high-level Petri nets, e.g. by adapting an approach used in (Kalinichenko et al., 2005).
In an opposite direction a translation is more complicated. For example, we can get more that one PN transition for one operation because of a non-deterministic nature of B-machine operations. Here we use Coloured PN, that match the modelling power of B-AMN while retaining valuable analytical properties. A step-by-step demonstration of the translation from B-AMN to Coloured PN can be found in (Korečko et al., 2008), where it is also shown how a structural analysis of the Petri net obtained can be used to reveal some additional invariant properties, not specified in the original B-machine.

6.4. Petri nets – Process Algebra Interface

Transformations of PN-PA interface, introduced in (Šimoňák, 2003), consist of two parts, namely: linguistic semantics preserving transformation of process algebra ACP specification into the corresponding Petri net and the operational semantics preserving transformation of (Ordinary) Petri net into the process algebra APC.
The first of two transformations mentioned, is based on construction of *elementary nets*, corresponding to atomic actions of the ACP specification, including the empty process (ε) and the deadlock (δ). Additionally, *net operations* are introduced, corresponding to operators of the ACP (alternative composition, sequential composition, parallel composition and encapsulation), allowing composition of Petri nets in order to obtain the resulting net, corresponding to the original specification. A description of the transformation, including an example can be found in (Šimoňák, 2006).
The aim of the second transformation is to construct the APC specification from the source Petri net. The approach is based on creating special variables (named E-variables) for every place of given Petri net, expressing processes initiated in those places. Algebraic semantics is given as a parallel composition of all such variables, whose corresponding places hold token(s) within the initial marking. A description and a short example of the transformation can be found in (Šimoňák et al., 2008).

7. Conclusion

In this work some results concerned the reachability analysis of time critical systems based on Petri Nets have been presented. The issue is very important, as the nowadays experience with computer based systems shows. The importance of the issue is not only from the practical point of view, but also from the theoretical one. As we know, and also it was demonstrated, the reachability analysis in the case the state space is large, or even infinite, is an intractable problem. Things get even worse, when the time issue comes under consideration. The results presented lay a foundation for coping with the problem. They are based on the original RP algorithm, and the de/compositional method of reachability analysis developed by the first author. The corner stone here are the properties of the finite state automaton of the type M_w, that was revealed by a convex analysis approach to the fsa (Hudák, 1999).

In the state diagram of fsa M_w ρ-simple loops play profound role in the reachability analysis of the ordinary PN as the results of this work demonstrate and it has been gathered enough arguments (Hudák & Teliopoulos, 1998b) that the role of ρ-simple loops remains in the issue of TRA of TB nets.

We distinguish two subclasses of loops in $M_{\tau w}$: *selffeeded* and *stabile* loops. The loop is selffeeded one if in a *t* - firing (*t* belongs to the loop) *t* "consumes" only tokens that was created solely by firings of loop's transitions. A loop can be called stabile if at any *t*-firing (*t* belongs to the loop) all tokens at precondition places are uniformly generated, i.e. at any *t*-firing at each repetition *t* consumes tokens from the same generators, i.e. transitions that generated tokens consumed by *t* (Hudák & Teliopoulos, 1998b) . There is a strong relation between the two types of loops (Hudák & Teliopoulos, 1998b). Each loop becomes stabile after some initialization, after that some TIP (we call it initial) is reached which starts stable part of computation.

The proposed algorithm for reachability problem has been partially implemented in the mFDT Enviroment and, thanks to the mFDTE interfaces, can be used also for specifications written in other formal specification languages.

After defining initial TIP for any loop on a path we can define the TIs of any chronos (tokens) in which it can exist in the future. The structure of those TIs reminds very much spectral image of time sequences (Hudák & Teliopoulos, 1998b). Great deal of work has been done already on the study of properties of different kinds of loops from the point of view of feeding transitions on the loop (Hudák & Teliopoulos, 1998a). There are some problems left, specifically from the point of view of different semantics (MWTS, STS) (Ghezzi et al., 1994).

Results of the theory presented show that once $M_{\tau w}$ has been constructed we can predict precisely future of any token and discover perhaps a moment when it disappears because of the emptiness or dummy feature of its TI. For any TI τ_t (*t* - generated TI with the name τ) we can construct a formula for the TI to be calculated. There are still some problems to be resolved, and we hope to deal with them in the future.

The RP algorithm works almost in the same way in the case of TB nets as it does in the case of ordinary Petri Nets. We hope that the questions raised above will be tackled upon as the subject of further research, and the results achieved will be published elsewhere.

8. References

Abrial, J.R. (1996). *The B-book: assigning programs to meanings*, Cambridge University Press, ISBN 0-521-49619-5, Cambridge, U.K.

Baeten, J.C.M. & Weijland W.P. (1990). *Process algebra*, Cambridge University Press, ISBN 0-521-40043-0, Cambridge, Great Britain

Baeten, J.C.M. & Bergstra, J.A. (1991). Real Time Process Algebra. *Formal Aspects of Computing*, Vol.3, No.2, (1991) pp.142-188, ISSN 0934-5043

Billington, J.; Wheeler, G. & Wilbur-Ham, M. (1988). Protean: A high-level Petri net tool for the specification and verification of communication protocols. *IEEE Trans.Software Eng*. Vol.14, No.3, (March 1988) pp. 301-316, ISSN 0098-5589

Bowen, J.P. & Hinchey, M.G. (2006). Ten commandments of formal methods... ten years later. *Computer,* Vol. 39, No. 1, (January 2006) pp. 40- 48, ISSN 0018-9162

Bruno, G. & Marchetto, G. (1986). Process-translatable Petri nets for the rapid prototyping of process control systems. *IEEE Trans.Software Eng.* Vol.12, No.2, (February 1986) pp. 346-357, ISSN 0098-5589

Genrich, H.J. (1986). Predicate/transition nets. In: *Advances in Petri Nets 1986*, Brauer, W.; Reisig, W. & Rozenberg, G. (Ed.), pp. 207-247, Springer Verlag, ISBN 0-387-17905-4, New York

Genrich ,H.J. & Lautenbach,K. (1981). System Modelling with High-Level Petri Nets. In: *Theoretical Computer Science 13*, pp. 109-136

Ghezzi, C.; Mandrioli, D.; Morasca, S. & Pezze,M. (1991). A unified high-level Petri net formalism for time-critical systems. *IEEE Trans.Software Eng*. Vol.17, No.2, (February 1991) pp. 160-172, ISSN 0098-5589

Ghezzi, C.; Morasca, S. & Pezze, M. (1994). Validating Timing Requirements for TB Net Specifications. *Journal of Systems and Software*, Vol.27, No.7, (November 1994) pp. 97-117, ISSN 0164-1212

Hudák, Š. (1980). *Extensions to Petri Nets*, Habilitation Thesis, Technical University of Košice, Slovakia

Hudák, Š. (1981). *The recursive decidability of the reachability problem for vector addition systems.* The University of Newcastle upon Tyne, Computing Laboratory, ASM/84, August 1981 (also in Proceedings of The Second European Workshop on the Theory and Applications of Petri Nets, Bad Honnef, Germany, September 1981).

Hudák, Š. (1994). De/compositional Reachability Analysis. *Journal of Electrical Engineering*, Vol.45, No.11, (1994) pp. 424-431, ISSN 0013-578X

Hudák, Š. (1996). Time Interval Semantics of TB nets, *Proceedings of the International Conference RSEE'96*, 12pp, Oradea, Romania, May 1996

Hudák, Š. (1999). *Reachability Analysis of Systems Based on Petri Nets*, elfa s.r.o., ISBN 80-88964-07-5, Košice, Slovakia

Hudák, Š. & Grofčík, J. (2001). An Environment for Design and Analysis of Time-Critical Systems, *Proceedings of EMES'2001*, pp. 66-75, Oradea, Romania, May 2001.

Hudák, Š. & Teliopoulos, K. (1997). TB Nets: properties of Time Interval Profiles, *Proceedings of the International Conference RSEE'97*, 8pp., Oradea, Romania, May 1997

Hudák, Š. & Teliopoulos, K. (1998a). Loop Spectral Analysis of Time Rechability Problem, *Proceedings of RSEE'98*, 11pp, Oradea, Romania, May 1998

Hudák, Š. & Teliopoulos, K. (1998b). TB Nets and TRA of Time-critical Systems, *Proceedings of the Scientific Conference Artificial Intelligence in Industry*, pp. 156-165, High Tatras, Slovakia, April 1998

Jensen, K. & Kristensen, L.M. (2009). *Coloured Petri Nets. Modelling and Validation of Concurrent Systems*, Springer Verlag, ISBN 978-3-642-00283-0

Kalinichenko, L.A.; Stupnikov, S.A. & Zemtsov N.A. (2005). Extensible Canonical Process Model Synthesis Applying Formal Interpretation. *Proceedings of ADBIS'05*, LCNS vol.3631 pp. 183-198, ISBN: 978-3-540-28585-4, Talin, Estonia, September 2005, ISBN 3-540-28585-7, Springer Verlag, Berlin-Heidelberg

Korečko, Š. (2006) *Integration of Petri Nets and B-Method for the mFDT Environment*. PhD thesis. DCI FEEI TU Košice, Slovakia, 2006 (in Slovak)

Korečko, Š.; Hudák, Š. & Šimoňák, S. (2008). Analysis of B-machine based on Petri Nets, P*roceedings of CSE 2008*, pp. 24-33, ISBN 978-80-8086-092-9, Stará Lesná, Slovakia, September 2008, elfa s.r.o, Košice

available from: hornad.fei.tuke.sk/~korecko/pblctns/CSE2008_SKor.pdf
Korečko, Š. (2009). *From Petri nets to B-Method*, Technical report DCI 1/2009, DCI FEEI TU Košice, 2009, available from: hornad.fei.tuke.sk/~korecko/pblctns/trEvPN_B.pdf
Murata, T. (1989). Petri Nets:Properties, Analysis and Applications, *Proceedings of the IEEE*, Vol. 77, No. 4., (April 1989) pp. 541-580, ISSN 0018-9219
Olderog, E.R. (1991). *Nets, Terms and Formulas*, Cambridge University Press, ISBN 0-521-40044-9, Cambridge, U.K.
Ostroff, J.S. (1989).*Temporal Logic for Real-Time Systems*, Research Studies Press Ltd., ISBN 0-08380-086-6, U.K.
Peterson, J.L (1981). *Petri Net Theory and the Modelling of Systems*, Prentice Hall PTR, ISBN 0-136-61983-5, Upper Saddle River, NJ, USA
Reisig, W. (1985). *Petri nets: An Introduction*, Springer Verlag, ISBN 0-387-13723-8, Heidelberg
Šimoňák, S. (2003). *Formal methods integration based on Petri nets and process algebra transformations*. PhD Thesis, DCI FEEI TU Košice, 2003 (in Slovak)
Šimoňák, S. (2006). Formal Methods Transformation Optimizations within the ACP2PETRI Tool. *Acta Electrotechnica Et Informatica*, Vol.6, No.1, (2006) pp. 75-80, ISSN 1335-8243, available from: www.aei.tuke.sk
Šimoňák, S.; Hudák, Š. & Korečko, Š. (2008). APC Semantics for Petri Nets. *Informatica*, Vol. 32, No.3, (2008) pp. 253-260, ISSN 0350-5596, available from: www.informatica.si
Voisinet, J.C.; Tatibouet, B. & Hammad, A. (2002). jBTools: An experimental platform for the formal B method. P*roceedings of PPPJ'02*, pp. 137—140, ISBN 0-901-51987-1 Dublin, Ireland, June 2002, National University Of Ireland, Maynooth
Atelier B website (2009). www.atelierb.eu

Petri Nets Applications

Edited by Pawel Pawlewski

ISBN 978-953-307-047-6

Hard cover, 752 pages

Publisher InTech

Published online 01, February, 2010

Published in print edition February, 2010

Petri Nets are graphical and mathematical tool used in many different science domains. Their characteristic features are the intuitive graphical modeling language and advanced formal analysis method. The concurrence of performed actions is the natural phenomenon due to which Petri Nets are perceived as mathematical tool for modeling concurrent systems. The nets whose model was extended with the time model can be applied in modeling real-time systems. Petri Nets were introduced in the doctoral dissertation by K.A. Petri, titled "„Kommunikation mit Automaten" and published in 1962 by University of Bonn. During more than 40 years of development of this theory, many different classes were formed and the scope of applications was extended. Depending on particular needs, the net definition was changed and adjusted to the considered problem. The unusual "flexibility" of this theory makes it possible to introduce all these modifications. Owing to varied currently known net classes, it is relatively easy to find a proper class for the specific application. The present monograph shows the whole spectrum of Petri Nets applications, from classic applications (to which the theory is specially dedicated) like computer science and control systems, through fault diagnosis, manufacturing, power systems, traffic systems, transport and down to Web applications. At the same time, the publication describes the diversity of investigations performed with use of Petri Nets in science centers all over the world.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Stefan Hudak, Stefan Korecko and Slavomir Simonak (2010). Reachability Analysis of Time-Critical Systems, Petri Nets Applications, Pawel Pawlewski (Ed.), ISBN: 978-953-307-047-6, InTech, Available from: http://www.intechopen.com/books/petri-nets-applications/reachability-analysis-of-time-critical-systems

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen