Time Petri Nets State Space Reduction Using Dynamic Programming

Louchka Popova-Zeugmann

Department of Computer Science
Humboldt-Universität zu Berlin
Unter den Linden 6
10099 Berlin, Germany
popova@informatik.hu-berlin.de

Abstract. In this paper a parametric description for the state space of an arbitrary TPN is given. An enumerative procedure for reducing the state space is introduced. The reduction is defined as a truncated multistage decision problem and solved recursively. A reachability graph is defined in a discrete way by using the reachable integer-states of the TPN.

Keywords: Time Petri Net, dynamic programming, state space reduction, integer-state, reachability graph

1 Introduction

For more than forty years Petri nets have been used in order to describe and study concurrent systems. At first sight, time and concurrence do not seem to have much in common. But if one looks closer, they are often connected, e.g. whenever local time dependencies between actions are relevant. There are endless examples from different areas showing this. For this reason, a large variety of time dependent Petri nets have been introduced and well studied. One of the first such nets defined is the Time Petri net (TPN), introduced in [1] in order to study recoverability problems in computer systems and to design communication protocols.

Time Petri nets (TPN) are derived from classical Petri nets. Additionally, each transition t is associated with a time interval $[a_t, b_t]$. Here a_t and b_t are relative to the time, when t was enabled last. When t becomes enabled, it can not fire before a_t time units have elapsed, and it has to fire not later than b_t time units, unless t got disabled in between by the firing of another transition. The firing itself of a transition takes no time. The time interval is designed by real numbers, but the interval bounds are nonnegative rational numbers. It is easy to see (cf. [2]) that w.l.o.g. the interval bounds can be considered as integers only. Thus, the interval bounds a_t and b_t of any transition t are natural numbers, including zero and $a_t \leq b_t$ or $b_t = \infty$.

Every possible situation in a given TPN can be described completely by a state $z = (m, h)$, consisting of a (place) marking m and a transition marking h. The (place) marking, which is a place vector (i.e. the vector has as many components as places in the considered TPN), is defined as the marking notion in classical Petri nets. The time marking, which is a transition vector (i.e. the vector has as many components as transitions in the considered TPN), describes the time circumstances in the considered situation. In general, each TPN has infinite number of states. Thus the central problem for analysis of a certain TPN is the knowing of its state space.

In this paper the state space is characterized parametrically and it is shown that knowledge of the integer-states, i.e. states whose time markings are (nonnegative) integers, is sufficient to determine the entire behavior of the net at any point in time. While the calculation of a single integer-state is easy, the proof that knowledge of the integer-states is sufficient for analysing a TPN is difficult. This problem is divided into a finite number of problems, which are solved recursively in a manner typical of the methodology of dynamic programming. A parametrical characterization of the state space has already been introduced in [3].

The new contributions presented in this paper are as follows. We extend the fundamental property to the case where time is described by nonnegative reals (instead of just rationals). Furthermore, we modify the definition of the reachability graph in the case that infinity is allowed for a latest firing time. This modified definition allows us to obtain a finite reachability graph if and only if the TPN is bounded. This property previously held only when all latest firing times were finite. Additionally, we elaborate the correlation between the above described problem and dynamic programming. Dynamic programming originated as a method for solving decision problems by Bellman, amongst others, in [4] and later studied in [5], [6] etc. The algorithm, proved here shows that the set of all reachable $p-$markings in a certain TPN is semi-decidable. The last set is in general not decidable because of the equivalence between TPNs and Turing machines (cf. [7]).

This paper is organised as follows. The next section gives some preliminary definitions and remarks. The third section introduces a parametric characterization of the state space. Afterwards, the special meaning of the integer-states is proved using the method of finite dynamic programming. This is followed by introducing the reachability graph. In the fourth section some related work is summarized. Finally, the last section gives some remarks including future outlook.

2 Basic Notations and Definitions

As usual, we use the following notations in this paper : $\mathbb{N}$ is the set of natural numbers, $\mathbb{N}^+ := \mathbb{N} \setminus \{0\}$. $\mathbb{Q}_0^+$, res. $\mathbb{R}_0^+$, is the set of nonnegative rational numbers, res. the set of nonnegative real numbers . T^* denotes the language of all words over the alphabet T, including the empty word e; $l(w)$ is the length of the word

w. $\wp(C)$ denotes the power set of a set C. $\mid D \mid$ stands for the number of elements of a finite set D. $C^{|D|}$ is the cartesian product $\underbrace{C \times \cdots \times C}_{|D|\ times}$. The "floor" of a real number r denoted by $\lfloor r \rfloor$ is the maximum of the set of integers that are not greater than r, respectively the "ceiling" of r denoted by $\lceil r \rceil$ is minimum of the set of integers that are not smaller than r.

And now the definition of a (classical) Petri net is as follows:

Definition 1 (Petri net). *The structure $N = (P, T, F, V, m_o)$ is called a Petri net (PN) iff*

(a) P, T, F *are finite sets with*
$P \cap T = \emptyset$, $P \cup T \neq \emptyset$, $F \subseteq (P \times T) \cup (T \times P)$ *and* $dom(F) \cup cod(F) = P \cup T$
(b) $V : F \longrightarrow \mathbb{N}^+$ *(weight of the arcs)*
(c) $m_o : P \longrightarrow \mathbb{N}$ *(initial marking)*

A **marking** of a PN is a function $m : P \longrightarrow \mathbb{N}$, such that $m(p)$ denotes the number of tokens at the place p. The **pre-sets** and **post-sets** of a transition t are given by ${}^\bullet t := \{p \mid p \in P \wedge (p,t) \in F\}$ and $t^\bullet := \{p \mid p \in P \wedge (t,p) \in F\}$, respectively. Each transition $t \in T$ induces the marking t^- and t^+, defined as follows:

$$t^-(p) = \begin{cases} V(p,t) & \text{,iff} \quad (p,t) \in F \\ 0 & \text{,iff} \quad (p,t) \notin F \end{cases} \qquad t^+(p) = \begin{cases} V(t,p) & \text{,iff} \quad (t,p) \in F \\ 0 & \text{,iff} \quad (t,p) \notin F \end{cases}$$

Moreover, Δt denotes $t^+ - t^-$. A transition $t \in T$ is **enabled (may fire)** at a marking m iff $t^- \leq m$ (e.g. $t^-(p) \leq m(p)$ for every place $p \in P$). When an enabled transition t at a marking m fires, this yields a new marking m' given by $m'(p) := m(p) + \Delta t(p)$ and denoted by $m \xrightarrow{t} m'$. Thus, the dynamical behavior of a classical PN is characterized by firing transitions that leads to change of the markings.

Example 1.

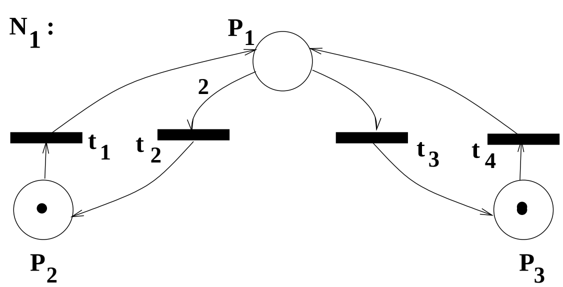

Fig. 1. N_1 - a (classical) Petri net

A marking m is a **reachable** one in N if there is a transition sequence which can fire starting at m_0 and ending at m. The set of all markings reachable in N is denoted by R_N.

Definition 2 (Time Petri net). *The structure* $Z = (P, T, F, V, m_o, I)$ *is called a Time Petri net (TPN) iff:*

(a) $S(Z) := (P, T, F, V, m_o)$ *is a PN.*
(b) $I : T \longrightarrow \mathbb{Q}_0^+ \times (\mathbb{Q}_0^+ \cup \{\infty\})$ *and* $I_1(t) \leq I_2(t)$ *for each* $t \in T$, *where* $I(t) = (I_1(t), I_2(t))$.

A TPN is called finite Time Petri net (FTPN) iff $I : T \longrightarrow \mathbb{Q}_0^+ \times \mathbb{Q}_0^+$.

I is the **interval function** of Z, $I_1(t)$ and $I_2(t)$ the **earliest firing time of** t ($eft(t)$) and the **latest firing time of** t ($lft(t)$), respectively. It is not difficult to see (cf. [3]) that considering TPNs with $I : T \longrightarrow \mathbb{N} \times (\mathbb{N} \cup \{\infty\})$ will not result in a loss of generality. Therefore, only such time functions I will be considered subsequently. Furthermore, conflict is used in the strong sense: two transitions t_1 and t_2 are in **conflict** iff $Ft_1 \cap Ft_2 \neq \emptyset$. The PN $S(Z)$ is referred to as the **skeleton** of Z.

Within this approach, the definition of a state is of fundamental importance for the ensuing theory. A state is characterized by a marking together with the momentary local time for enabled transitions or the sign $\sharp$ for the disabled transitions.

Example 2.

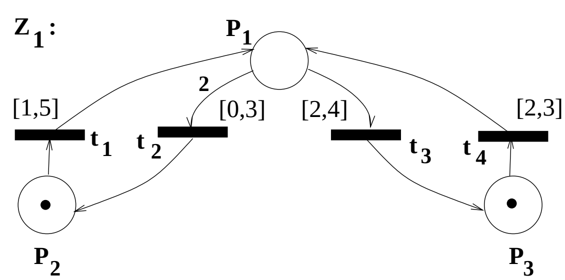

Fig. 2. Z_1 - a Time Petri net (with $S(Z_1) = N_1$)

Definition 3 (state). *Let* $Z = (P, T, F, V, m_o, I)$ *be a TPN and* $h : T \longrightarrow \mathbb{R}_0^+ \cup \{\#\}$. $z = (m, h)$ *is called a state in* Z *iff:*

(a) m *is a reachable marking in* $S(Z)$.
(b) $\forall t\ (\ (t \in T \wedge t^- \leq m) \ \longrightarrow\ h(t) \leq lft(t))$.
(c) $\forall t\ (\ (t \in T \wedge t^- \not\leq m) \ \longrightarrow\ h(t) = \#)$.

Interpretation of the notion "state" is as follows: within the net, each transition t has a clock $h(t)$. If t is enabled at a marking m, the clock of t $h(t)$ shows the time elapsed since t became most recently enabled. If t is disabled at m, the clock does not work (indicated by $h(t) = \#$). Thus, the vector h which is a vector of clocks is actually a transition marking and the already defined notion "marking" is in fact a place marking. In the following we call the places marking m a **p-marking** and the transitions marking h a **t-marking**.

The state $z_o := (m_o, h_o)$ with $h_o(t) := \begin{cases} 0 & \text{iff} \quad t^- \leq m_0 \\ \# & \text{iff} \quad t^- \not\leq m_0 \end{cases}$ is set as the **initial state** of the TPN Z.

Example 3.
The initial state in Z_1 , compare Fig. 2, is $z_0 = (\underbrace{(0,1,1)}_{p-marking}, \underbrace{(0,\sharp,\sharp,0)}_{t-marking})$.

Now the dynamic aspects of TPNs – changing from one state into another by firing a transition or by time elapsing – can be introduced:

Definition 4 (state changing). *Let $Z = (P,T,F,V,m_o,I)$ be a TPN, $\hat{t}$ be a transition in T and $z = (m,h)$, $z' = (m',h')$ be two states. Then*

(a) the transition $\hat{t}$ is **ready** *to fire in the state $z = (m,h)$, denoted by $z \xrightarrow{\hat{t}}$, iff*
  (i) $\hat{t}^- \leq m$ and
  (ii) $eft(\hat{t}) \leq h(\hat{t})$.

(b) the state $z = (m,h)$ is **changed** *into the state $z' = (m',h')$* **by firing the transition** *$\hat{t}$, denoted by $z \xrightarrow{\hat{t}} z'$, iff*
  (i) $\hat{t}$ is ready to fire in the state $z = (m,h)$
  (ii) $m' = m + \Delta\hat{t}$ and
  (iii) $\forall t \ (t \in T \longrightarrow h'(t) := \begin{cases} \# & \text{iff} \quad t^- \not\leq m' \\ h(t) & \text{iff} \quad t^- \leq m \wedge t^- \leq m' \wedge Ft \cap F\hat{t} = \emptyset \\ 0 & \text{otherwise} \end{cases})$.

(c) the state $z = (m,h)$ is **changed** *into the state $z' = (m',h')$* **by the time elapsing** *$\tau \in \mathbb{R}_0^+$, denoted by $z \xrightarrow{\tau} z'$, iff*
  (i) $m' = m$ and
  (ii) $\forall t \ (t \in T \wedge h(t) \neq \# \longrightarrow h(t) + \tau \leq lft(t))$ i.e. the time elapsing τ is possible, and
  (iii) $\forall t \ (t \in T \longrightarrow h'(t) := \begin{cases} h(t) + \tau & \text{iff} \quad t^- \leq m' \\ \# & \text{iff} \quad t^- \not\leq m' \end{cases})$.

The state $z = (m,h)$ is called an **integer state** iff $h(t)$ is an integer for each enabled transition t in m.

Example 4.
In the net Z_1, in the initial state, the transitions t_1 and t_4 are enabled, but neither t_1 nor t_4 may fire because of their time restrictions. Thus, z_0 can change into another state only as time elapses. For example, the change of states $z_0 \xrightarrow{1.3} z_1$ is feasible, where z_1 is given by $m_1 = m_0$ and $h_1 = (1.3, \sharp, \sharp, 1.3)$. Furthermore, z_1 can change into the state z_2 with $z_1 \xrightarrow{1.0} z_2$, where the state z_2 is given by $m_2 = m_1$ and $h_2 = (2.3, \sharp, \sharp, 2.3)$. In z_2 the transition t_4 can fire, yielding the state z_3 with: $m_3 = (1, 1, 0)$ and $h_3 = (2.3, \sharp, 0.0, \sharp)$. Now, as time progresses by 2, state z_3 changes into the state z_4, with $m_4 = m_3$, $h_4 = (4.3, \sharp, 2.0, \sharp)$. Subsequently, t_1 can fire and z_4 is changed into a state z_5 with $m_5 = (2, 0, 0)$ and $h_5 = (\sharp, 0.0, 2.0, \sharp)$. Afterwards t_2 is ready to fire. Firing t_2 at z_5 leads to $z_6 = (m_6, h_6)$ with $m_6 = (0, 1, 0)$ and $h_6 = (0.0, \sharp, \sharp, \sharp)$. Thus, the sequence $z_o \xrightarrow{1.3} z_1 \xrightarrow{1.0} z_2 \xrightarrow{t_4} z_3 \xrightarrow{2.0} z_4 \xrightarrow{t_1} z_5 \xrightarrow{t_2} z_6$ is executable in Z_1. The initial state and the states z_5 and z_6 are integer states, whereas the states z_1, z_2, z_3 and z_4 are not.

Definition 5 (reachable state, state space). *Let $Z = (P, T, F, V, m_o, I)$ be a TPN.*

(a) The state $z = (m, h)$ is called reachable in Z (starting at z_0), iff there exist states $z_1, z'_1, ..., z_n, z'_n$, transitions $t_1, ..., t_n$ and times $\tau_i \in \mathbb{R}_0^+, i = 1, ..., n$ and it holds

$$z_0 \xrightarrow{\tau_1} z_1 \xrightarrow{t_1} z'_1 \xrightarrow{\tau_2} z_2 \xrightarrow{t_2} z'_2 ... \xrightarrow{\tau_n} z_n \xrightarrow{t_n} z'_n.$$

(b) The set $StSp(Z)$ of all reachable states in Z is called the state space of Z.

It is easy to see that the set of all reachable p-markings in a TPN Z is the set $\{m \mid (m, h) \in StSp(Z)\}$, which will be denoted with R_Z.

The sequence of transitions $(t_1, ..., t_n)$ can fire in Z starting at z_0, because there is a sequence $(\tau_1, t_1, ..., \tau_n, t_n)$. We denote such a **transition sequence** $\sigma = (t_1, ..., t_n)$ **feasible**. The sequence $\sigma(\tau) = (\tau_1, t_1, ..., \tau_n, t_n)$ which is a concrete execution of σ in Z is called a **(feasible) run** of σ. It is clear that in a given TPN the state changes are achieved by alternating series of time elapsing and firing. Obviously, for a given run the transition sequence is well defined and for a given transition sequence there are infinitely many runs in general.

It is clear that the state space of a TPN is in general infinite and dense in terms of the time. On the one hand the set of reachable p-markings can be infinite. On the other hand, for a fixed p-marking the set of t-markings can be infinite. Nevertheless, it is possible to pick up some "essential" states only, so that qualitative and quantitative analysis is possible. In [3] it is shown, that the essential states are the integer states.

The state space can be considered as the union of all sets C_σ, which are defined below recursively:

Definition 6 (state class). *Let $Z = (P, T, F, V, m_o, I)$ be a TPN and σ be a feasible transition sequence. The set C_σ is called a state class, iff*

Basis: $C_e := \{z \mid \exists\tau(\tau \in \mathbb{R}_0^+ \wedge z_0 \xrightarrow{\tau} z)\}$

Step: *Let C_σ be already defined. Then $C_{\sigma t}$ is derived from C_σ by firing t (denoted by $C_\sigma \xrightarrow{t} C_{\sigma t}$), iff*

$$C_{\sigma t} := \{z \mid \exists z_1 \exists z_2 \exists \tau(z_1 \in C_\sigma \wedge \tau \in \mathbb{R}_0^+ \wedge z_1 \xrightarrow{t} z_2 \xrightarrow{\tau} z)\}.$$

In other words, the state class C_e is the set of all reachable states in Z that one gets after firing the empty transition sequence e at the initial state and afterwards all states that are reachable by state changing with all possible elapses of time. That is why, sometimes C_0 stands for C_e. Obviously, it holds $StSp(Z) = \bigcup_\sigma C_\sigma$.

3 Fundamental Property

The properties of a Petri net, both the classical one as well as the TPN, can be divided into two parts: There are static properties, like being pure, ordinary, free choice, extended simple, conservative, etc., and there are dynamic properties like being bounded, live, reachable, and having place- or transitions invariants, deadlocks, etc. While it is easy to prove the static behavior of a net using only the static definition, the dynamic behavior depends on both the static and dynamic definitions and is quite complicated to prove. That means that in order to get good knowledge of the dynamical behavior of the net, the set of all possible situations reachable for the net have to be known, i.e. the state space must be known. As already mentioned, this set is in general infinite and therefore hard to handle.

Nevertheless, it is possible to pick up some “essential” states only, so that qualitative and quantitative analysis is possible. In [3] it is shown, that the essential states are the integer-states.

The aim of this section is to justify the reduction of the state space of a certain TPN to a set of all its reachable integer-states as an adequate set for testing dynamical properties. To do this we use dynamic programming.

Notions, notations, definitions and approach referring to dynamic programing are used similar to [5]. We consider the problem as a non-optimization problem just like the abstract dynamic programming model considered in chapter 14.3 in [5] and solve it.

3.1 Parametric Description of the State Space

Let Z be an arbitrary TPN with the initial state $z_0 = (m_0, h_0)$. Let $\sigma = (t_1, ..., t_n)$ be a feasible transition sequence in Z and let $\sigma(\tau) = (\tau_1, t_1, ..., \tau_n, t_n)$ be a certain run. Obviously after firing $\sigma(\tau)$ a fix reachable state $z_{\sigma(\tau)}$ is yielded. When the times $\tau = (\tau_1, ..., \tau_n)$ are given parametrically with $X = (x_1, ..., x_n)$ then the achieved state $z_{\sigma(X)} = (m_{\sigma(X)}, h_{\sigma(X)})$ is a parametrical one. It is easy to see that the p-marking $m_{\sigma(X)}$ does not depend on X. It depends only on σ.

However, the t-marking $h_{\sigma(X)}$ depends on σ as well as on the parameter X. Furthermore, it is clear that for a concrete value of X with the additional condition that the refering run is feasible, the t-marking $h_{\sigma(X)}$ is well defined. Hence, an unique parametric state $z_\sigma := z_{\sigma(X)}$ can be assigned to each transition sequence σ , i.e. we can consider a function between the set of all transition sequences in Z and the set of all reachable states, defined parametrically. In order to make certain that σ is feasible we consider the parametric state z_σ together with a set of conditions for the values of X. Thus, we can consider the state space as the codomain of this function in connection with a set of certain additional conditions. It is clear that the codomain of this function is the parametric description of the state space of the TPN.

In order to define and investigate this function it is convenient to transfer the subject matter into the terminology of a first-order predicate calculus. In general this terminology is used similar to [8]. Let Z be an TPN and let $S := \{f^2, A^2, \$, u_t, v_t | t \in T\}$ be a set of a symbols where f^2 is a binary function symbol, A^2 is a binary relation symbol, \$ and u_t, v_t for each transition t are constant symbols (for short: $K := \{\$, u_t, v_t | t \in T\}$). Furthermore, we consider the S-structure $\mathcal{D} := [D, \omega]$ with $D := \mathbb{R}_0^+ \cup \{\sharp\}$ as a domain and

$$\omega(f^2) := +, \quad \omega(A^2) :=\leq, \quad \omega(u_t) := \mathrm{eft}(t), \quad \omega(v_t) := \mathrm{lft}(t), \quad \omega(\$) := \sharp.$$

Here, $+$ is a binary operation on D, which coincides with the well-known addition in $\mathbb{R}_0^+$. In this context, it is not necessary to specify $+$ any further. Similar considerations apply to $\leq$.

Let SUM be the union of the set of all terms, in which each variable appears at most once and constants do not appear at all, and the set which consists only of the constant symbol \$. COND denotes the set of all formulae $A^2 term_i term_j$ where $term_i \in \mathrm{SUM} \setminus \{\$\}$ and $term_j \in K \setminus \{\$\}$, or vice versa; the term in $\{term_i, term_j\} \cap (\mathrm{SUM} \setminus \{\$\})$ is denoted by $s(A^2 term_i term_j)$, the one in $\{term_i, term_j\} \cap (K \setminus \{\$\})$ with $r(A^2 term_i term_j)$. Let β be an arbitrary assignment for the set of variables $X_S := \{x_i \mid i \in \mathbb{N}\}$ into the domain D. Under the interpretation $\mathcal{I} := (\mathcal{D}, \beta)$, the value of a *term* with respect to β (which is an element in D) will be denoted by $[\![term]\!]_\beta$. In the following all used interpretations $\mathcal{I} = (\mathcal{D}, \beta)$ consists of the same S-structure $\mathcal{D}$, i.e. the interpretations differ to each other in the assignment β only. Thus, the assertion " β satisfy a formula c " means that the interpretation $\mathcal{I} = (\mathcal{D}, \beta)$ satisfy c.

Definition 1. *Let $Z = [P, T, F, V, m_0, I]$ be a TPN. The function $\delta : T^* \longrightarrow R_Z \times \mathrm{SUM}^{|T|} \times \wp(\mathrm{COND})$ is partially defined by induction:*

Basis: $\delta(e) := [m_e, \Sigma_e, B_e]$ *where*

(a) $m_e = m_0$

(b) $\Sigma_e(t) := \begin{cases} x_0 & \textit{iff } t^- \leq m_e \\ \$ & \textit{otherwise} \end{cases}$

(c) $B_e := \{A^2 \Sigma_e(t) v_t | t^- \leq m_e\}$.

Step: *Let* σ *be a transition sequence and assume that* $\delta(\sigma)$ *has been defined as* $[m_\sigma, \Sigma_\sigma, B_\sigma]$. *For a transition* $\hat{t}$ *with* $\Sigma_\sigma(\hat{t}) \neq \$$, $\delta(\sigma\hat{t}) = [m_{\sigma\hat{t}}, \Sigma_{\sigma\hat{t}}, B_{\sigma\hat{t}}]$ *is defined as follows:*

(a) $m_{\sigma\hat{t}} := m_\sigma + \Delta(\hat{t})$,

(b) $\Sigma_{\sigma\hat{t}}(t) := \begin{cases} \$ & \textit{iff } t^- \not\leq m_{\sigma\hat{t}} \\ x_{l(\sigma)+1} & \textit{iff } (t^- \not\leq m_\sigma \wedge t^- \leq m_{\sigma\hat{t}}) \vee \\ & (t^- \leq m_\sigma \wedge t^- \leq m_{\sigma\hat{t}} \wedge Ft \cap F\hat{t} \neq \emptyset) \\ f^2\, \Sigma_\sigma(t)\, x_{l(\sigma)+1} & \textit{otherwise} \end{cases}$

(c) $B_{\sigma\hat{t}} := B_{\sigma t} \cup \{A^2 u_t \Sigma_\sigma(\hat{t})\} \cup \{A^2 \Sigma_{\sigma\hat{t}}(t) v_t \mid t^- \leq m_{\sigma\hat{t}}\}$

With regard to the interpretation of the symbols for functions, predicates and constants in the logic defined above, the following notational conventions for terms in $SUM\backslash\{\$\}$, formulae in *COND*, and constants will be used for reasons of convenience and increased readability:

$$x_1 + \cdots + x_n := f^2 \ldots f^2 x_1 x_2 \ldots x_n,$$

$$term_1 \leq term_2 := A^2 term_1 term_2,$$

and instead of constant symbols their interpretation under ω.

Obviously, there is a close connection between the state classes and the mapping δ defined above: $C_\sigma = \{(m_\sigma, \Sigma_\sigma(t)) \mid B_\sigma\}$.

Example 1.
We consider Z_1 again (cf. Fig. 2). The state class C_0 has the parametric form

$$C_0 = \{((0,1,1),(x_1,\sharp,\sharp,x_1)) \mid 0 \leq x_1 \leq 3\}.$$

After firing t_4 from an arbitrary state, belonging to C_0, the set C_{t_4} will be achieved, and C_{t_4} has the parametric form

$$C_{t_4} = \{((1,1,0),(x_1+x_2,\sharp,x_2,\sharp)) \mid 2 \leq x_1 \leq 3, x_1+x_2 \leq 5, 0 \leq x_2 \leq 4\}.$$

The parametric state $((0,1,1),(x_1,\sharp,\sharp,x_1))$ defines the set C_0, and the parametric state $((1,1,0),(x_1+x_2,\sharp,x_2,\sharp))$ defines the set C_{t_4}. The parameter x_1 in C_0 has to satisfy the constraint $0 \leq x_1 \leq 3$, and the parameters x_1, x_2 in C_{t_4} have to satisfy the three constraints $2 \leq x_1 \leq 3, x_1+x_2 \leq 5, 0 \leq x_2 \leq 4$. At the same time, C_0 is the parametric description of the empty transition sequence, and C_{t_4} is the parametric description of the transition sequence t_4. Thus, the parametric description for the transition sequence (t_4,t_3) is

$$C_{t_4t_3} = \{((0,1,1),(x_1+x_2+x_3,\sharp,\sharp,x_3)) \mid \begin{array}{ll} 2 \leq x_1 \leq 3, & x_1+x_2 \leq 5, \\ 2 \leq x_2 \leq 4, & x_1+x_2+x_3 \leq 5, \\ 0 \leq x_3 \leq 3 & \end{array}\}.$$

Accordingly, for (t_4,t_3,t_4) the parametric description is $C_{t_4t_3t_4} =$

$$\{((1,1,0),(x_1+x_2+x_3+x_4),\sharp,x_4,\sharp)) \mid \begin{array}{ll} 2 \leq x_1 \leq 3, & x_1+x_2 \leq 5, \\ 2 \leq x_2 \leq 4, & x_1+x_2+x_3 \leq 5, \\ 2 \leq x_3 \leq 3, & x_1+x_2+x_3+x_4 \leq 5, \\ 0 \leq x_4 \leq 4 & \end{array}\}.$$

3.2 Properties and Dynamic Programming

For each feasible transition sequence σ with $\delta(\sigma) = [m_\sigma, \Sigma_\sigma, B_\sigma]$ it is easy to prove that the following holds:

Remark 1. For each state $z \in C_\sigma$, there is an assignment $\beta : X \longrightarrow \mathbb{R}_0^+$ such that: $z = (m_\sigma, [\![\Sigma_\sigma]\!]_\beta)$ and $\bigwedge_{b \in B_\sigma} \beta$ satisfies b.

Also easy to prove is the converse:

Remark 2. For each assignment $\beta : X \longrightarrow \mathbb{R}_0^+$ with $\bigwedge_{b \in B_\sigma} \beta$ satisfies b, the state $z := (m_\sigma, [\![\Sigma_\sigma]\!]_\beta)$ is in C_σ.

By induction on σ it can be proved, too, that remark 3, remark 4 and remark 5 are true:

Remark 3. For any two transitions t_i and t_j in Z with $\Sigma_\sigma(t_i) = x_{i_0} + x_{i_1} + \cdots + x_{i_k}$ and $\Sigma_\sigma(t_j) = x_{j_0} + x_{j_1} + \cdots + x_{j_l}$, it follows that $i_{k-r} = j_{l-r}$ for all $r = 0, 1, \ldots, \min\{k, l\}$.

Remark 4. For each transition $t \in T$ it is true that: if $\Sigma_\sigma(t) = x_i + \ldots + x_j$ then each variable x_k with $i \le k \le j$ also appears in $\Sigma_\sigma(t)$.

Remark 5. For each $\mathit{term} \in \mathit{SUM}$, which is a part of a formula b in B_σ, it is true that: if $\mathit{term} = x_i + \ldots + x_j$ then each variable x_k with $i \le k \le j$ also appears in term.

The following theorem supplies the fundamental property of the TPN that allows one to consider an essential reduced state space.

Theorem 1. *Let $Z = [P, T, F, V, m_0, I]$ be a TPN, σ a transition sequence of length n, with $\delta(\sigma) = [m_\sigma, \Sigma_\sigma, B_\sigma]$ and $\hat{\beta} : X \longrightarrow \mathbb{R}_0^+$ an assignment such that $\forall c (c \in B_\sigma \rightarrow \hat{\beta}$ satisfies $c)$. Then there exists an assignment $\beta^* : X \longrightarrow \mathbb{N}$ such that:*

(1) $\forall c (c \in B_\sigma \rightarrow \beta^*$ *satisfies* $c)$
(2) $\forall t (t \in T \wedge t^- \le m_\sigma \rightarrow [\![\Sigma_\sigma(t)]\!]_{\beta^*} \le [\![\Sigma_\sigma(t)]\!]_{\hat{\beta}})$
(3) $[\![\sum_{k=0}^{n} x_k]\!]_{\beta^*} \le [\![\sum_{k=0}^{n} x_k]\!]_{\hat{\beta}}$.

The meaning of the theorem is that if $\hat{\beta}$ supplies a feasible run for σ with real numbers for elapsed time then it is possible to find a further feasible run for σ with integer time elapses (meaning of (1)). The differences between the respective time elapses in both runs are allways smaller than 1 (follows from the construction of β^* given below). The difference between the clocks of each enabled transition at m_σ after the first run and after the second one is smaller than 1, too (meaning of (2)). And finally also the difference between the total times of both runs is smaller than 1 (meaning of (3)).

Idea of the proof: The integer values $\beta^*(x_1), \beta^*(x_2), \cdots, \beta^*(x_n)$ defined by the assignment β^* will be explicitly constructed out of the given assignment $\hat{\beta}$ by successively transforming each non-integral real number to the nearby integer in $(n+1)$ steps.

As the default value $\lfloor r \rfloor$ will be taken, in order to ensure that the second and third property stated in the theorem are satisfied. By doing so, the restriction yielded by the first property will be somewhat loosened, i.e., temporarily it is sufficient that a required condition is "almost" satisfied. This means, that for each formula c in B_σ, the value of the non-constant term $s(c)$ under the current assignment will only have to lie in a certain neighbourhood of the initial value $[\![s(c)]\!]_{\hat{\beta}}$.

If, by taking the integer part of the rational value for a certain variable, such a neighbourhood will be left for at least one condition, $\lceil r \rceil$ will be taken instead. The largest part of the proof aims to show that the three requirements stated above will also be satisfied (with the first one once again "loosened") in this case.

To complete the proof, it then remains to verify, that for the finally constructed assignment, which takes only integer values, the "loosened" version of the first requirement is equivalent to the original one.

Construction of β^*

Let X_σ be the set of all variables which appear in B_σ, i.e. $X_\sigma := \{x_0, x_1, \ldots, x_n\}$ define a finite sequence of assignments $\beta_i : X_\sigma \longrightarrow \mathbb{R}_0^+$ by induction:

Basis: $\beta_0 : X_\sigma \longrightarrow \mathbb{R}_0^+$ with $\beta_0(x) := \hat{\beta}(x)$ for all $x \in X_\sigma$.

Step: Assume that β_{i-1} has been defined. In order to describe the construction of β_i, the following function is used:

$$\underline{\beta_i}(x) := \begin{cases} \beta_{i-1}(x) & \text{iff } x \neq x_{n-(i-1)} \\ \lfloor \beta_{i-1}(x) \rfloor & \text{otherwise} \end{cases}$$

Now define $\beta_i : X_\sigma \longrightarrow \mathbb{R}_0^+$ by

$$\beta_i(x) := \begin{cases} \beta_{i-1}(x) & \text{iff } x \neq x_{n-i+1} \\ \lfloor \beta_{i-1}(x) \rfloor & \text{iff } x = x_{n-i+1} \wedge \forall c (c \in B_\sigma \rightarrow \lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1 < [\![s(c)]\!]_{\underline{\beta_i}}) \\ \lceil \beta_{i-1}(x) \rceil & \text{otherwise} \end{cases}$$

In words, $\underline{\beta_i}$ describes how the value of the variable $x_{n-(i-1)}$ currently considered should be modified in the default case. However, if for any condition c in B_σ, the value of the term $s(c)$ is decreased below the bound $\lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1$ as a consequence, $x_{n-(i-1)}$ will be set to $\lceil \beta_{i-1}(x_{n-(i-1)}) \rceil$ instead (cf. Fig. 3).

Note that in each step the value of exactly one variable is changed, and that the value changed in a specific step is not altered by other steps before or afterwards. This implies in particular, that for variables x_k with $k < n - (j-1)$

$$\beta_j(x_k) = \beta_{j-1}(x_k) = \ldots = \beta_0(x_k) \tag{1}$$

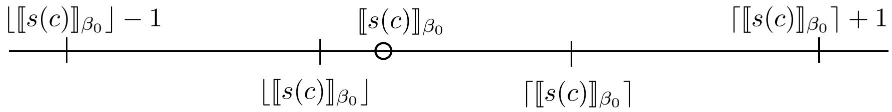

Fig. 3. Position of the real number $[\![s(c)]\!]_{\beta_0}$ and the integers $\lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1$, $\lfloor [\![s(c)]\!]_{\beta_0} \rfloor$, $\lceil [\![s(c)]\!]_{\beta_0} \rceil$ and $\lceil [\![s(c)]\!]_{\beta_0} \rceil + 1$

and that for variables x_k with $k \geq n - (j-1)$

$$\beta_j(x_k) = \beta_{j+1}(x_k) = \ldots = \beta_{n+1}(x_k) \tag{2}$$

Furthermore, if $\beta_0(x_{n-(i-1)})$ is already an integer, then β_i leaves $x_{n-(i-1)}$ unaltered, since for any integer k, $k = \lfloor k \rfloor = \lceil k \rceil$.

Obviously all values $\beta_{n+1}(x_0), \beta_{n+1}(x_1), \ldots, \beta_{n+1}(x_n)$ are integers. Thus

$$\beta^*(x_j) := \beta_{n+1}(x_j) \qquad \text{for each } j = 0, 1, \ldots, n.$$

The ensuing tableau points up the successive constructing of β^* from $\hat{\beta}$:

β	$\beta(x_0)$	$\beta(x_1)$	$\cdots$	$\beta(x_{n-j})$	$\beta(x_{n-(j-1)})$	$\cdots$	$\beta(x_{n-1})$	$\beta(x_n)$
$\hat{\beta} := \beta_0$	r	r	$\cdots$	r	r	$\cdots$	r	r
β_1	r	r	$\cdots$	r	r	$\cdots$	r	i
β_2	r	r	$\cdots$	r	r	$\cdots$	i	i
$\vdots$			$\vdots$			$\vdots$		
β_j	r	r	$\cdots$	r	i	$\cdots$	i	i
$\vdots$			$\vdots$			$\vdots$		
β_n	r	i	$\cdots$	i	i	$\cdots$	i	i
$\beta^* := \beta_{n+1}$	i	i	$\cdots$	i	i	$\cdots$	i	i

The r in the tableau above stands for (nonnegative) real number and i – for (nonnegative) integer.

The following three assertions about this sequence of assignments bundled in the next lemma are proved by induction for each i:

Lemma 1. *For all* $i \in \{0, 1, \ldots, n+1\}$ *it holds:*

(a) $\forall c (c \in B_\sigma \rightarrow [\![s(c)]\!]_{\beta_i} \in (\, \lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1, \lceil [\![s(c)]\!]_{\beta_0} \rceil + 1 \,)\,)$

(b) $\forall t (t \in T \wedge t^- \leq m_\sigma \rightarrow [\![\Sigma_\sigma(t)]\!]_{\beta_i} \leq [\![\Sigma_\sigma(t)]\!]_{\beta_0})$

(c) $[\![\sum_{k=0}^{n} x_k]\!]_{\beta_i} \leq [\![\sum_{k=0}^{n} x_k]\!]_{\beta_0}$

Before starting the proof please note that $(a), (b)$ and (c) from lemma 1 supply a finite number of inequalities. Namely (a) derives from each inequality of VC[1] two further inequalities:

$$[\![s(c)]\!]_{\beta_i} \geq \lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1 \text{ and}$$

$$[\![s(c)]\!]_{\beta_i} \leq \lceil [\![s(c)]\!]_{\beta_0} \rceil + 1,$$

(b) supplies at most as many inequalities as the number of transitions in the TPN and (c) delivers one inequality.

And as a last discussion before starting the proofs let us consider the assertion of the theorem 1 more precisely and elaborate the connection to dynamic programming. The theorem 1 solves the following **problem** :

Input: a TPN, a transition sequence $\sigma = (t_1, \ldots, t_n)$ and a sequence of $(n+1)$ real numbers, $(\hat{\beta}(x_0), \hat{\beta}(x_1), \cdots, \hat{\beta}(x_n))$ subject to a certain finite set VC[1] of conditions.

Output: a sequence of $(n+1)$ integers, $(\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_n))$ subject to VC.

Thus, let now consider the solving of the output as the problem P^*, that means:

Problem $\mathbf{P}^*$: *Compute a sequence of* $(n+1)$ *integers,* $(\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_n))$ *subject to* VC^*[2].

The solution strategy for the problem P^* is a typical dynamic programming's one:

In the following disquisition about the connection between solving the problem P^* and dynamic programming (DP) we use certain notions and notations typical for both the theory of DP, and the theory of TPN. However, the meanings of the notions state, state space, transition function etc. are different between the two theories. The meaning of these should however be clear from the context. Once again, DP notions and notations used here are the same as those in [5].

Thus, the *target problem,* now is *Problem P^**.

The *set of solutions* of this problem $\mathrm{z^O}$ is set of all sequences of integers $(\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_n))$ subject to VC.

---

[1] VC is derived from the set of formulae B_σ where relation- and function symbols are interpreted, i.e. VC is a finite set of inequalities with the variables $x_0, x_1, \ldots, x_n$. Thus $(\hat{\beta}(x_0), \hat{\beta}(x_1), \cdots, \hat{\beta}(x_n))$ subject to the finite set VC means that the real numbers $\hat{\beta}(x_0), \hat{\beta}(x_1), \cdots, \hat{\beta}(x_n)$ satisfy all inequalities of VC. The semantics of this is that the run $\sigma(\hat{\beta}) = (\hat{\beta}(x_0), t_1, \hat{\beta}(x_1), t_2, \cdots, t_n, \hat{\beta}(x_n)\ )$ is a feasible one in the TPN.

[2] VC^* is the union of the set VC and the finite set of inequalities supply by $(a), (b)$ and (c). The semantics of this is that if the run $\sigma(\hat{\beta})$ is feasible one in the TPN, than the run $\sigma(\beta^*)$ is feasible in the TPN, too, and the set of all transitions which are ready to fire after the run $\sigma(\beta^*)$ is the same as the set of all transitions which are ready to fire after the run $\sigma(\hat{\beta})$.

The *state space* (for P^*) is the set $S = \{0, 1, \ldots, n\}$.
The family of the *modified problems* $P^*(s)$, s∈S are obviously the problems

Problem P^*(s): *Compute a sequence of* $(n+1)$ *numbers,* $(\beta^s(x_0), \beta^s(x_1), \cdots, \beta^s(x_n)$ with $\beta^s(x_0), \beta^s(x_1), \cdots, \beta^s(x_{n-s}))$ *are reals and* $\beta^s(x_{n-(s-1)}), \cdots, \beta^s(x_n)$ *are integers subject to* VC^*.

The existence of the assignment β^s for each s is verified by lemma 1.

The *set of its critical states* is the singleton $\mathrm{S^O} = \{n\}$.
The *set of its terminal states* is the singleton $\mathrm{S^t} = \{0\}$.
Thus the *set of non-terminal states* is $\mathrm{S}'' = \mathrm{S} \setminus \mathrm{S^t} = \{1, 2, \ldots, n\}$.
The T-*linker* $\mathrm{L_T}$ has the form $\mathrm{L_T(z(s^O)) = z^O = z(s^O)}$.
The *transition function* t is defined as

$$\mathrm{t(s) := s - 1}, \qquad \mathrm{s} \in \mathrm{S}''.$$

And lastly the *linker* L is clearly given by

$$\begin{aligned} \mathrm{z(s)} &= \mathrm{L(s,\ \{(s',z(s')) \mid s' \in t(s)\})}, \qquad \forall s \in \mathrm{S}'' \\ &= \mathrm{L(s, z(t(s)))} \\ &= \mathrm{L(s,z(s\text{-}1))} := \beta_s \end{aligned}$$

and β_s is defined as in the constuction of β^*.

Now we are going to verify Lemma 1. The subsequent example 3.1 illustrates the use of DP for a concrete TPN.

Proof of Lemma 1:
Induction on i:
Basis: For $i = 0$, all three assertions are trivially true.
Step: Assume that the assertions have been justified for each of $1, \ldots, i$, and consider the case $i+1$. If $\beta_i(x_{n-i}) \in \mathbb{N}$, then $\beta_{i+1} = \beta_i$ and all assertions follow immediately from the induction hypothesis. Therefore, it may be assumed that $\beta_i(x_{n-i})$ is not an integer.
Two cases need to be considered:

Case 1: $\beta_{i+1}(x_{n-i}) = \lfloor \beta_i(x_{n-i}) \rfloor$
Hence, it holds:

$$\beta_{i+1}(x) \leq \beta_i(x) \ \text{ for each } \ x \in X_\sigma. \tag{3}$$

to *(a):*
Let b be any condition (i.e. a formula) in B_σ. If x_{n-i} does not appear in $s(b)$, then $[\![s(b)]\!]_{\beta_{i+1}} = [\![s(b)]\!]_{\beta_i}$, and the first assertion follows from the induction hypothesis. Hence, assume that x_{n-i} is in $s(b)$.

Since $\beta_{i+1}(x) \leq \beta_i(x)$ for each $x \in X_\sigma$, it is evident that

$$[\![s(b)]\!]_{\beta_{i+1}} \leq [\![s(b)]\!]_{\beta_i}$$

By the induction hypothesis, $[\![s(b)]\!]_{\beta_i} < \lceil [\![s(b)]\!]_{\beta_0} \rceil + 1$, so the previous inequality becomes

$$[\![s(b)]\!]_{\beta_{i+1}} < \lceil [\![s(b)]\!]_{\beta_0} \rceil + 1 \tag{4}$$

As $\beta_{i+1}(x_{n-i})$ has been set to $\lfloor \beta_i(x_{n-i}) \rfloor$, the corresponding criteria in the definition of β_{i+1}

$$\forall c(c \in B_\sigma \rightarrow \lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1 < [\![s(c)]\!]_{\underline{\beta_{i+1}}})$$

has been fulfilled. Since $\underline{\beta_{i+1}} = \beta_{i+1}$, it follows for the condition b in particular:

$$\lfloor [\![s(b)]\!]_{\beta_0} \rfloor - 1 < [\![s(b)]\!]_{\beta_{i+1}} \tag{5}$$

Because b was chosen arbitrarily, the inequalities (4) and (5) combine prove the first assertion in the case $i+1$, and therefore complete the induction step.

to *(b):*

The inequality (3) leads immediately to

$$[\![\Sigma_\sigma(t)]\!]_{\beta_{i+1}} \leq [\![\Sigma_\sigma(t)]\!]_{\beta_i}$$

for each transition t which is enabled after the firing of σ, and because of the induction hypothesis

$$[\![\Sigma_\sigma(t)]\!]_{\beta_i} \leq [\![\Sigma_\sigma(t)]\!]_{\beta_0}$$

the second assertion *(b)* is proved.

to *(c):*

The inequality (3) instantaneously yields *(c)*.

Case 2: $\beta_{i+1}(x_{n-i}) = \lceil \beta_i(x_{n-i}) \rceil$

i.e. a formula $\tilde{c}$ exists in B_σ, such that

$$[\![s(\tilde{c})]\!]_{\underline{\beta_{i+1}}} \leq \lfloor [\![s(\tilde{c})]\!]_{\beta_0} \rfloor - 1 \tag{6}$$

and thus x_{n-i} does appear in $\tilde{c}$.

Further it is true in this case that:

$$\beta_i(x) \leq \beta_{i+1}(x) \;\; \text{for each} \;\; x \in X_\sigma. \tag{7}$$

to *(a):*

Let b be any formula in B_σ again. Then it holds:

$$\begin{aligned} \lfloor [\![s(b)]\!]_{\beta_0} \rfloor - 1 &< [\![s(b)]\!]_{\beta_i} && \text{ind. hypothesis} \\ &\leq [\![s(b)]\!]_{\beta_{i+1}} && \text{because of (7)} \end{aligned} \tag{8}$$

On the other hand, it is true for the formula $\tilde{c}$:

$$
\begin{aligned}
[\![s(\tilde{c})]\!]_{\beta_{i+1}} &= [\![s(\tilde{c})]\!]_{\beta_i} - \beta_i(x_{n-i}) + \beta_{i+1}(x_{n-i}) \\
&= [\![s(\tilde{c})]\!]_{\beta_i} - \beta_i(x_{n-i}) + \lceil \beta_i(x_{n-i}) \rceil \\
&= [\![s(\tilde{c})]\!]_{\beta_i} - \beta_i(x_{n-i}) + \lfloor \beta_i(x_{n-i}) \rfloor + 1 \\
&= [\![s(\tilde{c})]\!]_{\underline{\beta_{i+1}}} + 1 \\
&\le \lfloor [\![s(\tilde{c})]\!]_{\beta_0} \rfloor \qquad\qquad \text{because of (6)}
\end{aligned}
$$

i.e.

$$[\![s(\tilde{c})]\!]_{\beta_{i+1}} \le \lfloor [\![s(\tilde{c})]\!]_{\beta_0} \rfloor \tag{9}$$

and therefore $\quad [\![s(\tilde{c})]\!]_{\beta_{i+1}} \le [\![s(\tilde{c})]\!]_{\beta_0} \quad$ is true, too. (10)

Because of (8) and (9) assertion (a) holds for the formula $\tilde{c}$.
Now suppose that

$$[\![s(b)]\!]_{\beta_{i+1}} \ge \lceil [\![s(b)]\!]_{\beta_0} \rceil + 1 \tag{11}$$

which in particular implies

$$[\![s(b)]\!]_{\beta_{i+1}} \ge [\![s(b)]\!]_{\beta_0} + 1. \tag{12}$$

Let $j_{\tilde{c}}$ and $k_{\tilde{c}}$ be the minimal and maximal variable index which appears in $s(\tilde{c})$, respectively. Refering to Remark 5. above it is clear that

$$s(\tilde{c}) = x_{j_{\tilde{c}}} + x_{j_{\tilde{c}}+1} + \ldots + x_{n-i} + x_{n-(i-1)} + \ldots + x_{k_{\tilde{c}}}. \tag{13}$$

Similarly, there are indices j_b and k_b such that

$$s(b) = x_{j_b} + x_{j_b+1} + \ldots + x_{n-i} + x_{n-(i-1)} + \ldots + x_{k_b}. \tag{14}$$

Hence, it holds for the indices $n-i, k_{\tilde{c}}$ and k_b:

$$
\begin{array}{rcc}
n - i \le k_{\tilde{c}} & \text{and} & n - i \le k_b \\
\text{i.e. } n - k_{\tilde{c}} < i + 1 & \text{and} & n - k_b < i + 1
\end{array}
$$

The values of the variables according to the two assignments β_0 and β_{i+1} are:

$$
\underbrace{\beta_{i+1}(x_{j_{\tilde{c}}}) \quad \cdots \quad \beta_{i+1}(x_{n-(i+1)})}_{\substack{\beta_{i+1} \,=\, \beta_0 \\ \uparrow \quad\;\; \uparrow \\ \text{real} \quad \text{real}}}
\underbrace{\beta_{i+1}(x_{n-i}) \quad \cdots \quad \beta_{i+1}(x_{k_{\tilde{c}}})}_{\substack{\beta_{i+1} \,\ne\, \beta_0 \\ \uparrow \quad\;\; \uparrow \\ \text{int.} \quad \text{real}}}
\tag{15}
$$

According to the definition (construction) of β^* it is clear, that the assignment β_{i+1} changes the value of the variable x_{n-i} and the assignment β_{n-r} changes the value of x_{r+1}.
Hence, refering to (13), (10) may be rewritten as

$$
\begin{aligned}
&(\beta_{i+1}(x_{j_{\tilde{c}}}) - \beta_0(x_{j_{\tilde{c}}}))+\\
&(\beta_{i+1}(x_{j_{\tilde{c}}+1}) - \beta_0(x_{j_{\tilde{c}}+1})) + \ldots +\\
&(\beta_{i+1}(x_{n-i}) - \beta_0(x_{n-i})) + (\beta_{i+1}(x_{n-i+1}) - \beta_0(x_{n-i+1})) + \ldots +\\
&(\beta_{i+1}(x_{k_{\tilde{c}}}) - \beta_0(x_{k_{\tilde{c}}})) \leq 0
\end{aligned}
\tag{16}
$$

and refering to (15), (16) may be rewritten as

$$
\begin{aligned}
&(\beta_{i+1}(x_{n-i}) - \beta_0(x_{n-i}) + (\beta_{i+1}(x_{n-i+1}) - \beta_0(x_{n-i+1})) + \ldots +\\
&(\beta_{i+1}(x_{k_{\tilde{c}}}) - \beta_0(x_{k_{\tilde{c}}})) \leq 0
\end{aligned}
\tag{17}
$$

Similarly, (12) and (14) yield

$$
\begin{aligned}
&(\beta_{i+1}(x_{n-i}) - \beta_0(x_{n-i})) + (\beta_{i+1}(x_{n-i+1}) - \beta_0(x_{n-i+1})) + \ldots +\\
&(\beta_{i+1}(x_{k_b}) - \beta_0(x_{k_b})) \geq 1
\end{aligned}
\tag{18}
$$

Three sub-cases need to be considered:

Case 2.1: $k_{\tilde{c}} = k_b$

Then it holds:

$$
\begin{aligned}
1 &\leq [\![s(b)]\!]_{\beta_{i+1}} - [\![s(b)]\!]_{\beta_0} && \text{because of (12)}\\
&= [\![s(\tilde{c})]\!]_{\beta_{i+1}} - [\![s(\tilde{c})]\!]_{\beta_0} && \text{because of (17) and 18}\\
&\leq 0 && \text{because of (10)}
\end{aligned}
$$

Clearly, this is a contradiction.

Case 2.2: $k_{\tilde{c}} < k_b$

In this case the two terms $s(\tilde{c})$ and $s(b)$ have the form:

$$
\begin{aligned}
s(\tilde{c}) &= x_{j_{\tilde{c}}} + \cdots + x_{n-i} + \ \cdots \ + x_{k_{\tilde{c}}}\\
s(b) &= x_{j_b} + \cdots + x_{n-i} + \ \cdots \ + x_{k_{\tilde{c}}} + \ \cdots \ + x_{k_b}
\end{aligned}
$$

Because of (1) and (2) this leads to the following values of the variables in $s(b)$ according to the assignments $\beta_0, \beta_{n-k_{\tilde{c}}}$ and β_{n+1} :

$$
s(b) \ = \ \overbrace{\underbrace{x_{j_b} + \ \cdots +}_{\substack{\beta_{i+1} = \beta_{n-k_{\tilde{c}}}\\ \uparrow \qquad \uparrow\\ \text{real} \quad \text{real}}} \ \underbrace{x_{n-i} + \ \cdots \ + x_{k_{\tilde{c}}}}_{\substack{\beta_{i+1} \neq \beta_{n-k_{\tilde{c}}}\\ \uparrow \qquad \uparrow\\ \text{int.} \quad \text{real}}}}^{\beta_0 = \beta_{n-k_{\tilde{c}}}} \overbrace{\underbrace{+ \ \cdots \ + x_{k_b}}_{\substack{\beta_{i+1} = \beta_{n-k_{\tilde{c}}}\\ \uparrow \qquad \uparrow\\ \text{int.} \quad \text{int.}}}}^{\beta_0 \neq \beta_{n-k_{\tilde{c}}}}. \tag{19}
$$

Now consider the values of the term $s(b)$ according to the assignments β_{i+1} and $\beta_{n-k_{\tilde{c}}}$.

Because of (19), β_o and $\beta_{n-k_{\tilde{c}}}$ agree on all variables with indices not greater then $k_{\tilde{c}}$. That is way inequality (17) leads to

$$\begin{aligned}&(\beta_{i+1}(x_{n-i}) - \beta_{n-k_{\tilde{c}}}(x_{n-i})+\\&(\beta_{i+1}(x_{n-i+1}) - \beta_{n-k_{\tilde{c}}}(x_{n-i+1})) + \ldots +\\&(\beta_{i+1}(x_{k_{\tilde{c}}}) - \beta_{n-k_{\tilde{c}}}(x_{k_{\tilde{c}}})) \leq 0\end{aligned} \tag{20}$$

Thus (19) and (20) yield

$$[\![s(b)]\!]_{\beta_{i+1}} - [\![s(b)]\!]_{\beta_{n-k_{\tilde{c}}}} \leq 0 \tag{21}$$

But (11) and (21) then yield

$$[\![s(b)]\!]_{\beta_{n-k_{\tilde{c}}}} \geq \lceil [\![s(b)]\!]_{\beta_0} \rceil + 1$$

which contradicts the induction hypothesis for $n-k_{\tilde{c}}$, then $n-k_{\tilde{c}} < i+1$.

Case 2.3: $k_{\tilde{c}} > k_b$

Now the two terms $s(\tilde{c})$ and $s(b)$ have the form:

$$\begin{aligned}s(\tilde{c}) &= x_{j_{\tilde{c}}} + \cdots + x_{n-i} + \ \cdots \ + x_{k_b} + \ \cdots \ x_{k_{\tilde{c}}}\\ s(b) &= x_{j_b} + \cdots + x_{n-i} + \ \cdots \ + x_{k_b}\end{aligned}$$

Analogosly to Case 2.2 this leads to the following values of the variables in $s(b)$ according to the assignments β_0, β_{n-k_b} and β_{n+1} :

$$s(\tilde{c}) \ = \ \overbrace{\underbrace{x_{j_b} + \ \cdots \ +}_{\substack{\beta_{i+1} = \beta_{n-k_b}\\ \uparrow \quad\quad \uparrow\\ \text{real} \quad \text{real}}} \ \underbrace{x_{n-i} + \ \cdots \ + x_{k_b}}_{\substack{\beta_{i+1} \neq \beta_{n-k_b}\\ \uparrow \quad\quad \uparrow\\ \text{int.} \quad \text{real}}}}^{\beta_0 = \beta_{n-k_b}} \overbrace{\underbrace{+ \ \cdots \ + x_{k_{\tilde{c}}}}_{\substack{\beta_{i+1} = \beta_{n-k_b}\\ \uparrow \quad\quad \uparrow\\ \text{int.} \quad \text{int.}}}}^{\beta_0 \neq \beta_{n-k_b}} . \tag{22}$$

Now the term $s(\tilde{c})$ will be evaluated by the assignments β_{i+1} and β_{n-k_b} and afterwards $[\![s(b)]\!]_{\beta_{i+1}}$ and $[\![s(b)]\!]_{\beta_{n-k_b}}$ will be compared:
Because of (22), β_{i+1} and β_{n-k_b} agree on all variables with indices smaller than $n-i$ and also agree on all variables with indices greater than k_b. That is why the inequality (18) leads to

$$\begin{aligned}&(\beta_{i+1}(x_{n-i}) - \beta_{n-k_b}(x_{n-i})+\\&(\beta_{i+1}(x_{n-i+1}) - \beta_{n-k_b}(x_{n-i+1})) + \ldots +\\&(\beta_{i+1}(x_{k_b}) - \beta_{n-k_b}(x_{k_b})) \geq 1\end{aligned} \tag{23}$$

Hence, (23) together with (13) show that

$$[\![s(\tilde{c})]\!]_{\beta_{i+1}} - [\![s(\tilde{c})]\!]_{\beta_{n-k_b}} \geq 1 \tag{24}$$

But (9) and (24) then yield

$$[\![s(\tilde{c})]\!]_{\beta_{n-k_b}} \leq \lfloor [\![s(\tilde{c})]\!]_{\beta_0} \rfloor - 1$$

which contradicts the induction hypothesis for $n-k_b$, then $n-k_b < i+1$. The assumption (11) has led to a contradiction in all three sub-cases. Therefore the following inequality must hold:

$$[\![s(b)]\!]_{\beta_{i+1}} < \lceil [\![s(b)]\!]_{\beta_0} \rceil + 1 \tag{25}$$

Because b was chosen arbitrarily, (8) and (25) prove the first assertion *(a)*.

to *(b):*
Let t be a transition which is not disabled after the firing of σ. If x_{n-i} does not appear in $\Sigma_\sigma(t)$, then $[\![\Sigma_\sigma(t)]\!]_{\beta_{i+1}} = [\![\Sigma_\sigma(t)]\!]_{\beta_i}$, and $[\![\Sigma_\sigma(t)]\!]_{\beta_{i+1}} \leq [\![\Sigma_\sigma(t)]\!]_{\beta_0}$ follows from the induction hypothesis. Therefore, assume that x_{n-i} does appear in $\Sigma_\sigma(t)$.

By the construction of Σ_σ (cf. remark 3), x_n appears in every component which is not \$. Referring to remark 4, this implies that there is an index j_t such that

$$\Sigma_\sigma(t) = x_{j_t} + x_{j_t+1} + \ldots + x_{n-i} + x_{n-(i-1)} + \ldots + x_n \tag{26}$$

Because of (19), β_{i+1} and $\beta_{n-k_{\tilde{c}}}$ agree on all variables with indices smaller than $n-i$ and on all variables with indices greater than $k_{\tilde{c}}$.

Hence, (20) together with (26) show that

$$[\![\Sigma_\sigma(t)]\!]_{\beta_{i+1}} - [\![\Sigma_\sigma(t)]\!]_{\beta_{n-k_{\tilde{c}}}} \leq 0 \tag{27}$$

Using the induction hypothesis for $n-k_{\tilde{c}}$, (27) yields

$$[\![\Sigma_\sigma(t)]\!]_{\beta_{i+1}} \leq [\![\Sigma_\sigma(t)]\!]_{\beta_0}$$

Since t was chosen arbitrarily, the second assertion *(b)* is also proved.
to *(c):*
Again, because of (19), β_{i+1} and $\beta_{n-k_{\tilde{c}}}$ agree on all variables with indices smaller than $n-i$ and on all variables with indices greater than $k_{\tilde{c}}$ it follows from (20):

$$[\![\sum_{k=0}^{n} x_n]\!]_{\beta_{i+1}} \leq [\![\sum_{k=0}^{n} x_n]\!]_{\beta_{n-k_{\tilde{c}}}}$$

But by the induction hypothesis for $n-k_{\tilde{c}}$,

$$[\![\sum_{k=0}^{n} x_n]\!]_{\beta_{n-k_{\tilde{c}}}} \leq [\![\sum_{k=0}^{n} x_n]\!]_{\beta_0}$$

which, together with the previous inequality, proves the third assertion *(c)*:

$$[\![\sum_{k=0}^{n} x_n]\!]_{\beta_{i+1}} < [\![\sum_{k=0}^{n} x_n]\!]_{\beta_0}.$$

With it the lemma 1 is proved. ■

Proof of theorem 1:

It is obvious, that $(a), (b)$ and (c) lead to the validation of the assertions of the theorem 1: It is immediately clear, that property (2) is the same as (b) and resp. (3) is the same as (c) for setting $\beta^* := \beta_{n+1}$.

Consider a condition c in B_σ. Since β^* assigns only integers to the variables, $[\![s(c)]\!]_{\beta^*}$ is also an integer. The first assertion in lemma 1 implies that

$$[\![s(c)]\!]_{\beta^*} \in (\lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1, \lceil [\![s(c)]\!]_{\beta_0} \rceil + 1)$$

But the only integers in the interval $(\lfloor [\![s(c)]\!]_{\beta_0} \rfloor - 1, \lceil [\![s(c)]\!]_{\beta_0} \rceil + 1)$ are $\lfloor [\![s(c)]\!]_{\beta_0} \rfloor$ and $\lceil [\![s(c)]\!]_{\beta_0} \rceil$.

$r(c)$ is a constant symbol, which is interpreted by ω as an integer, namely $\mathrm{eft}(t)$ or $\mathrm{lft}(t)$ for some transition t. Clearly, if for a given real number r and an integer i, the inequalities $r \le i$ or $i \le r$ hold, then $\lfloor r \rfloor \le i$ or $i \le \lfloor r \rfloor$ are also fulfilled, respectively, and the same applies to $\lceil r \rceil$.

Therefore, for both possible values $\lfloor [\![s(c)]\!]_{\beta_0} \rfloor$ and $\lceil [\![s(c)]\!]_{\beta_0} \rceil$ of $[\![s(c)]\!]_{\beta^*}$, it follows that β^* satisfies c.

Since c was chosen arbitrarily, β^* satisfies all conditions in B_σ, so that property (1) stated in the theorem follows from lemma 1(a) and hence theorem 1 is proved. ■

The next proposition immediately follows from theorem 1:

Corollary 1. *Let $z = (m, h)$ be an arbitrary reachable state in a TPN Z. Then the state $\underline{z} := (m, \lfloor h \rfloor)$ is also reachable in Z.*

Proof: The existence of $\underline{z} := (m, \lfloor h \rfloor)$ follows immediately from theorem 1(2). ■

The next theorem can be proved analogously to theorem 1.

Theorem 2. *Let $Z = [P, T, F, V, m_0, I]$ be a TPN, σ a transition sequence of length n, with $\delta(\sigma) = [m_\sigma, \Sigma_\sigma, B_\sigma]$ and $\hat{\beta} : X \longrightarrow \mathbb{R}_0^+$ an assignment such that $\forall c(c \in B_\sigma \rightarrow \hat{\beta}$ satisfies $c)$. Then there exists an assignment $\beta^* : X \longrightarrow \mathbb{N}$ such that:*

(a) $\forall c(c \in B_\sigma \rightarrow \beta^*$ *satisfies* $c)$

(b) $\forall t(t \in T \wedge t^- \le m_\sigma \rightarrow [\![\Sigma_\sigma(t)]\!]_{\beta^*} \ge [\![\Sigma_\sigma(t)]\!]_{\hat{\beta}})$

(c) $[\![\sum_{k=0}^{n} x_k]\!]_{\beta^*} \ge [\![\sum_{k=0}^{n} x_k]\!]_{\hat{\beta}}$

Corollary 2. *Let $z = (m, h)$ be an arbitrary reachable state in a TPN Z. Then the state $\overline{z} := (m, \lceil h \rceil)$ is also reachable in Z.*

Proof: The existence of $\overline{z} := (m, \lceil h \rceil)$ follows immediately from theorem 2(2). ■

Example 2.
Let us consider the following TPN Z_2

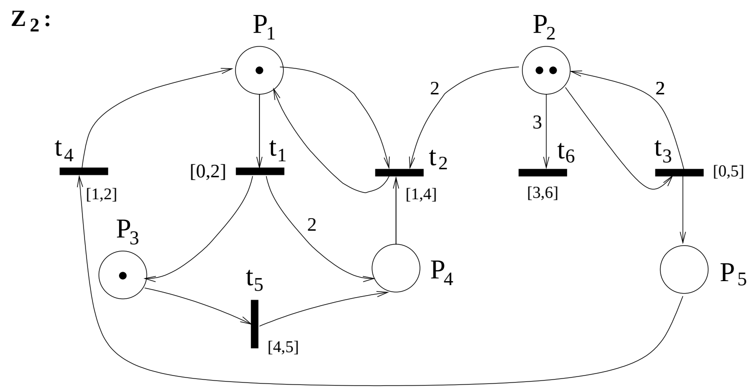

Fig. 4. Z_2 - a TPN

and the transition sequence $\sigma = (t_1 t_3 t_4 t_2 t_3)$. Successively it can be computed

$$m_\sigma = (1, 2, 2, 1, 1),$$

$$\Sigma_\sigma = (x_4 + x_5, x_5, x_5, x_5, x_0 + x_1 + x_2 + x_3 + x_4 + x_5, \sharp) \text{ and}$$

$$B_\sigma = \{ \begin{array}{lll} 0 \leq x_0, & x_0 \leq 2, & x_0 + x_1 + x_2 \leq 5, \\ 0 \leq x_1, & x_2 \leq 2, & x_2 + x_3 \leq 5, \\ 1 \leq x_2, & x_3 \leq 2, & x_0 + x_1 + x_2 + x_3 \leq 5, \\ 1 \leq x_3, & x_4 \leq 2, & x_0 + x_1 + x_2 + x_3 + x_4 \leq 5, \\ 0 \leq x_4, & x_5 \leq 2, & x_0 + x_1 + x_2 + x_3 + x_4 + x_5 \leq 5, \\ 0 \leq x_5, & x_0 + x_1 \leq 5, & x_4 + x_5 \leq 2 \end{array} \}.$$

It can be proved easily that the transition sequence σ is feasible, because the run $\sigma(\tau)$ with

$$\sigma(\tau) := z_0 \xrightarrow{0.7} \xrightarrow{t_1} \xrightarrow{0.0} \xrightarrow{t_3} \xrightarrow{0.4} \xrightarrow{t_4} \xrightarrow{1.2} \xrightarrow{t_2} \xrightarrow{0.5} \xrightarrow{t_3} \xrightarrow{1.4} z$$

is a feasible one in Z_2. This is the case, since the values 0.7, 0.0, 0.4, 1.2, 0.5, 1.4 assign to the variables x_1, x_2, x_3, x_4, x_5 (with an assignment $\hat{\beta}$) satisfy B_σ.

In the next two tableaus *I* and *II* the recursive construction of integer values for x_1, x_2, x_3, x_4, x_5 according to theorem 1 (tableau *I*), res. to theorem 2 (*II*)is shown. Since $\Sigma_\sigma(t_2) = \Sigma_\sigma(t_3) = \Sigma_\sigma(t_4)$ only $\Sigma_\sigma(t_2)$ is shown in the tableaus.

Actually, the tableaus *I* and *II* illustrate the solution of the input problem by using dynamic programming. If we consider, for example, tableau *I* in more detail, the following concrete problem is solved:

Input: The TPN Z_2, the transition sequence $\sigma = (t_1, t_3, t_4, t_2, t_3)$
and the six ($6 = n + 1$, i.e.$n = 5$) elapces of time
$\hat{\beta}(x_0) = 0.7, \hat{\beta}(x_1) = 0.0, \hat{\beta}(x_2) = 0.4,$
$\hat{\beta}(x_3) = 1.2, \hat{\beta}(x_4) = 0.5, \hat{\beta}(x_5) = 1.4,$
which are real numbers with the property that
$\sigma(\hat{\beta}) = (0.7, t_1, 0.0, t_3, 0.4, t_4, 1.2, t_2, 0.5, t_3, 1.4)$
is a feasible run in Z_2

Output: Six elapces of time $\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_5)$
which are integers, and $\sigma(\beta^*)$ is a feasible run in Z_2.
The set of transitions which are ready to fire after $\sigma(\hat{\beta})$
is the same as the set of transitions which are ready to fire
after $\sigma(\beta^*)$

Thus, the target problem P^* here is the computing of the six integers. Each row s ($s = 0, 1, \cdots, 6$) in the tableau I is a solution of one modified problem $P^*(s)$. $P^*(s)$ modifies one elapse of time, which is not integer in $P^*(s-1)$, to a number which is an integer such that the modified run remains feasible.

I	x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat{\beta} = \beta_0$	0.7	0.0	0.4	1.2	0.5	1.4	1.9	1.4	4.2
β_1	0.7	0.0	0.4	1.2	0.5	**1**	1.5	1.0	3.8
β_2	0.7	0.0	0.4	1.2	**0**	1	1.0		3.3
β_3	0.7	0.0	0.4	**1**	0	1			3.1
β_4	0.7	0.0	**1**	1	0	1			3.7
β_5	0.7	**0**	1	1	0	1			3.7
$\beta^* = \beta_6$	**1**	0	1	1	0	1			4.0

and

II	x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat{\beta} = \beta_0$	0.7	0.0	0.4	1.2	0.5	1.4	1.9	1.4	4.2
β_1	0.7	0.0	0.4	1.2	0.5	**2**	2.5	2.0	4.8
β_2	0.7	0.0	0.4	1.2	**0**	2	2.0		4.3
β_3	0.7	0.0	0.4	**2**	0	2			5.1
β_4	0.7	0.0	**0**	2	0	2			4.7
β_5	0.7	**0**	0	2	0	2			4.7
$\beta^* = \beta_6$	**1**	0	0	2	0	2			5.0

Hence, the runs

$$\sigma(\tau_1^*) := z_0 \xrightarrow{1} \xrightarrow{t_1} \xrightarrow{0} \xrightarrow{t_3} \xrightarrow{1} \xrightarrow{t_4} \xrightarrow{1} \xrightarrow{t_2} \xrightarrow{0} \xrightarrow{t_3} \xrightarrow{1} \underline{z}$$

and

$$\sigma(\tau_2^*) := z_0 \xrightarrow{1} \xrightarrow{t_1} \xrightarrow{0} \xrightarrow{t_3} \xrightarrow{0} \xrightarrow{t_4} \xrightarrow{2} \xrightarrow{t_2} \xrightarrow{0} \xrightarrow{t_3} \xrightarrow{2} \overline{z}$$

are feasible in Z_2, too.

Furthermore, it is easy to see that $\underline{z} = (m_\sigma, \underline{h})$, res. $\overline{z} = (m_\sigma, \overline{h})$ where $\underline{h} = (1, 1, 1, 1, 4, \sharp)$ and $\overline{h}) = (2, 2, 2, 2, 5, \sharp)$.

Obviously, both sequences of every six integers $1, 0, 1, 1, 0, 1$ and $1, 0, 0, 2, 0, 2$ belong to the solution set z^0 of the problem P^*. It can be easily shown that the set z^0 contains more solutions in general.

3.3 Reachability graph of a TPN

In the previous subsection we saw that the integer-states are "girders" of the net. Obviously, (it immediately follows from theorem 1) all feasible transition sequences in a certain TPN and all reachable p-markings of the net are well known when all integer-states are known. The set of all integer-states ($IS(Z)$) can be computed considering **only** state changes by firing and by **integer** time elapsing. However, this set can be finite as well as infinite.

Though, when the TPN is bounded (i.e. the set of reachable p-markings is finite) and finite (i.e. the lft's for all transitions are finite) the set $IS(Z)$ is finite, too. Then almost all properties of such TPNs can be proved using this set. In this case a reachability graph $RG(Z)$ for a TPN Z can be defined in such a way that its vertices are the reachable integer-states. The edges are defined by the triples (z, t, z') and $(z, 1, z')$, where $z \xrightarrow{t} z'$ and $z \xrightarrow{1} z'$, respectively. This graph is finite if and only if the set of the reachable markings of the net is finite.

An enumeration procedure for computing the reachable graph of a given TPN can be constructed **easily**: Starting at the initial state all integer-state successors of a reached integer-state can be derived in a successive way (i.e. breadth-first search):

Basis)
$z_0 \in RG(Z)$

Step)
Let z be in $RG(Z)$ already.
1. <u>for</u> i=1 <u>to</u> n <u>do</u>
<u>if</u> $z \xrightarrow{t_i} z'$ possible in Z (cf. def 4) <u>then</u> $z' \in RG(Z)$ <u>end</u>
2. <u>if</u> $z \xrightarrow{1} z'$ possible in Z (cf. def 4) <u>then</u> $z' \in RG(Z)$

The reachability graph defined above can be reduced: 1st stage – all vertices that have input and output edges labeled with time only can be ignored. Their input edges are merged with their output edges (each input edge with each output one) and labeled with the sum of both labels. A further reduction can be accomplished as follow: 2nd stage – all vertices which have input edges with time only can be ignored, too. Their input edges are merged with their output edges labeled with the both labels (from the input and from the output edges). However the second reduction decreases the number of vertices but the labels of the merged edges are more complex then the remained labels. For more see e.g. [2] or [9]. From now on we are going to use the notion reachability graph in the sense of (double) reduced reachability graph.

Unfortunately, the reachability graph defined above is not finite if there is a transition t with $lft(t) = \infty$, even though the net is bounded. Nonetheless, the time after reaching the $eft(t)$ is not important for t actually. Thus, when the clock $h(t)$ reaches the eft, i.e. $h(t) = eft(t)$ than it is not necessary to increment the time for this transition anymore (even though the time is going on). That is why we modify the definition 4(c)(iii) as follows:

$$(iii)' \quad \forall t \; (\; t \in T \longrightarrow h'(t) := \begin{cases} h(t) + \tau & \text{iff} \quad t^- \leq m' \wedge eft(t) \geq h(t) + \tau \\ eft(t) & \text{iff} \quad t^- \leq m' \wedge eft(t) < h(t) + \tau \\ \# & \text{iff} \quad t^- \not\leq m' \end{cases} \;).$$

It can be proved by induction that in a TPN a transition sequence is feasible according to definition 4(c)(iii) if and only if it is feasible according to definition $4'$. The crucial advantage of the modified reachable graph which is obtained by using the modified definition $4'$ is the property from above: it is finite if and only if the TPN is bounded.

Furthermore, it is easy to see that in the case of a finite TPN both definitions deliver the same set of reachable integer-states, i.e. the modified definition $4'$ is a consistent extension of definition 4 if the class of considered TPNs is extended.

The enumeration procedure for computing the reachable graph of a given TPN defined above can be performed using the modified definition $4'$. The result is the modified reachability graph. This graph can be reduced in the same manner as $RG(Z)$, too.

Example 3.

Let us consider the finite TPN Z_2. The full reachability graph $RG^{(1)}(Z_2)$ as well as the reduced reachability graphs $RG^{(2)}(Z_2)$ (1st stage of reduction) and $RG(Z_2)$ (2nd stage of reduction) are illustrated below:

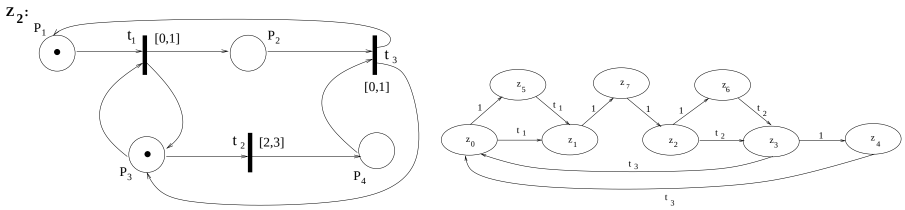

Fig. 5. Z_2 and its full reachability graph $RG^{(1)}(Z_2)$

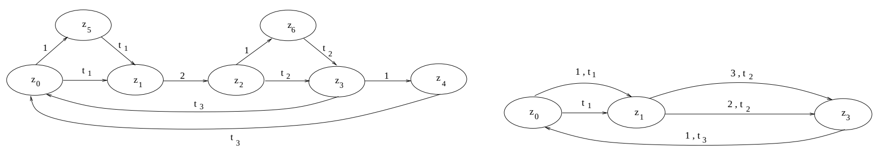

Fig. 6. The reduced reachability graphs $RG^{(2)}(Z_2)$ and $RG(Z_2)$

$$
\begin{array}{lllll}
m_0 = (1,0,1,0) & h_1 = (\sharp,0,\sharp)^T & h_5 = (1,1,\sharp)^T & z_1 = (m_1,h_1) & z_4 = (m_2,h_4) \\
m_1 = (0,1,1,0) & h_2 = (\sharp,2,\sharp)^T & h_6 = (\sharp,\sharp,3)^T & z_2 = (m_1,h_2) & z_5 = (m_0,h_5) \\
m_2 = (0,1,0,1) & h_3 = (\sharp,\sharp,0)^T & z_0 = (m_0,h_0) & z_3 = (m_2,h_3) & z_6 = (m_1,h_6) \\
h_0 = (0,0,\sharp)^T & h_4 = (\sharp,\sharp,1)^T & & &
\end{array}
$$

Finally, we modify the finite TPN Z_2 to Z_3. The TPN Z_3 arises from Z_2 by changing the $lft(t_2)$ to ∞. Thus, the net Z_3 is not finite. Its reduced reachability graph $RG(Z_3)$ according to def. $4'$ is to be seen bellow:

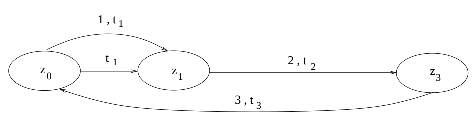

Fig. 7. The reachability graph $RG(Z_3)$

And as a last remark here note that the set of all reachable integer-states of a certain TPN is finite, if the set of reachable markings of its skeleton – the timeless net – is finite. The other direction is not true in general.

4 Related

Time Petri nets were introduced in the early seventies as already mentioned. Berthomieu and Menasche in [10] res. Berthomieu and Diaz in [11] provide a method for analyzing the qualitative behavior of the net. They divide the state space in state classes which are described by a marking and time domain given by inequalities. The reachability graph that they defined consists of these classes as vertices and edges labeled by transitions. Thus, the edges of this graph contain essential time information (systems of inequalities). This is in contrast to the reachability graph used in this paper, which is an usual weighted digraph, and the time appears explicitly as weights on some edges. The reachability graph defined in [11] has also the property that the graph is finite iff the TPN is bounded. A similar definition for a reachability graph for a TPN delivers [12].

A new direction of investigation was started at the beginning of the nineties with the deployment of timed automata. Several authors, i.e. recently in [13], [14] etc., translate a given TPN into a timed automata and then analyse the timed automata in order to gain knowledge about the TPN. In this case well proved algorithms in the area of timed automata (mainly for model checking) can be used.

Only few papers are published connecting the theory of Petri Nets and dynamic programming. Mostly, they consider quantitative properties of systems, e.g. [15].

5 Conclusions

In this paper a methodology that deploys dynamic programming in order to reduce the state space of a TPN is used. Thus, an enumeration procedure can compute a reachability graph for a given TPN. While the graph is a usual directed weighted graph, the behaviour of the net can be studied by means of prevalent methods of graph theory. This is especially fruitful if the considered TPN is bounded. Now in order to accomplish quantitative analysis effective algorithms can be used, e.g., for computing minimal and maximal time length of runs, existence of a certain run with a given time length, etc.

The author would like to thank Doratha Drake for many discussions in preparing this paper.

References

1. Merlin, P.M.: A Study of the Recoverability of Computing Systems. PhD thesis, University of California, Computer Science Dept., Irvine (1974)
2. Popova, L.: On Time Petri Nets. J. Inform. Process. Cybern. EIK 27(1991)4 (1991) 227–244
3. Popova-Zeugmann, L., Schlatter, D.: Analyzing Path in Time Petri Nets. Fundamenta Informaticae (FI) 37, IOS Press, Amsterdam (1999) 311–327

4. Bellman, R.: Dynamic programming. Princeton University Press, Princeton, New Jersey (1957)
5. Sniedovich, M.: Dynamic programming. Marcel Dekker, New York (1992)
6. Bertsekas, D.: Dynamic programming and optimal control, Vol. I, 2nd edition. Athena Scient., Belmont, Mass. (2000)
7. Popova-Zeugmann, L.: Zeit-Petri-Netze. PhD thesis, Humboldt-Universität zu Berlin (1989)
8. Ebbinghaus, H.D., Flumm, J., Thomas, W.: Mathematical Logic. Springer-Verlag, New York-Berlin-Heidelberg-London-Paris-Tokyo- Hong Kong-Barcelona-Budapest (1994)
9. Popova-Zeugmann, L., Werner, M.: Extreme runtimes of schedules modelled by time petri nets. Fundamenta Informaticae (FI) 67, IOS Press, Amsterdam (2005) 163–174
10. Berthomieu, B., Menasche, M.: An Enumerative Approach for Analyzing Time Petri Nets. In: Proceedings IFIP Congress. (1983)
11. Berthomieu, B., Diaz, M.: Modeling and Verification of Time Dependent Systems Using Time Petri Nets. In: Advances in Petri Nets 1984. Volume 17, No. 3 of IEEE Trans. on Software Eng. (1991) 259–273
12. Boucheneb, H., Berthelot, G.: Towards a simplified building of time petri net reachability graphs. In: Proceedings of Petri Nets and Performance Models PNPM 93, Toulouse France, IEEE Computer Society Press (1993)
13. Cassez, F., Roux, O.H.: Structural translation from time Petri nets to timed automata. In: Fourth International Workshop on Automated Verification of Critical Systems (AVoCS'04). Electronic Notes in Theoretical Computer Science, London (UK), Elsevier (2004)
14. Penczek, W.: Partial order reductions for checking branching properties of time petri nets. Proc. of the Int. Workshop on CS&P'00 Workshop, Informatik-Berichte Nr.140(2) (2000) 189–202
15. Yee, S., Ventura, J.: A dynamic programming algorithm to determine optimal assembly sequences using petri nets. International Journal of Industrial Engineering - Theory, Applications and Practice, Vol.6, No.1 (1999) 27–37