Time Petri Net State Space Reduction Using Dynamic Programming and Time Paths

Louchka Popova-Zeugmann

Humboldt-Universität zu Berlin
Institut of Computer Science
Unter den Linden 6, 10099 Berlin, Germany

IFORS 2005, Hawaii
July 11-15, 2005

Outline

Petri Net

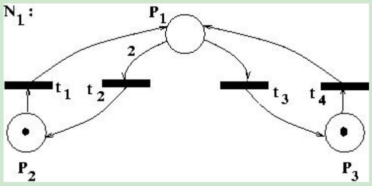
Definition (Petri Net)

The structure $N = (P, T, F, V, m_0)$ is a **Petri Net (PN)**, iff

- P, T and F are finite sets,
  P−set of places } set of vertices(nodes)
  T−set of transitions }
  $P \cap T = \emptyset, \quad P \cup T \neq \emptyset,$
  F – set of edges (arcs)
  $F \subseteq (P \times T) \cup (T \times P)$ and $dom(F) \cup cod(F) = P \cup T$
- $V : F \longrightarrow \mathbb{N}^+$ (weights of edges)
- $m_0 : P \longrightarrow \mathbb{N}$ (initial marking)

Petri Net

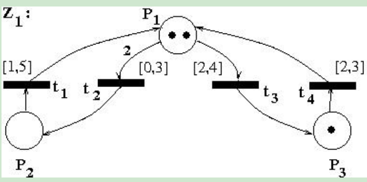
Example



- $m_0 = (0, 1, 1)$

Time Petri Net

Example



- $m_0 = (0, 1, 1)$ p-marking
- $h_0 = (\sharp, 0, 0, 0)$ t-marking

Time Petri Net

Definition (Time Petri net)

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

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

state

Definition (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:

- m is a p-marking in Z .
- h is a t-marking in Z.

Definition (state changing)

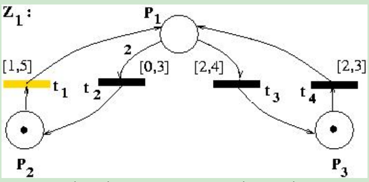
Let $Z = (P, T, F, V, m_o, I)$ be a TPN,
$z = (m, h)$, $z' = (m', h')$ be two states.
Then

$z = (m, h)$ changes into $z' = (m', h')$ by

firing a transition / \ **time elapsing**

Time Petri Net

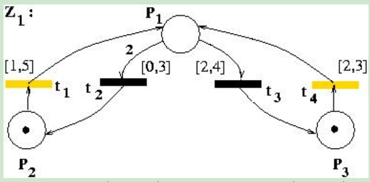
Example



$$(m_0, \begin{pmatrix} 0 \\ \sharp \\ \sharp \\ 0 \end{pmatrix}) \xrightarrow{1.3} (m_1, \begin{pmatrix} 1.3 \\ \sharp \\ \sharp \\ 1.3 \end{pmatrix})$$

Time Petri Net

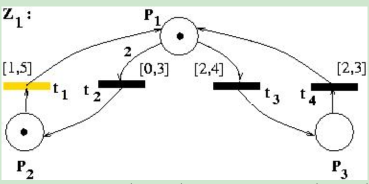
Example



$$z_0 \xrightarrow{1.3} (m_1, \begin{pmatrix} 1.3 \\ \sharp \\ \sharp \\ 1.3 \end{pmatrix}) \xrightarrow{1.0} (m_2, \begin{pmatrix} 2.3 \\ \sharp \\ \sharp \\ 2.3 \end{pmatrix})$$

Time Petri Net

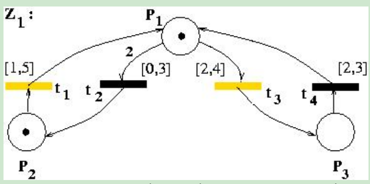
Example



$$z_0 \xrightarrow{1.3} \xrightarrow{1.0} (m_2, \begin{pmatrix} 2.3 \\ \sharp \\ \sharp \\ 2.3 \end{pmatrix}) \xrightarrow{t_4} (m_3, \begin{pmatrix} 2.3 \\ \sharp \\ 0.0 \\ \sharp \end{pmatrix})$$

Time Petri Net

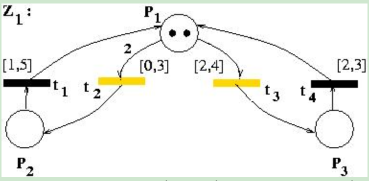
Example



$$z_0 \xrightarrow{1.3} \xrightarrow{1.0} \xrightarrow{t_4} (m_3, \begin{pmatrix} 2.3 \\ \sharp \\ 0.0 \\ \sharp \end{pmatrix}) \xrightarrow{2.0} (m_4, \begin{pmatrix} 4.3 \\ \sharp \\ 2.0 \\ \sharp \end{pmatrix})$$

Time Petri Net

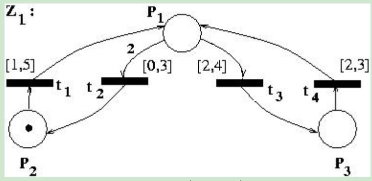
Example



$$z_0 \xrightarrow{1.3}\xrightarrow{1.0}\xrightarrow{t_4}\xrightarrow{2.0} \left(m_4, \begin{pmatrix}4.3\\ \sharp\\ 2.0\\ \sharp\end{pmatrix}\right) \xrightarrow{t_1} \left(m_5, \begin{pmatrix}\sharp\\ 0.0\\ 2.0\\ \sharp\end{pmatrix}\right)$$

Time Petri Net

Example



$$z_0 \xrightarrow{1.3}\xrightarrow{1.0}\xrightarrow{t_4}\xrightarrow{2.0}\xrightarrow{t_1} \left(m_5, \begin{pmatrix}\sharp\\ 0.0\\ 2.0\\ \sharp\end{pmatrix}\right) \xrightarrow{t_2} \left(m_6, \begin{pmatrix}0.0\\ \sharp\\ \sharp\\ \sharp\end{pmatrix}\right)$$

Transition sequences, Runs

Definition

- **transition sequence:** $\sigma = (t_1, \cdots, t_n)$
- **run:** $\sigma(\tau) = (\tau_0, t_1, \tau_1, \cdots, \tau_{n-1}, t_n, \tau_n)$
- **feasible run:** $z_0 \xrightarrow{\tau_0} z_0^* \xrightarrow{t_1} z_1 \xrightarrow{\tau_1} z_1^* \cdots \xrightarrow{t_n} z_n \xrightarrow{\tau_n} z_n^*$
- **feasible transition sequence :** σ is feasible if there ex. a feasible run $\sigma(\tau)$

Reachable state, Reachable marking, State space

Definition

- z is **reachable state** in Z if there ex. a feasible run $\sigma(\tau)$ and $z_0 \xrightarrow{\sigma(\tau)} z$
- m is **reachable marking** in Z if there ex. a reachable state z in Z with $z = (m, h)$
- The set of all reachable states in Z is the **state space** of Z (denoted: $StSp(Z)$).

Qualitative Properties

- static properties: being/having
  - homogenous
  - ordinary
  - free choice
  - extended simple
  - conservative
  - deadlocks, etc.

  decidable **without knowledge** of the state space!
- dynamic properties: being/having
  - bounded
  - live
  - reachable marking/state
  - place- or transitions invariants, etc.

  decidable, if at all (TPN is equiv. to TM!), **with implicit/explicit knowledge** of the state space

Quantitative Properties

each time proposition as having/computing

- (min-/max) time length of path
- path between two states/markings with min-/max time length
- set of all reachable markings within a period
- looking for *efts* and *lfts* leading to certain qualitative/quantitative properties etc.

decidable, if at all, **with implicit/explicit knowledge** of the state space

Parametric Description of the State Space

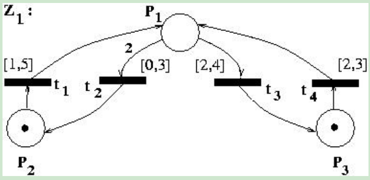
Let $Z = [P, T, F, V, m_0, I]$ be a TPN and $\sigma = (t_1, \cdots, t_n)$ be a transition sequence in Z.
$\delta(\sigma) = [m_\sigma, \Sigma_\sigma, B_\sigma]$ is the parametric description of σ, if

- $m_0 \xrightarrow{\sigma} m_\sigma$
- $\Sigma_\sigma(t)$ is a parametrical $t-$marking
- B_σ is a set of conditions (a system of inequalities)

Obviously

- $z_0 \xrightarrow{\sigma} (m_\sigma, \Sigma_\sigma) =: z_\sigma$,
- $StSp(Z) = \bigcup\limits_{\sigma} z_\sigma$.

Example



$$\sigma = (e) \quad \Longrightarrow$$

$$\delta(\sigma) = C_e = \{(\underbrace{(0,1,1)}_{m_\sigma}, \underbrace{(x_1, \sharp, \sharp, x_1)}_{\Sigma_\sigma}) \mid \underbrace{0 \le x_1 \le 3}_{B_\sigma}\}$$

State Space Reduction

Theorem (1)

Let Z be a TPN and $\sigma = (t_1, \cdots, t_n)$ be a feasible transition sequence in Z, with a run $\sigma(\tau)$ as an execution of σ, i.e.

$$z_0 \xrightarrow{\tau_0} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n} \xrightarrow{t_n} z_n = (m_n, h_n),$$

and all $\tau_i \in \mathbb{R}_0^+$.
Then, there exists a further feasible run $\sigma(\tau^*)$ of σ with

$$z_0 \xrightarrow{\tau_0^*} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n^*} \xrightarrow{t_n} z_n^* = (m_n^*, h_n^*).$$

such that

State Space Reduction

Theorem (1 – continuation)

$z_0 \xrightarrow{\tau_0} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n} \xrightarrow{t_n} z_n = (m_n, h_n),\ \tau_i \in \mathbb{R}_0^+.$
$z_0 \xrightarrow{\tau_0^*} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n^*} \xrightarrow{t_n} z_n^* = (m_n^*, h_n^*),\ \tau_i^* \in \mathbb{N}.$

1. *For each $i, 0 \le i \le n$ the time τ_i^* is a natural number.*
2. *For each enabled transition t at marking $m_n (= m_n^*)$ it holds:*
   2.1 $h_n(t)^* = \lfloor h_n(t) \rfloor$.
   2.2 $\sum\limits_{i=1}^{n} \tau_i^* = \lfloor \sum\limits_{i=1}^{n} \tau_i \rfloor$
3. *For each transition $t \in T$ holds:*
   t is ready to fire in z_n iff t is ready to fire in $\lfloor z_n \rfloor$, too.

State Space Reduction

Theorem (2 – similar to 1)

Let Z be a TPN and $\sigma = (t_1, \cdots, t_n)$ be a feasible transition sequence in Z, with a run $\sigma(\tau)$ as an execution of σ, i.e.

$$z_0 \xrightarrow{\tau_0} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n} \xrightarrow{t_n} z_n = (m_n, h_n),$$

and all $\tau_i \in \mathbb{R}_0^+$.
Then, there exists a further feasible run $\sigma(\tau^*)$ of σ with

$$z_0 \xrightarrow{\tau_0^*} \xrightarrow{t_0} \cdots \xrightarrow{\tau_n^*} \xrightarrow{t_n} z_n^* = (m_n^*, h_n^*).$$

such that

State Space Reduction

Theorem (2 – continuation)

1. *For each $i, 0 \le i \le n$ the time τ_i^* is a natural number.*
2. *For each enabled transition t at marking $m_n (= m_n^*)$ it holds:*
   2.1 $h_n(t)^* = \lceil h_n(t) \rceil$.
   2.2 $\sum\limits_{i=1}^{n} \tau_i^* = \lceil \sum\limits_{i=1}^{n} \tau_i \rceil$
3. *For each transition $t \in T$ holds:*
   t is ready to fire in z_n i t is ready to fire in $\lceil z_n \rceil$, too.

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 of conditions (inequalities).

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

Dynamic Programming

The solving of the output is the problem P*:

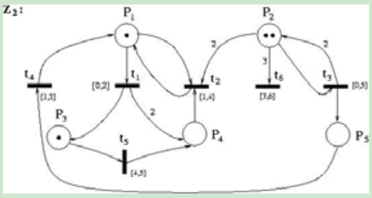
Problem P*: *Compute a sequence of* $(n+1)$ *integers,* $(\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_n))$ *subject to* VC^*[1].

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

[1] VC^* is a certain finite superset of the set VC

State Space Reduction

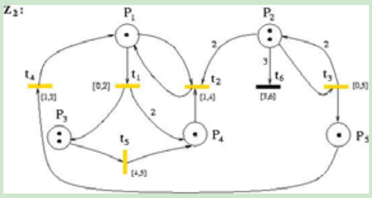
Example



$$\sigma = (t_1\ t_3\ t_4\ t_2\ t_3)$$

State Space Reduction

Example

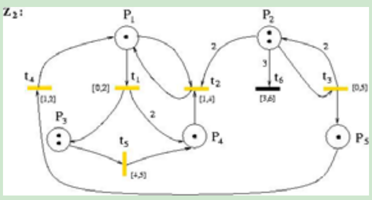


$$\sigma = (t_1 t_3 t_4 t_2 t_3)$$

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

State Space Reduction

Example

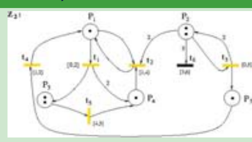


$$\sigma = (t_1 t_3 t_4 t_2 t_3)$$

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

State Space Reduction

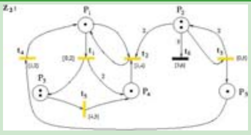
Example (continuation)



$$\Sigma_\sigma = \begin{pmatrix} x_4 + x_5 \\ x_5 \\ x_5 \\ x_5 \\ x_0 + x_1 + x_2 + x_3 + x_4 + x_5 \\ \sharp \end{pmatrix} \quad \text{and}$$

State Space Reduction

Example (continuation)



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

State Space Reduction

Example (continuation)

The run $\sigma(\tau)$ with
$\sigma(\tau) =$

$$z_0 \xrightarrow{\mathbf{0.7}} \xrightarrow{t_1} \xrightarrow{\mathbf{0.0}} \xrightarrow{t_3} \xrightarrow{\mathbf{0.4}} \xrightarrow{t_4} \xrightarrow{\mathbf{1.2}} \xrightarrow{t_2} \xrightarrow{\mathbf{0.5}} \xrightarrow{t_3} \xrightarrow{\mathbf{1.4}} \left(m, \begin{pmatrix} 1.9 \\ 1.4 \\ 1.4 \\ 1.4 \\ 4.2 \\ \sharp \end{pmatrix}\right)$$

is feasible.

State Space Reduction

Example (continuation)

$$\underbrace{\left(m, \begin{pmatrix} \mathbf{1.0} \\ \mathbf{1.0} \\ \mathbf{1.0} \\ \mathbf{1.0} \\ \mathbf{4.0} \\ \sharp \end{pmatrix}\right)}_{z_0 \xrightarrow{\sigma(?)} \lfloor z \rfloor} \qquad \underbrace{\left(m, \begin{pmatrix} 1.9 \\ 1.4 \\ 1.4 \\ 1.4 \\ 4.2 \\ \sharp \end{pmatrix}\right)}_{z_0 \xrightarrow{\sigma(\hat\beta)} z} \qquad \underbrace{\left(m, \begin{pmatrix} \mathbf{2.0} \\ \mathbf{2.0} \\ \mathbf{2.0} \\ \mathbf{2.0} \\ \mathbf{5.0} \\ \sharp \end{pmatrix}\right)}_{z_0 \xrightarrow{\sigma(?)} \lceil z \rceil}$$

State Space Reduction

Example (continuation)

I		x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat\beta$ =	β_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
	β_6	**1**	0	1	1	0	1			**4.0**

State Space Reduction

Example (continuation)

II		x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat\beta$ =	β_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
	β_6	**1**	0	0	2	0	2			**5.0**

State Space Reduction

Example (continuation)

Hence, the runs

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

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

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

are feasible in Z, too.

Dynamic programming

Where is the Dynamic Programming here?

Let us consider the tableau **I** again!

Dynamic programming

Input:

- **The TPN Z_2,**
- **the transition sequence $\sigma = (t_1, t_3, t_4, t_2, t_3)$**
- **the six ($6 = n+1$, i.e. $n = 5$) elapses 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 and**
- **the run $\sigma(\hat{\beta}) = (0.7, t_1, 0.0, t_3, 0.4, t_4, 1.2, t_2, 0.5, t_3, 1.4)$ is a feasible one in Z_2.**

Dynamic programming

Output:

- **Six elapses of time $\beta^*(x_0), \beta^*(x_1), \cdots, \beta^*(x_5)$ which are integers,**
- **$\sigma(\beta^*)$ 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 $\sigma(\beta^*)$.**

$= \mathbf{P}^*$

Dynamic Programming

I		x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat{\beta} =$	β_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**			
	β_2	0.7	0.0	0.4	1.2	**0**	1			
	β_3	0.7	0.0	0.4		0	1			

Dynamic Programming

I		x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat{\beta} =$	β_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**			
	β_2	0.7	0.0	0.4	1.2	**0**	1			
	β_3	0.7	0.0	0.4	**1**	0	1			

Dynamic Programming

I		x_0	x_1	x_2	x_3	x_4	x_5	$\Sigma_\sigma(t_1)$	$\Sigma_\sigma(t_2)$	$\Sigma_\sigma(t_5)$
$\hat{\beta} =$	β_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
$\beta^* =$	β_6	**1**	0	1	1	0	1			**4.0**

$\Sigma_\sigma(t_1) = x_4 + x_5,$ $\Sigma_\sigma(t_2) = \Sigma_\sigma(t_3) = \Sigma_\sigma(t_4) = x_5$

$\Sigma_\sigma(t_5) = x_1 + x_2 + x_3 + x_4 + x_5$

Dynamic Programming

- The *set of its critical states* is the singleton $S^o = \{5\}$.
- The *set of its terminal states* is the singleton $S^t = \{0\}$.
- The *set of non-terminal states* is $S'' = S \setminus S^t = \{1, 2, \dots, 5\}$.
- The T-*linker* L_T has the form $L_T(z(s^o)) = z^o = z(s^o)$.
- The *transition function* t is defined as

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

Dynamic Programming

- The *linker* L is clearly given by

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

Dynamic Programming

The time length of the run $\sigma(\hat{\beta})$ is
$l_{\sigma(\beta^*)} = \hat{\beta}(x_0) + \hat{\beta}(x_1) + \hat{\beta}(x_2) + \hat{\beta}(x_3) + \hat{\beta}(x_4) + \hat{\beta}(x_5) = 4.2$

In tableau I: The time length of the run $\sigma(\beta^*)$ is $l_{\sigma(\beta^*)} = 4$

In tableau II: The time length of the run $\sigma(\beta^*)$ is $l_{\sigma(\beta^*)} = 5$

i.e. $l_{\sigma(\beta^*)} = 4 \leq 4.2 = l_{\sigma(\beta^*)} = 4.2 \leq 5 = l_{\sigma(\beta^*)}$

State Space Reduction

Corollary

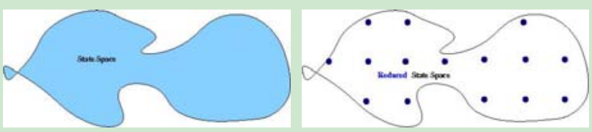
- *Each feasible t-sequence σ in Z can be realized with an "integer" run.*
- *Each reachable marking in Z can be found using "integer" runs only.*
- *If z is reachable in Z, then $\lfloor z \rfloor$ and $\lceil z \rceil$ are reachable in Z, too.*
- *The length of the shortest and longest time path between two arbitrary p-markings are natural numbers.*

State Space Reduction

Definition

A state $z = (m, h)$ in a TPN is an **integer** one iff for all enabled transitions t at m holds: $h(t) \in \mathbb{N}$.

Example (State Space Reduction)



State Space Reduction

Theorem (3)

Let Z be a FTPN.
The set of all reachable integer states in Z is finite

if and only if

the set of all reachable markings in Z is finite.

Remark: Theorem 3 can be generalized for all TPNs (applying a further reduction).

Reachability Graph

Definition

Basis) *1* $z_0 \in RG(Z)$

Step)

Let z *be in* $RG(Z)$ *already.*

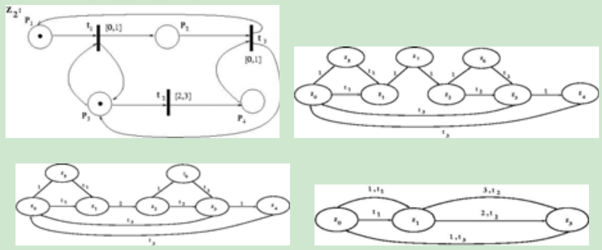
1. *<u>for</u> i=1 <u>to</u> n <u>do</u>*

<u>if</u> $z \xrightarrow{t_i} z'$ *possible in* Z *<u>then</u>* $z' \in RG(Z)$ *<u>end</u>*

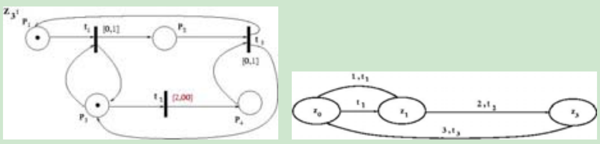
2. *<u>if</u>* $z \xrightarrow{1} z'$ *possible in* Z *<u>then</u>* $z' \in RG(Z)$

$\Longrightarrow$ The reachability graph is a weighted directed graph.

Example (The FTPN Z_2 and its reachability graph(s))



Example (The infinite TPN Z_3 and its reachability graph $RG(Z_3)$)



Let $Z = (P, T, F, V, I, m_o)$ be a bounded TPN. The following problems can be decided/computed with the knowledge of its RG, **amongst others**:

Result:

Input: z and z' - two states (in Z).

Output: – Is there a path between z and z' in $RG(Z)$?
– If yes, compute the path with the shortest time length.

Solution: By means of prevalent methods of the graph theory, e.g. Bellman-Ford algorithm (the running time is $\mathcal{O}(|V| \cdot |E|)$ and $RG(Z) = (V, E)$)

Result:

Input: m and m' - two markings (in Z).

Output: – Is there a path between m and m' in $RG(Z)$?
– If yes, compute the path with the shortest time length.

Solution: By means of prevalent methods of the graph theory, for computing all-pairs shortest paths.
The running time is polynomial, too.

Definition

The **longest path** between two states (vertices in $RG(Z)$) z and z' is $lp(z, z')$ with

$$lp(z,z') := \begin{cases} \infty & \text{, if a cycle is reachable starting on } z \text{ before reaching } z' \\ \max\limits_{\sigma(\tau)} \sum \tau_i & \text{, if } z \xrightarrow{\sigma(\tau)} z' \end{cases}$$

Result:

Input: z and z' - two states (in Z).

Output: – Is there a path between z and z' in $RG(Z)$?
– If yes, compute the path with the longest time length.

Solution: By means of prevalent methods of the graph theory, e.g. Bellman-Ford algorithm (polyn. running time). or by computing all strongly connected components of $RG(Z)$. (linear running time)

Result:

Input: m and m' - two states (in Z).

Output: – Is there a path between z and z' in $RG(Z)$?
– If yes, compute the path with the longest time length.

Solution: By means of prevalent methods of the graph theory, e.g. Bellman-Ford algorithm. or by computing all strongly connected components of $RG(Z)$.

Conclusion

- The State Space Reduction of a TPN is a nonoptimization truncated decision problem
- The minimal and the maximal time length of a path between two markings in a TPN is a natural number (if finite)

$\Longrightarrow$

it can be computed in polynomial/linear time (with res. to the RG)

Thank you!