On Parametric DBMs and their applications to time Petri nets[1]

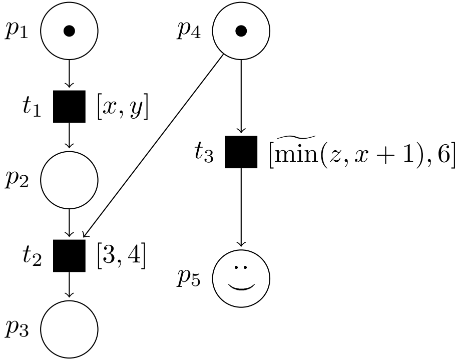
Loriane Leclercq, Didier Lime and Olivier H. Roux

Qest-Formats 2024
September 11

[1]This work has been partially funded by ANR projects ProMiS ANR-19-CE25-0015 and BisoUS ANR-22-CE48-0012

Parametric Time Petri Net (PTPN)

Example



Introduction

- Parameters for early verification during the design of timed systems
- *Parametric Time Automata (PTA)* and *Parametric Time Petri Nets (PTPN)*
  - great expressiveness
  - lots of undecidable problems (existence of a parameter valuation to reach a given configuration)
  - semi-algorithms and decidable fragments/restrictions

Introduction

- Use of *convex polyhedra*
  - only two clocks in each constraint
  - DBM extension to parameters (PDBMs for L/U-TA[2])
  - comparison between two constraints leads to splitting PDBMs

[2] Thomas Hune et al. "Linear parametric model checking of timed automata". In: *Journal of Logic and Algebraic Programming* 52-53 (2002), pp. 183–220.

Introduction

- Use of *convex polyhedra*
  - only two clocks in each constraint
  - DBM extension to parameters (PDBMs for L/U-TA[2])
  - comparison between two constraints leads to splitting PDBMs

- Tropical PDBMs with minimum of linear expressions to represent state classes

[2] Thomas Hune et al. “Linear parametric model checking of timed automata”. In: *Journal of Logic and Algebraic Programming* 52-53 (2002), pp. 183–220.

Tropical semi-ring and expressions

The *parametric tropical semi-ring*:

$$\mathbb{T} = (\mathbb{P} \cup -\mathbb{P} \cup \mathbb{Z} \cup \{+\infty\}, \widetilde{\min}, \widetilde{+})$$

where $-\mathbb{P}$ contains the inverse of each parameter with respect to $\widetilde{+}$.

A *parametric tropical expression* is a value in $\mathbb{T}$, i.e. a term generated by grammar:

$$e ::= e_1 \mathbin{\widetilde{+}} e_2 \mid \widetilde{\min}(e_1, e_2) \mid a \mid p \mid -p$$

where :

- $p \in \mathbb{P}$ is a parameter,
- $a \in \mathbb{Z} \cup \{+\infty\}$ is a constant and
- e_1, e_2 are tropical expressions.

A *parametric tropical constraint* is a term of the form $e \leq 0$.

Parametric Time Petri Net (PTPN)

Semantics

States of the PTPN $\mathcal{N} = (P, T, \mathbb{P}, F, m_0, I_s)$ is (m, v_p, θ) with

- $m \subseteq P$ a marking,
- v_p a parameter valuation and
- θ the firing dates for every transition enabled by m

Parametric Time Petri Net (PTPN)

Semantics

States of the PTPN $\mathcal{N} = (P, T, \mathbb{P}, F, m_0, I_s)$ is (m, v_p, θ) with

- $m \subseteq P$ a marking,
- v_p a parameter valuation and
- θ the firing dates for every transition enabled by m

The transition relation $\rightarrow$:

- either $(m, v_p, \theta) \xrightarrow{d} (m, v_p, \theta')$ for the time delay transition
- or $(m, v_p, \theta) \xrightarrow{t_f} (m', v_p, \theta')$ for the firing of a transition t_f

Parametric Time Petri Net (PTPN)

Semantics

States of the PTPN $\mathcal{N} = (P, T, \mathbb{P}, F, m_0, I_s)$ is (m, v_p, θ) with

- $m \subseteq P$ a marking,
- v_p a parameter valuation and
- θ the firing dates for every transition enabled by m

The transition relation $\rightarrow$:

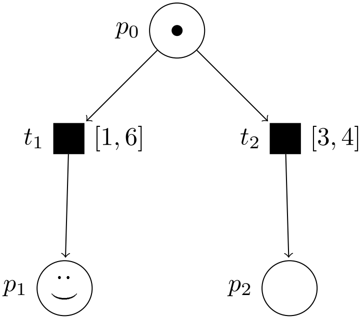
- either $(m, v_p, \theta) \xrightarrow{d} (m, v_p, \theta')$ for the time delay transition
- or $(m, v_p, \theta) \xrightarrow{t_f} (m', v_p, \theta')$ for the firing of a transition t_f

Firing dates are chosen when the transition become enable instead of at the firing time (moment of firing)[3].

[3] Loriane Leclercq, Didier Lime, and Olivier H. Roux. "A state class based controller synthesis approach for Time Petri Nets". In: *Petri Nets 2023*. LNCS. Springer.

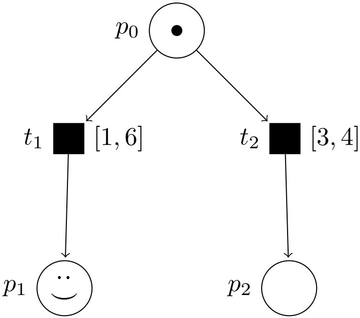
Example

Without parameter



Example

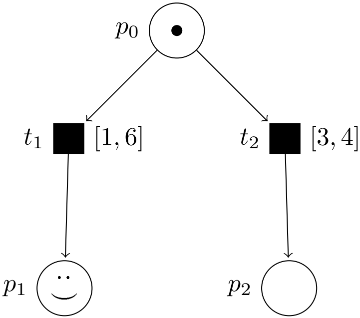
Without parameter

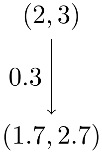


$(2, 3)$

Example

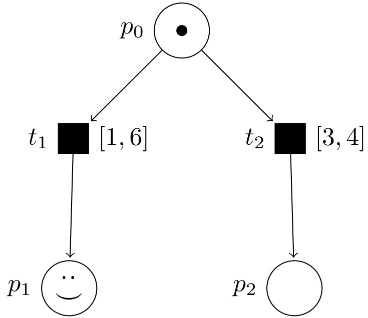
Without parameter





Example

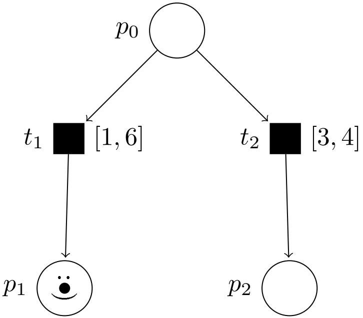
Without parameter



$$
\begin{tikzcd}
(2, 3) \arrow[d, "0.3"'] \\
(1.7, 2.7) \arrow[d, "1.7"'] \\
(0, 1)
\end{tikzcd}
$$

Example

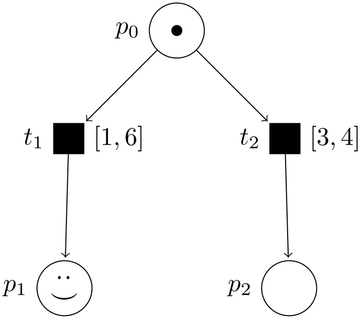
Without parameter



$$(2, 3) \xrightarrow{0.3} (1.7, 2.7) \xrightarrow{1.7} (0, 1) \xrightarrow{t_1} (\perp, \perp)$$

Example

Without parameter

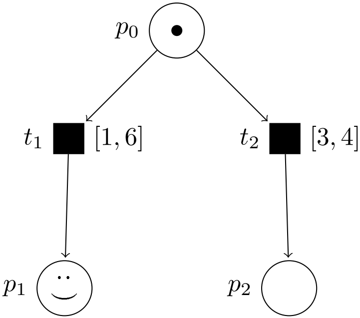


$$(2,3) \qquad (5,3)$$

$$(2,3) \xrightarrow{0.3} (1.7, 2.7) \xrightarrow{1.7} (0,1) \xrightarrow{t_1} (\perp, \perp)$$

Example

Without parameter

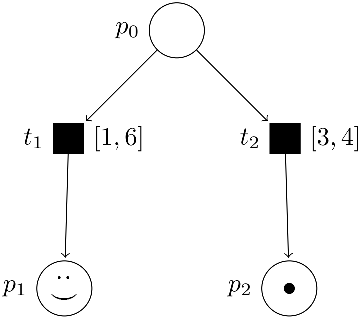


$$(2, 3) \xrightarrow{0.3} (1.7, 2.7) \xrightarrow{1.7} (0, 1) \xrightarrow{t_1} (\perp, \perp)$$

$$(5, 3) \xrightarrow{3} (2, 0)$$

Example

Without parameter

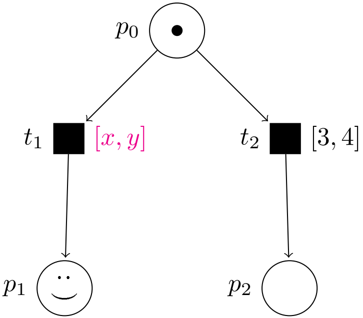


$$(2,3) \xrightarrow{0.3} (1.7,2.7) \xrightarrow{1.7} (0,1) \xrightarrow{t_1} (\perp,\perp)$$

$$(5,3) \xrightarrow{3} (2,0) \xrightarrow{t_2} (\perp,\perp)$$

Example

The parametric case



$$(2,3) \xrightarrow{0.3} (1.7, 2.7) \xrightarrow{1.7} (0, 1) \xrightarrow{t_1} (\perp, \perp)$$

$$(5,3) \xrightarrow{3} (2, 0) \xrightarrow{t_2} (\perp, \perp)$$

Infinite number of states

The non-parametric case

As an initial state we can take $(1, 3)$, $(1.1, 3)$, $(1.01, 3)$, $(1.1, 2.9)$, $(\frac{4}{3}, 3)$,...

Infinite number of states

The non-parametric case

As an initial state we can take $(1, 3)$, $(1.1, 3)$, $(1.01, 3)$, $(1.1, 2.9)$, $(\frac{4}{3}, 3)$,...

The state space in infinite !

Infinite number of states

The non-parametric case

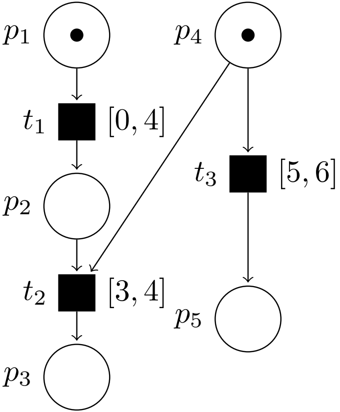
As an initial state we can take $(1, 3)$, $(1.1, 3)$, $(1.01, 3)$, $(1.1, 2.9)$, $(\frac{4}{3}, 3)$,...

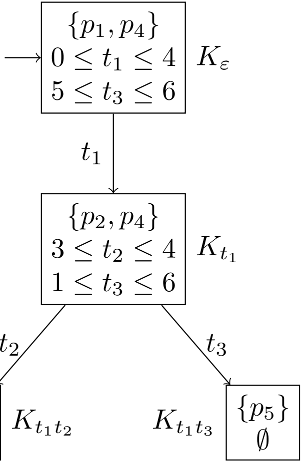
The state space in infinite !

We need finite representations: *state classes* and the state class graph

State Class Graph (SCG)

The non-parametric case





Parametric state classes

A finite representation of the state space

The *parametric state class* $K_{t_1 \ldots t_n}$ is the set of all states obtained by firing $t_1 \ldots t_n$ in order,

Notation : $K_\sigma = (m, D)$

The *firing domain* D:

- is the union of all possible firing dates and parameters,
- points in D are (v_p, θ) with v_p a valuation of $\mathbb{P}$ and θ the firing dates of enabled transitions.

Parametric State Class Graph (PSCG)

The *Parametric State Class Graph (PSCG)* is the directed graph whose states are the state classes whose transitions are such that:

$$K_\sigma \xrightarrow{t} K_{\sigma \cdot t}$$

Convex polyhedra to DBM

Constraints of a state classe (m, D) :

$$\forall t_i, t_j \in \mathsf{en}(m) : \begin{array}{c} -e_{0i} \leq \theta_i \leq e_{i0} \\ -e_{ji} \leq \theta_i - \theta_j \leq e_{ij} \end{array}$$

Convex polyhedra to DBM

Constraints of a state classe (m, D) :

$$\forall t_i, t_j \in \mathsf{en}(m) : \begin{array}{c} -e_{0i} \leq \theta_i \leq e_{i0} \\ -e_{ji} \leq \theta_i - \theta_j \leq e_{ij} \end{array}$$

Difference Bound Matrix (DBM):

$$\theta_i - \theta_j \prec_{ij} e_{ij} \iff M_c[i,j] = (e_{ij}, \prec_{ij})$$

with $e_{ij} \in \mathbb{N}$ and $\prec_{ij} \in \{\leq, <\}$

Convex polyhedra to tPDBM

Constraints of a parametric state classe (m, D) :

$$\forall t_i, t_j \in \mathsf{en}(m) : \begin{array}{c} -e_{0i} \leq \theta_i \leq e_{i0} \\ -e_{ji} \leq \theta_i - \theta_j \leq e_{ij} \end{array}$$

Tropical Parametric Difference Bound Matrix (PDBM):

$$\theta_i - \theta_j \prec_{ij} e_{ij} \iff M_c[i,j] = (e_{ij}, \prec_{ij})$$

with $e_{ij} \in \mathbb{T}$ and $\prec_{ij} \in \{\leq, <\}$

Convex polyhedra to tPDBM

Constraints of a parametric state classe (m, D) :

$$\forall t_i, t_j \in \mathsf{en}(m) : \begin{array}{c} -e_{0i} \leq \theta_i \leq e_{i0} \\ -e_{ji} \leq \theta_i - \theta_j \leq e_{ij} \end{array}$$

Tropical Parametric Difference Bound Matrix (PDBM):

$$\theta_i - \theta_j \prec_{ij} e_{ij} \iff M_c[i,j] = (e_{ij}, \prec_{ij})$$

with $e_{ij} \in \mathbb{T}$ and $\prec_{ij} \in \{\leq, <\}$

Parametric state classes can be represented by tPDBM using tropical constraints as bounds.

Efficient State Class Graph (SCG)

Efficient way to compute the successor domain of a class using DBM[4][5]

$$\mathsf{Succ_p}((m, M_c), t_f) = \begin{cases} M'_c[o,i] = \min_{k \in \mathsf{en}(m)}(0, M_c[k,i]) \\ M'_c[i,o] = M_c[i,f] \\ M'_c[i,j] = \min(M_c[i,j], M'_c[i,o] + M'_c[o,j]) \\ \qquad\qquad\qquad\qquad\qquad (\mathit{if}\ i,j \notin \mathsf{newen}(m,t_f)) \\ M'_c[i,j] = M'_c[i,o] + M'_c[o,j] \qquad (\mathit{otherwise}) \end{cases}$$

[4] Boucheneb and Mullins, "Analyse des réseaux temporels : Calcul des classes en $O(n^2)$ et des temps de chemin en $O(m \times n)$".

[5] Bourdil et al., "Symmetry reduction for time Petri net state classes".

Efficient Parametric State Class Graph (PSCG)

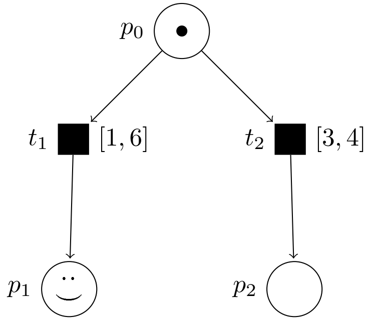
Efficient way to compute the successor domain of a class using DBM[4][5]

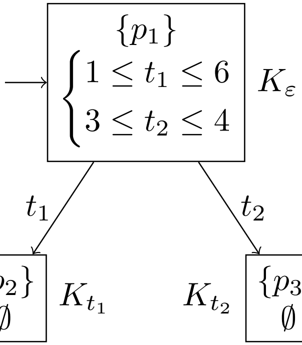
$$\mathsf{Succ_p}((m, M_c), t_f) = \begin{cases} M'_c[o,i] = \widetilde{\min}_{k \in \mathsf{en}(m)}(0, M_c[k,i]) \\ M'_c[i,o] = M_c[i,f] \\ M'_c[i,j] = \widetilde{\min}(M_c[i,j], M'_c[i,o] \,\widetilde{+}\, M'_c[o,j]) \\ \qquad\qquad\qquad\qquad (\mathit{if}\ i,j \notin \mathsf{newen}(m,t_f)) \\ M'_c[i,j] = M'_c[i,o] \,\widetilde{+}\, M'_c[o,j] \qquad (\mathit{otherwise}) \end{cases}$$

[4] Boucheneb and Mullins, "Analyse des réseaux temporels : Calcul des classes en $O(n^2)$ et des temps de chemin en $O(m \times n)$".

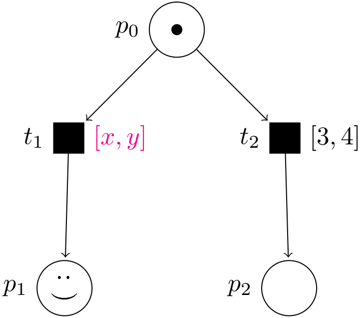
[5] Bourdil et al., "Symmetry reduction for time Petri net state classes".

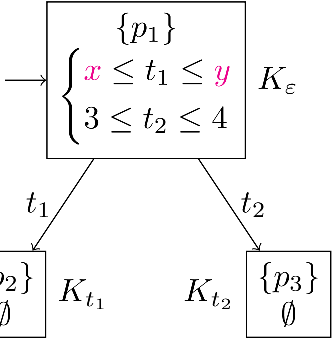
Example for SCG





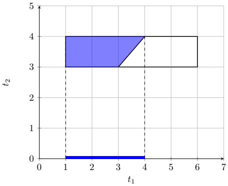
Example for PSCG



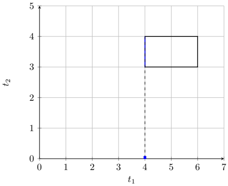


The importance of parameter selection for reachability

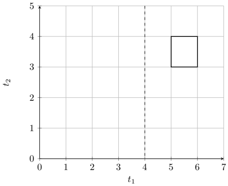
We suppose that $y = 6$ and try different values for x:



(a) $x = 1$

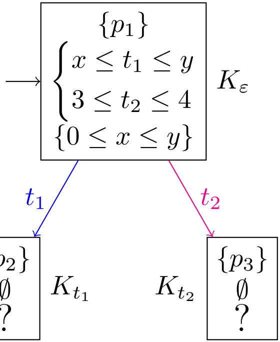


(b) $x = 4$



(c) $x = 5$

Computing successor for reachability

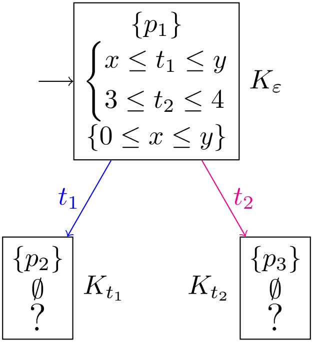


$$D = \begin{pmatrix} \{p_1\} \\ \begin{cases} x \leq t_1 \leq y \\ 3 \leq t_2 \leq 4 \end{cases} \\ \{0 \leq x \leq y\} \end{pmatrix}$$

$$M_c = \begin{pmatrix} 0 & \text{-x} & \text{-3} \\ \text{y} & 0 & \text{y-3} \\ 4 & \text{4-x} & 0 \end{pmatrix}$$

$$P_c = \{0 \leq x \leq y\}$$

Computing successor for reachability



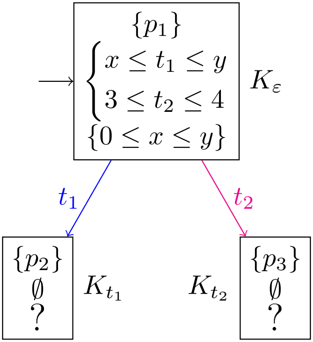
$$D = \begin{pmatrix} \{p_1\} \\ \begin{cases} x \leq t_1 \leq y \\ 3 \leq t_2 \leq 4 \end{cases} \\ \{0 \leq x \leq y\} \end{pmatrix}$$

$$M_c = \begin{pmatrix} 0 & \text{-x} & \text{-3} \\ \text{y} & 0 & \text{y-3} \\ 4 & \text{4-x} & 0 \end{pmatrix}$$

$$P_c = \{0 \leq x \leq y\}$$

$$\mathsf{fir}_{\mathsf{p}}(D, t_1) = \{0 \leq x \leq y\} \cap \{4 - x \geq 0\}$$
$$P_c^{t_1} = \{0 \leq x \leq y, x \leq 4\}$$

Computing successor for reachability



$$D = \begin{pmatrix} \{p_1\} \\ \begin{cases} x \leq t_1 \leq y \\ 3 \leq t_2 \leq 4 \end{cases} \\ \{0 \leq x \leq y\} \end{pmatrix}$$

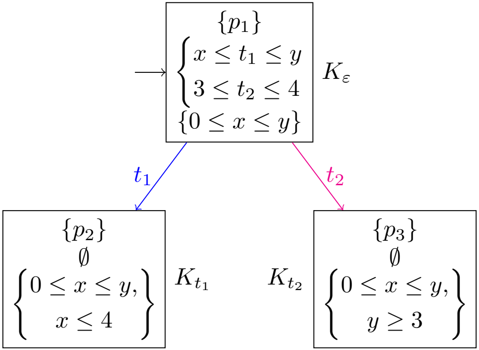
$$M_c = \begin{pmatrix} 0 & \text{-x} & \text{-3} \\ \text{y} & 0 & \text{y-3} \\ 4 & \text{4-x} & 0 \end{pmatrix}$$

$$P_c = \{0 \leq x \leq y\}$$

$\mathsf{fir}_{\mathsf{p}}(D, t_1) = \{0 \leq x \leq y\} \cap \{4 - x \geq 0\}$
$P_c^{t_1} = \{0 \leq x \leq y, x \leq 4\}$

$\mathsf{fir}_{\mathsf{p}}(D, t_2) = \{0 \leq x \leq y\} \cap \{y - 3 \geq 0\}$
$P_c^{t_2} = \{0 \leq x \leq y, y \geq 3\}$

Parameter constraints for reachability



Inclusion for reachability

Two constrained tPDBMs $D^1 = (P_c^1, M_c^1)$ and $D^2 = (P_c^2, M_c^2)$

$$[\![D^1]\!] \subseteq [\![D^2]\!] \iff \begin{cases} [\![P_c^1]\!] \subseteq [\![P_c^2]\!] \text{ and} \\ \forall v_p \in [\![P_c^1]\!], \forall i, j, v_p(M_c^1[i,j]) \leq v_p(M_c^2[i,j]) \end{cases}$$

Implementation

The tool Roméo:

https://romeo.ls2n.fr

Tested on several examples from the litterature:

- Fischer's mutual exclusion protocol
- level crossing
- producer-consumer

Case study: Fischer's mutual-exclusion protocol

Nb. of par.	Nb. of proc.	General Polyhedron Time (s)	General Polyhedron Mem. (Mb)	Tropical PDBM Time (s)	Tropical PDBM Mem. (Mb)	(split) PDBM Time (s)	(split) PDBM Mem. (Mb)
1	4	1.6	16	**0.4**	**9**	1.2	50
1	5	8.2	71	**1.8**	**47.7**	5.2	260
1	6	49	367	**11**	**283**	26	1460
1	7	296	2155	**67**	**1745**	133	4293
1	8	DNF	DNF	**404**	**9616**	DNF	DNF
2	4	1.6	16	**0.6**	**12**	1	62
2	5	8	71	**3.4**	**61**	4.8	322
2	6	48	360	**23**	**350**	26	1740
2	7	305	**2086**	148	2104	**143**	9700
2	8	DNF	DNF	DNF	DNF	DNF	DNF

Table: Results for Fischer's protocol with one parameter (fixing $B = 1$) and two parameters

Future work

- take advantage of tPDBM specific form to compute integer hulls more efficiently
- efficient comparison of tropical expressions
- extend operations on tPDBMs for more complex algorithims: controller synthesis with parameter for reachability, safety and ω-properties
- applications to real-life problems

Thank you!

References

Boucheneb, Hanifa and John Mullins. “Analyse des réseaux temporels : Calcul des classes en $O(n^2)$ et des temps de chemin en $O(m \times n)$”. In: *TSI. Technique et science informatiques* 22.4 (2003), pp. 435–459.

Bourdil, Pierre-Alain et al. “Symmetry reduction for time Petri net state classes”. In: *Science of Computer Programming* 132 (2016), pp. 209–225.

Hune, Thomas et al. “Linear parametric model checking of timed automata”. In: *Journal of Logic and Algebraic Programming* 52-53 (2002), pp. 183–220.

Leclercq, Loriane, Didier Lime, and Olivier H. Roux. “A state class based controller synthesis approach for Time Petri Nets”. In: *Petri Nets 2023*. LNCS. Springer.

Appendix

Parametric Time Petri Net (PTPN)

Definition

A *parametric time Petri net* (TPN) is a tuple $\mathcal{N} = (P, T, \mathbb{P}, F, m_0, I_s)$:

- P is a finite non-empty set of *places*,
- T is a finite set of *transitions* such that $T \cap P = \emptyset$,
- $\mathbb{P}$ is a finite set of *parameters* such that $\forall q \in \mathbb{P}, q \in \mathbb{R}$
- $F : (P \times T) \cup (T \times P)$ is the *flow function*,
- $m_0 \subset P$ the *initial marking*
- $I_s : T \to \mathcal{I}(\mathbb{T})$ is the *static firing interval* function,

Fischer's mutual-exclusion protocol

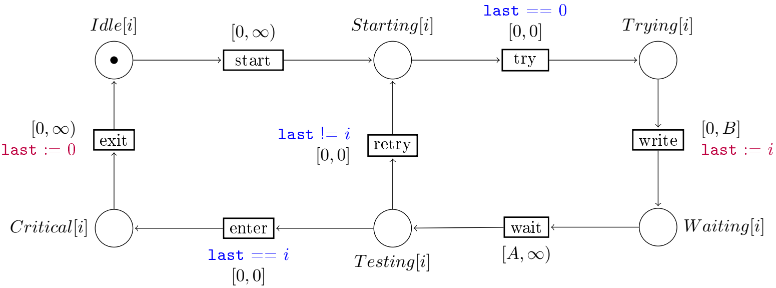


Figure: Fischer's mutual-exclusion protocal, PTPN model of the process i

Semantics

Timed Transition System $(S, s_0, \Sigma, \rightarrow)$ with:

- S the set of states (m, θ),
- initial state $s_0 = (\{p_0\}, \theta_0) \in S$ with $\theta_0(t_{\mathsf{init}}) = 0$
- a labelling alphabet Σ containing letters $t_f \in T$ and $d \in \mathbb{R}_{\geq 0}$,
- the transition relation $\rightarrow \subseteq S \times \Sigma \times S$:
  $s \xrightarrow{a} s' \Leftrightarrow$
  - either $(m, \theta) \xrightarrow{t_f} (m', \theta')$ for $t_f \in T$ when:
    1. $t_f \in \mathsf{en}(m)$ and $\theta_f = 0$
    2. $m' = (m \setminus \mathsf{Pre}(t_f)) \cup \mathsf{Post}(t_f)$
    3. $\forall t_k \in T, \begin{pmatrix} \theta'_k \in I_s(t_k) & \text{if } t_k \in \mathsf{newen}(m, t_f) \\ \wedge\theta'_k = \theta_k & \text{if } t_k \in \mathsf{pers}(m, t_f) \\ \wedge\theta'_k = \bot & \text{otherwise} \end{pmatrix}$
  - or $(m, \theta) \xrightarrow{d} (m, \theta')$ when:
    - $d \in \mathbb{R}_{\geq 0} \setminus \{0\}$,
    - $\forall t_k \notin \mathsf{en}(m), \theta_k = \bot$, and
    - $\forall t_k \in \mathsf{en}(m), \theta_k - d \geq 0$ and $\theta'_k = \theta_k - d$.

State Class Graph

algorithm from Berthomieu et al.[6]

- Initial system $K_\epsilon = \{\theta_k \in I_s(k) \mid t_k \in \mathsf{en}(m_0)\}$
- if σ firable then $\sigma.t_k$ firable if and only if :
  1. $t \in \mathsf{en}(m)$
  2. $K_\sigma \wedge \{\theta_k \leq \theta_i \mid i \neq k \wedge t_i \in \mathsf{en}(m)\}$ consistent
- If $\sigma.t_k$ is firable, then $K_{\sigma.t_k}$ is computed from K_σ :
  - Add $\{\theta_k \leq \theta_i \mid i \neq k \wedge t_i \in \mathsf{en}(m)\}$ to K_σ
  - $\forall t_i \in \mathsf{en}(m')$ we add θ'_i such that:
    $\theta'_i = \theta_i - \theta_k$ if $k \neq i \wedge t_i \notin \mathsf{newen}(m, t_k)$
    $\theta'_i \in I_s(i)$ otherwise
  - Eliminate θ_i variables $\forall i$

[6] **BM83**.