A state class based controller synthesis approach for Time Petri Nets[1]

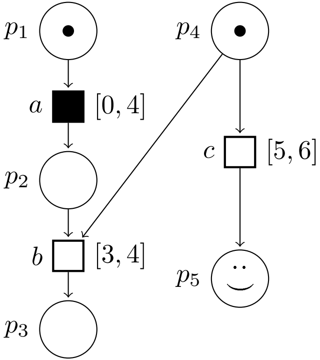
Loriane Leclercq, Didier Lime and Olivier H. Roux

Petri Nets 2023
June 29, 2023

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

Time Petri Net (TPN)

Example



Introduction

- Controller synthesis for Time Petri Nets
- Timed game for reachability
- Explicit firing dates semantics
- State classes

Time Petri Net (TPN)

Definition

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

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

Semantics

States of the TPN: (m, θ) with

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

Semantics

States of the TPN: (m, θ) with

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

The transition relation $\rightarrow \subseteq S \times \Sigma \times S$:

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

Semantics

States of the TPN: (m, θ) with

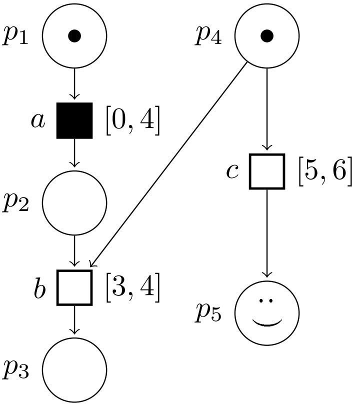
- $m \subseteq P$ a marking and
- θ the firing dates for every transition enabled by m

The transition relation $\rightarrow \subseteq S \times \Sigma \times S$:

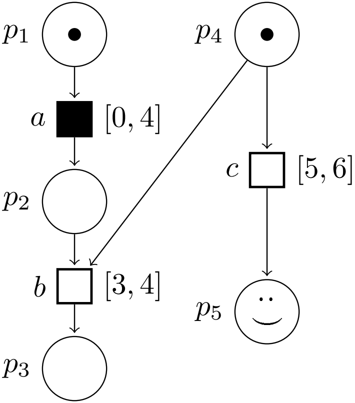
- either $(m, \theta) \xrightarrow{d} (m, \theta')$ for the time delay transition
- or $(m, \theta) \xrightarrow{t_f} (m', \theta')$ for the firing of a transition t_f

Firing dates are choosen when the transition become enable instead of at the firing time (moment of firing).

Example

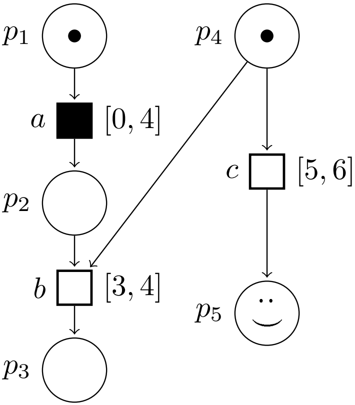


Example



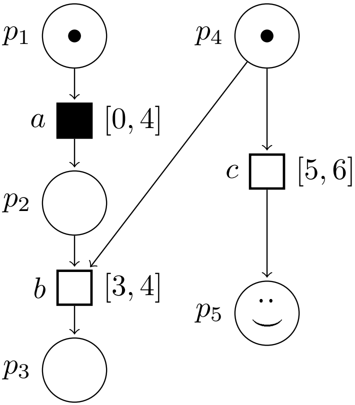
$(1, \perp, 5)$

Example



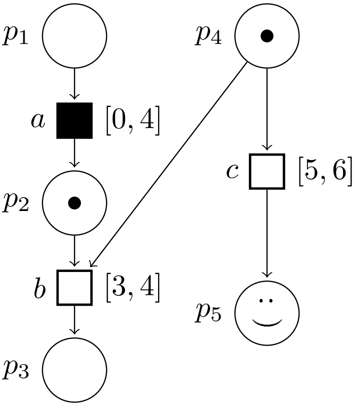
$$
\begin{array}{c}
(1, \perp, 5) \\
\downarrow 0.3 \\
(0.7, \perp, 4.7)
\end{array}
$$

Example



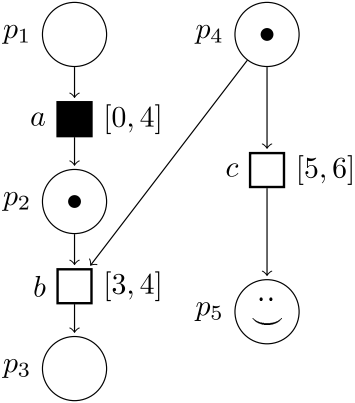
$$\begin{array}{c} (1, \perp, 5) \\ \big\downarrow{\scriptstyle 0.3} \\ (0.7, \perp, 4.7) \\ \big\downarrow{\scriptstyle 0.7} \\ (0, \perp, 4) \end{array}$$

Example



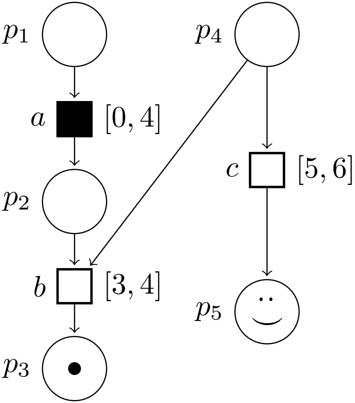
$$(1, \perp, 5) \xrightarrow{0.3} (0.7, \perp, 4.7) \xrightarrow{0.7} (0, \perp, 4) \xrightarrow{a} (\perp, \mathbf{3}, 4)$$

Example



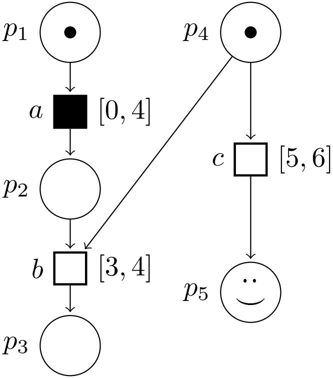
$$
\begin{array}{c}
(1, \perp, 5) \\
\big\downarrow{\scriptstyle 0.3} \\
(0.7, \perp, 4.7) \\
\big\downarrow{\scriptstyle 0.7} \\
(0, \perp, 4) \\
\big\downarrow{\scriptstyle a} \\
(\perp, \mathbf{3}, 4) \\
\big\downarrow{\scriptstyle 3} \\
(\perp, 0, 1)
\end{array}
$$

Example

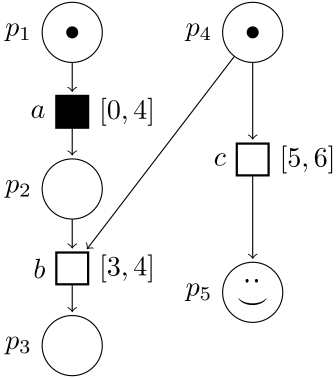


$$
\begin{array}{c}
(1, \bot, 5) \\
\Big\downarrow{\scriptstyle 0.3} \\
(0.7, \bot, 4.7) \\
\Big\downarrow{\scriptstyle 0.7} \\
(0, \bot, 4) \\
\Big\downarrow{\scriptstyle a} \\
(\bot, \mathbf{3}, 4) \\
\Big\downarrow{\scriptstyle 3} \\
(\bot, 0, 1) \\
\Big\downarrow{\scriptstyle b} \\
(\bot, \bot, \bot)
\end{array}
$$

Initial states



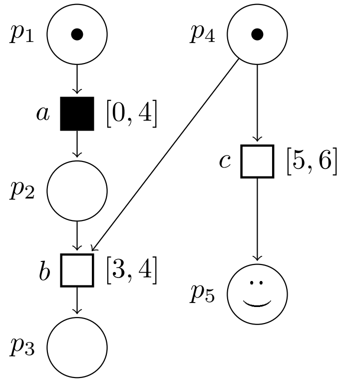
Initial states



In interval-based semantics :

$$S_0 = \left(\{p_1, p_4\}, \begin{matrix} 0 \leq a \leq 4 \\ 5 \leq c \leq 6 \end{matrix} \right)$$

Initial states



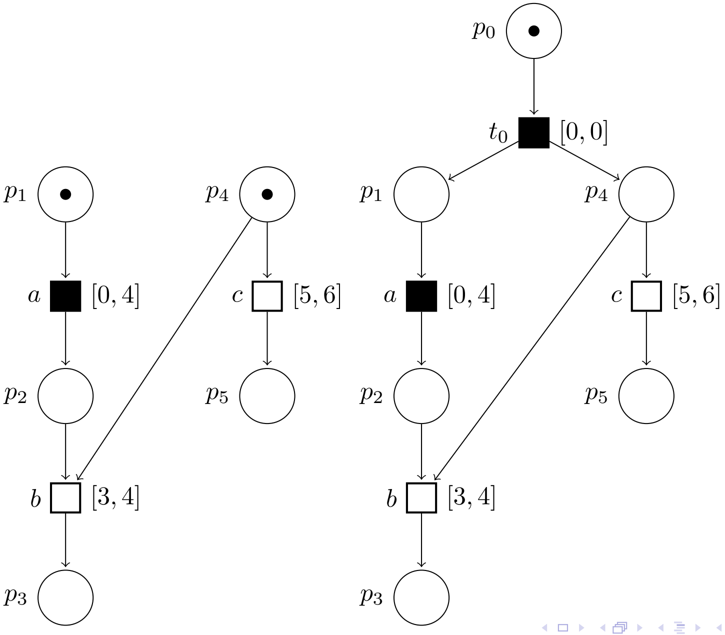
In interval-based semantics :

$$S_0 = \left(\{p_1, p_4\}, \begin{array}{c} 0 \leq a \leq 4 \\ 5 \leq c \leq 6 \end{array} \right)$$

In explicit firing-dates semantics :

$$S_0 \supseteq \left(\{p_1, p_4\}, \begin{array}{r} 0 \\ \bot \\ 5 \end{array} \right), \left(\{p_1, p_4\}, \begin{array}{r} 3 \\ \bot \\ 6 \end{array} \right), \ldots$$

Initial transition



Reachability game

Definition

$\mathcal{R} = (\mathcal{A}, \mathsf{Goal})$ with:

- an arena $\mathcal{A} = (S, \rightarrow, Pl, (Mov_i)_{i \in Pl}, Trans)$
  - Pl_c : controllable transitions
  - Pl_u : uncontrollable transitions
- a set of target states $\mathsf{Goal} \in S$ that Pl_c wants to reach and Pl_u wants to avoid.

Reachability game

Definition

$\mathcal{R} = (\mathcal{A}, \mathsf{Goal})$ with:

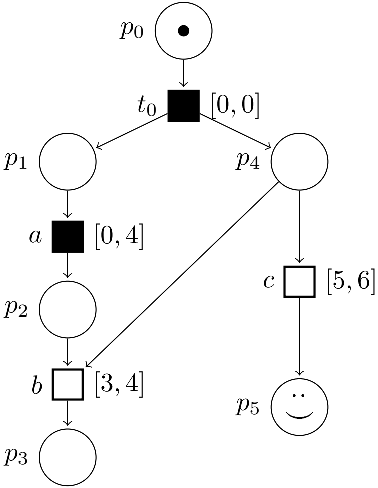
- an arena $\mathcal{A} = (S, \rightarrow, Pl, (Mov_i)_{i \in Pl}, Trans)$
  - Pl_c : controllable transitions
  - Pl_u : uncontrollable transitions
- a set of target states $\mathsf{Goal} \in S$ that Pl_c wants to reach and Pl_u wants to avoid.

Turn :

1. Pl_c chooses $t_c \in T_c$
2. Pl_u chooses $t_u \in T_u \cup \{t_c\}$
3. Both player chooses firing dates for their newly enabled transitions, controllable or uncontrollable.

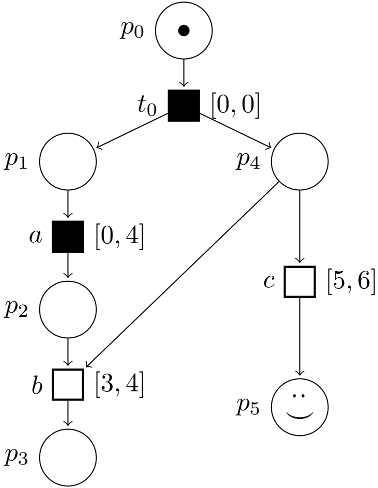
Reachability game

Example



Reachability game

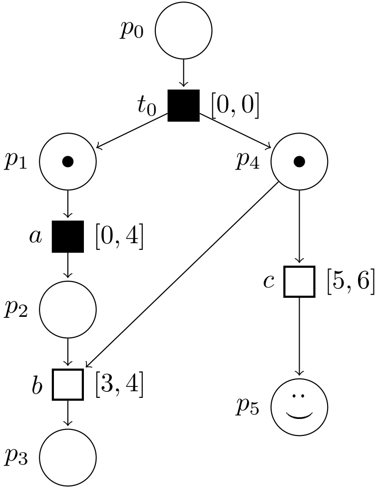
Example



$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

Reachability game

Example

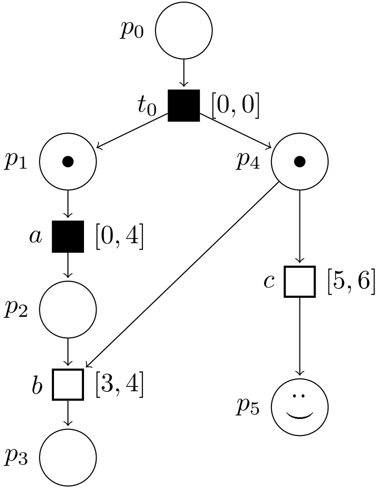


$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

$$t_c = t_0, t_u = t_c, \theta(a) = 2, \theta(c) = 6$$

Reachability game

Example

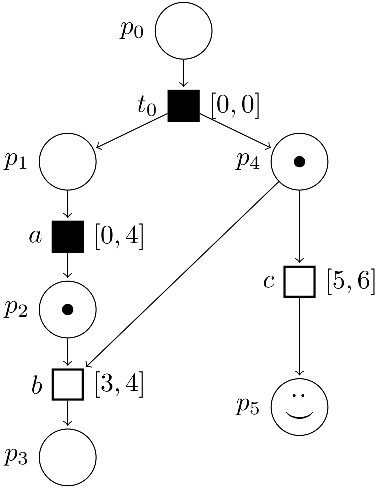


$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

$$s_1 = (\{p_1, p_4\}, \theta(a) = 2, \theta(c) = 6)$$

Reachability game

Example



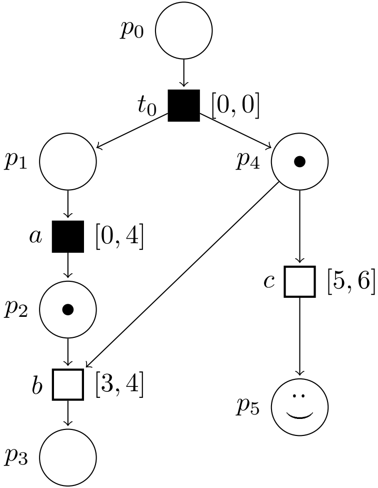
$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

$$s_1 = (\{p_1, p_4\}, \theta(a) = 2, \theta(c) = 6)$$

$$t_c = a, t_u = t_c, \theta(b) = 4$$

Reachability game

Example



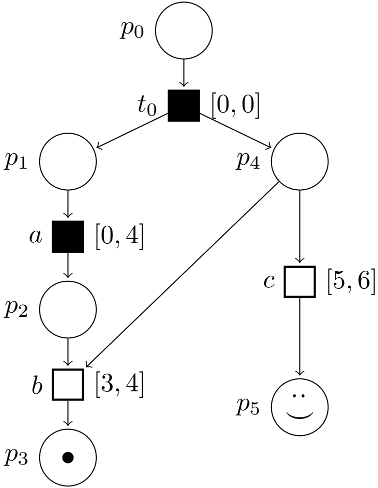
$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

$$s_1 = (\{p_1, p_4\}, \theta(a) = 2, \theta(c) = 6)$$

$$s_2 = (\{p_2, p_4\}, \theta(b) = 4, \theta(c) = 4)$$

Reachability game

Example



$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

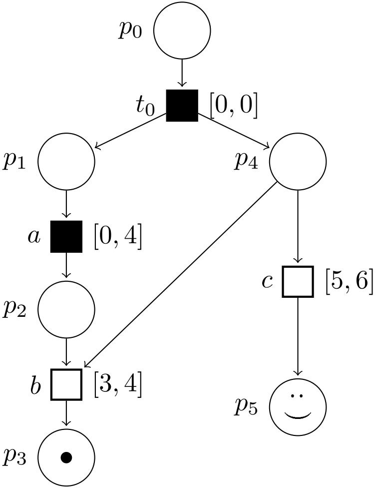
$$s_1 = (\{p_1, p_4\}, \theta(a) = 2, \theta(c) = 6)$$

$$s_2 = (\{p_2, p_4\}, \theta(b) = 4, \theta(c) = 4)$$

$$t_u = b$$

Reachability game

Example



$$s_0 = (\{p_0\}, \theta(t_0) = 0)$$

$$s_1 = (\{p_1, p_4\}, \theta(a) = 2, \theta(c) = 6)$$

$$s_2 = (\{p_2, p_4\}, \theta(b) = 4, \theta(c) = 4)$$

$$t_u = b$$

$$s_3 = (\{p_3\}, \perp)$$

State Class Graph

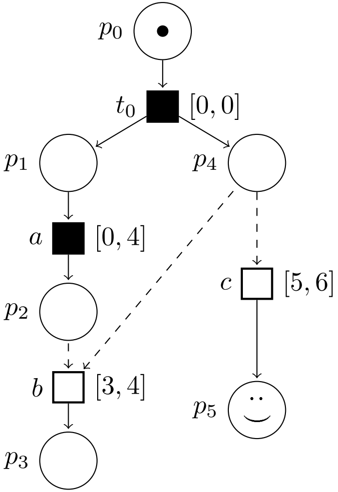
Algorithm from Berthomieu et al.[2]

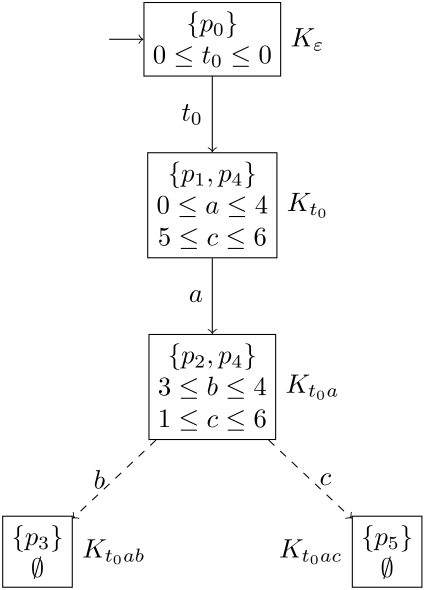
Algorithm Successor (m', D') of (m, D) by firing firable transition t_f

1: $m' \leftarrow (m \setminus \mathsf{Pre}(t_f)) \cup \mathsf{Post}(t_f)$
2: $D' \leftarrow D \wedge \bigwedge_{i \neq f, i \in \mathsf{en}(m)} \theta_f \leq \theta_i$
3: for all $i \in \mathsf{en}(m \setminus \mathsf{Pre}(t_f)), i \neq f$, add variable θ'_i to D', constrained by $\theta'_i = \theta_i - \theta_f$
4: eliminate (by existential projection) variables θ_i for all i from D'
5: for all $i \in \mathsf{newen}(m, t_f)$, add variable θ''_i to D', constrained by $\theta''_i \in I_s(i)$

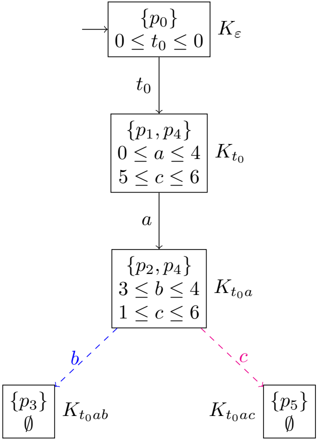
[2] Berthomieu and Menasche, "An Enumerative Approach For Analyzing Time Petri Nets".

State Class Graph (SCG)



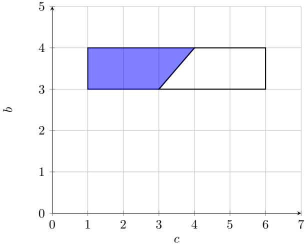


Subset of winning states

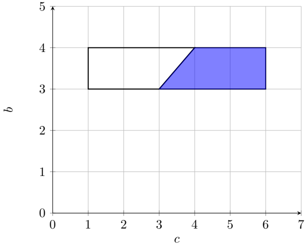


$$C = \begin{pmatrix} \{p_2, p_4\} \\ 3 \leq b \leq 4 \\ 1 \leq c \leq 6 \\ c < b \end{pmatrix}$$

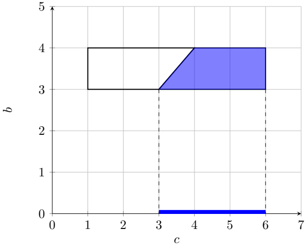
Universal projection : $\pi^{\forall}_{\{c\}}(D_{t_0 a}, D)$



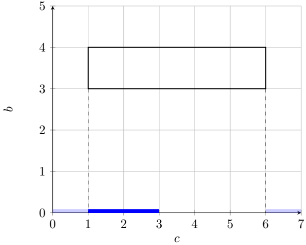
(a) Set D in the rectangle $D_{t_0 a}$



(b) $D_{t_0 a} \cap \overline{D}$

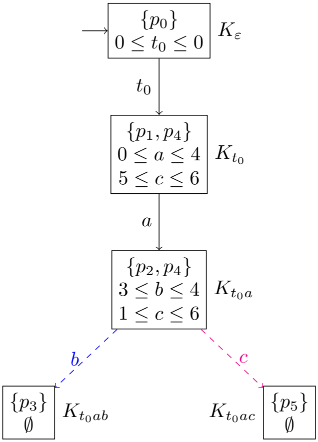


(c) $\pi^{\exists}_{\{c\}}(D_{t_0 a} \cap \overline{D})$



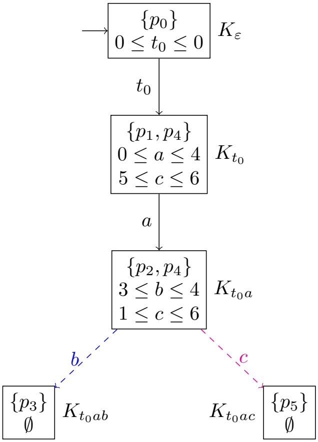
(d) $\pi^{\exists}_{\{c\}}(D_{t_0 a}) \cap \overline{\pi^{\exists}_{\{c\}}(D_{t_0 a} \cap \overline{D})}$

Subset of winning states



$$C = \begin{pmatrix} \{p_2, p_4\} \\ 3 \leq b \leq 4 \\ 1 \leq c \leq 6 \\ c < b \end{pmatrix} = \begin{pmatrix} \{p_2, p_4\} \\ 3 \leq b \leq 4 \\ 1 \leq c < 3 \end{pmatrix}$$

Subset of winning states



$$C' = \begin{pmatrix} \{p_1, p_4\} \\ 0 \le a \le 4 \\ 5 \le c \le 6 \\ 1 \le c - a < 3 \end{pmatrix} = \begin{pmatrix} \{p_1, p_4\} \\ 2 < a \le 4 \\ 5 \le c \le 6 \\ 1 \le c - a < 3 \end{pmatrix}$$

$$C = \begin{pmatrix} \{p_2, p_4\} \\ 3 \le b \le 4 \\ 1 \le c \le 6 \\ c < b \end{pmatrix} = \begin{pmatrix} \{p_2, p_4\} \\ 3 \le b \le 4 \\ 1 \le c < 3 \end{pmatrix}$$

Implementation

The tool Roméo https://pagesperso.ls2n.fr/∼lime-d/romeo.html

Future work

- reachability with parametric firing-time constraints: using an efficient linear programming algorithm when the dimension is small and/or fixed
- explicit firing dates semantics expressivity
- applications to concrete problems

Thank you!

References

Berthomieu, Bernard and Miguel Menasche. "An Enumerative Approach For Analyzing Time Petri Nets". In: *Proceedings IFIP*. Elsevier Science Publishers, 1983, pp. 41–46.

Appendix

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.[3]

- 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 $\forall i$

[3] Berthomieu and Menasche, "An Enumerative Approach For Analyzing Time Petri Nets".

Game definition

Loriane: TODO describe mov, trans, pl, ...

A reachability game $\mathcal{R} = (\mathcal{A}, \mathsf{Win})$ with:

- an arena $\mathcal{A} = (S, \rightarrow, Pl, (Mov_i)_{i \in Pl}, Trans)$
- a target set $\mathsf{Win} \in S$

Game definition

Moves

$$\mathsf{Movt}_c(m,\theta) = \{t_i \mid t_i \in \mathsf{en_c}(m) \wedge \theta_i = \min_{t_k \in \mathsf{en}(m)} \theta_k\}$$

$$\mathsf{Movt}_u((m,\theta),t_c) = \{t_i \mid t_i \in \mathsf{en_u}(m) \wedge \theta_i = \min_{t_k \in \mathsf{en}(m)} \theta_k\} \cup \{t_c\}$$

$$\mathsf{Movf}_c((m,\theta),t_i) = \left\{ \theta^c \in \mathbb{R}^{T_c}_{\geq 0} \;\middle|\; \begin{array}{l} \theta^c_k \in I_s(t_k) \text{ if } t_k \in \mathsf{newen}(m,t_i) \\ \theta^c_k = \theta_k - \theta_i \text{ if } t_k \in \mathsf{pers}(m,t_i) \\ \theta^c_k = \perp \text{ otherwise} \end{array} \right\}$$

$$\mathsf{Movf}_u((m,\theta),t_i) = \left\{ \theta^u \in \mathbb{R}^{T_u}_{\geq 0} \;\middle|\; \begin{array}{l} \theta^u_k \in I_s(t_k) \text{ if } t_k \in \mathsf{newen}(m,t_i) \\ \theta^u_k = \theta_k - \theta_i \text{ if } t_k \in \mathsf{pers}(m,t_i) \\ \theta^u_k = \perp \text{ otherwise} \end{array} \right\}$$

Game definition

Trans

$$Trans : S \times T \times T \times \mathbb{R}_{\geq 0}^{|T_c|}, \mathbb{R}_{\geq 0}^{|T_u|} \to S$$

$$Trans((m,\theta), t_c, t_u, \theta^c, \theta^u) = \text{if } \begin{pmatrix} t_c \in \mathsf{Movt}_c(m,\theta) \\ \wedge t_u \in \mathsf{Movt}_u((m,\theta), t_c) \\ \wedge \theta(t_u) = min_k(\theta(t_k)) \\ \wedge (t_u \in T_u \vee t_c = t_u) \\ \wedge \theta^c \in \mathsf{Movf}_c(s, t_u) \\ \wedge \theta^u \in \mathsf{Movf}_u(s, t_u) \end{pmatrix}$$

$$\text{then } ((m \setminus Pre(t)) \cup Post(t), \theta^c \sqcup \theta^u)$$

$$\text{otherwise } \bot$$

Predecessor

$$\mathsf{Pred}_{C\xrightarrow{t_f}C'}(B) = \{s \in C \mid \exists s'.s \xrightarrow{t_f} s' \in B\}$$

$$\mathsf{cPred}_{C\xrightarrow{t_f}C'}(B) = \left\{ (m,\theta) \in C \left| \begin{array}{l} \forall t_i \in \mathsf{newen_c}(C,t_f), \exists \theta'_i \in I_s(t_i) \text{ s. t.} \\ \forall t_{n+j} \in \mathsf{newen_u}(C,t_f), \forall \theta'_{n+j} \in I_s(t_{n+j}), \\ s \xrightarrow{t_f} s' = (m',\theta') \in B \\ \text{where } \forall i \in [\![1,n+k]\!], \theta'(t_i) = \theta'_i \\ \text{ and } \forall i \in [\![1,l]\!], \theta'(t_{n+k+i}) = \theta(t_{n+k+i}) - \theta(t_f) \end{array} \right. \right\}$$

$$\mathsf{uPred}_{C\xrightarrow{t_f}C'}(B) = \left\{ (m,\theta) \in C \left| \begin{array}{l} \forall t_i \in \mathsf{newen_c}(C,t_f), \forall \theta'_i \in I_s(t_i) \text{ s. t.} \\ \forall t_{n+j} \in \mathsf{newen_u}(C,t_f), \exists \theta'_{n+j} \in I_s(t_{n+j}), \\ s \xrightarrow{t_f} s' = (m',\theta') \in B \\ \text{where } \forall i \in [\![1,n+k]\!], \theta'(t_i) = \theta'_i \\ \text{ and } \forall i \in [\![1,l]\!], \theta'(t_{n+k+i}) = \theta(t_{n+k+i}) - \theta(t_f) \end{array} \right. \right\}$$

Proj ∃ ∀

We first define the classical *existential projection*:
For any set of valuations D s. t. $\forall\theta \in D, \mathsf{tr}(\theta) = \{t_1, \ldots, t_{n+k}\}$,

$$\pi^{\exists}_{\{t_1,\ldots,t_n\}}(D) = \{(\theta_1 ... \theta_n) \mid \exists\theta_{n+1}, ..., \theta_{n+k}, (\theta_1 ... \theta_{n+k}) \in D\}$$

We also define a less usual *universal projection* of D' inside D:
For any two sets of valuations D and D' such that $D' \subseteq D$ and $\forall\theta \in D, \mathsf{tr}(\theta) = \{t_1, \ldots, t_{n+k}\}$,

$$\pi^{\forall}_{\{t_1,\ldots,t_n\}}(D, D') = \left\{ (\theta_1 ... \theta_n) \,\middle|\, \begin{array}{l} \exists\theta_{n+1}, ..., \theta_{n+k}, (\theta_1 ... \theta_{n+k}) \in D \\ \wedge\, \forall\theta_{n+1}, ..., \theta_{n+k}, (\theta_1 ... \theta_{n+k}) \in D \\ \qquad\qquad \implies (\theta_1 ... \theta_{n+k}) \in D' \end{array} \right\}$$

Extension and substitution operations

We also need an *extension operation*:
For any set of valuations D s. t. $\forall\theta \in D, \mathsf{tr}(\theta) = \{t_1, \ldots, t_n\}$,

$$\pi^{-1}_{\{t_1,\ldots,t_{n+k}\}}(D) = \{(\theta_1 \ldots \theta_{n+k}) \mid (\theta_1 \ldots \theta_n) \in D \text{ and } \forall i, \theta_{n+i} \geq 0\}$$

Finally, we define a *backward in time* operator:
For any set of valuations D s. t. $\forall\theta \in D, \mathsf{tr}(\theta) = \{t_1, \ldots, t_n\}$ and for $t_f \neq t_i$ for all $i \in [\![1, n]\!]$,

$$D + t_f = \{(\theta'_1 \ldots \theta'_n \theta'_f) \mid (\theta_1 \ldots \theta_n) \in D, \theta'_f \geq 0 \text{ and } \forall i, \theta'_i = \theta_i + \theta'_f\}$$

The universal projection is expressible with set complements and existential projections only, as stated in the following proposition. We denote by $\overline{D}$ the complement of D, i. e., $\overline{D} = \{s \mid s \notin D\}$.

Let $\tau = \{t_1, ..., t_n\}$ $\forall \tau \subseteq T, \pi_\tau^\forall(D, D') = \pi_\tau^\exists(D) \cap \overline{\pi_\tau^\exists(\overline{D'} \cap D)}$.

Symbolic computing of predecessors

Let $C = (m, D)$ and $C' = (m', D')$.

- Controllable predecessors:

  Consider $B = (m', D'') \subseteq C'$, and let $\mathsf{cPred}_{C \xrightarrow{t_f} C'}(B) = (m, D_p)$. Then:

$$D_p = D \cap \pi^{-1}_{\mathsf{en}(C)} \left(\pi^{\exists}_{\mathsf{pers}(C,t_f)} \left(\pi^{\forall}_{\substack{\mathsf{newen_c}(C,t_f) \\ \cup \mathsf{pers}(C,t_f)}} (D', D'') \right) + t_f \right)$$

- Uncontrollable predecessors:

  Let $\mathsf{cPred}_{C \xrightarrow{t_f} C'}(B) = (m, D_p)$:

$$D_p = D \cap \pi^{-1}_{\mathsf{en}(C)} \left(\pi^{\forall}_{\mathsf{pers}(C,t_f)} \left(\pi^{\exists}_{\substack{\mathsf{newen_u}(C,t_f) \\ \cup \mathsf{pers}(C,t_f)}} (D'), \pi^{\exists}_{\substack{\mathsf{newen_u}(C,t_f) \\ \cup \mathsf{pers}(C,t_f)}} (D'') \right) + t_f \right)$$

Good/bad states

$$\mathsf{uGood}_k(C) = \bigcup_{\substack{(C \xrightarrow{t_f} C') \in \mathcal{G}, \\ t_f \in \mathsf{en_u}(C)}} \left(\mathsf{cPred}_{C \xrightarrow{t_f} C'} (\mathsf{Win}_k \cap C') \right)$$

$$\mathsf{cGood}_k(C) = \bigcup_{\substack{(C \xrightarrow{t_f} C') \in \mathcal{G}, \\ t_f \in \mathsf{en_c}(C)}} \left(\mathsf{cPred}_{C \xrightarrow{t_f} C'} (\mathsf{Win}_k \cap C') \right)$$

$$\mathsf{uBad}_k(C) = \bigcup_{\substack{(C \xrightarrow{t_f} C') \in \mathcal{G}, \\ t_f \in \mathsf{en_u}(C)}} \left(\mathsf{uPred}_{C \xrightarrow{t_f} C'} (\overline{\mathsf{Win}_k} \cap C') \right)$$

$$\mathsf{cBad}_k(C) = \bigcup_{\substack{(C \xrightarrow{t_f} C') \in \mathcal{G}, \\ t_f \in \mathsf{en_c}(C)}} \left(\mathsf{uPred}_{C \xrightarrow{t_f} C'} (\overline{\mathsf{Win}_k} \cap C') \right)$$

Winning states construction

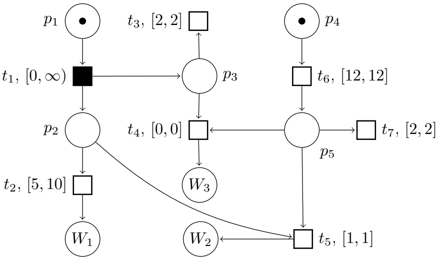
$$\mathsf{Win}_0 = \mathsf{Goal}$$

$$\mathsf{Win}_{k+1} = \mathsf{Win}_k \cup \bigcup_{C \in \mathcal{G}} \Big(\Big[\big(\mathsf{uGood}_k(C) \setminus \mathsf{cBad}_k(C) \big) \cup \mathsf{cGood}_k(C) \Big] \setminus \mathsf{uBad}_k(C) \Big)$$

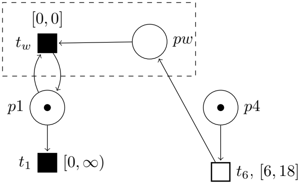
For all state s of $\mathcal{N}$, $s \in \mathsf{Win}_n$ if and only if from s the controller has a strategy to reach Goal in at most n steps.

Case studies

Supply chain



Petri Net model



Reinitializing the firing date of t_1 when t_6 is fired

Strategies :

- If the goal is W_1, initialize t_1 such that: $\theta_1 \in [0,3)$ or $\theta_1 \in (10,+\infty)$
- If the goal is W_2, initialize t_1 such that: $\theta_1 \in (0,3)$
- If the goal is W_3, initialize t_1 such that: $\theta_1 \in (10,12)$ or $\theta_1 \in (12,14)$