Model checking timed linear properties of time Petri nets using the state class method

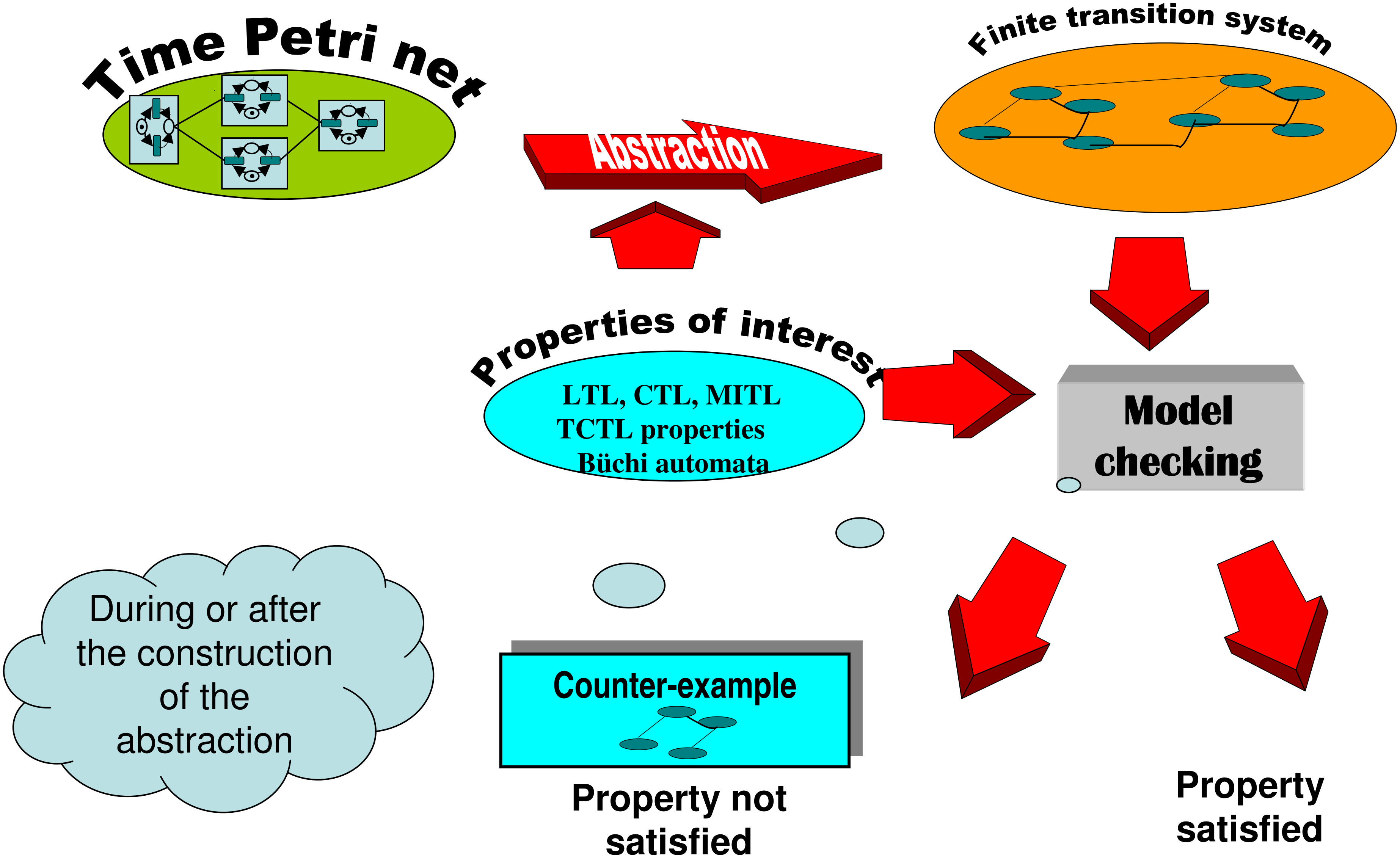
Hanifa Boucheneb
Laboratoire VeriForm

CNAM, March 2008

ÉCOLE
POLYTECHNIQUE
MONTRÉAL

Hanifa Boucheneb EPM

Model checking of time Petri nets based on the state class method



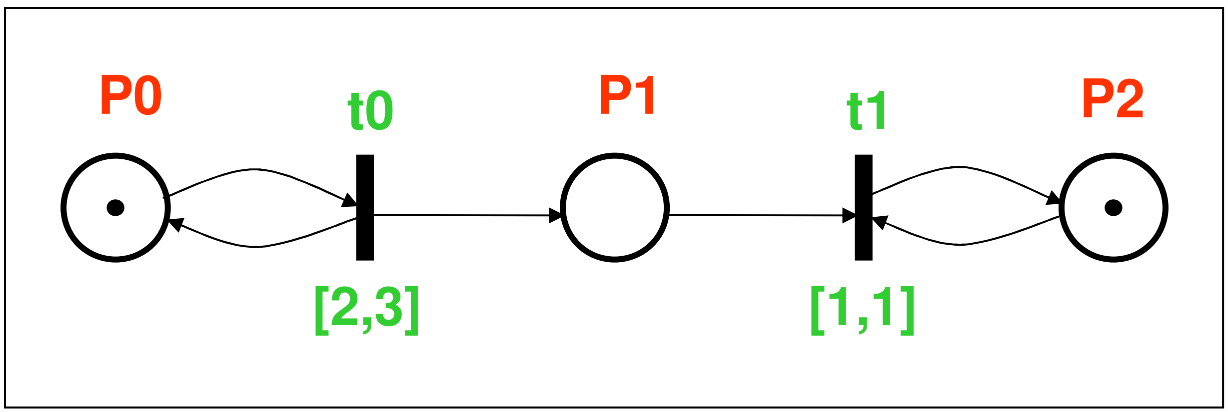
Plan

- Time Petri Nets (TPN model)
- TPN state space abstractions
- The state class graph method (SCG)
- Contracting the state class graph (CSCG)
- Model checking using an interval timed extension of Büchi Automata (ITBA)
- Conclusion

Hanifa Boucheneb EPM

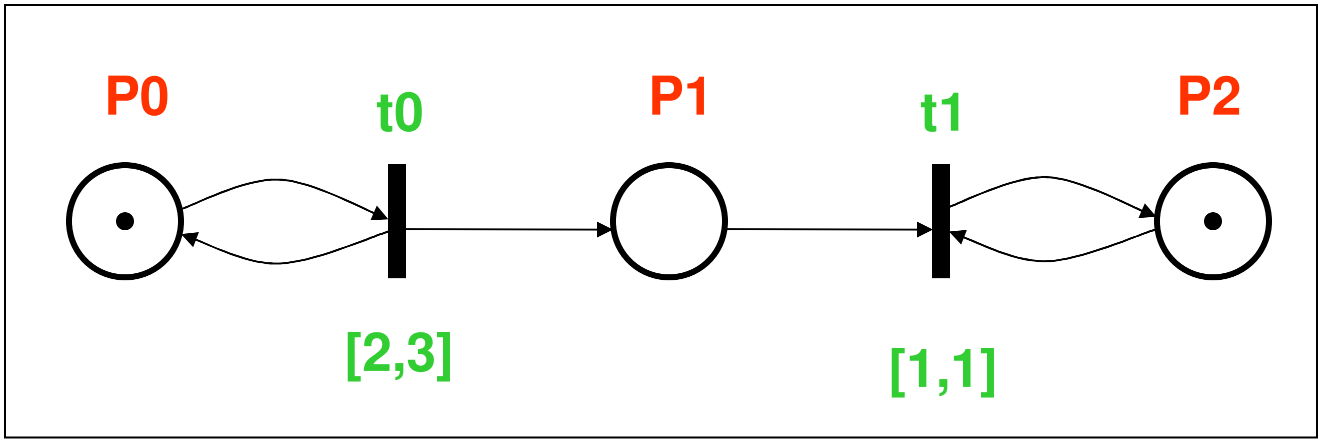
Time Petri Nets (Merlin & Farber 1976)

- Extension of Petri Nets by associating timing constraints with transitions in the form of intervals (static firing intervals)



➔ a good compromise between modeling Power and verification possibilities

TPN State : Interval state vs. Clock state



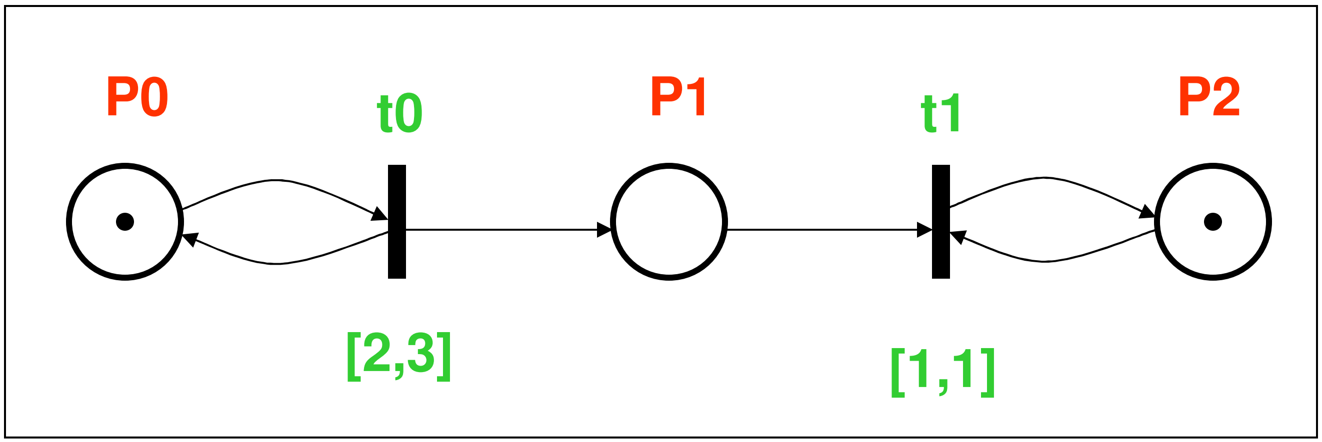
Two state definitions:

Interval State (M,I)	Clock State (M,v)
P0=1, P1=0, P2=1	P0=1, P1=0, P2=1
I(t0) = [2 , 3]	v(t0) = 0

Hanifa Boucheneb EPM

State evolution

TPN state evolves either by time progression or by firing transitions



bounds of intervals decrease with time

Interval State (M,I)

P0=1, P1=0, P2=1

I(t0) = [0 , 0.5]

Clock State (M,v)

P0=1, P1=0, P2=1

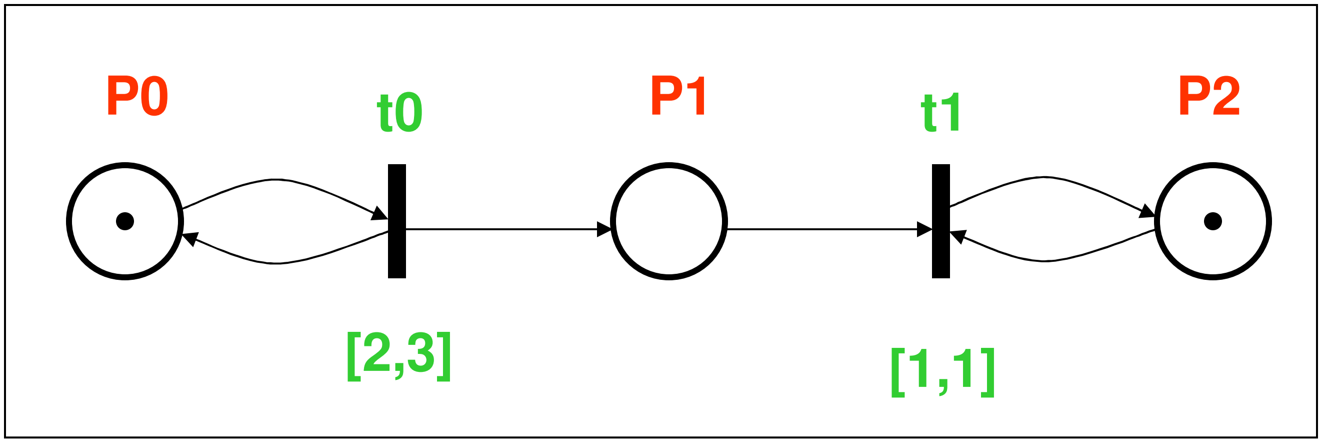
v(t0) = 2.5

Clocks increase with time

Hanifa Boucheneb EPM

State evolution

TPN state evolves either by time progression or by firing transitions



lower bound reaches 0
➔ t0 is firable

Interval State (M,I)

P0=1, P1=0, P2=1

I(t0) = [0 , 0.5]

Clock State (M,v)

P0=1, P1=0, P2=1

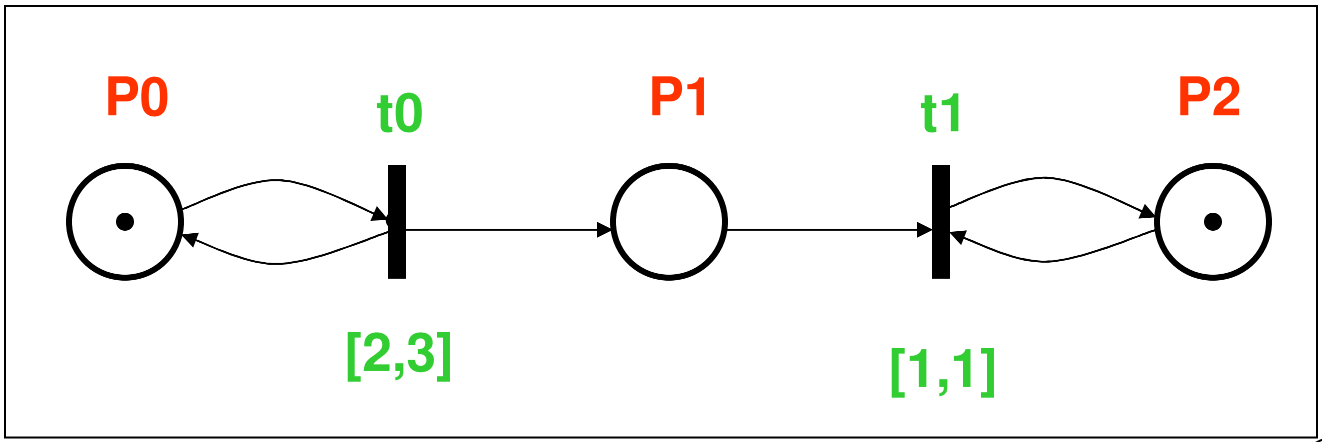
v(t0) = 2.5

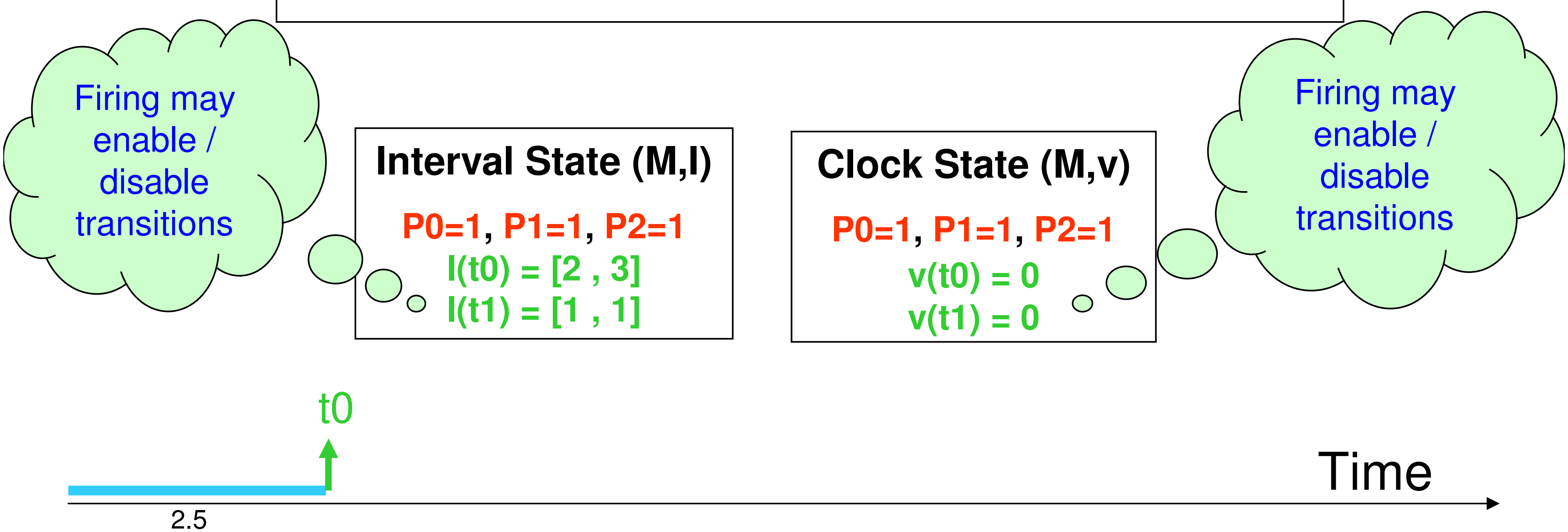
Clock reaches the interval of t0
➔ t0 is fiable

Hanifa Boucheneb EPM

State evolution

TPN state evolves either by time progression or by firing transitions

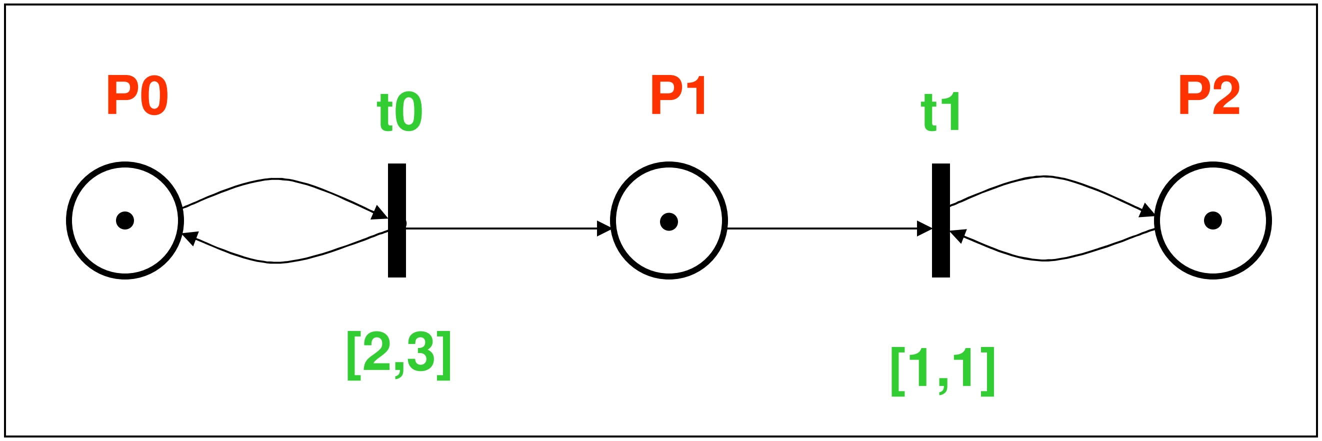




Hanifa Boucheneb EPM

State evolution

TPN state evolves either by time progression or by firing transitions



Upper bound reaches 0
➔ t1 must be fired

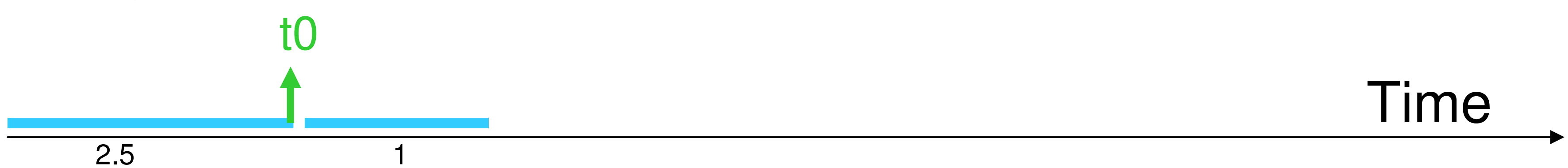
Interval State (M,I)

P0=1, P1=1, P2=1
I(t0) = [1 , 2]
I(t1) = [0 , 0]

Clock State (M,v)

P0=1, P1=1, P2=1
v(t0) = 1
v(t1) = 1

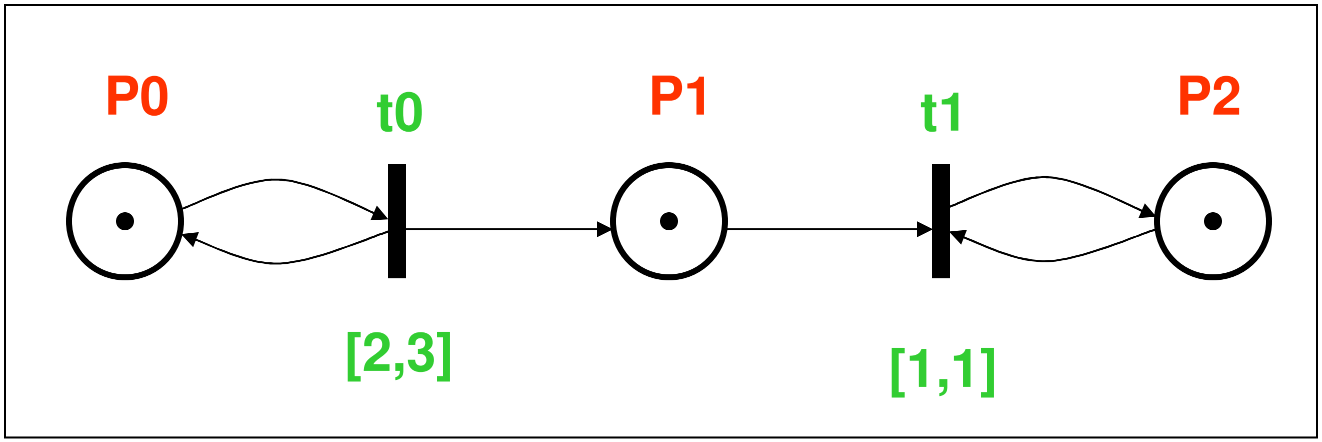
Clock reaches the upper bound
➔ t1 must be fired



Hanifa Boucheneb EPM

State evolution

TPN state evolves either by time progression or by firing transitions

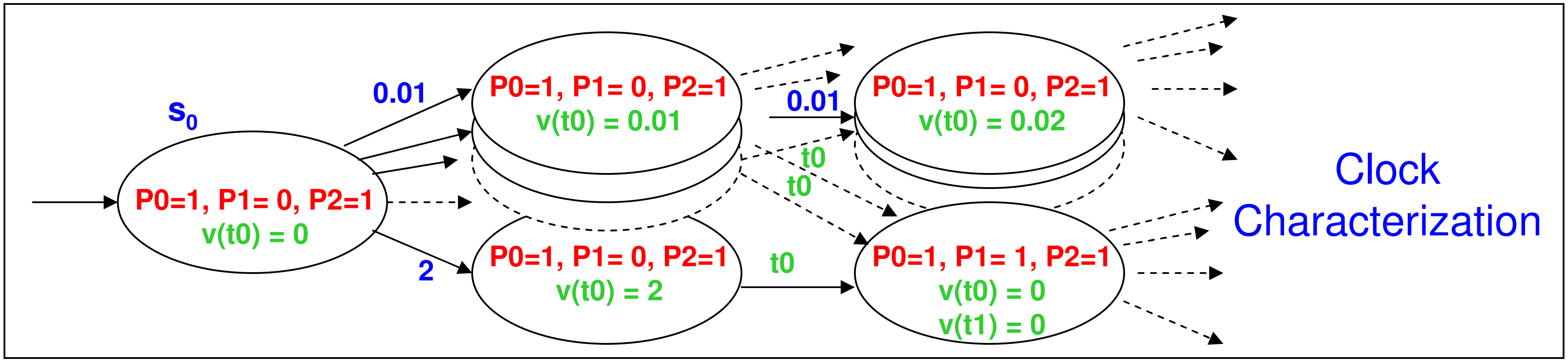


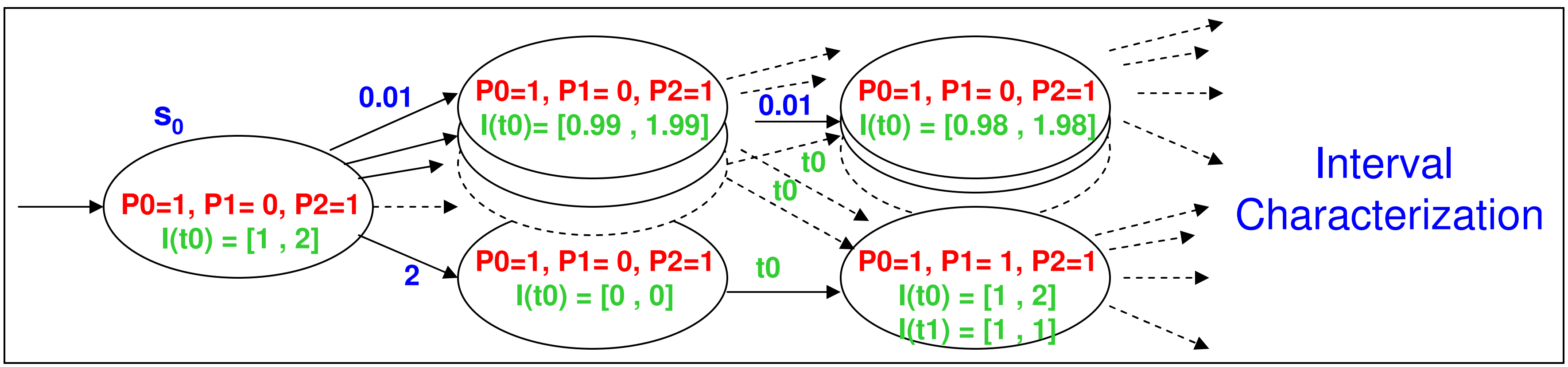
Interval State (M,I)	Clock State (M,v)
P0=1, P1=0, P2=1	P0=1, P1=0, P2=1
I(t0) = [1 , 2]	v(t0) = 1

Hanifa Boucheneb EPM

TPN state space

Transition system $(S, \rightarrow, s_0)$

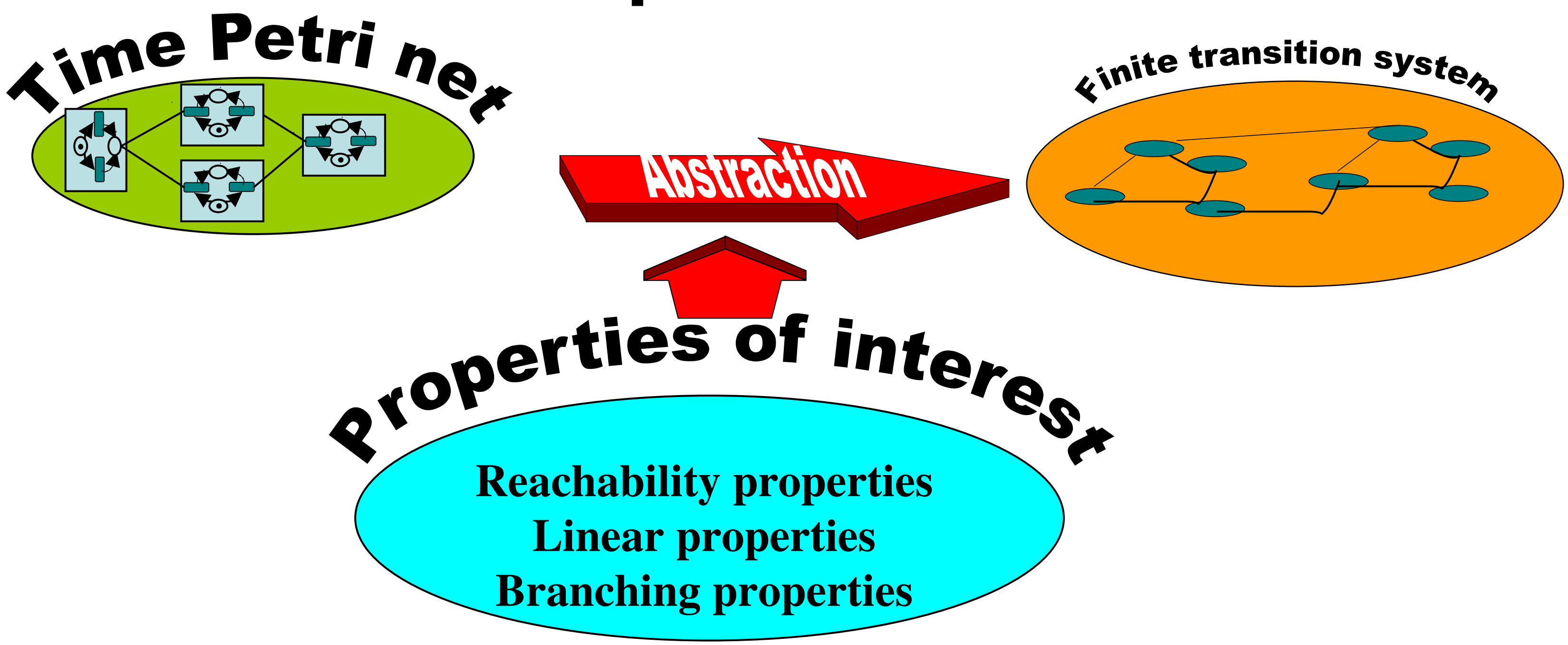




Infinite state space with infinite branching

How to abstract the TPN state space ?

TPN state space **abstractions**



Abstraction → Abstract irrelevant information while preserving properties of interest:

- Reachability: markings or states of the model.
- **Linear properties: firing sequences of the model.**
- Branching properties: execution trees of the model.

Challenge:

- **More coarser finite abstraction preserving properties of interest.**
- **Computed with minor resources (time and space).**

TPN state space abstractions

TPN state space abstractions in the literature preserving linear properties:

- **SCG [Berthomieu & Diaz 1991],**
- **GRG [Yoneda & Ryuba 1998],**
- **SSCG [Berthomieu & Vernadat 2003],**
- **ZBG [Gardey & Roux 2003],**
- **CSZG [Hadjidj & Boucheneb 2005].**
- **ASCGs**

→ may differ in:

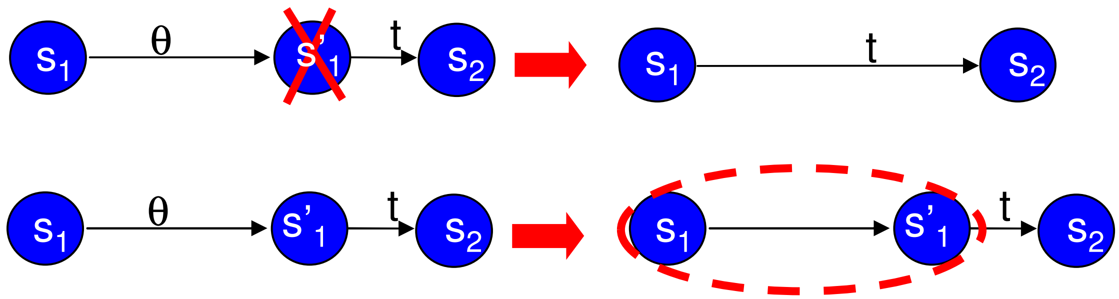
- **Characterization of states (interval state abstractions, clock state abstractions) ,**
- **Agglomeration criteria of states**
- **Preserved properties.**

Hanifa Boucheneb EPM

TPN state space **abstractions**

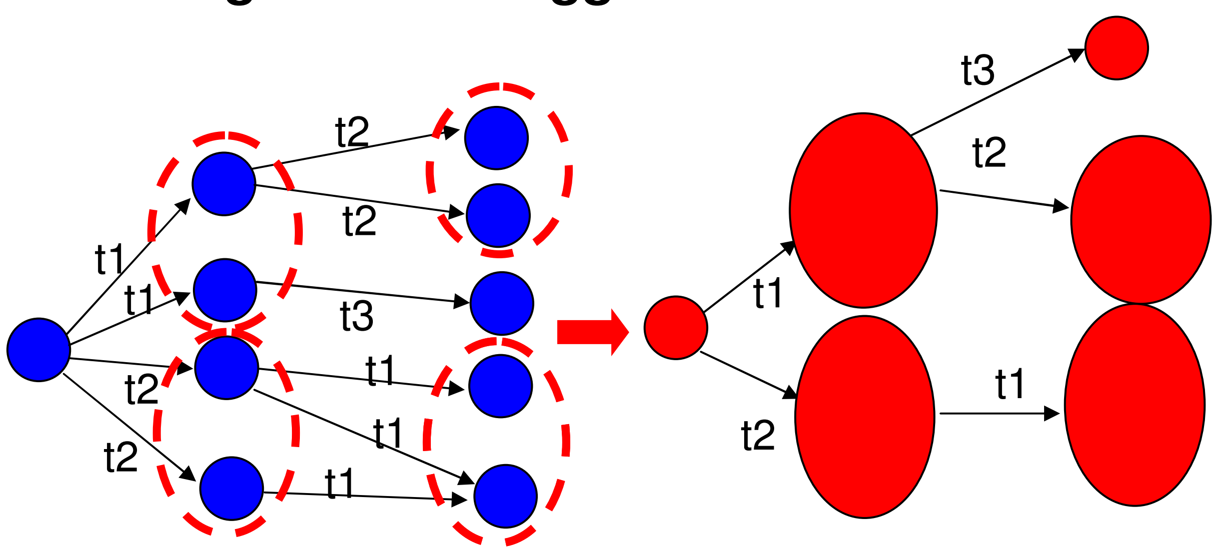
Three levels of abstraction:

1. **Time abstraction**



OR

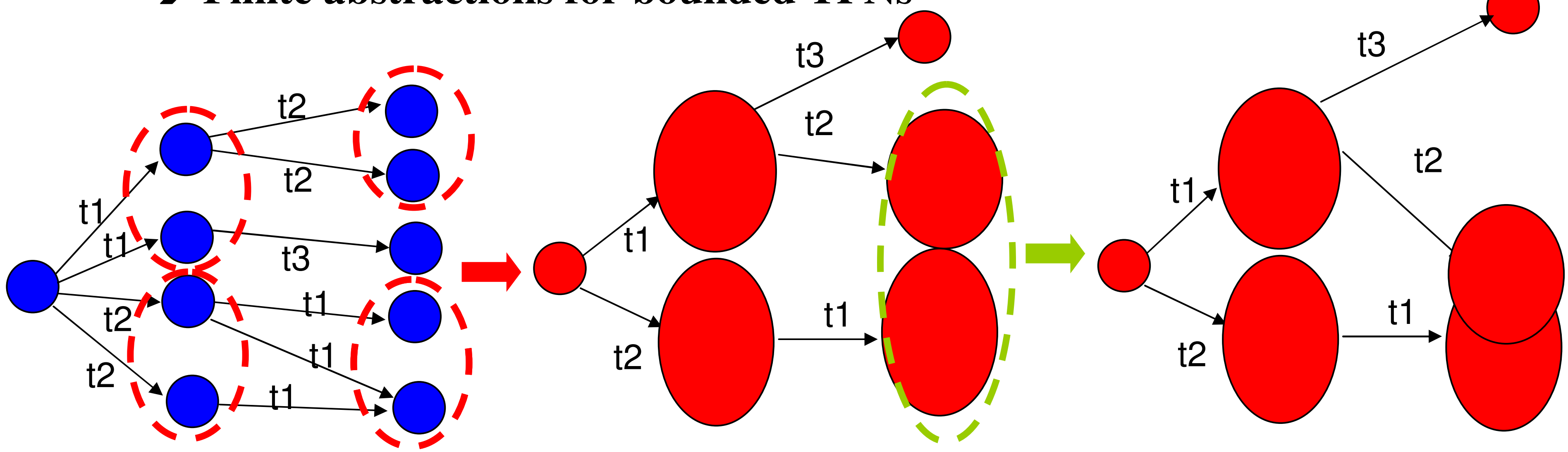
2. **States reachable by the same firing sequence independently of their firing times are agglomerated in the same node.**



TPN state space **abstractions**

3. **Agglomerated states are then considered modulo some relation of equivalence:**
   - **Firing domain of the SCG** [Berthomieu & Diaz 1991],
   - **k-approximation of the ZBG** [Gardey & Roux 2003],
   - **Approximation of the SSCG** [Berthomieu & Vernadat 2003],
   - **k-normalization of the CSZG** [Hadjidj & Boucheneb 2005],
   - **Relation of equivalence of the GRG** [Yoneda & Ryuba 1998].

➔ **Finite abstractions for bounded TPNs**



TPN state space abstractions

The SCG is the more interesting abstraction. Why?

- **The SCG is an interval state abstraction; all others are clock state abstractions.**

- **Abstractions based on clocks do not enjoy naturally the finiteness property for bounded TPNs with unbounded intervals. The finiteness is enforced using:**
  - k-approximation of the ZBG [Gardey & Roux 2003],
  - Approximation of the SSCG [Berthomieu & Vernadat 2003],
  - k-normalization of the CSZG [Hadjidj & Boucheneb 2005].

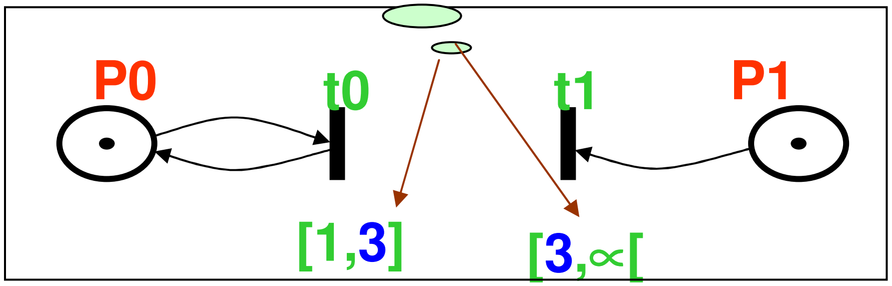
➔ **may involve some overhead computation.**

- **The interval characterization of states has a more abstracting power than the clock characterization, and allows to construct more compact abstractions.**

Hanifa Boucheneb EPM

TPN state space **abstractions**

The number of zones depends on the value of these bounds (3+1)



SCG state classes for t0⁺

$\alpha 0 = (P0{=}1, P1{=}1, \; 1 \le t0 \le 3 \wedge 3 \le t1 < \propto)$

t0 [1,3]

$\alpha 1 = (P0{=}1, P1{=}1, \; 1 \le t0 \le 3 \wedge 0 \le t1 < \propto)$

t0 [1,3]

Firing domain

ZBG zones for t0⁺

$z0 = (P0{=}1, P1{=}1, \; 0 \le t0 = t1 \le 3)$

t0 [1,3]

$z1 = (P0{=}1, P1{=}1, \; 0 \le t0 \le 3 \wedge 1 \le t1 \wedge 1 \le t1 - t0 \le \propto)$

t0 [1,3]

$z2 = (P0{=}1, P1{=}1, \; 0 \le t0 \le 3 \wedge 2 \le t1 \wedge 2 \le t1 - t0 \le \propto)$

t0 [1,3]

$z3 = (P0{=}1, P1{=}1, \; 0 \le t0 \le 3 \wedge 3 \le t1 \wedge 3 \le t1 - t0 \le \propto)$

t0 [1,3]

Clock domain

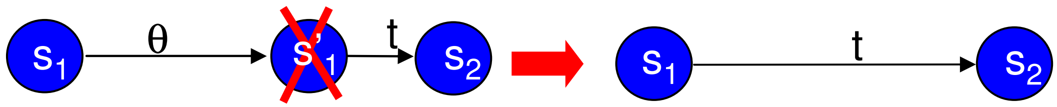
- Many ZBG zones may map to a single SCG state class.

The state class graph method

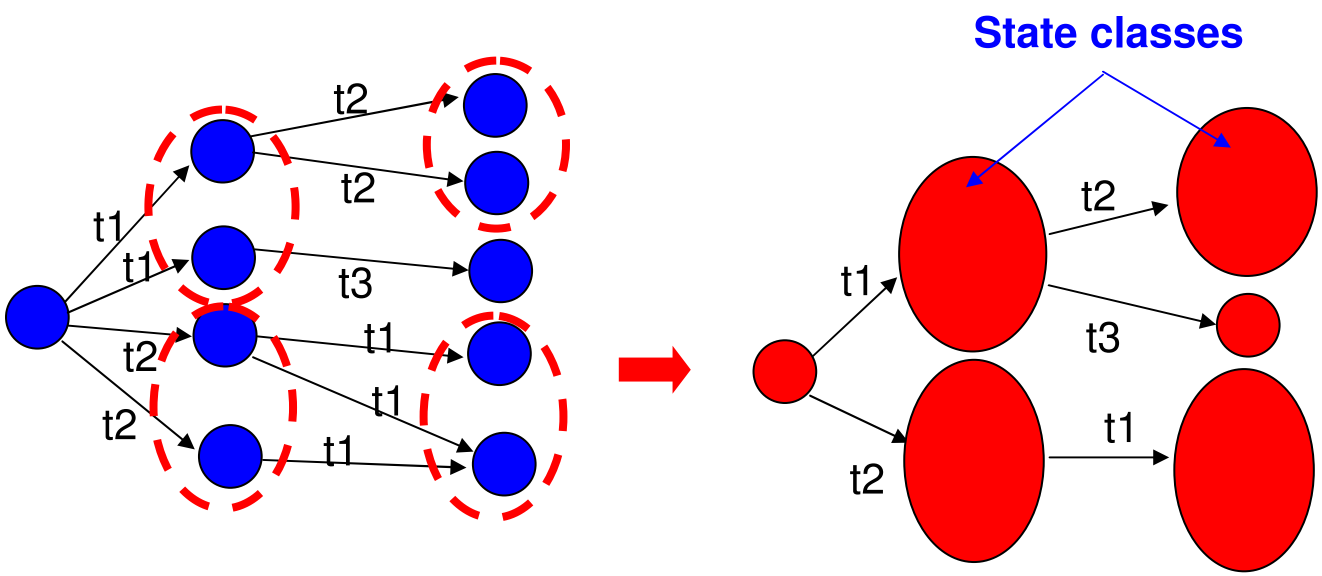
The state class graph method

1. State characterization: interval state (M,I)

2. Time abstraction:

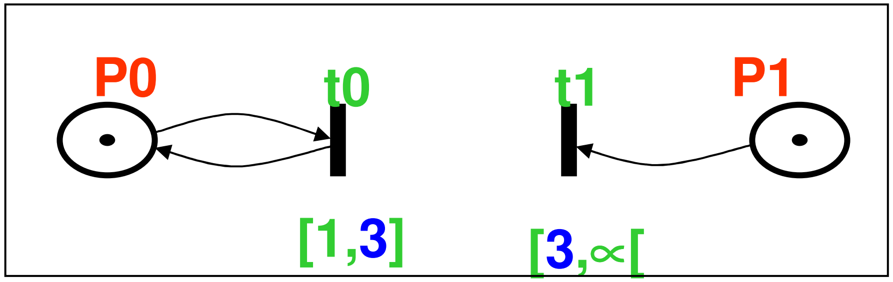


3. States reachable by the same firing sequence are agglomerated and considered modulo some relation of equivalence → **firing domain**



→ A state class = (marking, set of constraints which characterizes its states)

The state class graph method



- The initial state class: $\alpha 0 = ($ P0=1,P1=1, $1 \leq t0 \leq 3 \wedge 3 \leq t1 < \infty)$

- succ(α0,t0) $\neq \emptyset$ iff t0 is enabled and the following formula is consistent

$1 \leq t0 \leq 3 \wedge 3 \leq t1 < \infty \wedge t0 \leq t1$

- The firing of t0 leads to the state class $\alpha 1$ = (P0=1,P1=1, F1) where F1 is computed in four steps:

  1. **F1 ← $1 \leq t0 \leq 3 \wedge 3 \leq t1 < \infty \wedge t0 \leq t1$**
  2. Replace t1 with t1+t0: **$1 \leq t0 \leq 3 \wedge 3 \leq t1+t0 < \infty \wedge t0 \leq t1+t0$**
  3. Eliminate by substitution t0 and conflicting transitions: **$0 \leq t1 < \infty$**
  4. Add constraints of newly enabled transitions: **$0 \leq t1 < \infty \wedge 1 \leq t0 \leq 3$**

t1= new t1+t0

The state class graph method

- *F* is a conjunction of atomic constraints: $t_i - t_j \prec c$, $t_i \prec c$, $- t_j \prec c$,
  where $t_i, t_j \in T$, $c \in Q \cup \{\propto, -\propto\}$, $\prec \in \{=, \leq, \geq\}$
- Canonical form of F: $$\bigwedge_{x,x' \in En(M) \cup \{o\}} x - x' \leq Sup_F(x - x')$$

  where *o* represents the value zero and
  $Sup_F(x\text{-}x')$ is the least upper bound of ***x-x'*** in the domain of ***F***

- It can be represented by a Difference Bound Matrix (DBM) *D*:
  $$\forall x,x' \in En(M) \cup \{o\}, \quad D_{xx'} = Sup_F(x\text{-}x')$$
- *F* and all formulae equivalent to *F* have **the same canonical form.**
- Canonical form of *F* may be computed using:
  - The shortest path Floyd-Warshall's algorithm in $O(n^3)$, n being the number of variables in F, or
  - The algorithm, proposed in [Boucheneb & Mullins 2003], in $O(n^2)$.

The state class graph method

Algorithm proposed in [Boucheneb & Mullins 2003]

Let α=*(M,D)* be a state class in canonical form and t_f a transition.

- t_f *is firable from* (M, D) *(i.e.:* $Succ(\alpha, t_f) \neq \emptyset$*) iff* $t_f \in En(M)$ *and* $Min_{t \in En(M)} D_{t\, t_f} = 0$.

- *If* t_f *is firable from* (M, D)*, its firing leads to the state class* $Succ(\alpha, t_f) = (M', D')$ *computed as follows:*

$\forall p \in P, M'(p) = M(p) - Pre(p, t_f) + Post(p, t_f)$;

$\forall t \in En(M')$,

$$D'_{t\, o} = \begin{cases} tmax(t) & \text{if } t \in New(M', t_f); \\ D_{t\, t_f} & \text{otherwise} \end{cases}$$

$$D'_{o\, t} = \begin{cases} -tmin(t) & \text{if } t \in New(M', t_f); \\ Min_{u \in En(M)} D_{u\, t} & \text{otherwise} \end{cases}$$

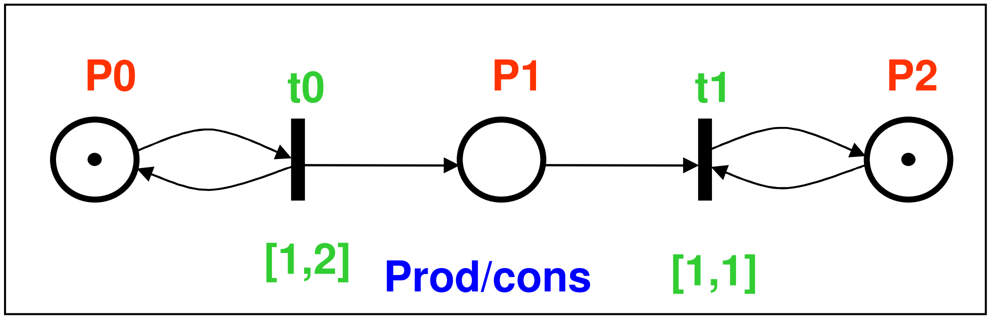
$\forall (t, t') \in (En(M'))^2$,

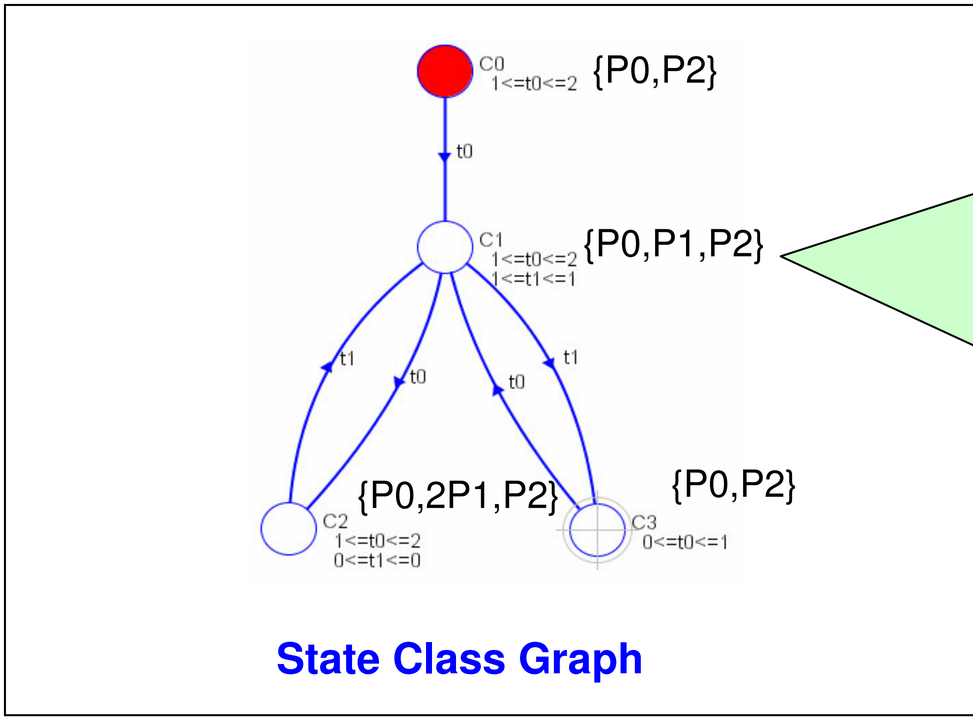
$$D'_{t\, t'} = \begin{cases} 0 & \text{if t is t}' \\ D'_{t\, o} + D'_{o\, t'} & \text{if t or t}' \in New(M', t_f); \\ Min(D_{t\, t'}, D'_{t\, o} + D'_{o\, t'}) & \text{otherwise} \end{cases}$$

Time complexity O(n)

Time complexity O(n²)

An example





Reachable markings:

1. P0 =1, P2=1
2. P0=1, P1=1, P2=1
3. P0=1, P1=2, P2=1

Firing sequences: t0; { t0; t1, t1;t0} $^{\infty}$

Path / cycle times of firing sequences [Boucheneb & Mullins 2003]:

From C0: t0; t1 → [2,3]

From C1: t0; t1 → [1,2]

→ SCG is finite iff the TPN is bounded.

→ SCG preserves markings and traces of the TPN → Linear properties

Contracting the state class graph

Hanifa Boucheneb EPM

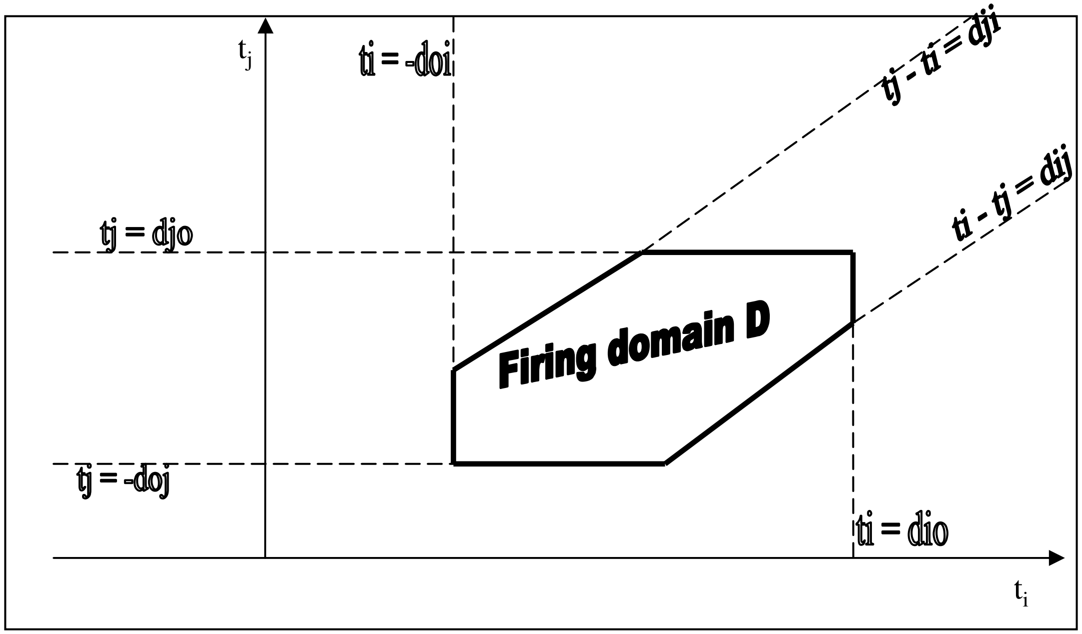
Contracting the state class graph

- *Define a relation of equivalence over state classes such that equivalent classes have the same firing sequences.*

- *Construct a quotient graph w.r.t. the relation of equivalence.*

→ *Over-approximation of state classes [Boucheneb & Rakkay; ACSD 2007]:*

D: $- doi \le ti \le dio$
$- doj \le tj \le djo$
$- dji \le ti - tj \le dij$



Over-approximation of state classes

- *The idea of this operation comes from:*

  1- The line and column o of each DBM D are not necessary to compute successor state classes of (M,D) (they can be eliminated)

  ➔ a gain in space

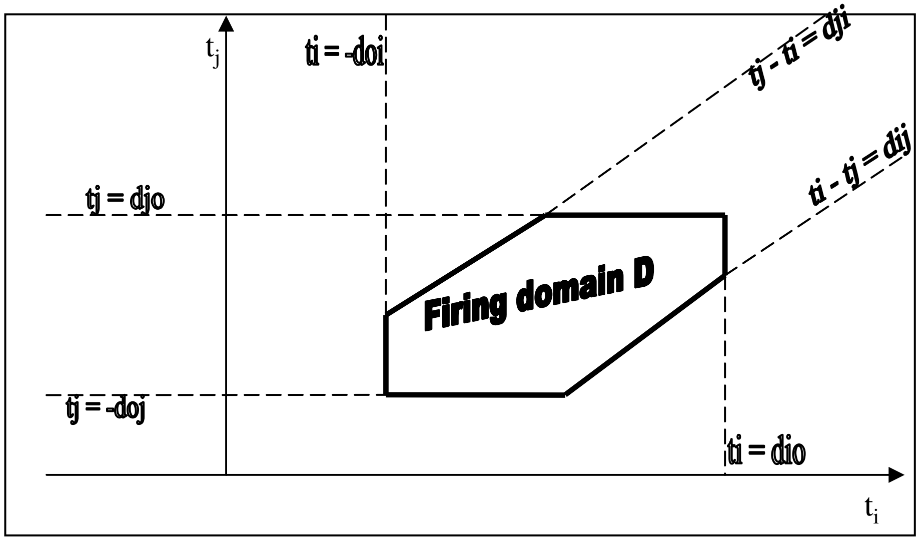
  2- The firing of any transition t_f from a state class α=*(M,D),* will disable all transitions conflicting with t_f.

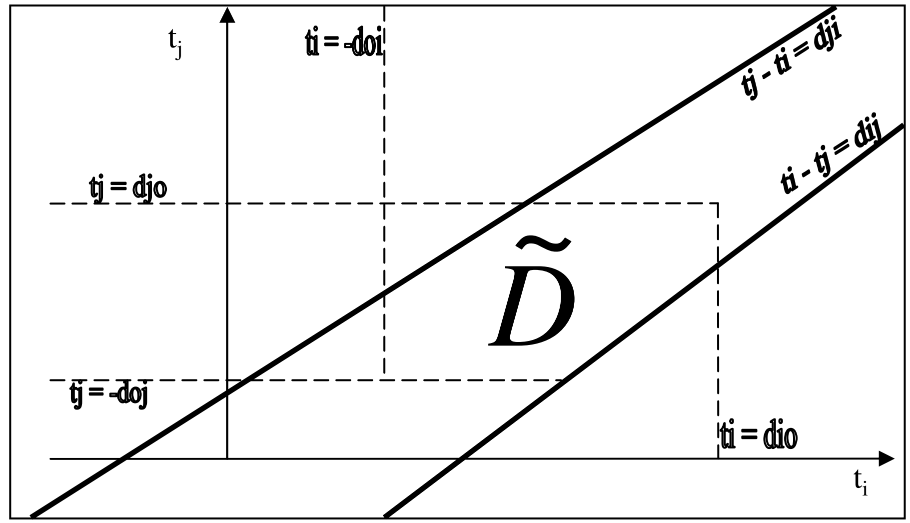
  ➔ There is no need to compute, for the s*uccessor of* α *by* t_f distances between t_f and its conflicting transitions,

  ➔ but we need to know whether t_f *is firable or not before all enabled transitions (including those in conflict with* t_f*).*

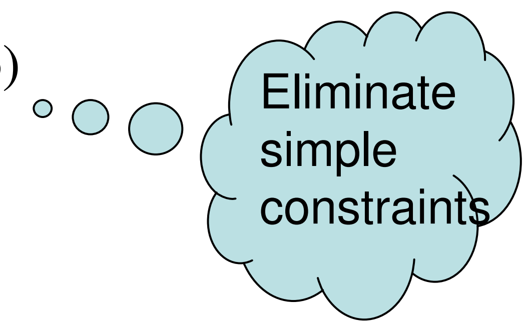
$\Rightarrow$ *State classes with lesser constraints*

Over-approximation of state classes





$$\tilde{d}ij = \begin{cases} \infty & \text{if } (i = o \text{ and } j \neq o) \text{ or } (j = o \text{ and } i \neq o) \\ 0 & \text{if } (i, j \neq o) \text{ and } dij \geq 0 \\ & \text{and (ti and tj in conflict)} \\ dij & \text{otherwise} \end{cases}$$



CSCG is the quotient graph of SCG w.r.t $\approx$:

$$\forall \alpha_1, \alpha_2 \in C,\ \alpha_1 \approx \alpha_2 \text{ iff } \tilde{\alpha}_1 = \tilde{\alpha}_2$$

Lemma: $\approx$ is a bisimulation over the SCG

Contracting the state class graph

Proposition 2 *Let $\alpha = (M, D)$ be a state class, $\widetilde{\alpha} = (M, \widetilde{D})$ its equivalence class and t_f a transition.*
$Succ(\alpha, t_f) \neq \emptyset$ iff $Min_{u \in En(M)} \widetilde{D}_{u\, t_f} = 0$.
If $Succ(\alpha, t_f) \neq \emptyset$ and $Succ(\alpha, t_f) = (M', D')$ then the equivalence class $(M', \widetilde{D}')$ of (M', D') can be computed using $\widetilde{D}$ as follows:

$\forall (t, t') \in (En(M'))^2 - conf(M'), \quad \widetilde{D}'_{t\, t'} =$

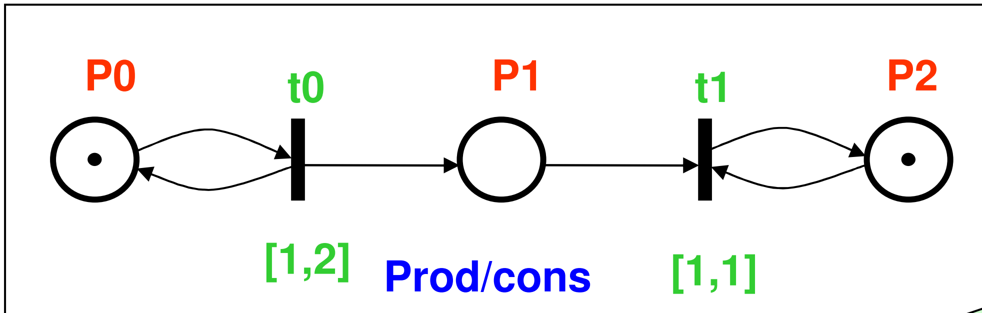
$$\begin{cases} 0 & \text{if t is t}' \\ tmax(t) - tmin(t') & \text{if t, t}' \in \text{New}(\text{M}', \text{t}_\text{f}); \\ tmax(t) + Min_{u \in En(M)} \widetilde{D}_{u\, t'} & \text{if t} \in \text{New}(\text{M}', \text{t}_\text{f}); \\ \widetilde{D}_{t\, t_f} - tmin(t') & \text{if t}' \in \text{New}(\text{M}', \text{t}_\text{f}); \\ Min(\widetilde{D}_{t\, t'}, \widetilde{D}_{t\, t_f} + Min_{u \in En(M)} \widetilde{D}_{u\, t'}) & \text{otherwise} \end{cases}$$

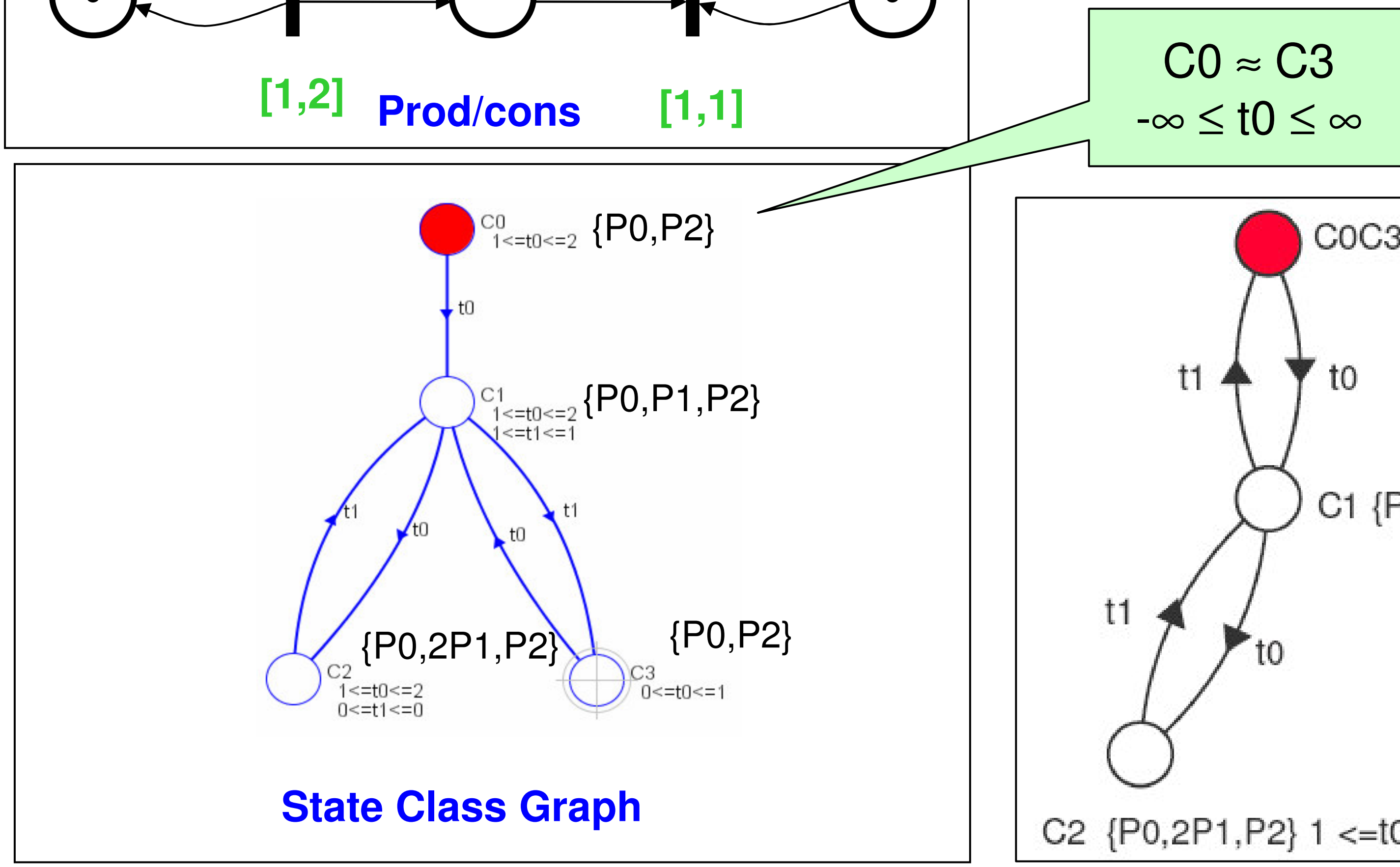
$\forall (t, t') \in conf(M'), \quad \widetilde{D}'_{t\, t'} =$

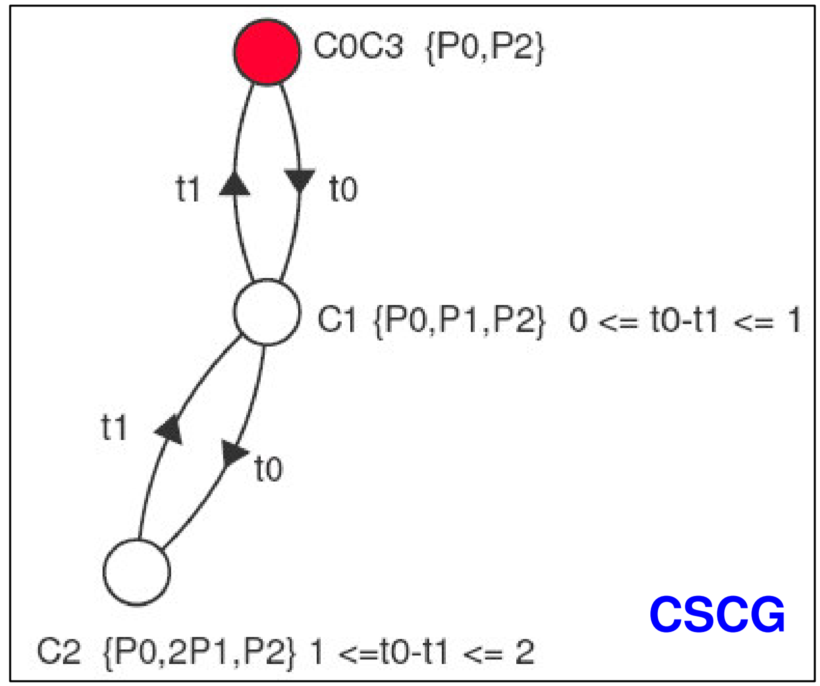
$$\begin{cases} 0 & \text{if t is t}' \\ Min(0, tmax(t) - tmin(t')) & \text{if t, t}' \in \text{New}(\text{M}', \text{t}_\text{f}); \\ Min(0, tmax(t) + Min_{u \in En(M)} \widetilde{D}_{u\, t'}) & \text{if t} \in \text{New}(\text{M}', \text{t}_\text{f}); \\ Min(0, \widetilde{D}_{t\, t_f} - tmin(t')) & \text{if t}' \in \text{New}(\text{M}', \text{t}_\text{f}); \\ Min(0, \widetilde{D}_{t\, t'}, \widetilde{D}_{t\, t_f} + Min_{u \in En(M)} \widetilde{D}_{u\, t'}) & \text{otherwise} \end{cases}$$

Classes of equivalence can be computed in $O(n^2)$ directly from equivalence classes of their predecessors

An example (no conflicting transitions)





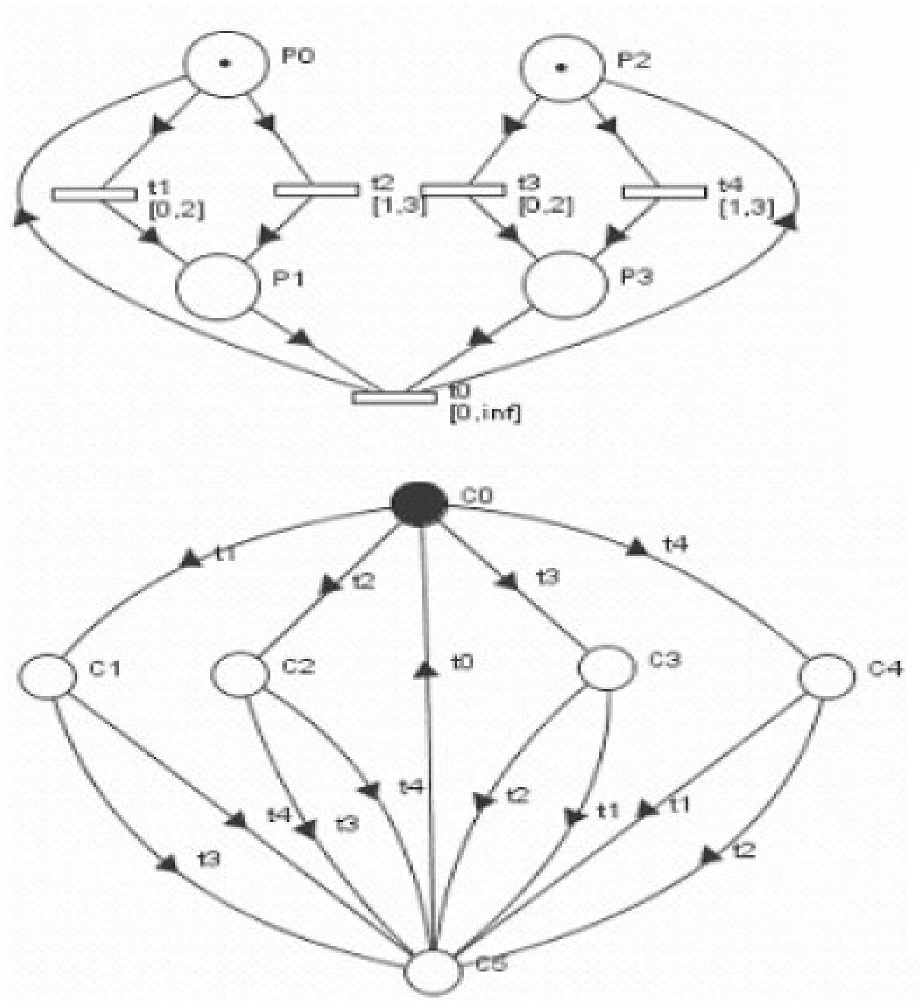


→ CSCG is smaller than CSG.

→ CSCG preserves markings and traces of the TPN → Linear properties

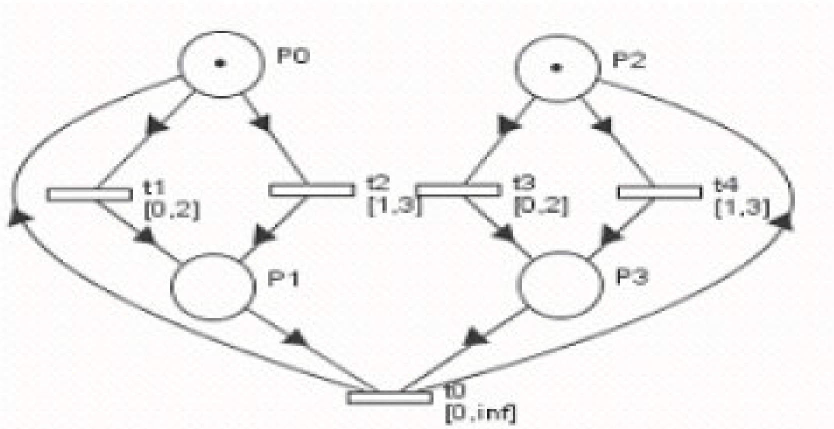
Another example with conflicting transitions

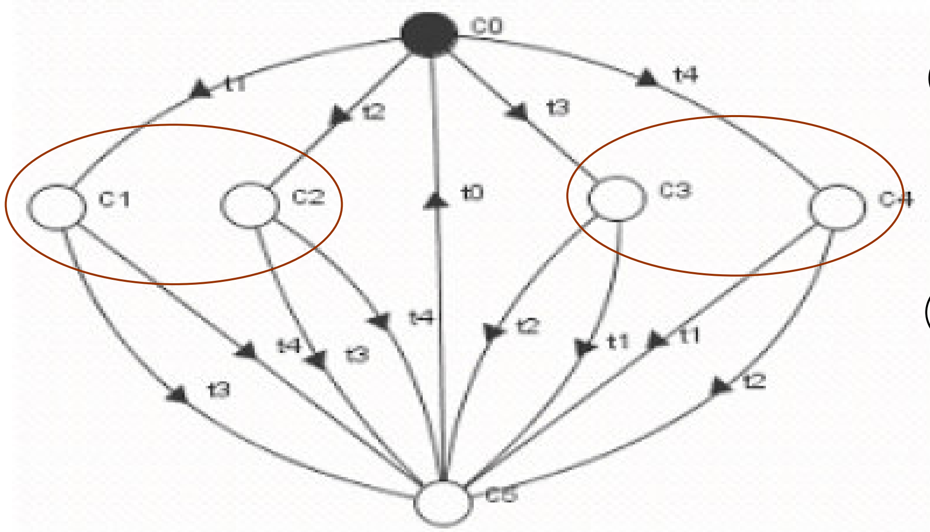
Eliminating simple constraints



- $C0 : P0 + P2$ — ~~$0 \le t1 \le 2$~~, ~~$1 \le t2 \le 3$~~, ~~$0 \le t3 \le 2$~~, ~~$1 \le t4 \le 3$~~, $-3 \le t1 - t2 \le 1$, $-2 \le t1 - t3 \le 2$, $-3 \le t1 - t4 \le 1$, $-1 \le t2 - t3 \le 3$, $-2 \le t2 - t4 \le 2$, $-3 \le t3 - t4 \le 1$
- $C1 : P1 + P2$ — ~~$0 \le t3 \le 2$~~, ~~$0 \le t4 \le 3$~~, $-3 \le t3 - t4 \le 1$
- $C2 : P1 + P2$ — ~~$0 \le t3 \le 1$~~, ~~$0 \le t4 \le 2$~~, $-2 \le t3 - t4 \le 1$
- $C3 : P0 + P3$ — ~~$0 \le t1 \le 2$~~, ~~$0 \le t2 \le 3$~~, $-3 \le t1 - t2 \le 1$
- $C4 : P0 + P3$ — ~~$0 \le t1 \le 1$~~, ~~$0 \le t2 \le 2$~~, $-2 \le t1 - t2 \le 1$
- $C5 : P1 + P3$ — ~~$0 \le t0 \le \infty$~~

Another example with conflicting transitions



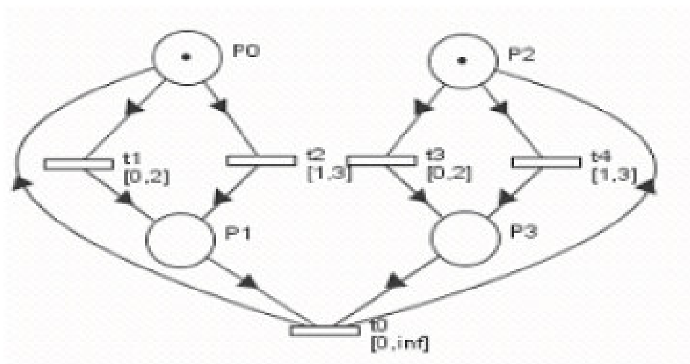


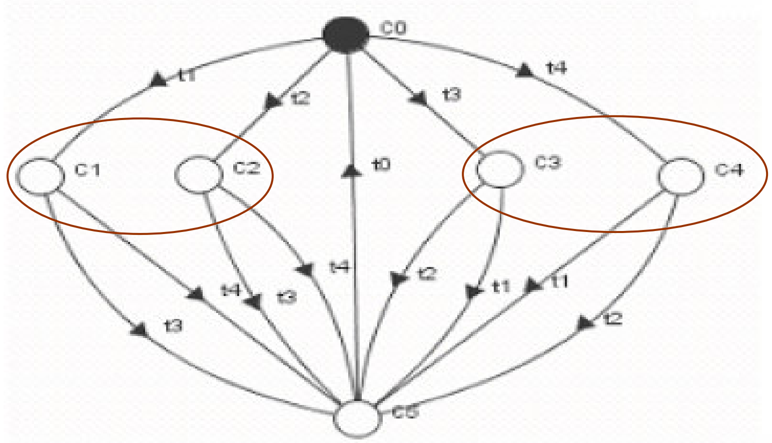
t3, t4 are in conflict

$$
\begin{cases}
C0: P0+P2 & 0 \le t1 \le 2 & 1 \le t2 \le 3 \\
 & 0 \le t3 \le 2 & 1 \le t4 \le 3 \\
 & -3 \le t1 - t2 \le 1 & -2 \le t1 - t3 \le 2 \\
 & -3 \le t1 - t4 \le 1 & -1 \le t2 - t3 \le 3 \\
 & -2 \le t2 - t4 \le 2 & -3 \le t3 - t4 \le 1 \\
C1: P1+P2 & 0 \le t3 \le 2 & 0 \le t4 \le 3 \\
 & -3 \le t3 - t4 \le 1 & \\
C2: P1+P2 & 0 \le t3 \le 1 & 0 \le t4 \le 2 \\
 & -2 \le t3 - t4 \le 1 & \\
C3: P0+P3 & 0 \le t1 \le 2 & 0 \le t2 \le 3 \\
 & -3 \le t1 - t2 \le 1 & \\
C4: P0+P3 & 0 \le t1 \le 1 & 0 \le t2 \le 2 \\
 & -2 \le t1 - t2 \le 1 & \\
C5: P1+P3 & 0 \le t0 \le \infty &
\end{cases}
$$

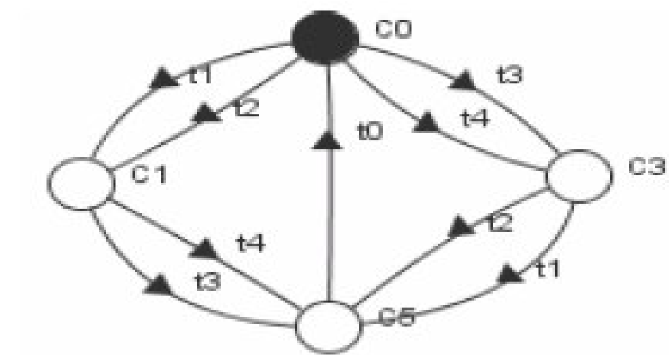
t1, t2 are in conflict

Another example with conflicting transitions





SCG



CSCG

$$
\begin{cases}
C0 : P0 + P2 & 0 \leq t1 - t2 \leq 0 \quad -2 \leq t1 - t3 \leq 2 \\
 & -3 \leq t1 - t4 \leq 1 \quad -1 \leq t2 - t3 \leq 3 \\
 & -2 \leq t2 - t4 \leq 2 \quad 0 \leq t3 - t4 \leq 0 \\
C1 : P1 + P2 & 0 \leq t3 - t4 \leq 0 \\
C3 : P0 + P3 & 0 \leq t1 - t2 \leq 0 \\
C5 : P1 + P3 &
\end{cases}
$$

Contracting the state class graph

Experimental results

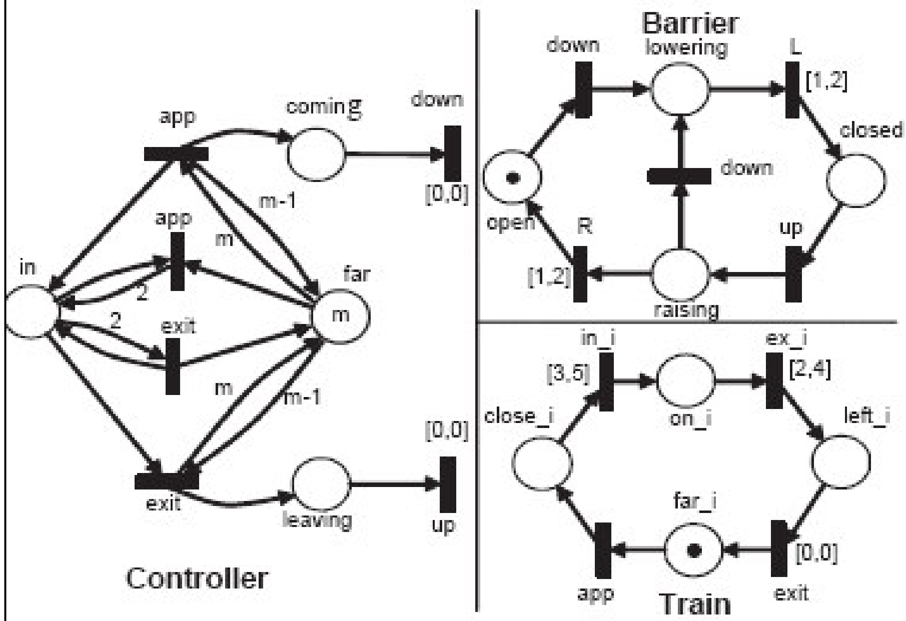
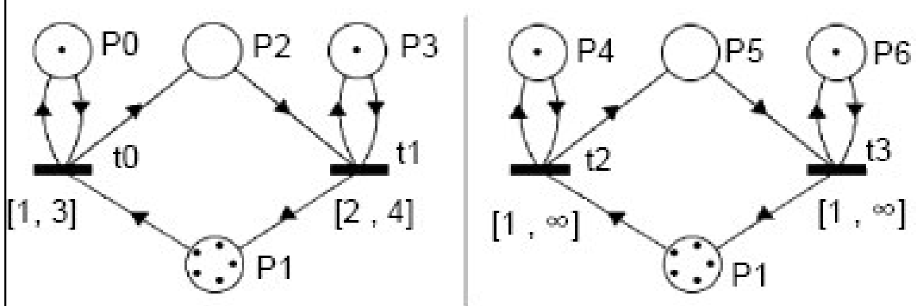


Figure 2. The level crossing models



a) ProdCons1

b) ProdCons2

Figure 3. Producer consumer model

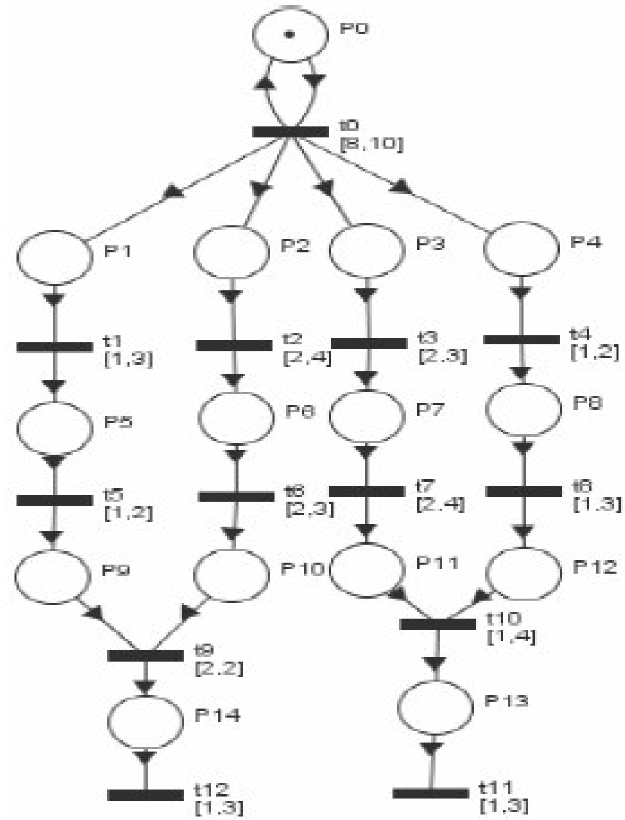


Figure 4. A TPN model of some periodical tasks

Hanifa Boucheneb EPM

Contracting the state class graph

Experimental results

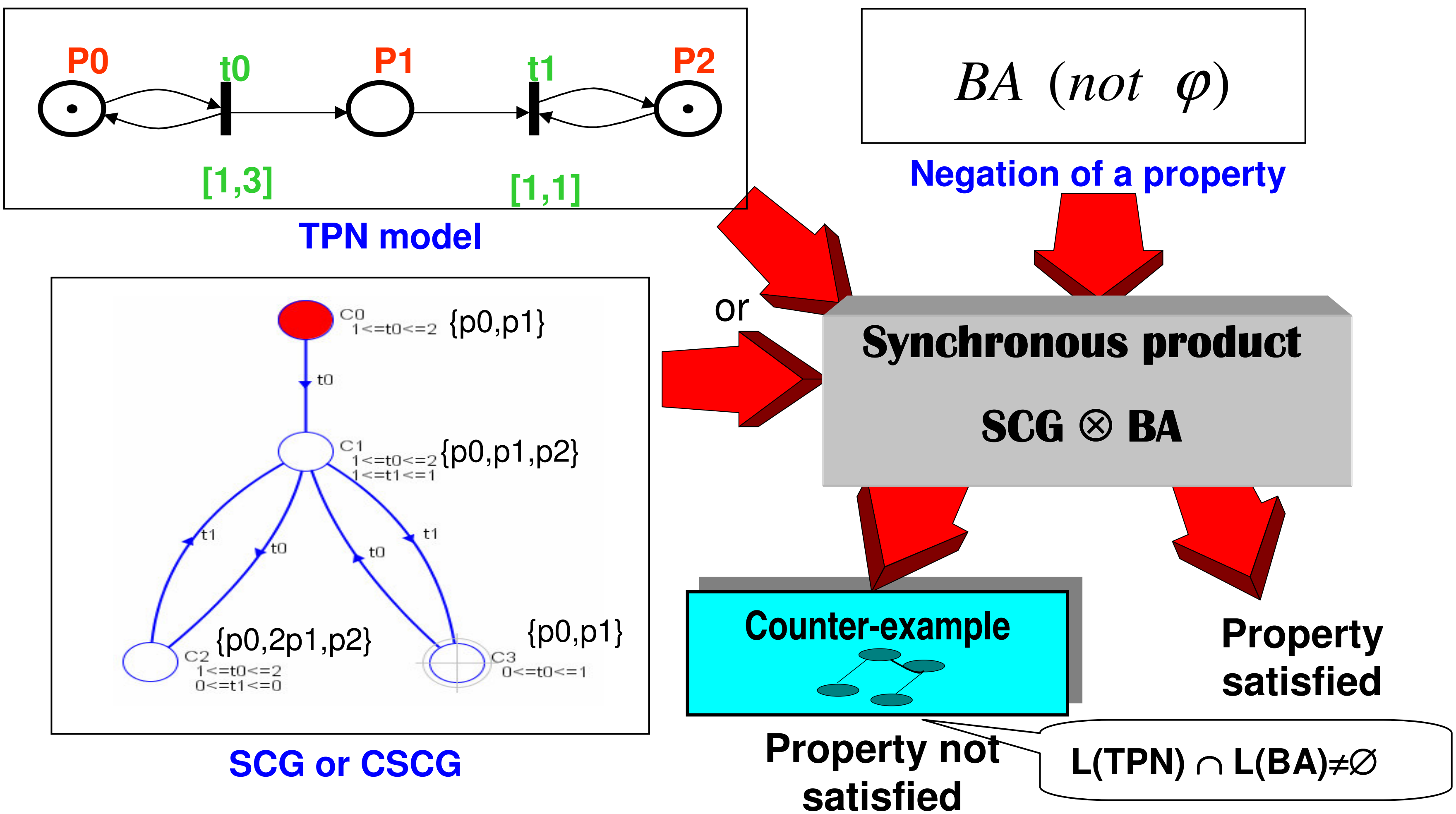
TPN	ZBG	SCG	*CSCG*	*Ratio*1
Fig.2(m=2)	147	123	113	1.09
Arcs	266	218	198	1.10
CPU(s)	0	0	0	-
Fig.2(m=3)	6299	3101	2816	1.10
Arcs	16565	7754	6941	1.12
CPU(s)	0.22	0.04	0.02	2
Fig.2(m=4)	?	134501	122289	1.10
Arcs		436896	391240	1.11
CPU(s)		4.47	2.41	1.85
Fig.3(n=2)	2941	748	519	1.44
Arcs	9952	2460	1678	1.47
CPU(s)	0.02	0.01	0	-
Fig.3(n=3)	100060	4604	2834	1.62
Arcs	485732	21891	13208	1.66
CPU(s)	20.57	0.09	0.04	2.25
Fig.3(n=4)	?	14086	8159	1.73
Arcs		83375	47592	1.75
CPU(s)		1.41	0.59	2.39
Fig.3(n=5)	?	31657	17643	1.79
Arcs		217423	120804	1.80
CPU(s)		8.77	3.22	2.72
Fig.3(n=6)	?	77208	37876	2.04
Arcs		624158	294363	2.12
CPU(s)		53.47	16.37	3.27
Fig.4	24015	18543	12910	1.44
Arcs	86621	65403	45970	1.42
CPU(s)	5.75	3.27	1.62	2.02

Ratio1 = SCG / CSCG

Gain in time and size grows with the size of the model

Model checking of SCG/CSCG using Büchi Automata method

Büchi Automata (BA) method



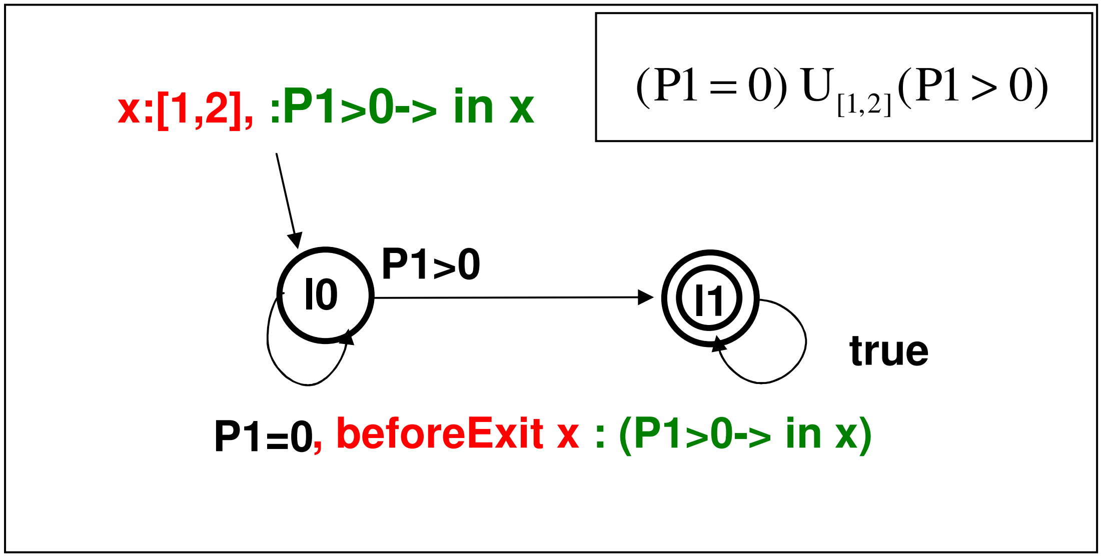
Interval Timed Büchi Automata (ITBA)

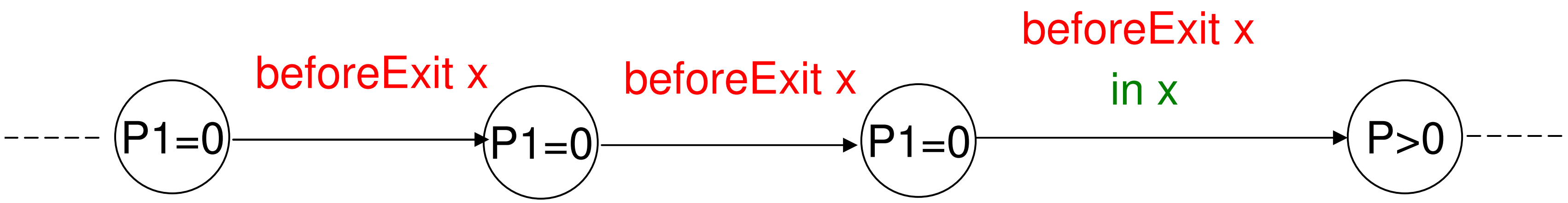
Hanifa Boucheneb EPM

Interval Timed Büchi Automata

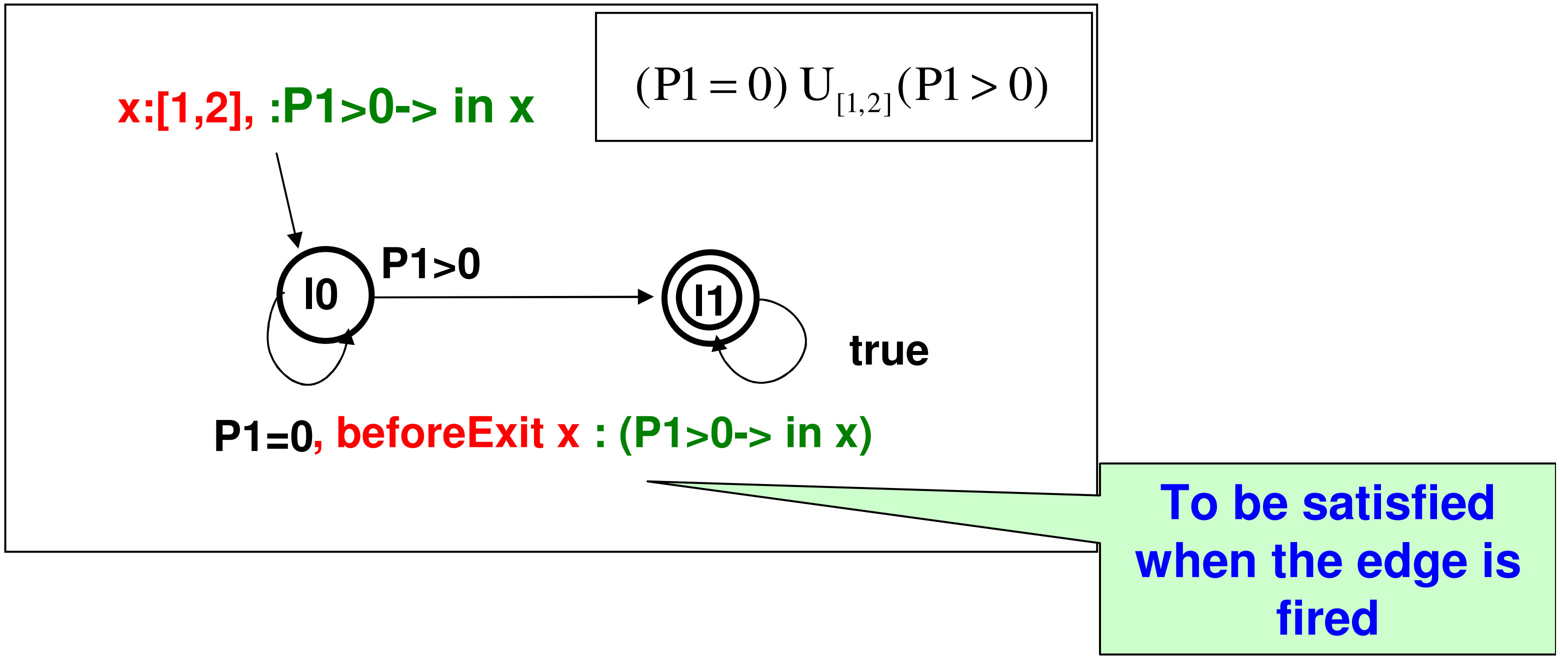
BA + **{interval delays}** **+**
{guards on delays} **+**
{**(re)setting /deactivation of delays**}

Guards on delays: **{** **before x**,
in x,
beforeExit x,
after x,
true}





Interval Timed Büchi Automata



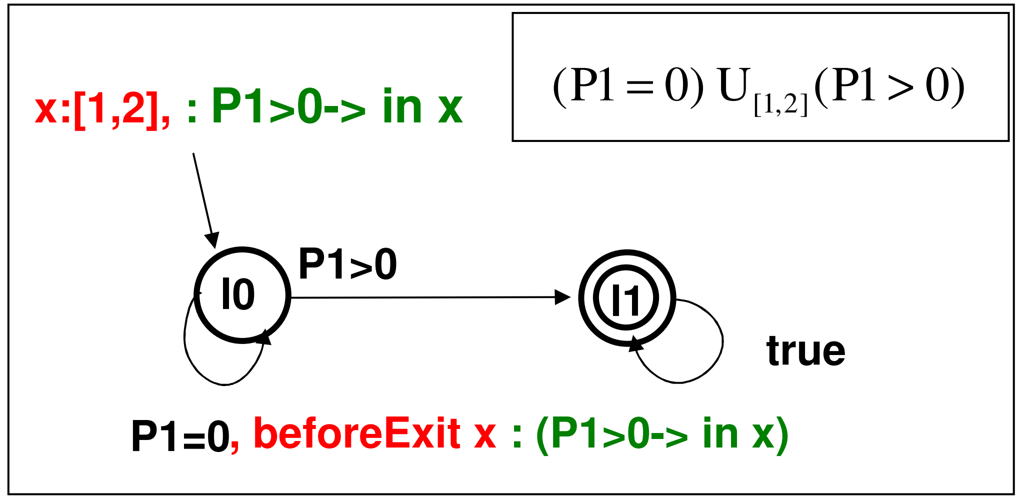
- With each delay x is associated a dynamic interval I(x).
- I(x) can be set to an interval at the beginning or when an edge is fired.
- When I(x) is set to an interval, its bounds decrease with time until it is deactivated.

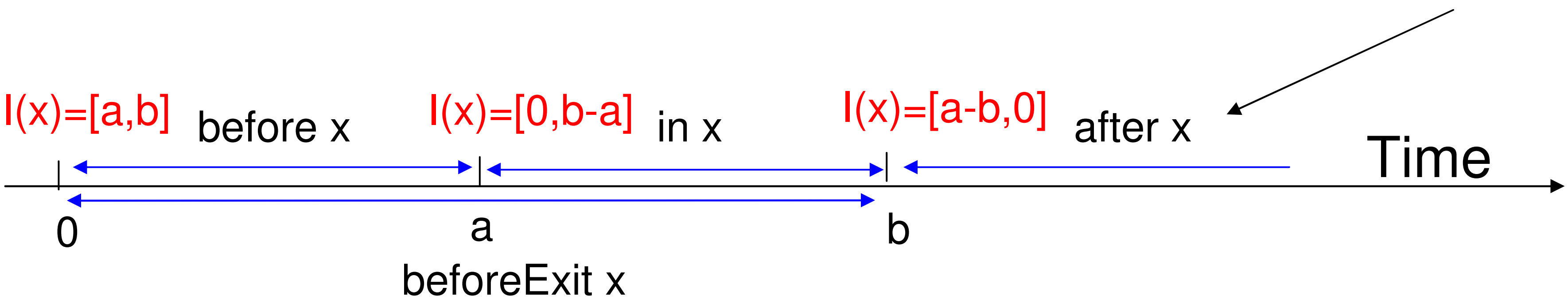
State: (l_i, A, I)

Hanifa Boucheneb EPM

Interval Timed Büchi Automata

State: (l_j, A, I)

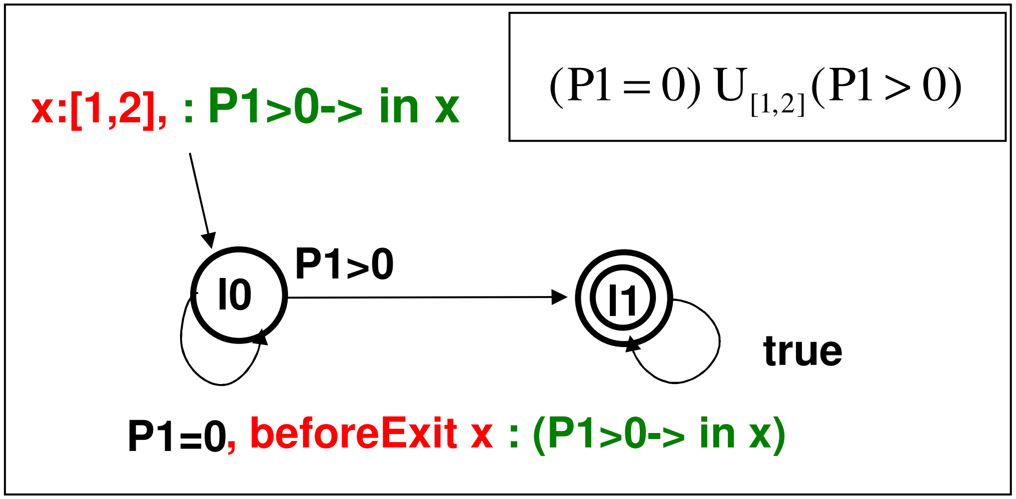




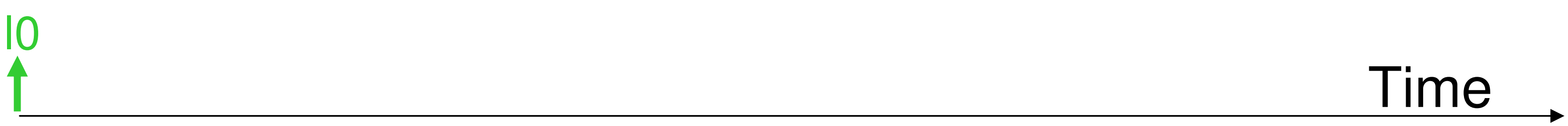
before x	in x	beforeExit x	after x
$\downarrow I(x) > 0$	$\downarrow I(x) \leq 0,\ \uparrow I(x) \geq 0$	$\uparrow I(x) \geq 0$	$\uparrow I(x) < 0$

Hanifa Boucheneb EPM

Interval Timed Büchi Automata



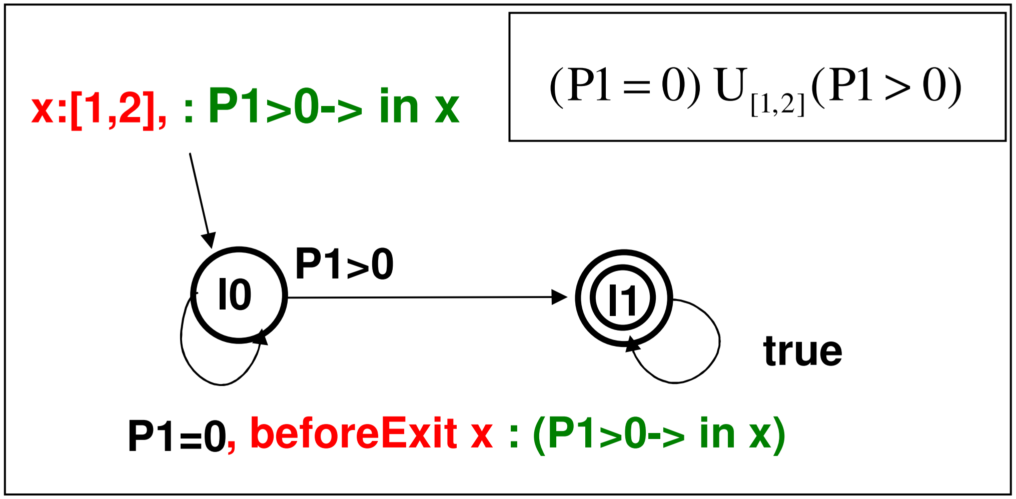
State q = (l0, A, I), A={x}, I(x)=[1,2]



State q = (l0, A, I), A={x}, I(x)=[1,2]

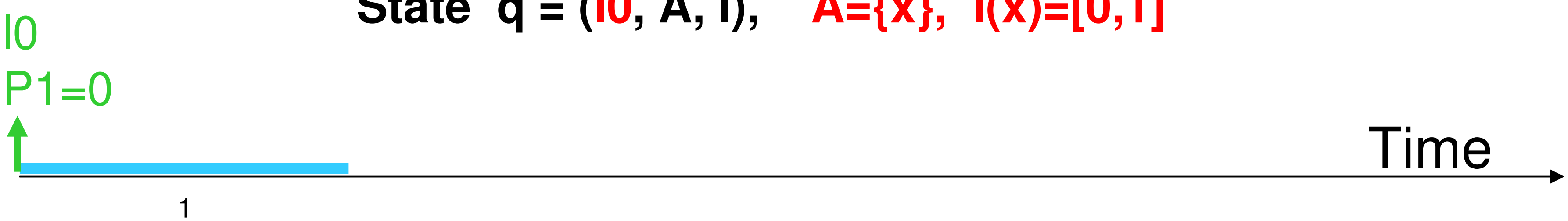
Hanifa Boucheneb EPM

Interval Timed Büchi Automata

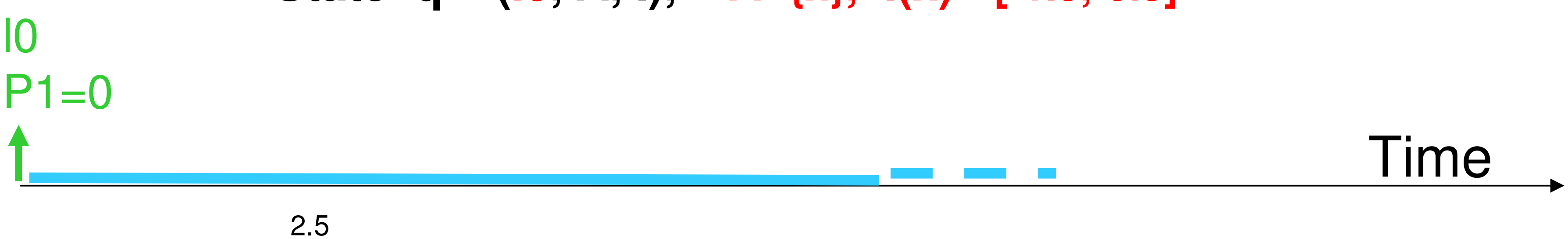


bounds of intervals decrease with time

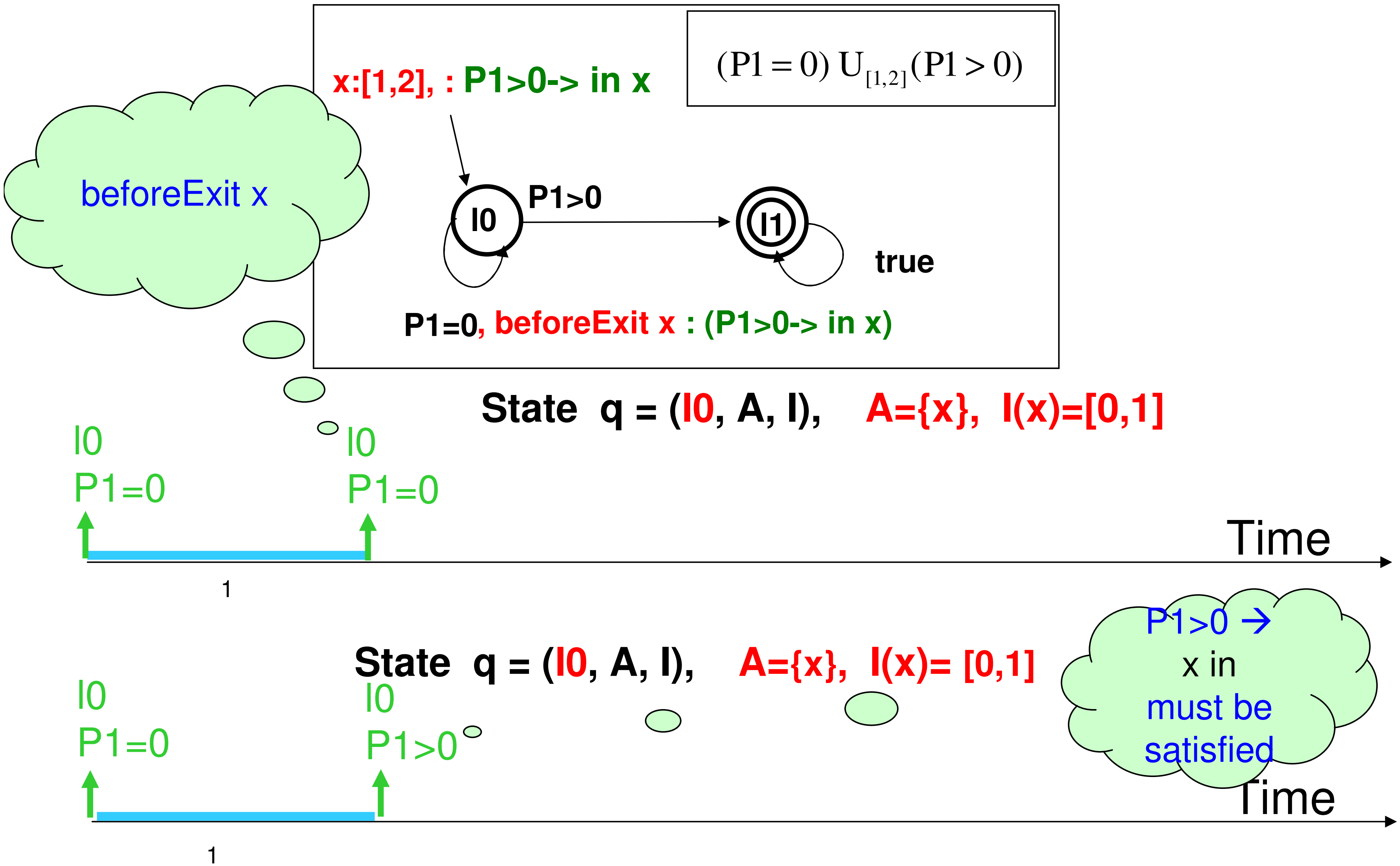
State q = (l0, A, I), A={x}, I(x)=[0,1]



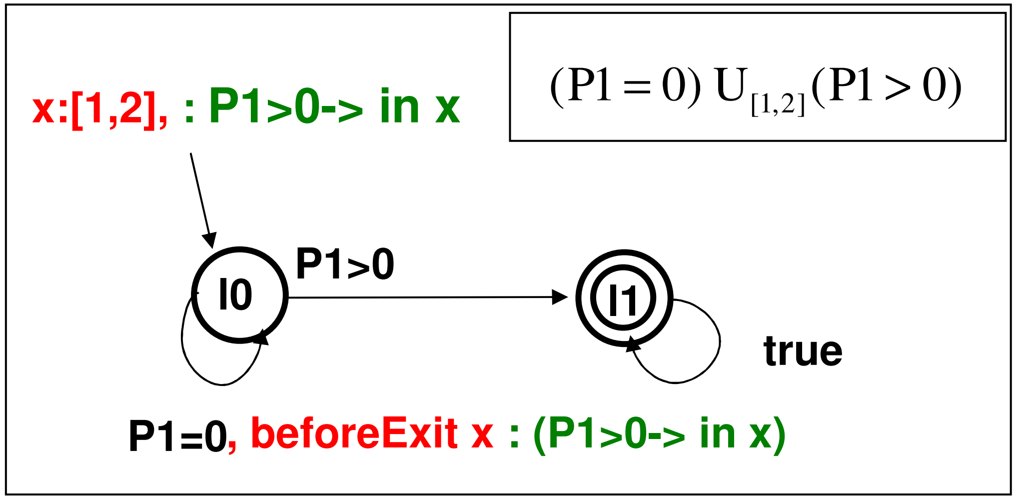
State q = (l0, A, I), A={x}, I(x)= [-1.5,-0.5]



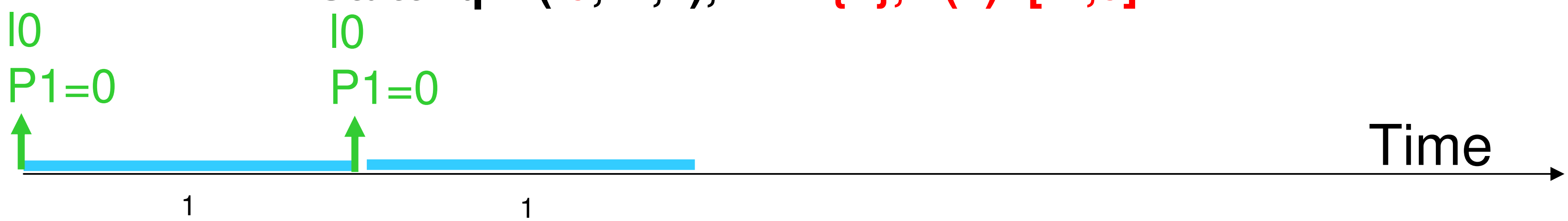
Interval Timed Büchi Automata



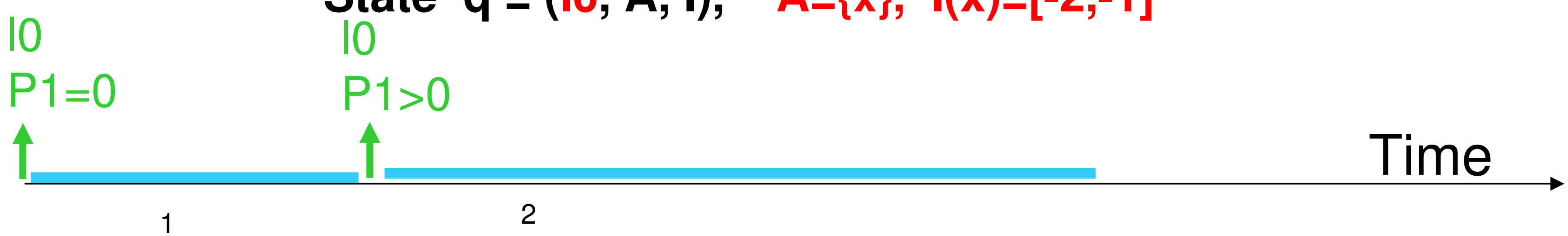
Interval Timed Büchi Automata



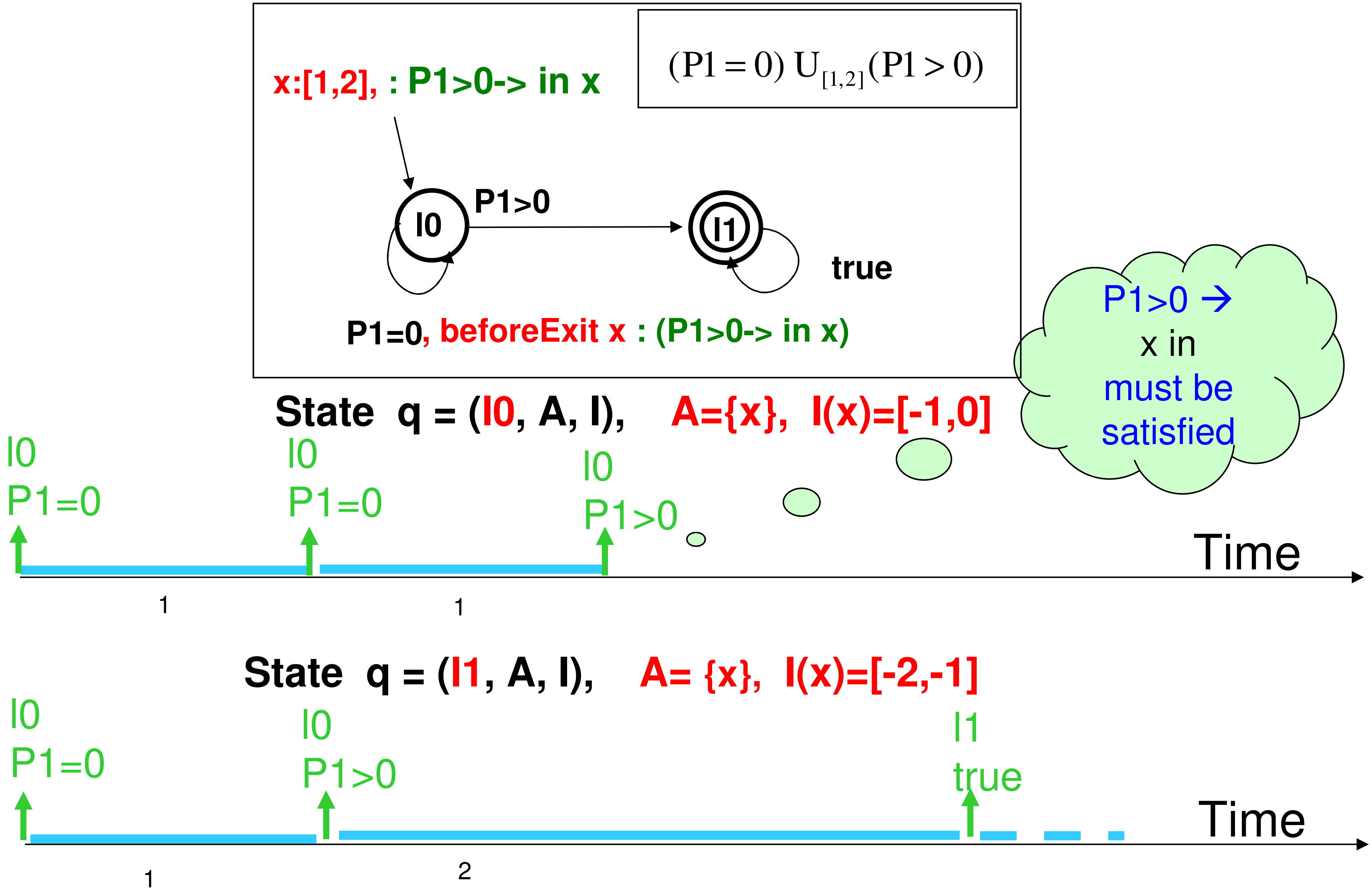
State q = (l0, A, I), A={x}, I(x)=[-1,0]



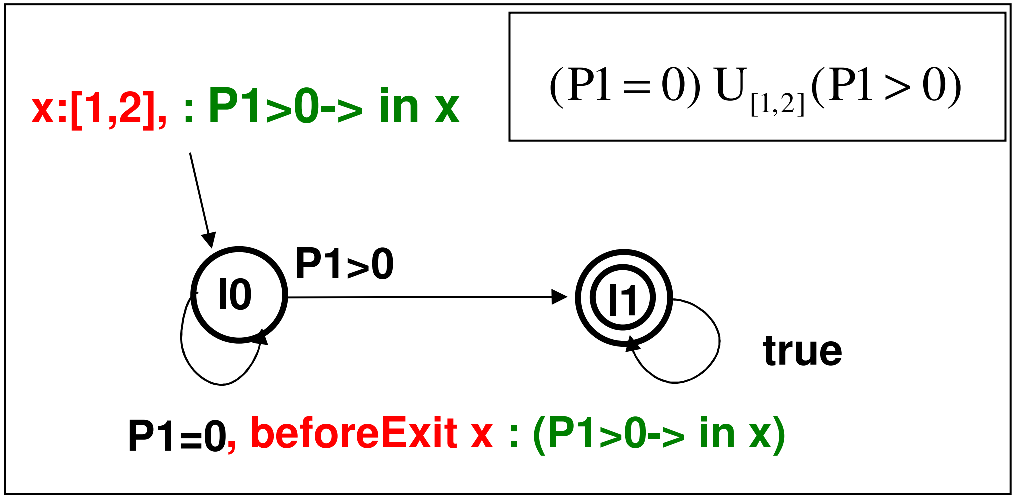
State q = (l0, A, I), A={x}, I(x)=[-2,-1]



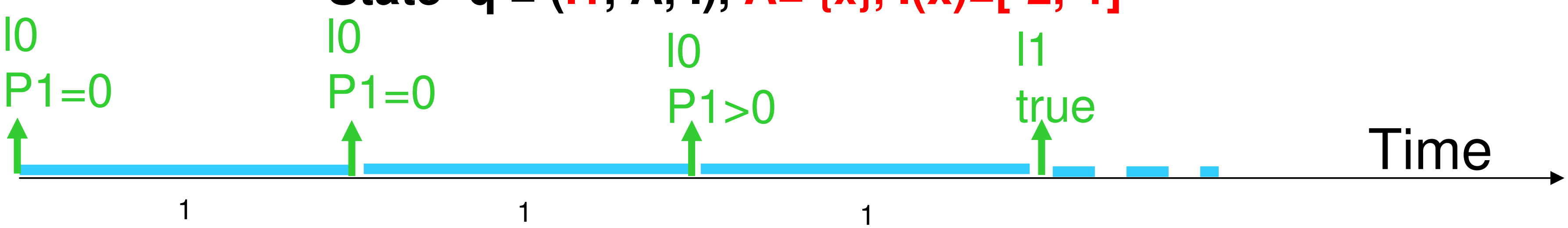
Interval Timed Büchi Automata



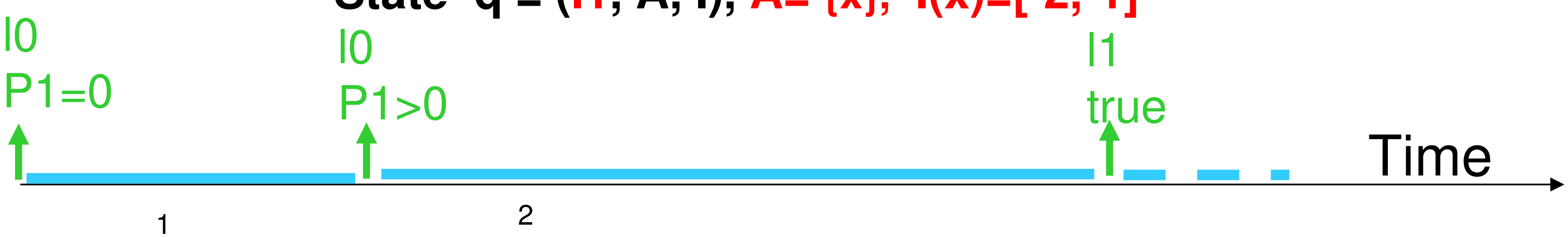
Interval Timed Büchi Automata



State q = (l1, A, I), A= {x}, I(x)=[-2,-1]

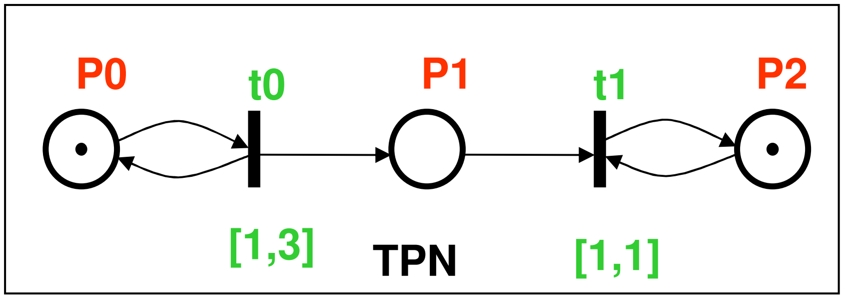


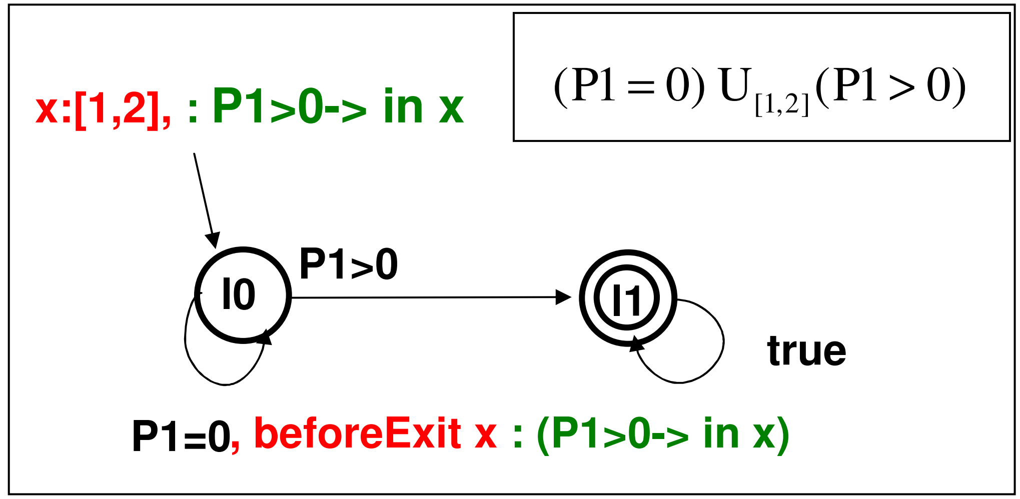
State q = (l1, A, I), A= {x}, I(x)=[-2,-1]

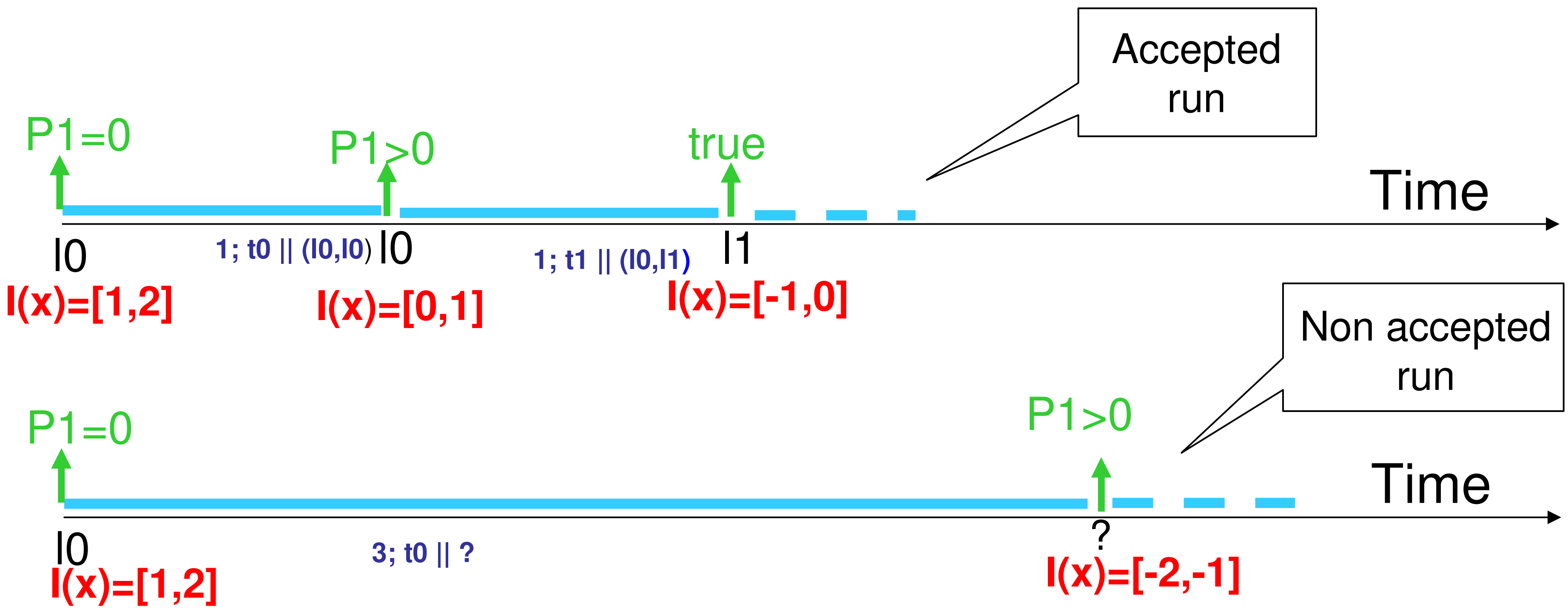


Hanifa Boucheneb EPM

Synchronous product TPN⊗ITBA

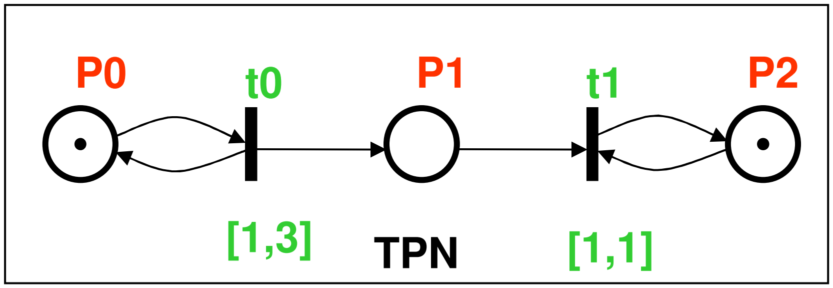


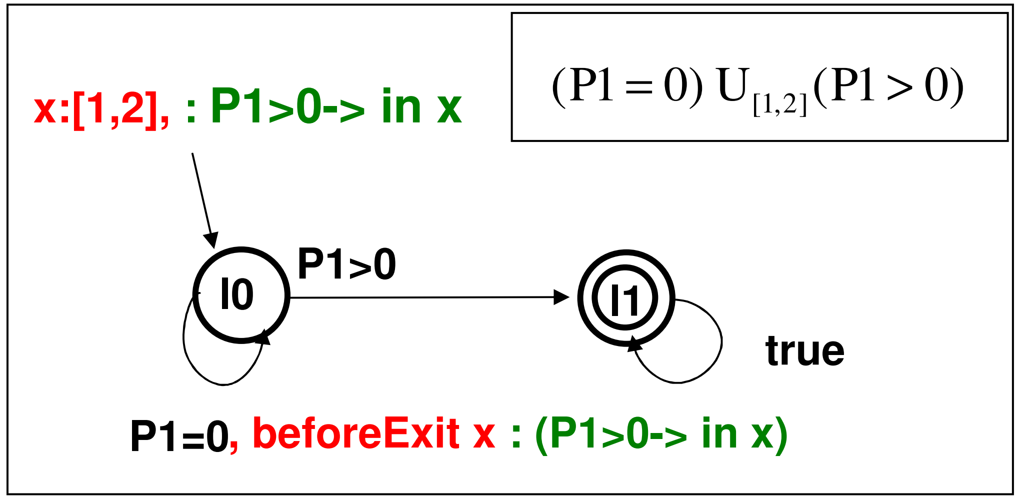




Hanifa Boucheneb EPM

SCG / CSCG of TPN⊗ITBA





State class of TPN⊗ITBA: ***(M, I_i, A, F)***

En(M) ∪ A

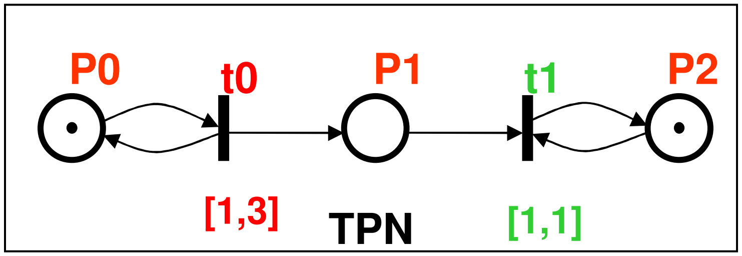
Firing some transition t_f:

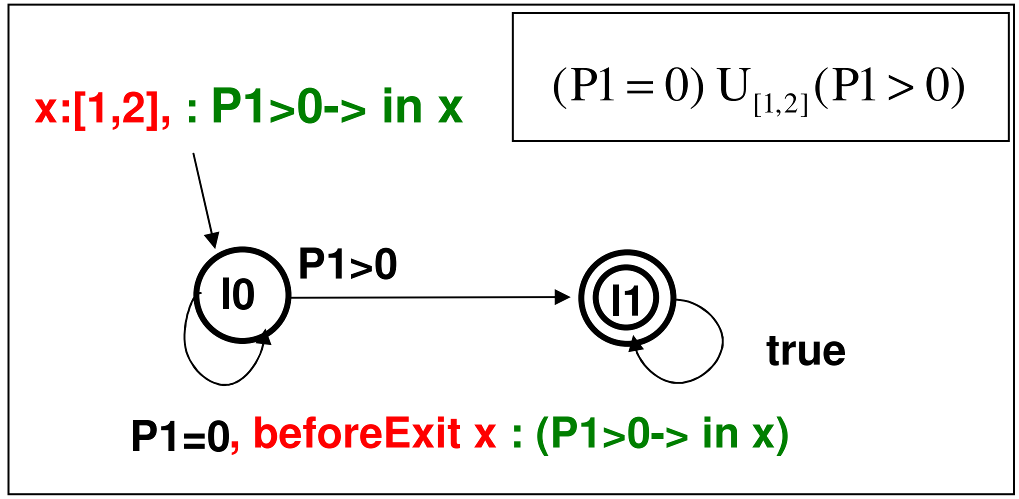
before x	*in x*	*beforeExit x*	*after x*
$x - t_f > 0$	$x - t_f = 0$	$x - t_f \geq 0$	$x - t_f < 0$

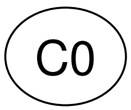
before x is satisfied when t_f is fired

Hanifa Boucheneb EPM

TPN⊗ITBA







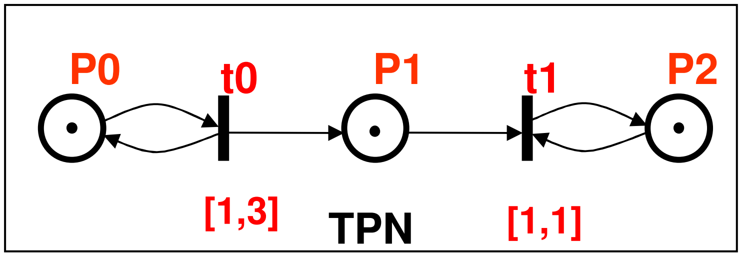
$(\{P0, P2\}, l0), \{x\},$

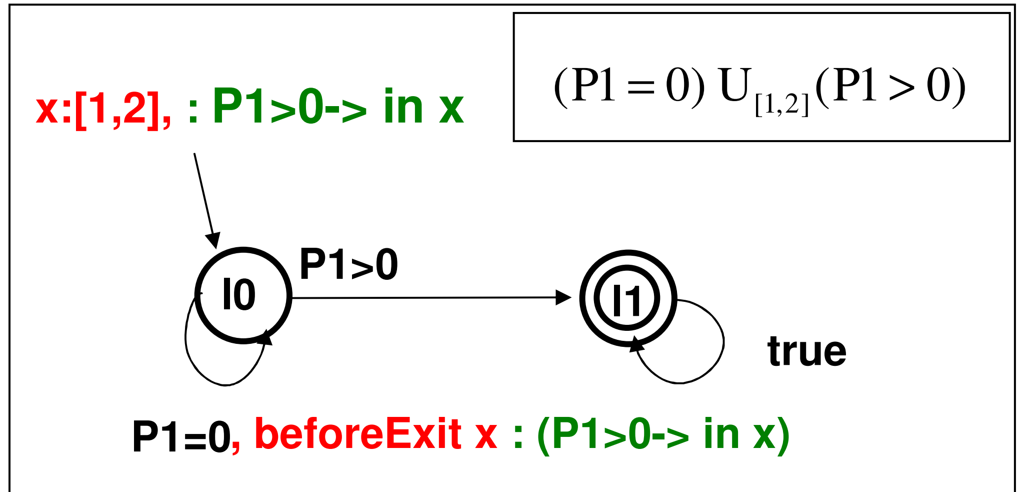
$1 \leq x \leq 2 \wedge$

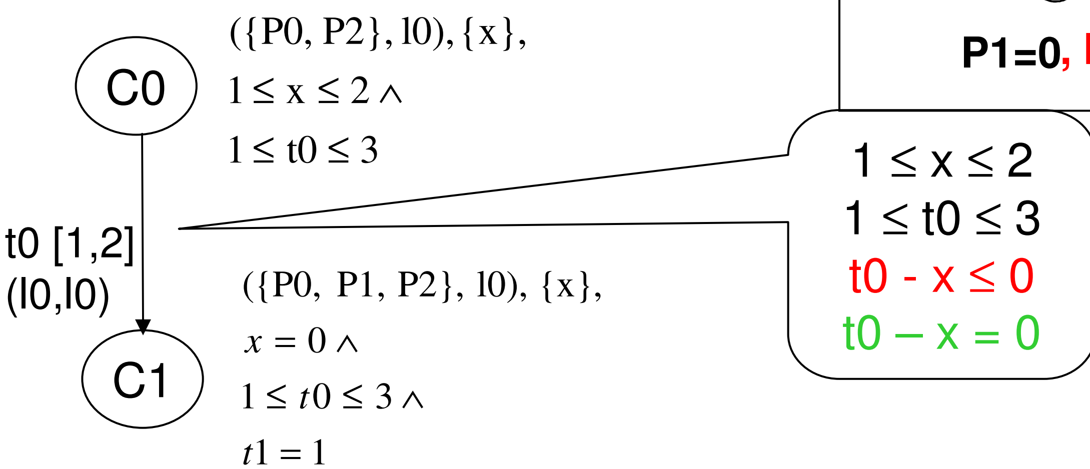
$1 \leq t0 \leq 3$

Hanifa Boucheneb EPM

TPN⊗ITBA

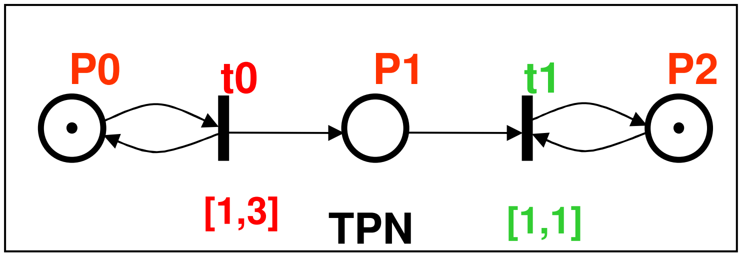


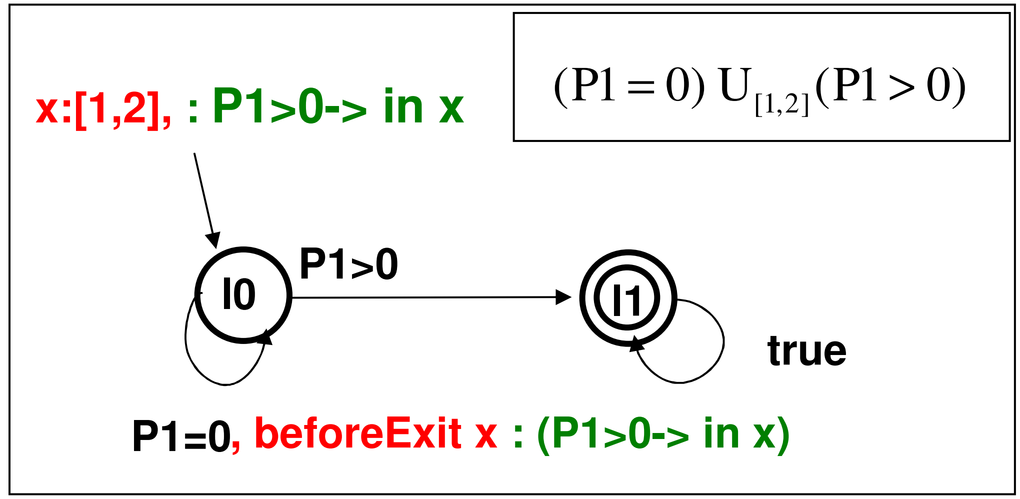


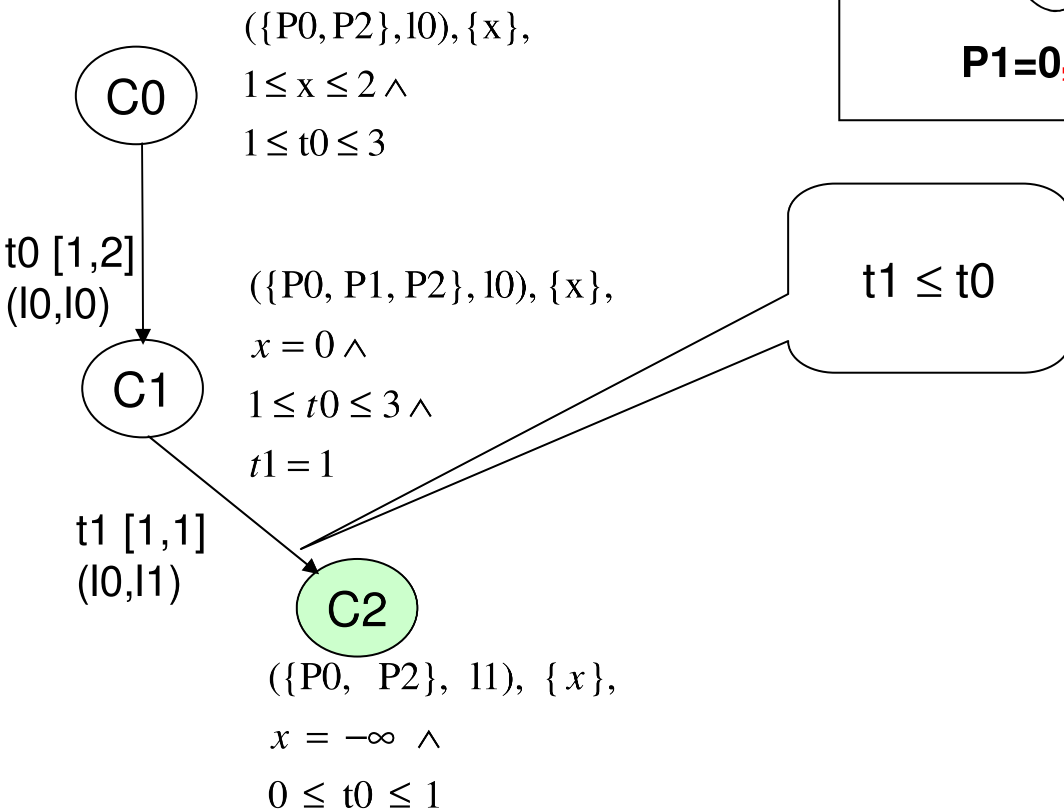


Hanifa Boucheneb EPM

TPN⊗ITBA

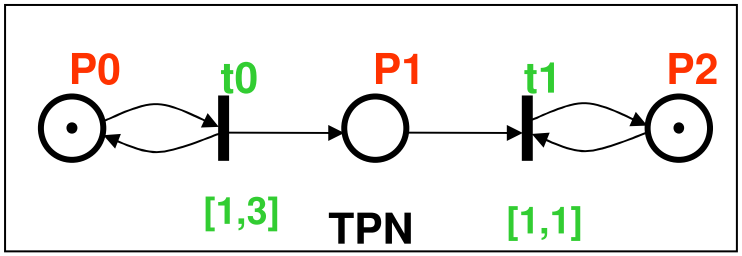


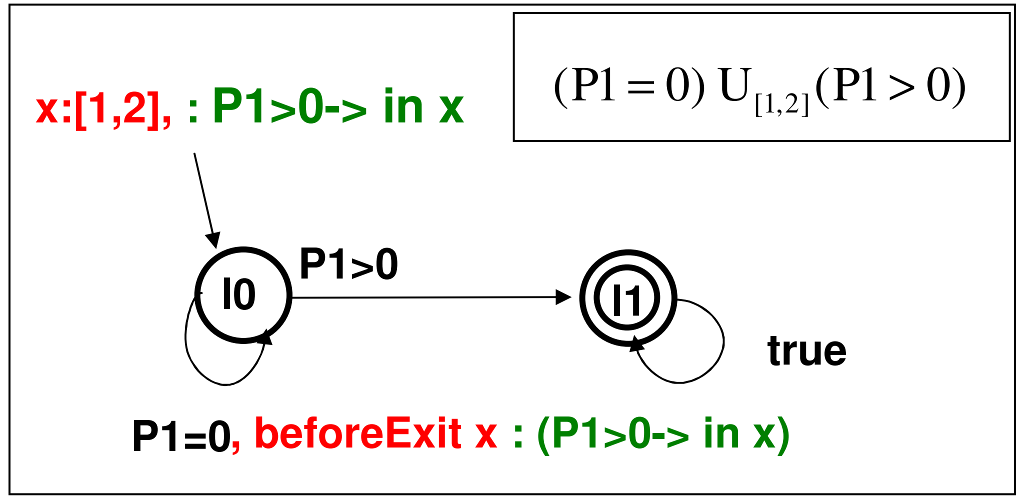


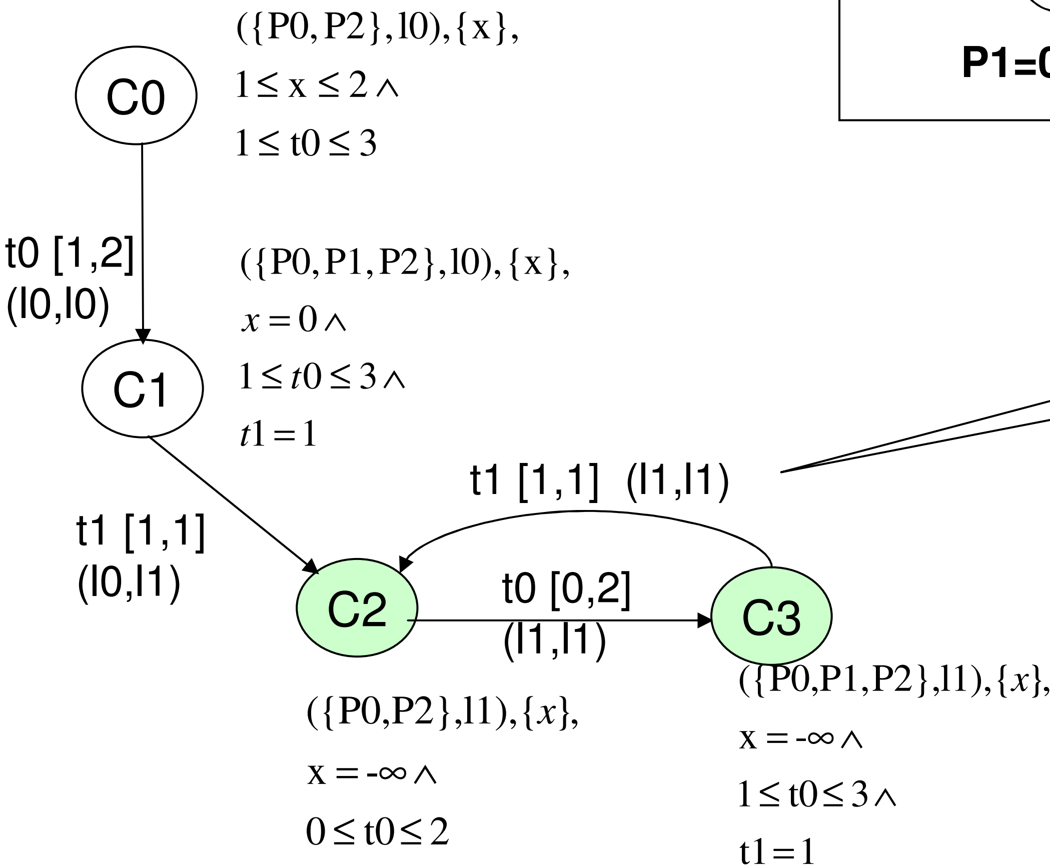


Hanifa Boucheneb EPM

TPN⊗ITBA







L(TPN) ∩ L(ITBA) ≠∅

Conclusion

- CSCG
- An Interval Timed Büchi Automata (ITBA)
- A verification technique using ITBA and SCG / CSCG

Merci!