Verification of time-critical systems : abstraction, reduction and distribution based on Timed Aggregate Graph

Amal Chamakh

Laboratoire d'Informatique de Paris Nord
UNIVERSITÉ PARIS 13, SORBONNE PARIS CITÉ

February 11, 2016

Introduction

- Complicated systems routinely built today
- Failures are costly
- Verification techniques needed

Introduction

- Complicated systems routinely built today
- Failures are costly
- Verification techniques needed

Formal verification

1. Mathematical model of the system (*i.e.*: Petri Nets)
2. Specification : write correctness requirements
3. Analysis and verification : check that model satisfies specification $\rightarrow$ prove or disprove the correctness claim

   $\rightarrow$ Model checking

Model checking

Principle

1. Design the system with a model $\mathcal{M}$ and design a property φ
2. $\mathcal{M} \models \varphi$? if no, a counter-example σ
3. Analyse the result:
   - If yes, **OK**
   - If no, refine $\mathcal{M}$ using σ and go to (1).

Model checking

Principle

1. Design the system with a model $\mathcal{M}$ and design a property φ
2. $\mathcal{M} \models \varphi$? if no, a counter-example σ
3. Analyse the result:
    - If yes, **OK**
    - If no, refine $\mathcal{M}$ using σ and go to (1).

Approach

- State space traversal

Model checking

Principle

1. Design the system with a model $\mathcal{M}$ and design a property φ
2. $\mathcal{M} \models \varphi$? if no, a counter-example σ
3. Analyse the result:
   - If yes, **OK**
   - If no, refine $\mathcal{M}$ using σ and go to (1).

Approach

- State space traversal

State space explosion problem

Exponential w.r.t size of the description

State space explosion problem

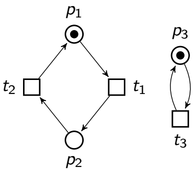
To tackle the state space explosion problem

- On-the-fly construction
- Partial order reduction
- Stuttering equivalence
- Modularity
- Symbolic representations (e.g., BDDs)

Petri Nets

Consists of places and transitions, connected by arcs. Tokens can be placed in places and manipulated by firing of transitions.

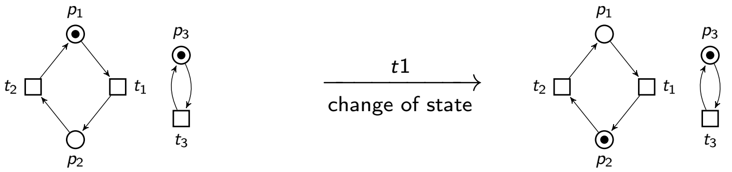
- Places (circles): represent possible states of the system
- Transitions (squares): are events or actions which cause the change of state
- Tokens (black dots): ressources
- State/Marking: a distribution of tokens over places



Petri Nets

Consists of places and transitions, connected by arcs. Tokens can be placed in places and manipulated by firing of transitions.

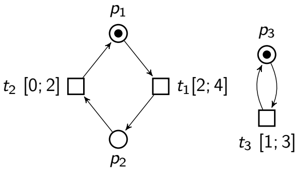
- Places (circles): represent possible states of the system
- Transitions (squares): are events or actions which cause the change of state
- Tokens (black dots): ressources
- State/Marking: a distribution of tokens over places



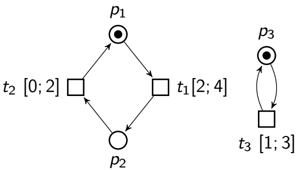
Time Petri Nets (TPN)

Temporal condition of transitions firing

- Each transition is associated with a time interval $[e(t), l(t)]$
- Implicit clock $v(t)$ per enabled transition t
- An enabled transition t is firable if $v(t)$ lies in $[e(t), l(t)]$
- A state of the TPN (configuration) is a couple (m, V)

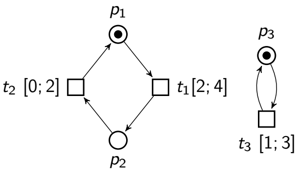


TPN: Change of state (Strong Semantics)



$$\begin{cases} m_0 = (1, 0, 1) \\ V_0 = (0, \perp, 0) \end{cases}$$

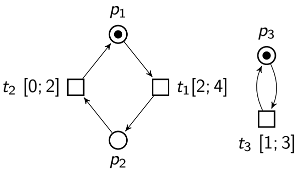
TPN: Change of state (Strong Semantics)



Elapsing of time

$$\begin{cases} m_0 = (1,0,1) \\ V_0 = (0,\perp,0) \end{cases} \xrightarrow{0.5} \begin{cases} (1,0,1) \\ (0.5,\perp,0.5) \end{cases}$$

TPN: Change of state (Strong Semantics)

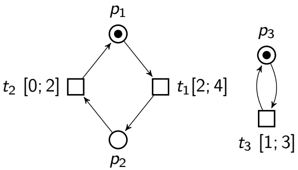


Elapsing of time

Firing of transitions

$$\begin{cases} m_0 = (1,0,1) \\ V_0 = (0,\perp,0) \end{cases} \xrightarrow{0.5} \begin{cases} (1,0,1) \\ (0.5,\perp,0.5) \end{cases} \xrightarrow{t_1} \begin{cases} (0,1,1) \\ (\perp,0,0.5) \end{cases}$$

TPN: Change of state (Strong Semantics)



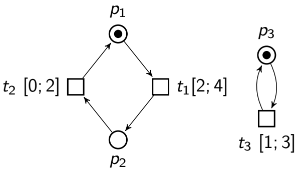
Elapsing of time

Firing of transitions

$$\begin{cases} m_0 = (1,0,1) \\ V_0 = (0,\bot,0) \end{cases} \xrightarrow{0.5} \begin{cases} (1,0,1) \\ (0.5,\bot,0.5) \end{cases} \xrightarrow{t_1} \begin{cases} (0,1,1) \\ (\bot,0,0.5) \end{cases}$$

Newly enabled transition VS persistent transition

TPN: Change of state (Strong Semantics)



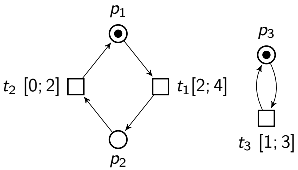
Elapsing of time — Firing of transitions

$$\begin{cases} m_0 = (1,0,1) \\ V_0 = (0,\perp,0) \end{cases} \xrightarrow{0.5} \begin{cases} (1,0,1) \\ (0.5,\perp,0.5) \end{cases} \xrightarrow{t_1} \begin{cases} (0,1,1) \\ (\perp,0,0.5) \end{cases}$$

Newly enabled transition VS persistent transition

Infinite State Space Systems

TPN: Change of state (Strong Semantics)



Elapsing of time   Firing of transitions

$$\begin{cases} m_0 = (1,0,1) \\ V_0 = (0,\perp,0) \end{cases} \xrightarrow{0.5} \begin{cases} (1,0,1) \\ (0.5,\perp,0.5) \end{cases} \xrightarrow{t_1} \begin{cases} (0,1,1) \\ (\perp,0,0.5) \end{cases}$$

Newly enabled transition VS persistent transition

Infinite State Space Systems

- Build a finite abstraction of the infinite state space
  - State Class Graph (SCG) [Berthomieu'83] — TINA
  - Region Graph (RG) [Alur'93] —> Zone Based Graph (ZBG) [Gardey'03] — Romeo

Timed Aggregate Graph (TAG)

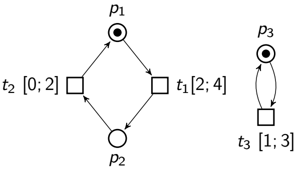
Principle

- Nodes are aggregates: elapsing of time is hidden
- A dynamic time interval for each enabled transition
  - $\delta(a)$=minimum time at a
  - $\Delta(a)$=maximum time at a
- Arcs are labeled with transitions only

Timed Aggregate $a = (m, E, \mathfrak{M})$

1. m : current marking
2. $E = \{(t, \alpha_t, \beta_t), tenabledtransitionbya.m\}$
3. $\mathfrak{M}(t_i, t_j) = [\alpha_{(t_i,t_j)}, \beta_{(t_i,t_j)}]$ when t_i met t_j for the last time

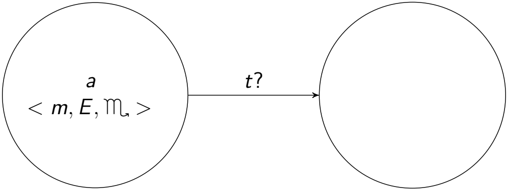
Timed aggregate



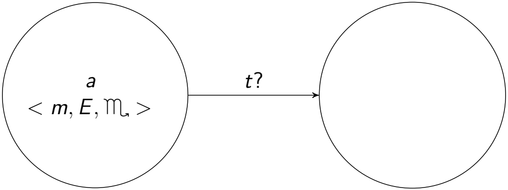
$$a_0 : \begin{cases} m = (1, 0, 1) \\ E = \{(t_1, 2, 4), (t_3, 1, 3)\} \\ Meet = \begin{pmatrix} - & [2 : 4] \\ [1 : 3] & - \end{pmatrix} \end{cases}$$

$$\begin{cases} m_0 = (1, 0, 1) \\ V_0 = (0, \bot, 0) \end{cases} \xrightarrow{0.5} \begin{cases} (1, 0, 1) \\ (0.5, \bot, 0.5) \end{cases} \xrightarrow{1} \begin{cases} (1, 0, 1) \\ (1.5, \bot, 1.5) \end{cases} \ldots \begin{cases} (1, 0, 1) \\ (3, \bot, 3) \end{cases}$$

Fireability



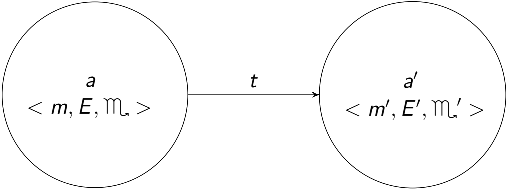
Fireability



$(t, \alpha_t, \beta_t) \in a.E$

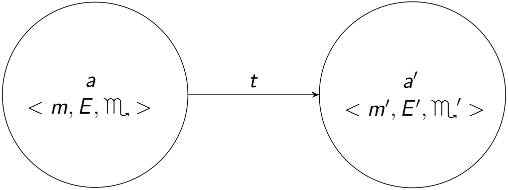
- $\mathfrak{M}(t, t') = [\alpha^{\mathfrak{M}(t,t')} : \beta^{\mathfrak{M}(t,t')}]$
- t is firable at a, denoted by $a \overset{t}{\leadsto}$, iff
  - $\forall t'$ enabled by a, $\alpha^{\mathfrak{M}(t,t')} \leq \beta^{\mathfrak{M}(t',t)}$

Computing the successor aggregate



$$a \overset{t}{\rightsquigarrow} a'$$

Computing the successor aggregate

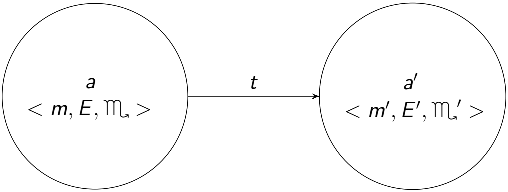


$$a \overset{t}{\rightsquigarrow} a'$$

New marking

$$m' = m - Pre(t) + Post(t)$$

Computing the successor aggregate



$$a \overset{t}{\rightsquigarrow} a'$$

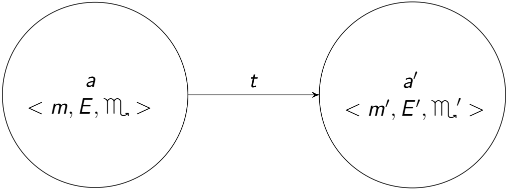
New marking

$m' = m - Pre(t) + Post(t)$

Dynamic firing intervals

- **if t' is newly enabled at a'**
  - $\alpha_{t'} \longleftarrow t'_{min}$
  - $\beta_{t'} \longleftarrow t'_{max}$

Computing the successor aggregate



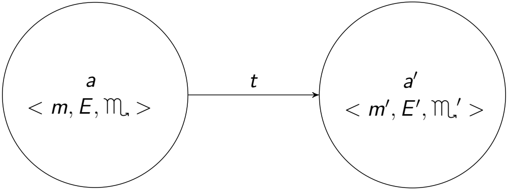
Dynamic firing intervals

if t' is not newly enabled at a'

The more the system can stay at a, the less it can stay at a'

- $\alpha'_{t'} = \begin{cases} 0 \text{ if } t' \text{ firable at } a \\ Max(\alpha^{\mathfrak{m}(t',t'')} - \beta^{\mathfrak{m}(t'',t')}),\ \text{otherwise} \\ \forall t'' \text{ enabled at } a \end{cases}$
- $\beta'_{t'} = Min(\beta_{t'}, (\beta^{\mathfrak{m}(t',t)} - \alpha^{\mathfrak{m}(t,t')}))$

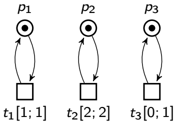
Computing the successor aggregate



♏

$$♏'(t_1, t_2) \leftarrow \begin{cases} [t_{1,min} : t_{1,max}] \textit{ if } t_1 \textit{ is newly enabled} \\ [\alpha'_{t_1} : \beta'_{t_1}] \textit{ if } t_1 \textit{ is old and } t_2 \textit{ is new} \\ ♏(t_1, t_2) \textit{ if } t_1 \textit{ and } t_2 \textit{ are both old} \end{cases}$$

Fireability

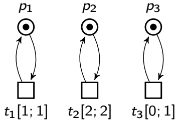


Fireability

$$\{(t_1, 1, 1), (t_2, 2, 2), (t_3, 0, 1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

$\boldsymbol{a = (m, E, Meet)}$

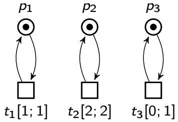


Fireability

$\{(t_1, 1, 1), (t_2, 2, 2), (t_3, 0, 1)\}$

$$\begin{pmatrix} - & [1 : 1] & [1 : 1] \\ [2 : 2] & - & [2 : 2] \\ [0 : 1] & [0 : 1] & - \end{pmatrix}$$

t_2 is not fireable

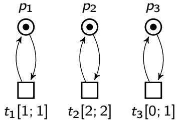


Fireability

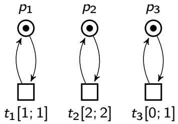
$$\{(t_1, 1, 1), (t_2, 2, 2), (t_3, 0, 1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

t_2 is not fireable



Firing



$$\{(t_1,1,1),(t_2,2,2),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

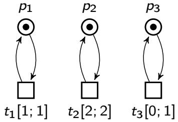
$$\downarrow t_3$$

$$\{(t_1,0,1),(t_2,1,2),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

Timed Aggregate Graph: Construction example

t_2 is not fireable



$$\{(t_1,1,1),(t_2,2,2),(t_3,0,1)\}$$

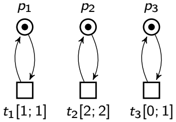
$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

$\downarrow t_3$

$$\{(t_1,0,1),(t_2,1,2),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

Timed Aggregate Graph: Construction example



$$\{(t_1,1,1),(t_2,2,2),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

$$\downarrow t_3$$

$$\{(t_1,0,1),(t_2,1,2),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

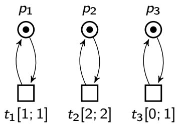
$$\downarrow t_1$$

$$\{(t_1,1,1),(t_2,1,1),(t_3,0,1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [1:1] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

Timed Aggregate Graph: Construction example

t_2 is firable now



$$\{(t_1, 1, 1), (t_2, 2, 2), (t_3, 0, 1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

$\downarrow t_3$

$$\{(t_1, 0, 1), (t_2, 1, 2), (t_3, 0, 1)\}$$

$$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

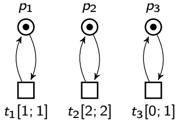
$\downarrow t_1$

$$\{(t_1, 1, 1), (t_2, 1, 1), (t_3, 0, 1)\}$$

$$\begin{pmatrix} - & [1:1] & [1:1] \\ [1:1] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

Timed Aggregate Graph: Construction example

t_2 is firable now



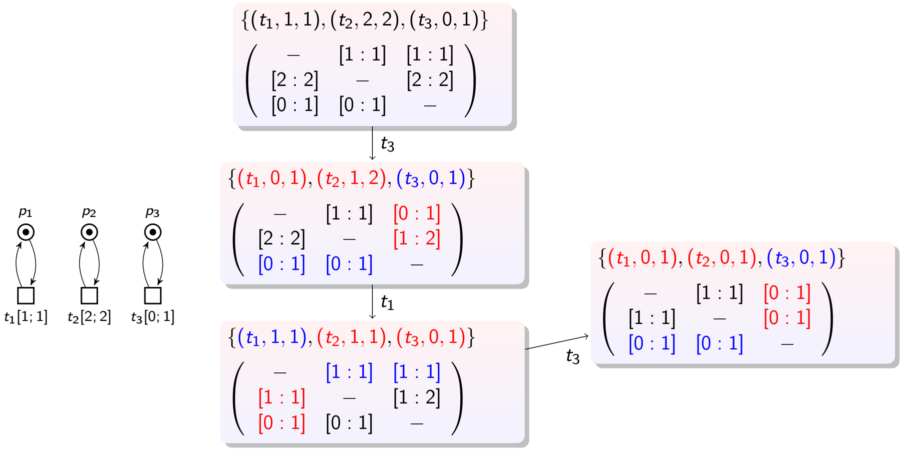
$$\{(t_1,1,1),(t_2,2,2),(t_3,0,1)\}$$
$$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

$\downarrow t_3$

$$\{(t_1,0,1),(t_2,1,2),(t_3,0,1)\}$$
$$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$

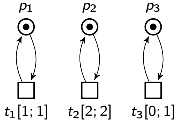
$\downarrow t_1$

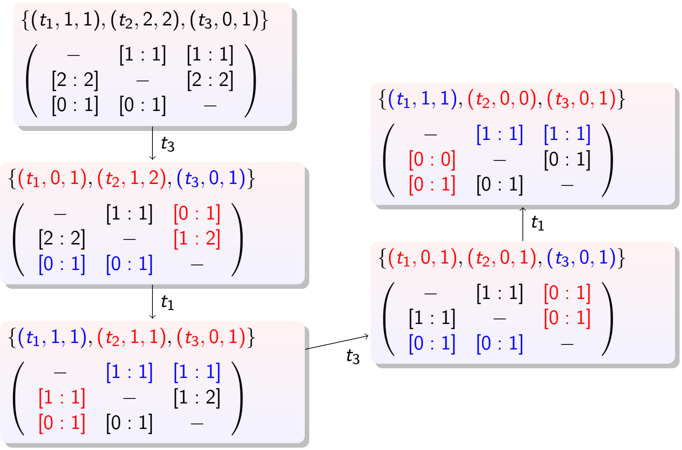
$$\{(t_1,1,1),(t_2,1,1),(t_3,0,1)\}$$
$$\begin{pmatrix} - & [1:1] & [1:1] \\ [1:1] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$$



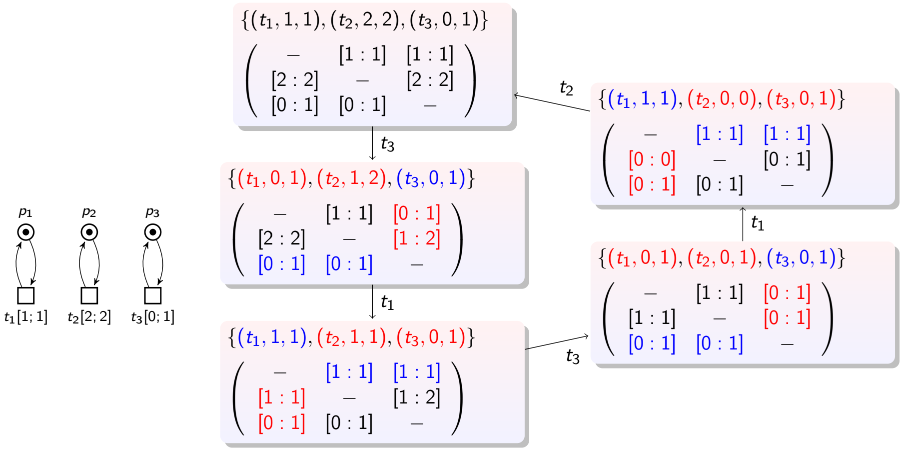
p1
p2
p3
t1 [1; 1]
t2 [2; 2]
t3 [0; 1]
{(t1, 1, 1), (t2, 2, 2), (t3, 0, 1)}
– [1 : 1] [1 : 1]
[2 : 2] – [2 : 2]
[0 : 1] [0 : 1] –
t3
{(t1, 0, 1), (t2, 1, 2), (t3, 0, 1)}
– [1 : 1] [0 : 1]
[2 : 2] – [1 : 2]
[0 : 1] [0 : 1] –
t1
{(t1, 1, 1), (t2, 1, 1), (t3, 0, 1)}
– [1 : 1] [1 : 1]
[1 : 1] – [1 : 2]
[0 : 1] [0 : 1] –
t3
{(t1, 0, 1), (t2, 0, 1), (t3, 0, 1)}
– [1 : 1] [0 : 1]
[1 : 1] – [0 : 1]
[0 : 1] [0 : 1] –

Timed Aggregate Graph: Construction example



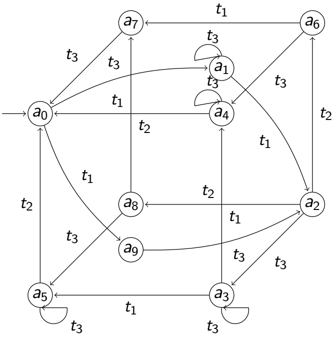


Timed Aggregate Graph: Construction example



Timed Aggregates Graph: Example

aggregate	E	Meet
a_0	$\{\langle t_1, 1, 1\rangle, \langle t_2, 2, 2\rangle, \langle t_3, 0, 1\rangle\}$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_1	$\{\langle t_1, 0, 1\rangle, \langle t_2, 1, 2\rangle, \langle t_3, 0, 1\rangle\}$	$\begin{pmatrix} - & [1:1] & [0:1] \\ [2:2] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_2	$\{\langle t_1, 1, 1\rangle\}, \langle t_2, 1, 1\rangle, \langle t_3, 0, 1\rangle$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [1:1] & - & [1:2] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_3	$\{\langle t_1, 0, 1\rangle, \langle t_2, 0, 1\rangle, \langle t_3, 0, 1\rangle\}$	$\begin{pmatrix} - & [1:1] & [0:1] \\ [1:1] & - & [0:1] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_4	$\{\langle t_1, 0, 0\rangle, \langle t_2, 2, 2\rangle, \langle t_3, 0, 1\rangle\}$	$\begin{pmatrix} - & [0:0] & [0:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_5	$\{\langle t_1, 1, 1\rangle, \langle t_2, 0, 0\rangle, \langle t_3, 0, 1\rangle\}$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [0:0] & - & [0:1] \\ [0:1] & [0:1] & - \end{pmatrix}$
a_6	$\{\langle t_1, 0, 0\rangle, \langle t_2, 2, 2\rangle, \langle t_3, 0, 0\rangle\}$	$\begin{pmatrix} - & [0:0] & [1:1] \\ [2:2] & - & [2:2] \\ [0:1] & [0:0] & - \end{pmatrix}$
a_7	$\{\langle t_1, 1, 1\rangle, \langle t_2, 2, 2\rangle, \langle t_3, 0, 0\rangle\}$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [2:2] & - & [2:2] \\ [0:0] & [0:0] & - \end{pmatrix}$
a_8	$\{\langle t_1, 1, 1\rangle, \langle t_2, 0, 0\rangle, \langle t_3, 0, 0\rangle\}$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [0:0] & - & [1:2] \\ [0:0] & [0:1] & - \end{pmatrix}$
a_9	$\{\langle t_1, 1, 1\rangle, \langle t_2, 1, 1\rangle, \langle t_3, 0, 0\rangle\}$	$\begin{pmatrix} - & [1:1] & [1:1] \\ [1:1] & - & [2:2] \\ [0:0] & [0:1] & - \end{pmatrix}$



Theoretical results

- The TAG associated with a **bounded** TPN is finite
- Each timed sequence of a TPN corresponds to an untamed sequence of the TAG, and vice versa
- Algorithm for building an explicit run from a path on the TAG
- The maximal and the minimal access time /firing time of a marking/transition
- On-The-Fly verification of reachability timed properties by exploring the TAG

Experimental results

Parameters	SCG (with Tina) (nodes / arcs)	ZBG (with Romeo) (nodes / arcs)	TAG-TPN (nodes / arcs)
Nb. prod/cons	TPN model of producer/consumer		
1	34 / 56	34 / 56	34 / 56
2	748 / 2460	593 / 1 922	740 / 2438
3	4604 / 21891	3240 / 15200	4553 / 21443
4	14086 / 83375	9504 / 56038	13878 / 80646
5	31657 / 217423	20877 / 145037	30990 / 207024
6	61162 / 471254	39306 / 311304	60425 / 449523
7	107236 / 907 708	67224 / 594795	106101 / 856050

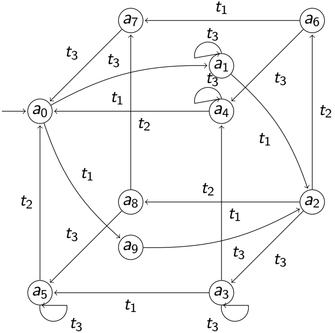
Experimental results

Parameters	SCG (with Tina) (nodes / arcs)	ZBG (with Romeo) (nodes / arcs)	TAG-TPN (nodes / arcs)
Nb. processes	Fischer protocol		
1	4 / 4	4 / 4	4 / 4
2	18 / 29	19 / 32	20 / 32
3	65 / 146	66 / 153	74 / 165
4	220 / 623	221 / 652	248 / 712
5	727 / 2536	728 / 2 615	802 / 2825
6	2378 / 9154	2379 / 10098	2564 / 10728
7	7737 / 24744	7738 / 37961	8178 / 39697
8	25080 / 102242	25081 / 139768	26096 / 144304

Experimental results

Parameters	SCG (with Tina) (nodes / arcs)	ZBG (with Romeo) (nodes / arcs)	TAG-TPN (nodes / arcs)
Nb. processes	Train crossing		
1	11 / 1 4	11 / 1 4	11 / 14
2	123 / 218	114 / 200	123 / 218
3	3101 / 7754	2817 / 6944	2879 / 7280
4	134501 / 436896	122290 / 391244	105360 / 354270

Partial-order reduction



- exploiting the commutativity of concurrently executed transitions
- result in the same state when executed in different orders
- reducing the size of the state-space

Modularity and Distribution

- Two (or many) TPNs sharing many transitions = one joint TPN
- Modular construction of the TAG : a TAG for each modular TPN $\Rightarrow$ one synchronized TAG for the joint TPN
- In multicore systems, distributing the whole construction based on modularity

Conclusion and Perspectives

- Timed Aggregate Graph
  - A finite abstraction for bounded Time Petri nets state space
  - Preserves reachable markings and timed traces
  - Allowing on the fly verification of **timed** reachability properties
  - Comparable to Tina/Romeo Tools performances

Conclusion and Perspectives

- Timed Aggregate Graph
  - A finite abstraction for bounded Time Petri nets state space
  - Preserves reachable markings and timed traces
  - Allowing on the fly verification of **timed** reachability properties
  - Comparable to Tina/Romeo Tools performances
- Next Steps
  - Improve the implementation w.r.t. construction time
  - Develop dedicated verification algorithms
    - Reachability properties
    - Full TCTL ?
  - Further reductions
    - Depending on the property
    - Partial Order
    - Aggregate the TAG's aggregates ?
  - Extend to Timed automata

THANK YOU