NWPT 2012

31st October 2012
Bergen, Norway

Robustness Analysis of Time Petri Nets

Étienne André[1], **Shweta Garg**[2]

[1]LIPN, Université Paris 13, Sorbonne Paris Cité, France
[2]Dept. of Computer Science, IIT Bombay, Mumbai, India

Context: Verifying Complex Timed Systems (1/2)

- Need for early bug detection
  - Bugs discovered when final testing: expensive
  - Need for thorough modeling and verification

Context: Verifying Complex Timed Systems (1/2)

- Need for early bug detection
  - Bugs discovered when final testing: expensive
  - Need for thorough modeling and verification

- Input

A timed concurrent system

Context: Verifying Complex Timed Systems (1/2)

- Need for early bug detection
  - Bugs discovered when final testing: expensive
  - Need for thorough modeling and verification

- Input

A timed concurrent system

A good behavior expected for the system

Context: Verifying Complex Timed Systems (1/2)

- Need for early bug detection
  - Bugs discovered when final testing: expensive
  - Need for thorough modeling and verification
- Input

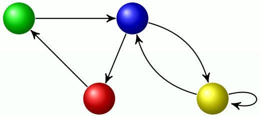
A timed concurrent system

A good behavior expected for the system

- Question: does the system behave well?

Context: Verifying Complex Timed Systems (2/2)

- Use formal methods



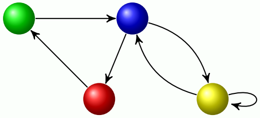
A finite model of the system

$AG\neg$ 

A formula to be satisfied

Context: Verifying Complex Timed Systems (2/2)

- Use formal methods



$\overset{?}{\models}$

AG¬

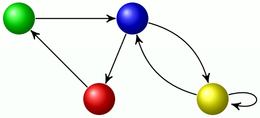
A finite model of the system

A formula to be satisfied

- Question: does the model of the system satisfy the formula?

Context: Verifying Complex Timed Systems (2/2)

- Use formal methods



$\overset{?}{\models}$

$AG\neg$ 

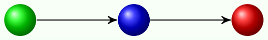
A finite model of the system

A formula to be satisfied

- Question: does the model of the system satisfy the formula?

Yes

No



Counterexample

Motivation: Robustness Analysis

- Timed systems are characterized by a set of timing constants
  - "The packet transmission lasts for $50\,ms$"
  - "The sensor reads the value every $10\,s$"
- Verification for one set of constants does not guarantee the correctness for other values
- Challenge: Robustness [Markey, 2011]
  - What happens if 50 is implemented with 49.99?
  - Until which value can we increase or decrease 50 such that the system still behaves well?

Motivation: Robustness Analysis

- Timed systems are characterized by a set of timing constants
  - "The packet transmission lasts for 50 ms"
  - "The sensor reads the value every 10 s"
- Verification for one set of constants does not guarantee the correctness for other values
- Challenge: Robustness [Markey, 2011]
  - What happens if 50 is implemented with 49.99?
  - Until which value can we increase or decrease 50 such that the system still behaves well?
- Parametric analysis
  - Consider that timing constants are parameters
  - Find good values for the parameters, such that the system still behaves well

Outline

Outline

1. Parametric Inhibitor Time Petri Nets
2. Robustness Analysis Using the Inverse Method
3. Perspectives

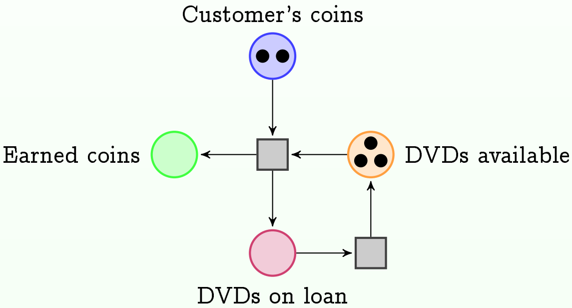
Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

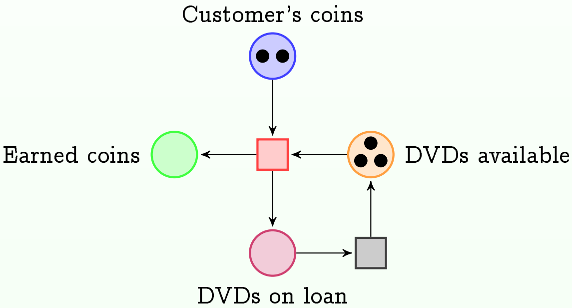
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

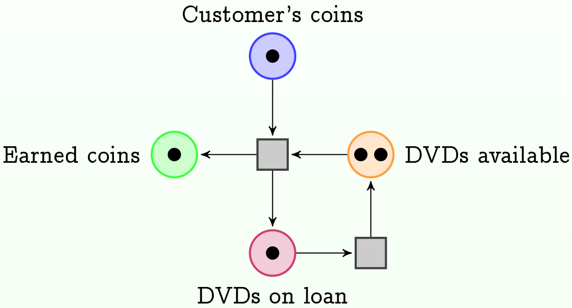
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

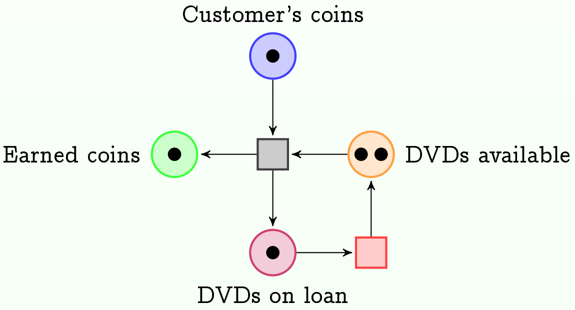
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

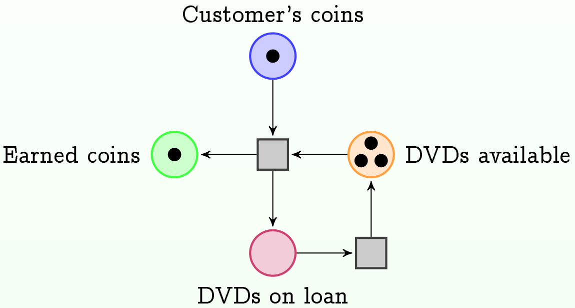
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

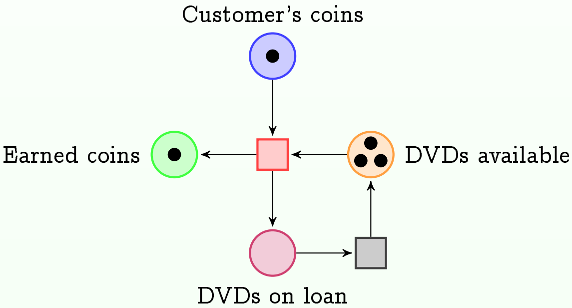
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

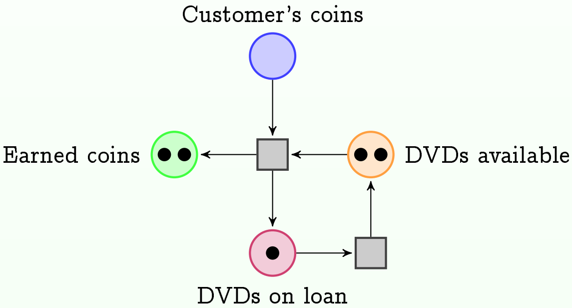
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

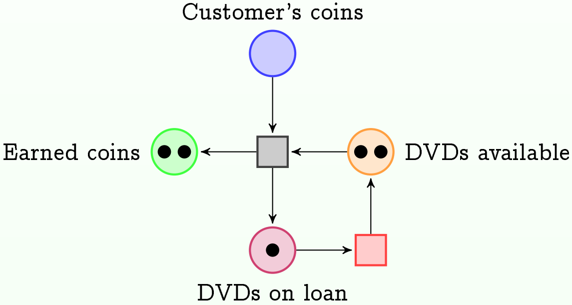
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

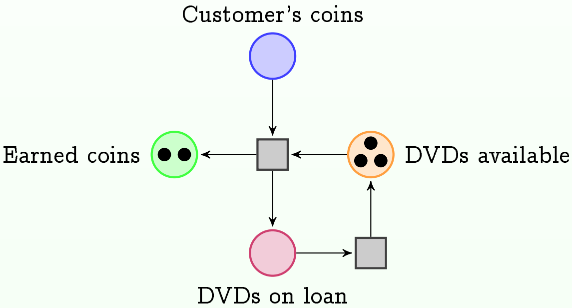
- Example: A DVD renting machine



Petri Nets [Petri, 1962]

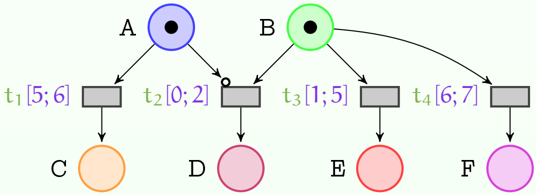
- Advantages of Petri nets
  - Detailed view of the process with an expressive graphical representation based on places and transitions
  - A formal semantics
  - Powerful model checking tools

- Example: A DVD renting machine

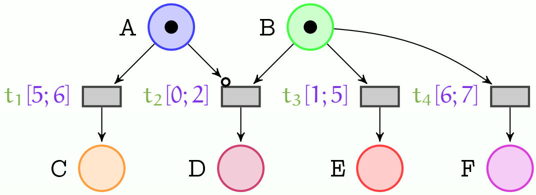


Time Petri Nets With Inhibitor Arcs

- Powerful formalism for verifying real-time systems [Merlin, 1974]
- Transition t_1 can be fired from 5 to 6 units of time after it is enabled
- An enabled transition must fire before (or at) its upper bound
  - Except if another transition fires before
- An inhibitor arc (t_2) enables its transition once its predecessor place (A) is empty



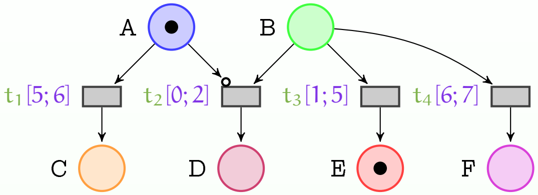
Time Petri Nets With Inhibitor Arcs: Example



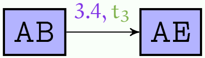
Some possible runs

AB

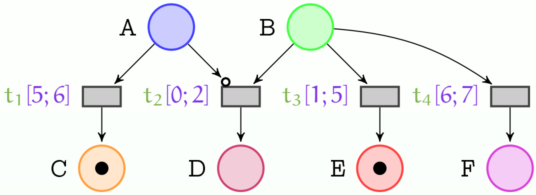
Time Petri Nets With Inhibitor Arcs: Example



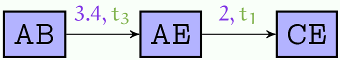
Some possible runs



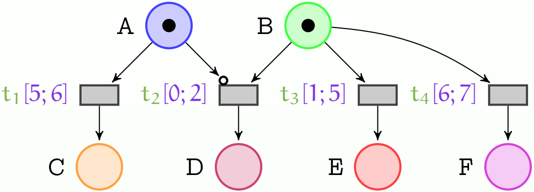
Time Petri Nets With Inhibitor Arcs: Example



Some possible runs



Time Petri Nets With Inhibitor Arcs: Example

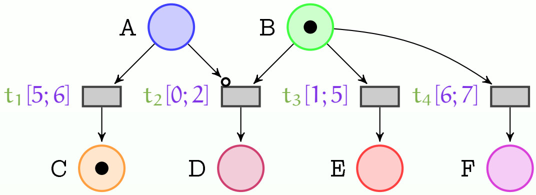


Some possible runs

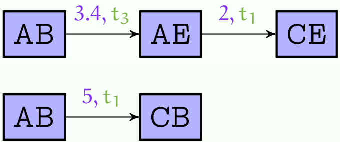
$$\boxed{AB} \xrightarrow{3.4,\ t_3} \boxed{AE} \xrightarrow{2,\ t_1} \boxed{CE}$$

$$\boxed{AB}$$

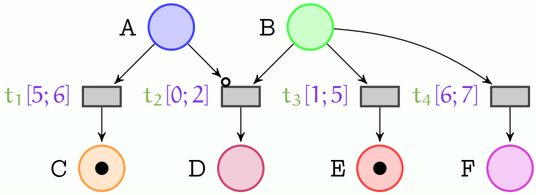
Time Petri Nets With Inhibitor Arcs: Example



Some possible runs



Time Petri Nets With Inhibitor Arcs: Example

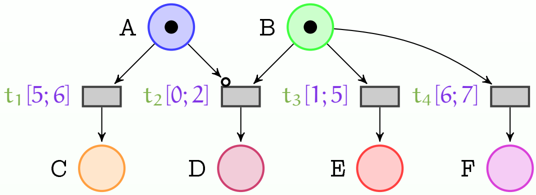


Some possible runs

$$\boxed{AB} \xrightarrow{3.4,\ t_3} \boxed{AE} \xrightarrow{2,\ t_1} \boxed{CE}$$

$$\boxed{AB} \xrightarrow{5,\ t_1} \boxed{CB} \xrightarrow{0,\ t_3} \boxed{CE}$$

Time Petri Nets With Inhibitor Arcs: Example



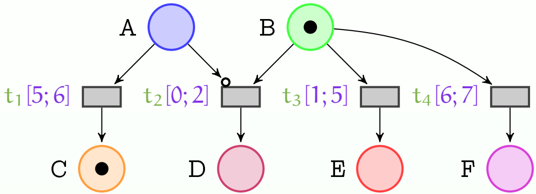
Some possible runs

$$AB \xrightarrow{3.4,\ t_3} AE \xrightarrow{2,\ t_1} CE$$

$$AB \xrightarrow{5,\ t_1} CB \xrightarrow{0,\ t_3} CE$$

$$AB$$

Time Petri Nets With Inhibitor Arcs: Example



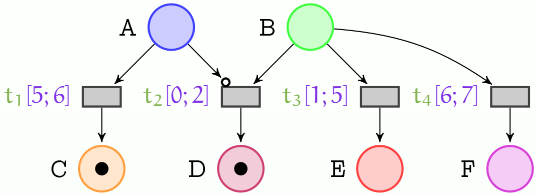
Some possible runs

$$\boxed{AB} \xrightarrow{3.4,\ t_3} \boxed{AE} \xrightarrow{2,\ t_1} \boxed{CE}$$

$$\boxed{AB} \xrightarrow{5,\ t_1} \boxed{CB} \xrightarrow{0,\ t_3} \boxed{CE}$$

$$\boxed{AB} \xrightarrow{5,\ t_1} \boxed{CB}$$

Time Petri Nets With Inhibitor Arcs: Example



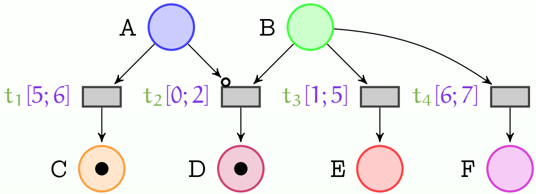
Some possible runs

$$\boxed{AB} \xrightarrow{3.4,\ t_3} \boxed{AE} \xrightarrow{2,\ t_1} \boxed{CE}$$

$$\boxed{AB} \xrightarrow{5,\ t_1} \boxed{CB} \xrightarrow{0,\ t_3} \boxed{CE}$$

$$\boxed{AB} \xrightarrow{5,\ t_1} \boxed{CB} \xrightarrow{0,\ t_2} \boxed{CD}$$

Time Petri Nets With Inhibitor Arcs: Example



Some possible runs

$AB \xrightarrow{3.4,\, t_3} AE \xrightarrow{2,\, t_1} CE$

$AB \xrightarrow{5,\, t_1} CB \xrightarrow{0,\, t_3} CE$

$AB \xrightarrow{5,\, t_1} CB \xrightarrow{0,\, t_2} CD$

Set of traces

$AB \xrightarrow{t_3} AE \xrightarrow{t_1} CE$

$AB \xrightarrow{t_1} CB \xrightarrow{t_3} CE$

$AB \xrightarrow{t_1} CB \xrightarrow{t_2} CD$

Trace: time-abstract behavior

Objectives

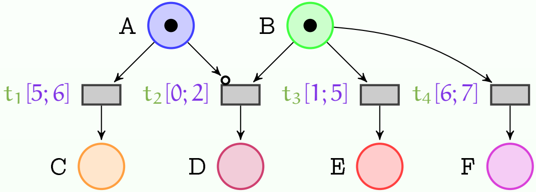
- We consider that the system behavior (good or bad) depends on the traces
- Questions
  - Until which value can we minimize the upper bound of t_3 (5) so that the system behavior remains the same?
  - Can we quantify the system robustness?

Objectives

- We consider that the system behavior (good or bad) depends on the traces
- Questions
  - Until which value can we minimize the upper bound of t_3 (5) so that the system behavior remains the same?
  - Can we quantify the system robustness?
- Idea
  - Reason with parametric time Petri nets
  - Synthesize a constraint on the parameters that guarantees the same behavior

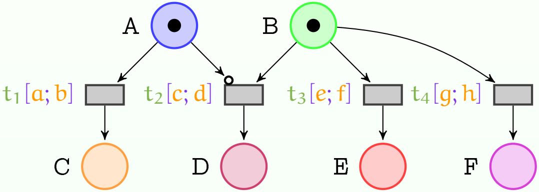
Parametric Time Petri Nets

- Constants in firing intervals replaced with parameters [Traonouez et al., 2009]



Parametric Time Petri Nets

- Constants in firing intervals replaced with parameters [Traonouez et al., 2009]



Outline

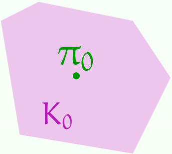
The Inverse Method

- Input
  - A PITPN $\mathcal{P}$
  - A reference valuation π_0 of all the parameters of $\mathcal{P}$

π_0

The Inverse Method

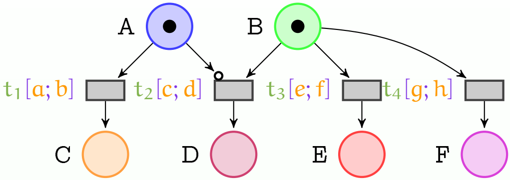
- Input
  - A PITPN $\mathcal{P}$
  - A reference valuation π_0 of all the parameters of $\mathcal{P}$
- Output: K_0
  - Convex constraint on the parameters such that
    - $\pi_0 \models K_0$
    - For all points $\pi \models K_0$, $\mathcal{P}[\pi]$ and $\mathcal{P}[\pi_0]$ have the same trace sets



The Inverse Method: General Idea

- Initially defined for timed automata [André et al., 2009]
- Extended to PITPNs [André and Garg, 2012]
- The idea
  - Exploration of the parametric state space
  - Instead of negating bad states (as in "CEGAR" approaches), remove π_0-incompatible states
  - Return the intersection of all constraints on the parameters

Application to an Example



π_0

$a = 5 \quad b = 6$

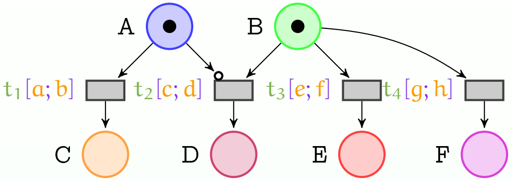
$c = 0 \quad d = 2$

$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

- Forward analysis

Application to an Example



π_0

$a = 5$ $b = 6$

$c = 0$ $d = 2$

$e = 1$ $f = 5$

$g = 6$ $h = 7$

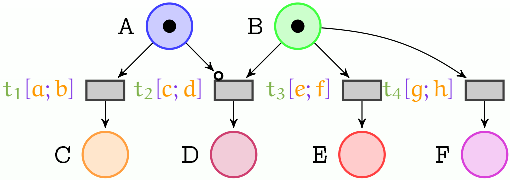
- Forward analysis

K :

true

AB	
$a \leq b$	$c \leq d$
$e \leq f$	$g \leq h$

Application to an Example



π_0

$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

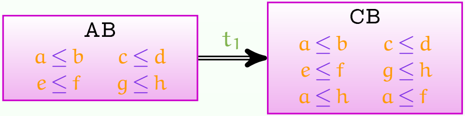
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

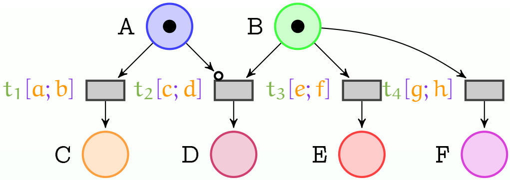
- Forward analysis

$K :$

`true`



Application to an Example



π_0

$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

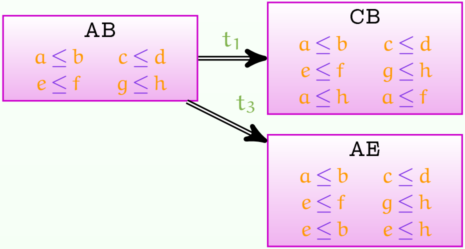
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

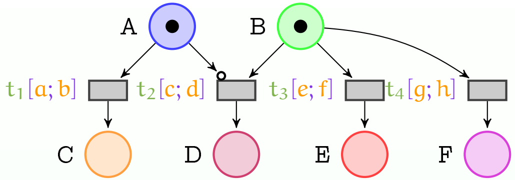
- Forward analysis

K :

`true`



Application to an Example



π_0

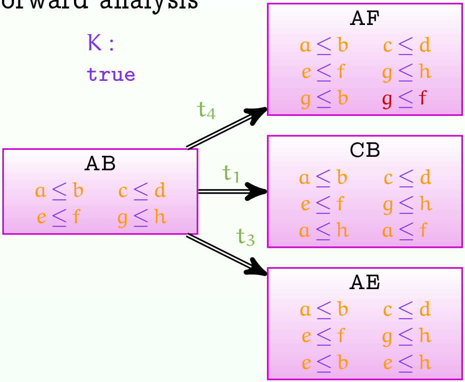
$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

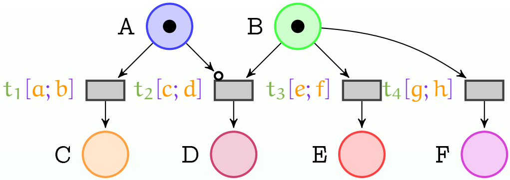
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

- Forward analysis



Application to an Example



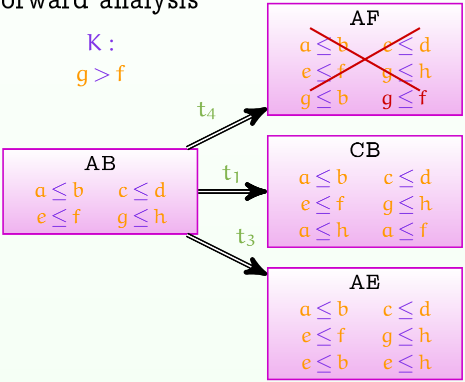
π_0

$a = 5$	$b = 6$
$c = 0$	$d = 2$
$e = 1$	$f = 5$
$g = 6$	$h = 7$

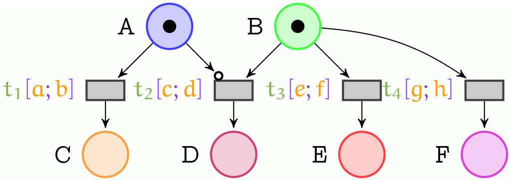
- Forward analysis

K :

$g > f$



Application to an Example



π_0

$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

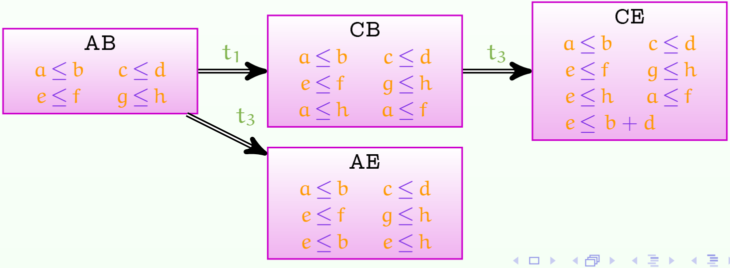
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

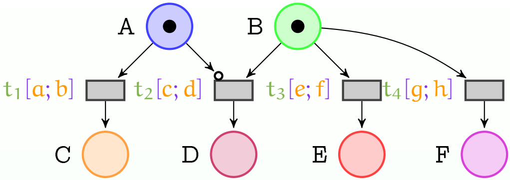
- Forward analysis

$K:$

$g > f$



Application to an Example



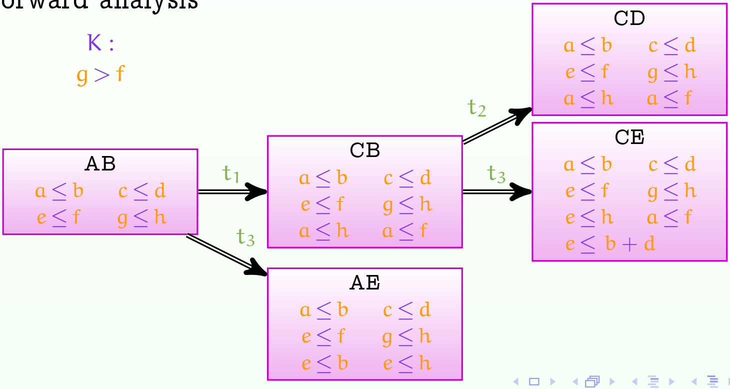
π_0

$a = 5$	$b = 6$
$c = 0$	$d = 2$
$e = 1$	$f = 5$
$g = 6$	$h = 7$

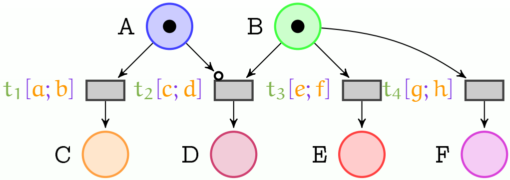
- Forward analysis

K :

$g > f$



Application to an Example



π_0

$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

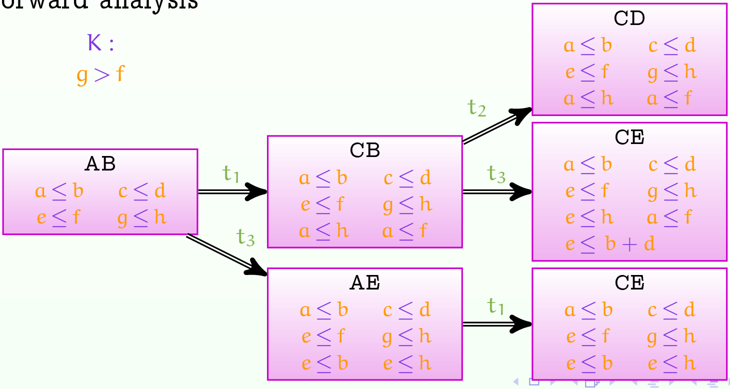
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

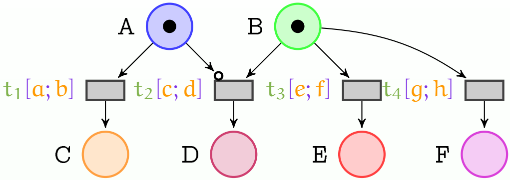
- Forward analysis

$K:$

$g > f$



Application to an Example



π_0

$a = 5 \quad b = 6$

$c = 0 \quad d = 2$

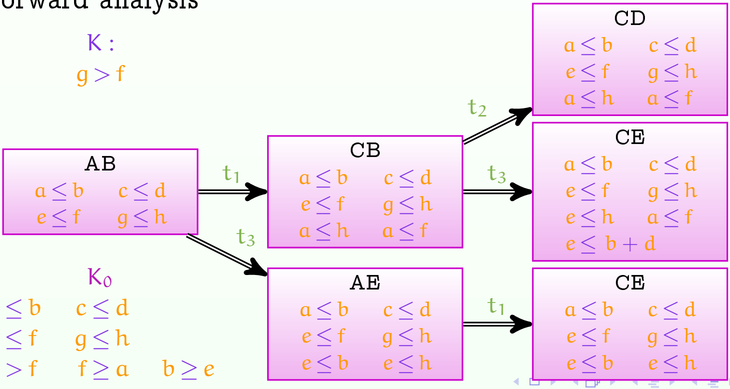
$e = 1 \quad f = 5$

$g = 6 \quad h = 7$

- Forward analysis

$K:$

$g > f$



K_0

$a \leq b \quad c \leq d$

$e \leq f \quad g \leq h$

$g > f \quad f \geq a \quad b \geq e$

Application to an Example: Interpretation

- Resulting constraint K_0

$$a \leq b \quad c \leq d \quad e \leq f \quad g \leq h$$
$$g > f \quad f \geq a \quad b \geq e$$

- Interpretation
  - For any $\pi \models K_0$, the trace set is the same as for π_0

- Remark
  - c and d do not appear within K_0 (except $c \leq d$): for any $\pi \models K_0$, the values of c and d do not influence the trace set

Application to an Example: Interpretation

- Resulting constraint K_0

$$a \leq b \quad c \leq d \quad e \leq f \quad g \leq h$$
$$g > f \quad f \geq a \quad b \geq e$$

- Interpretation
  - For any $\pi \models K_0$, the trace set is the same as for π_0
- Remark
  - c and d do not appear within K_0 (except $c \leq d$): for any $\pi \models K_0$, the values of c and d do not influence the trace set
- Application
  - Until which value can we minimize the upper bound f of t_3 (5) so that the system behavior remains the same?

Application to an Example: Interpretation

- Resulting constraint K_0

$$a \leq b \quad c \leq d \quad e \leq f \quad g \leq h$$
$$g > f \quad f \geq a \quad b \geq e$$

- Interpretation
  - For any $\pi \models K_0$, the trace set is the same as for π_0
- Remark
  - c and d do not appear within K_0 (except $c \leq d$): for any $\pi \models K_0$, the values of c and d do not influence the trace set
- Application
  - Until which value can we minimize the upper bound f of t_3 (5) so that the system behavior remains the same?
  - Due to $f \geq a$ with $a = f = 5$, one cannot decrease f
    $\rightsquigarrow$ The system is not robust w.r.t. small variations of f or a

Correctness

Theorem (Correctness)

Let $\mathcal{P}$ *be a PITPN, and* π_0 *be a reference valuation.*
Let $K_0 = IM(\mathcal{P}, \pi_0)$. *Then:*

1. $\pi_0 \models K_0$ *and*
2. $\forall \pi \models K_0$, $\mathcal{P}[\pi]$ *and* $\mathcal{P}[\pi_0]$ *have the same trace set.*

Correctness

Theorem (Correctness)

Let $\mathcal{P}$ be a PITPN, and π_0 be a reference valuation.
Let $K_0 = IM(\mathcal{P}, \pi_0)$. Then:

1. $\pi_0 \models K_0$ *and*
2. $\forall \pi \models K_0$, $\mathcal{P}[\pi]$ *and* $\mathcal{P}[\pi_0]$ *have the same trace set.*

Proof.

By induction on the length of the runs. □

Advantages

- Quantification of the system robustness
- Allows timing optimizations
- Allows the replacement of a component with another one
  - As long as the new timings satisfy K_0

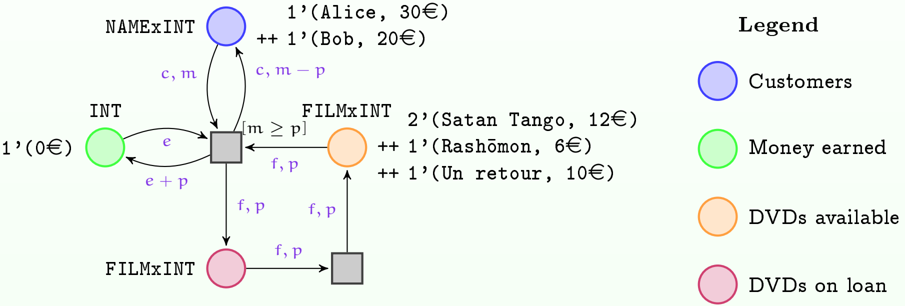
Outline

Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type (“color set”)
  - Arcs are labeled with expressions
  - Transitions can have a guard

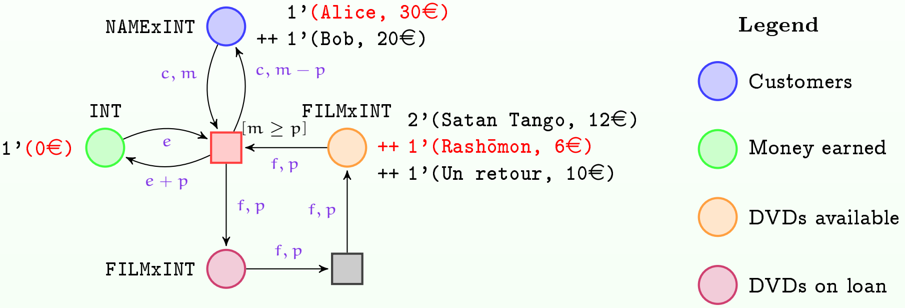
Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type ("color set")
  - Arcs are labeled with expressions
  - Transitions can have a guard
  - Example: A more complex version of the DVD renting machine



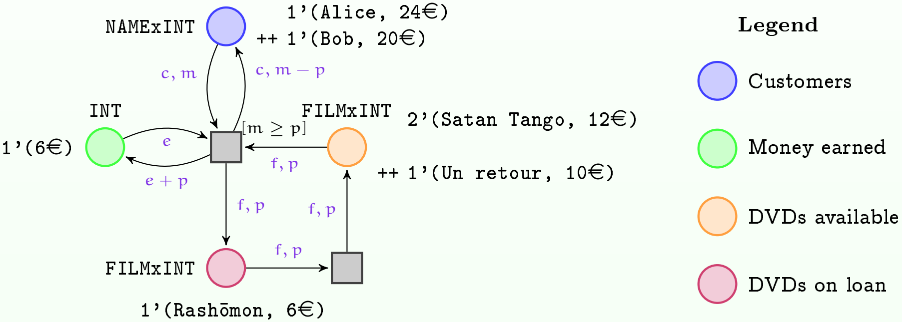
Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type (“color set”)
  - Arcs are labeled with expressions
  - Transitions can have a guard
  - Example: A more complex version of the DVD renting machine



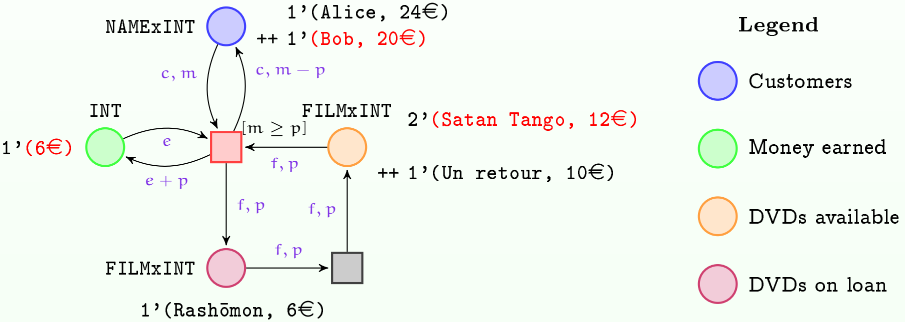
Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
    - Tokens and places have a type ("color set")
    - Arcs are labeled with expressions
    - Transitions can have a guard
    - Example: A more complex version of the DVD renting machine



Perspectives (1/2)

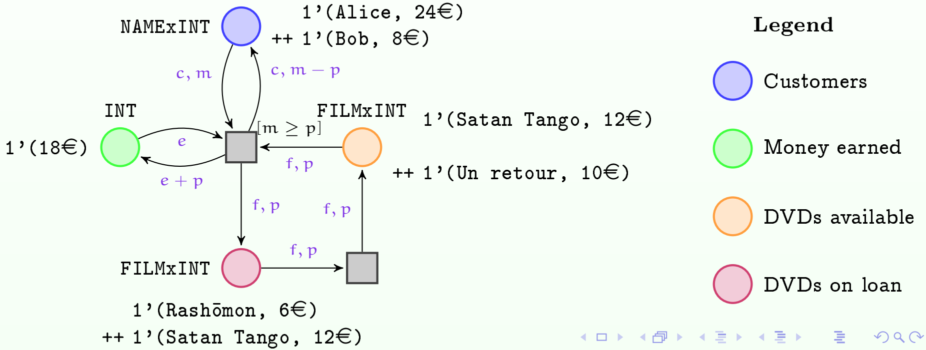
- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type ("color set")
  - Arcs are labeled with expressions
  - Transitions can have a guard
  - Example: A more complex version of the DVD renting machine



Perspectives (1/2)

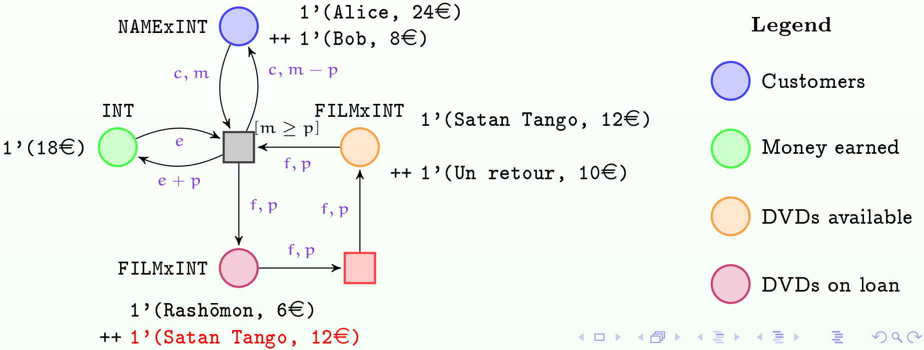
- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type ("color set")
  - Arcs are labeled with expressions
  - Transitions can have a guard

  - Example: A more complex version of the DVD renting machine



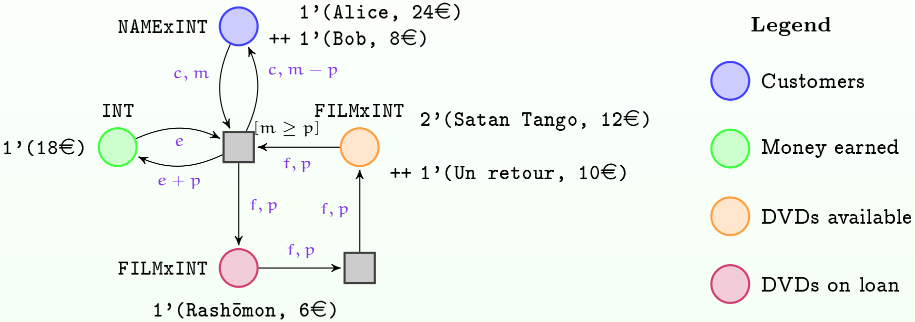
Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
    - Tokens and places have a type ("color set")
    - Arcs are labeled with expressions
    - Transitions can have a guard
    - Example: A more complex version of the DVD renting machine



Perspectives (1/2)

- Extension to colored Petri nets [Jensen and Kristensen, 2009]
  - Tokens and places have a type ("color set")
  - Arcs are labeled with expressions
  - Transitions can have a guard
  - Example: A more complex version of the DVD renting machine



Perspectives (2/2)

- Termination
  - Probably does not terminate in the general case
  - . . . but no example exhibited so far

- Implementation
  - To do

- Modular analysis
  - Combine the inverse method with modular state space exploration for timed Petri nets [Lakos and Petrucci, 2007]
  - Idea: apply the inverse method to separate modules, then combine the result
  - Challenge: identify subclasses of time(d) Petri nets such that this applies

References I

André, É., Chatain, Th., Encrenaz, E., and Fribourg, L. (2009).
An inverse method for parametric timed automata.
International Journal of Foundations of Computer Science, 20(5):819–836.

André, É. and Garg, S. (2012).
Robustness analysis of time Petri nets.
In *NWPT'12*, Bergen, Norway.

Jensen, K. and Kristensen, L. M. (2009).
Coloured Petri Nets – Modelling and Validation of Concurrent Systems.
Springer.

Lakos, C. and Petrucci, L. (2007).
Modular state space exploration for timed Petri nets.
Journal of Software Tools for Technology Transfer, 9(3-4):393–411.

Markey, N. (2011).
Robustness in real-time systems.
In *SIES'11*, pages 28–34, Västerås, Sweden. IEEE Computer Society Press.

Merlin, P. M. (1974).
A study of the recoverability of computing systems.
PhD thesis, University of California, Irvine.

References II

Petri, C. A. (1962).
Kommunikation mit Automaten.
PhD thesis, Darmstadt University of Technology, Germany.

Traonouez, L.-M., Lime, D., and Roux, O. H. (2009).
Parametric model-checking of stopwatch Petri nets.
Journal of Universal Computer Science, 15(17):3273–3304.

The Algorithm

Algorithm 1: $IM(\mathcal{P}, \pi_0)$

$i \leftarrow 0$; $K \leftarrow K_{init}$; $C \leftarrow \{\mathbf{c}_0\}$
while `true` **do**
    while $\exists$ π_0-*incompatible classes in* C **do**
        Select a π_0-incompatible class (M, D) of C
        Select a π_0-incompatible J in $D{\downarrow_P}$
        $K \leftarrow K \wedge \neg J$
        $C \leftarrow \bigcup_{j=0}^{i} Post^{j}_{\mathcal{P}(K)}(\{\mathbf{c}_0\})$
    if $Post_{\mathcal{P}(K)}(C) \subseteq C$ **then**
        return $K_0 \leftarrow \bigcap_{(M,D) \in C} D{\downarrow_P}$
    $i \leftarrow i + 1$; $C \leftarrow C \cup Post_{\mathcal{P}(K)}(C)$

License