Towards reasoning about concurrency: a logical approach

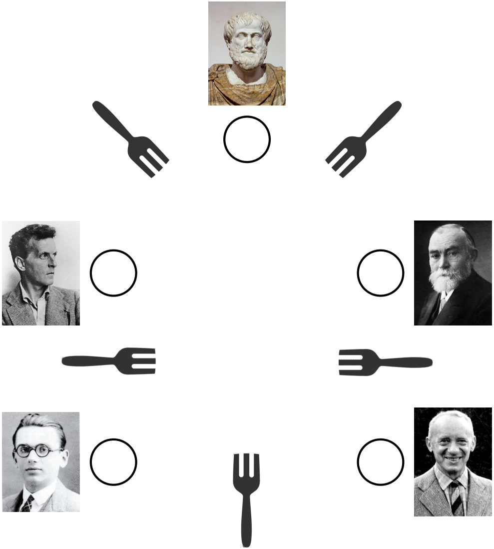
XIII Seminário Informal(, mas Formal!)

Bruno Lopes

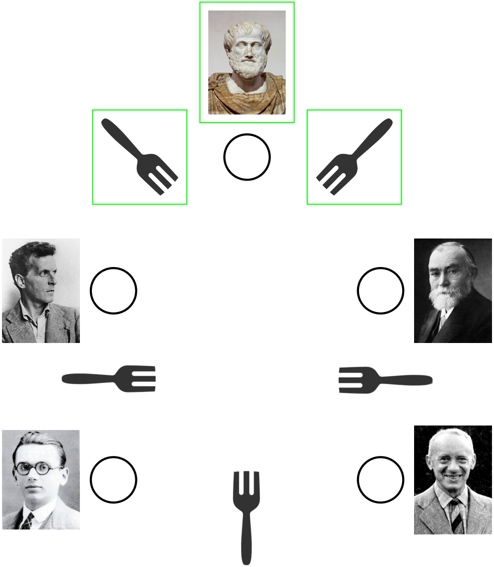
FRAME lab.
Instituto de Computação
Universidade Federal Fluminense

January, 2016

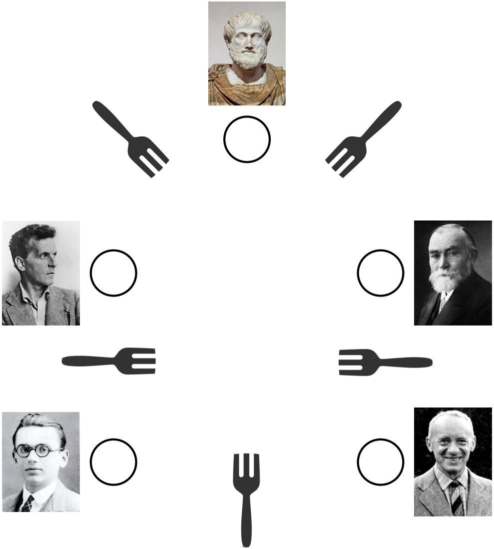
Instituto de Computação



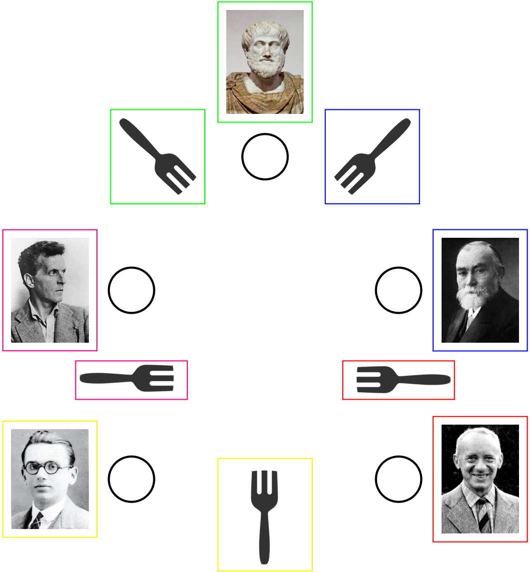
E. Dijkstra & C. A. R. Hoare (1965)



E. Dijkstra & C. A. R. Hoare (1965)



E. Dijkstra & C. A. R. Hoare (1965)



E. Dijkstra & C. A. R. Hoare (1965)

Instituto de Computação uff

Petri nets

C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Petri nets

C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

Petri nets

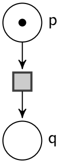
C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

- Place
- Transition
- Tokens
- Edge

Firing



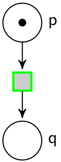
Petri nets

C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

Firing



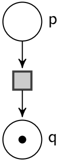
Petri nets

C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

Firing



Petri nets

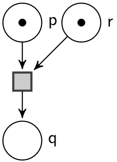
C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

- Place
- Transition
- Tokens
- Edge

Firing



Petri nets

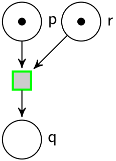
C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

- Place
- Transition
- Tokens
- Edge

Firing



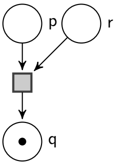
Petri nets

C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

Firing



Instituto de Computação

Petri nets

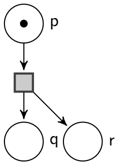
C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

- Place
- Transition
- Tokens
- Edge

Firing



Petri nets

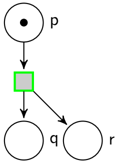
C. A. Petri (1939)

A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

- ○ Place
- □ Transition
- ∴ Tokens
- → Edge

Firing



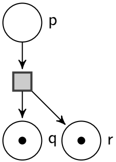
Petri nets

C. A. Petri (1939)

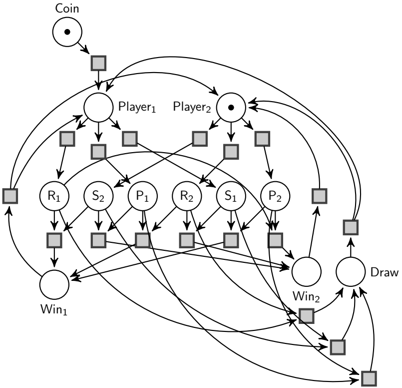
A bipartite graph with two types of nodes: **places** and **transitions**.

Elements

Firing



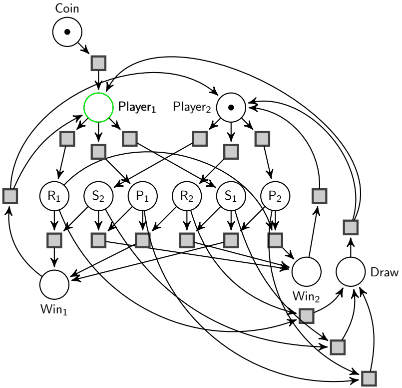
Petri nets: usage example



Modelling

Instituto de Computação

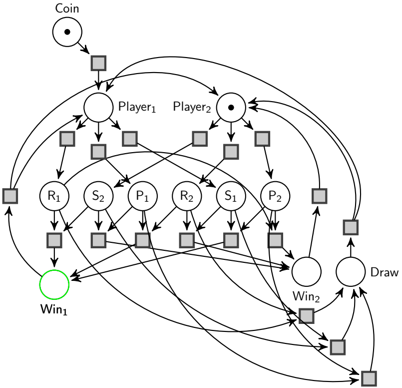
Petri nets: usage example



Modelling

Once a coin is inserted in a supposed machine, “Player 1” will able to begin his game.

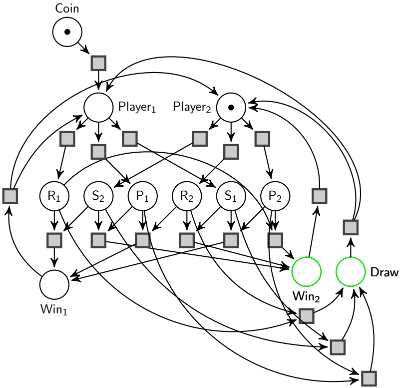
Petri nets: usage example



Modelling

If the user wins, a token will be placed at "Win$_1$" and the user will be able to play again.

Petri nets: usage example



Modelling

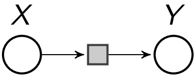
If he loses, a token will be placed at "Win$_2$" or the game restarts if there is a draw match.

Petri nets: reasoning challenges

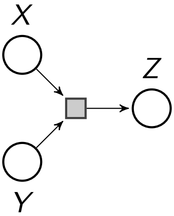
- [✗] State explosion
- [✗] Undecidability
- [✗] Incompleteness

Petri nets model

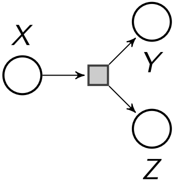
Type 1:



Type 2:



Type 3:

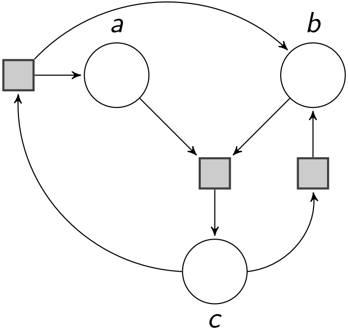


E. S. de Almeida & E. H. Haeusler (1999)

Instituto de Computação uff

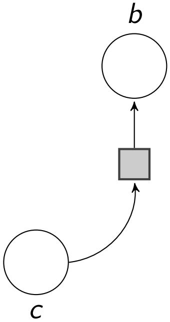
As an example...

For a Petri net with the three types of transitions



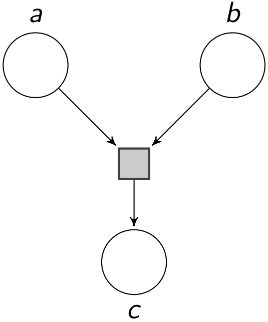
As an example...

We may decompose the Type 1 transition



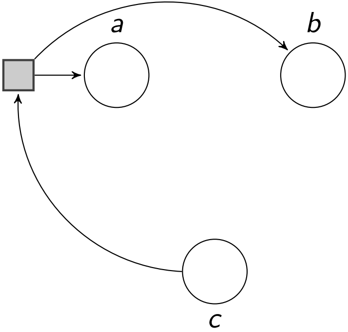
As an example...

The Type 2 transition



As an example...

And the Type 3 transition



Instituto de Computação

Propositional Dynamic Logic

PDL
Is a multi-modal logic used for specifying and reasoning on sequential programs. It uses one modality $\langle\pi\rangle$ for each program π.

Propositional Dynamic Logic

PDL

Is a multi-modal logic used for specifying and reasoning on sequential programs. It uses one modality $\langle\pi\rangle$ for each program π.

Language

Syntax: Let p be an atomic proposition and α a basic program

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \boxed{\langle\pi\rangle}\varphi$$
$$\pi ::= \alpha \mid \pi;\pi \mid \pi \cup \pi \mid \pi^\star$$

Propositional Dynamic Logic

PDL

Is a multi-modal logic used for specifying and reasoning on sequential programs. It uses one modality $\langle\pi\rangle$ for each program π.

Language

Syntax: Let p be an atomic proposition and α a basic program

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \boxed{\langle\pi\rangle}\varphi \quad \rightsquigarrow \text{"generator"}$$

$$\pi ::= \alpha \mid \pi ; \pi \mid \pi \cup \pi \mid \pi^\star$$

Usage example

```
if p then
    α;
    β;
    while q do
        β;
    end
end
```

Instituto de Computação

Usage example

if p **then**
    α;
    β;
    while q **do**
        β;
    end
end

Modelled in PDL

$p \to [\alpha;\beta](q \to [\beta^\star]\neg q)$

Instituto de Computação

Petri nets

[✔] Native support to concurrence

[✔] Intuitive graphical interpretation

Propositional Dynamic Logic

[✔] Formal system to verify properties in programs

[✔] Deductive systems

Petri nets

[✔] Native support to concurrence

[✔] Intuitive graphical interpretation

Propositional Dynamic Logic

[✔] Formal system to verify properties in programs

[✔] Deductive systems

Our approach

Unify these formalisms!

Petri-PDL

PDL Language

Syntax: Let p be an atomic proposition and α a basic program

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\pi\rangle\varphi$$

$$\pi ::= \alpha \mid \pi;\pi \mid \pi \cup \pi \mid \pi^\star$$

Petri-PDL

PDL Language

Syntax: Let p be an atomic proposition and α a basic program

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle \pi \rangle \varphi$$

$$\pi ::= \alpha \mid \pi ; \pi \mid \pi \cup \pi \mid \pi^{\star}$$

Instituto de computação uff

Petri-PDL

Petri-PDL Language

Syntax: Let p be an atomic proposition and α a basic program

$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle \boxed{\pi} \rangle \varphi \quad \rightsquigarrow \text{now a Petri net}$$

$$\pi ::= \pi \odot \pi \mid \eta$$

Petri-PDL

PDL Language

Syntax: Let p be an atomic proposition and α a basic program

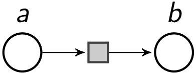
$$\varphi ::= p \mid \top \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle \boxed{s, \pi} \rangle\varphi \quad \rightsquigarrow \text{marked!}$$

$$\pi ::= \pi \odot \pi \mid \eta$$

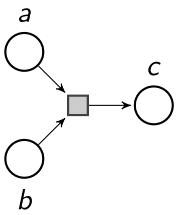
s : a sequence of names

Basic Petri nets

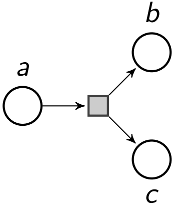
Type 1: at_1b



Type 2: a,bt_2c



Type 3: at_3b,c



Firing function

T_1

$$f(s, at_1b) = \begin{cases} s_1bs_2, & \text{if } s = s_1as_2 \\ \epsilon, & \text{if } a \not\prec s \end{cases}$$

T_3

$$f(s, at_3bc) = \begin{cases} s_1s_2bc, & \text{if } s = s_1as_2 \\ \epsilon, & \text{if } a \not\prec s \end{cases}$$

T_2

$$f(s, abt_2c) = \begin{cases} s_1cs_2s_3, & \text{if } s = s_1as_2bs_3 \\ \epsilon, & \text{if } a, b \not\prec s \end{cases}$$

ϵ

$$f(\epsilon, \pi) = \epsilon$$

Axiomatic System

(PL) Enough propositional logic tautologies

(K) $[s,\pi](p \to q) \to ([s,\pi]p \to [s,\pi]q)$

(Du) $[s,\pi]p \leftrightarrow \neg\langle s,\pi\rangle\neg p$

(Sub) If $\Vdash \varphi$, then $\Vdash \varphi^\sigma$, where σ uniformly substitutes proposition symbols by arbitrary formulas.

(MP) If $\Vdash \varphi$ and $\Vdash \varphi \to \psi$, then $\Vdash \psi$.

(Gen) If $\Vdash \varphi$, then $\Vdash [s,\pi]\varphi$.

(PC) $\langle s,\eta\rangle\varphi \leftrightarrow \langle s,\eta_1\rangle\langle s_1,\eta\rangle\varphi \vee \langle s,\eta_2\rangle\langle s_2,\eta\rangle\varphi \vee \cdots \vee \langle s,\eta_n\rangle\langle s_n,\eta\rangle\varphi$,
where $s_i = f(s,\eta_i)$, for all $1 \le i \le n$

(R_ϵ) $\langle s,\eta\rangle\varphi \leftrightarrow \varphi$, if $f(s,\eta) = \epsilon$

Axiomatic System

(PL) Enough propositional logic tautologies

(K) $[s,\pi](p \to q) \to ([s,\pi]p \to [s,\pi]q)$

(Du) $[s,\pi]p \leftrightarrow \neg\langle s,\pi\rangle\neg p$

(Sub) If $\Vdash \varphi$, then $\Vdash \varphi^{\sigma}$, where σ uniformly substitutes proposition symbols by arbitrary formulas.

(MP) If $\Vdash \varphi$ and $\Vdash \varphi \to \psi$, then $\Vdash \psi$.

(Gen) If $\Vdash \varphi$, then $\Vdash [s,\pi]\varphi$.

(PC) $\langle s,\eta\rangle\varphi \leftrightarrow \langle s,\eta_1\rangle\langle s_1,\eta\rangle\varphi \vee \langle s,\eta_2\rangle\langle s_2,\eta\rangle\varphi \vee \cdots \vee \langle s,\eta_n\rangle\langle s_n,\eta\rangle\varphi$,
where $s_i = f(s,\eta_i)$, for all $1 \leq i \leq n$

(R_ϵ) $\langle s,\eta\rangle\varphi \leftrightarrow \varphi$, if $f(s,\eta) = \epsilon$

Instituto de Computação uff

Instituto de Computação

Axiomatic System

(PL) Enough propositional logic tautologies

(K) $[s,\pi](p \to q) \to ([s,\pi]p \to [s,\pi]q)$

(Du) $[s,\pi]p \leftrightarrow \neg\langle s,\pi\rangle\neg p$

(Sub) If $\Vdash \varphi$, then $\Vdash \varphi^\sigma$, where σ uniformly substitutes proposition symbols by arbitrary formulas.

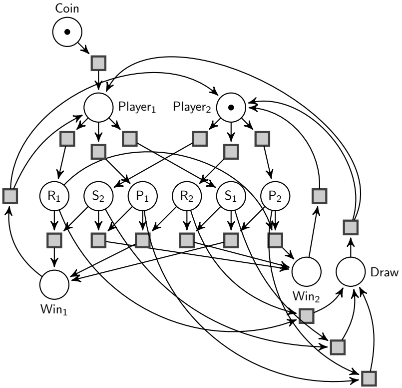
(MP) If $\Vdash \varphi$ and $\Vdash \varphi \to \psi$, then $\Vdash \psi$.

(Gen) If $\Vdash \varphi$, then $\Vdash [s,\pi]\varphi$.

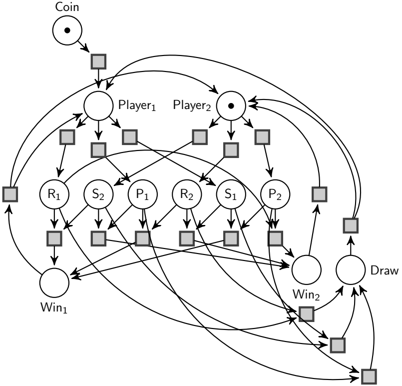
(PC) $\langle s,\eta\rangle\varphi \leftrightarrow \langle s,\eta_1\rangle\langle s_1,\eta\rangle\varphi \vee \langle s,\eta_2\rangle\langle s_2,\eta\rangle\varphi \vee \cdots \vee \langle s,\eta_n\rangle\langle s_n,\eta\rangle\varphi$, $\rightsquigarrow$ firing
where $s_i = f(s,\eta_i)$, for all $1 \leq i \leq n$

($\mathbf{R}_\epsilon$) $\langle s,\eta\rangle\varphi \leftrightarrow \varphi$, if $f(s,\eta) = \epsilon$ $\rightsquigarrow$ stop

Usage example

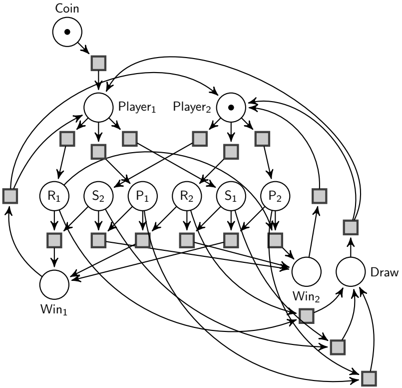


Usage example



Petri-PDL formula:

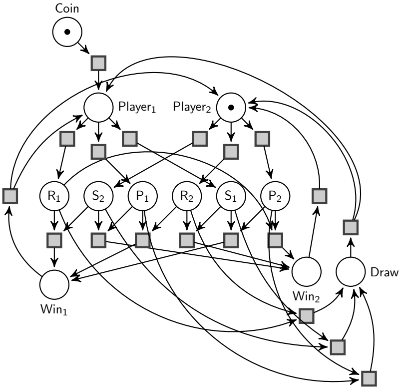
Usage example



Petri-PDL formula:

$\langle(\mathsf{Coin},\mathsf{Player}_2), \mathsf{Coin}\, t_1 \mathsf{Player}_1 \odot \Upsilon\rangle\psi$.

Usage example



Petri-PDL formula:

$\langle(\text{Coin},\text{Player}_2),\text{Coin}t_1\text{Player}_1\rangle\langle f((\text{Coin},\text{Player}_2),\text{Coin}t_1\text{Player}_1),\text{Coin}t_1\text{Player}_1\odot\Upsilon\rangle\psi.$

Instituto de Computação uff

Petri-PDL model

Frame: $\mathcal{F} = \langle W, R_\pi, M \rangle$

- W is a non-empty set of states
- R_π is a binary relation on W for each program π
- M is a function $M\colon W \to S$ that returns a sequence of names for each state

A labeled Natural Deduction

$$\dfrac{\dfrac{\dfrac{\{w : [s,\pi](p \to q)\}^3, \{w : [s,\pi]p\}^2 \qquad \dfrac{\dfrac{\{w : [s,\pi](p \to q)\}^1 \; \{wR_\pi u\}^1}{u : p \to q}\,\pi\Box_e \qquad \dfrac{\{w : [s,\pi]p\}^1 \; \{wR_\pi u\}^1}{u : p}\,\pi\Box_e}{u : q}\,\to_e}{w : [s,\pi]q}\,\pi\Box_i^1}{w : [s,\pi]p \to [s,\pi]q}\,\to_i^2}{w : [s,\pi](p \to q) \to ([s,\pi]p \to [s,\pi]q)}\,\to_i^3$$

Anti-Prenex Normal Form

APNF

The modalities are moved inwards a formula and only applied to modal literals.

Anti-Prenex Normal Form

APNF

The modalities are moved inwards a formula and only applied to modal literals.

A formula χ is in Anti-Prenex Normal Form (APNF) if, and only if

Let φ and ψ be formula in the language of Petri-PDL.

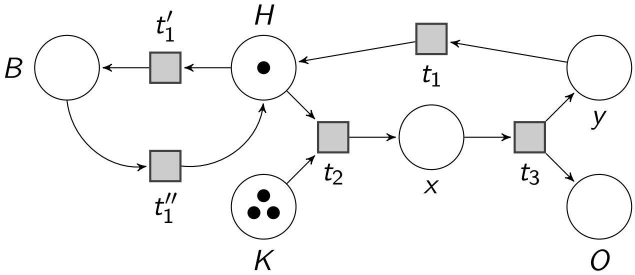
1. χ is a modal term; or
2. χ is of the form $(\varphi \wedge \psi)$, $(\varphi \vee \psi)$, or $(\varphi \to \psi)$, and φ and ψ are in APNF;
3. χ is of the form $[s, \pi]\varphi$, φ is disjunctive, and φ is in APNF; or
4. χ is of the form $\langle s, \pi\rangle\varphi$, φ is conjunctive, and φ is in APNF.

Divided Separated Normal Form for Petri-PDL

PPDL
Separates the contexts

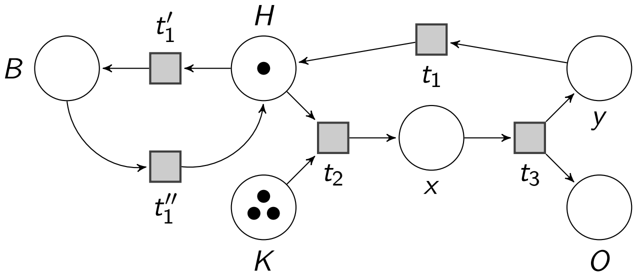
- formulae which are true only at the initial state
- formulae which are true in all states

Resolution based calculus



An RPG game

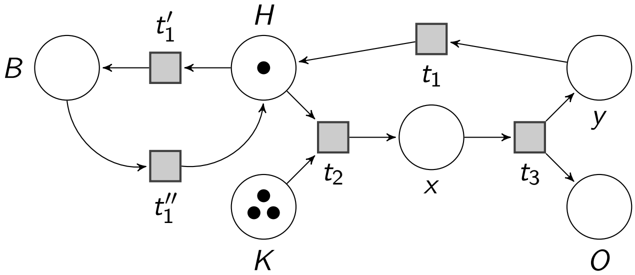
Resolution based calculus



An RPG game

A player walks through scenarios, taking a key (token in K) in his hand (a token in H) to open doors.

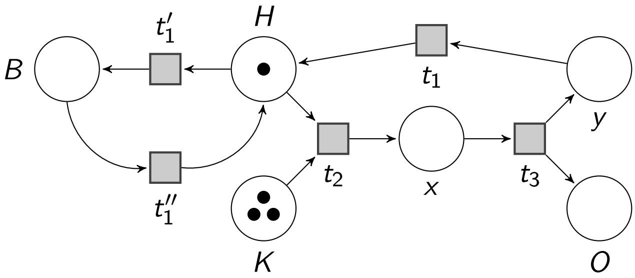
Resolution based calculus



An RPG game

If his hand is busy (a token in B) he can not open the door.

Resolution based calculus



An RPG game

Is it possible that after three rounds the player has opened one door, has a free hand and still has two keys to continue?

Instituto de Computação uff

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
4.	$\neg p_2 \vee \neg[(KKyO), \pi]\neg p_3$	$[\mathcal{U}]$
5.	$\neg p_3 \vee \neg p$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
4.	$\neg p_2 \vee \neg[(KKyO), \pi]\neg p_3$	$[\mathcal{U}]$
5.	$\neg p_3 \vee \neg p$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
4.	$\neg p_2 \vee \neg[(KKyO), \pi]\neg p_3$	$[\mathcal{U}]$
5.	$\neg p_3 \vee \neg p$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$

$$\text{ser2}\quad \frac{\begin{array}{l} D \vee \neg[s,\pi]l \in \mathcal{U} \\ l_1 \vee \cdots \vee l_n \vee l \in \mathcal{U} \end{array}}{D \vee \neg[s,\pi]\neg l_1 \vee \cdots \vee \neg[s,\pi]\neg l_n \in \mathcal{U}}$$

Instituto de Computação

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
7.	$\neg p_2 \vee \neg[KKyO, \pi]p$	$[\mathcal{U}], (ser2), (4, 5)$

Instituto de Computação uff

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
7.	$\neg p_2 \vee \neg[KKyO, \pi]p$	$[\mathcal{U}], (ser2), (4, 5)$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
3.	$\neg p_1 \vee \neg[(KKx), \pi]\neg p_2$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
7.	$\neg p_2 \vee \neg[KKyO, \pi]p$	$[\mathcal{U}], (ser2), (4,5)$

$$
\begin{array}{llllll}
\text{comp} & & D & \vee & \neg[s,\pi]l & \in \mathcal{U} \\
\text{if } \eta \subseteq \pi \text{ and} & & D' & \vee & [f(s,\pi_b),\eta]l & \in \mathcal{U} \\
\hline
\text{for any } \pi_b \in \pi, & & D & \vee & \neg[s,\pi]\neg D' & \in \mathcal{U}
\end{array}
$$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
8.	$\neg p_1 \vee \neg[KKx, \pi]p$	$[\mathcal{U}], (comp), (7, 3)$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
8.	$\neg p_1 \vee \neg[KKx, \pi]p$	$[\mathcal{U}], (comp), (7, 3)$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
2.	$\neg p_0 \vee \neg[(KKKH), \pi]\neg p_1$	$[\mathcal{U}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
8.	$\neg p_1 \vee \neg[KKx, \pi]p$	$[\mathcal{U}], (comp), (7, 3)$

$$\begin{array}{llll} \text{comp} & D \vee \neg[s,\pi]l & \in \mathcal{U} \\ \text{if } \eta \subseteq \pi \text{ and} & D' \vee [f(s,\pi_b),\eta]l & \in \mathcal{U} \\ \hline \text{for any } \pi_b \in \pi, & D \vee \neg[s,\pi]\neg D' & \in \mathcal{U} \end{array}$$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
9.	$\neg p_0 \vee \neg[KKKH, \pi]p$	$[\mathcal{U}], (comp), (8, 2)$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
9.	$\neg p_0 \vee \neg[KKKH, \pi]p$	$[\mathcal{U}], (comp), (8, 2)$

Instituto de Computação

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
6.	$\neg p_0 \vee [KKHO, \pi]p$	$[\mathcal{U}]$
9.	$\neg p_0 \vee \neg[KKKH, \pi]p$	$[\mathcal{U}], (comp), (8, 2)$

$$\text{ures} \quad \frac{\begin{array}{cccc} D & \vee & m & \in \mathcal{U} \\ D' & \vee & \neg m & \in \mathcal{U} \end{array}}{\begin{array}{cccc} D & \vee & D' & \in \mathcal{U} \end{array}}$$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
10.	$\neg p_0$	$[\mathcal{U}], (ures), (9, 6)$

Instituto de Computação

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
10.	$\neg p_0$	$[\mathcal{U}], (ures), (9, 6)$

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

1.	p_0	$[\mathcal{I}]$
10.	$\neg p_0$	$[\mathcal{U}], (ures), (9, 6)$

$$\text{ires} \quad \frac{\begin{array}{lll} C \vee l & \in \mathcal{I} \cup \mathcal{U} \\ C' \vee \neg l & \in \mathcal{I} \end{array}}{C \vee C' \quad \in \mathcal{I}}$$

Instituto de Computação

Resolution based calculus

An RPG game

Modelling in Petri-PDL and applying APNF and DSNF we have

$$11. \quad \bot \quad [\mathcal{I}], (ires), (10, 1)$$

Instituto de Computação

Marked Petri nets characteristics

- ✗ There is no way to control fire rate
- ✗ There is no way to model different timings

Instituto de Computação

Marked Petri nets characteristics

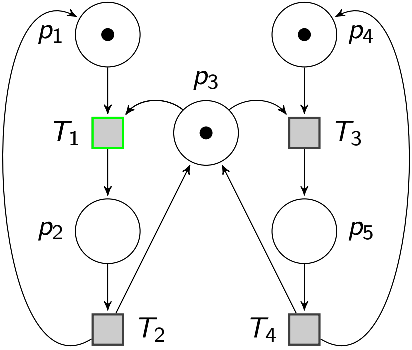
✗ There is no way to control fire rate

✗ There is no way to model different timings

Extend Petri net model
Stochastic Petri nets

An example...

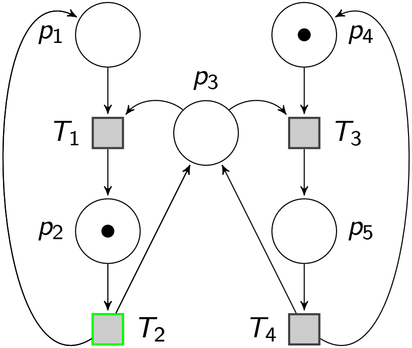
Two processes: I/O bound and CPU bound



Instituto de Computação uff

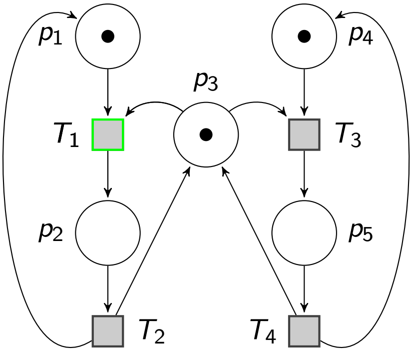
An example. . .

Two processes: I/O bound and CPU bound



An example...

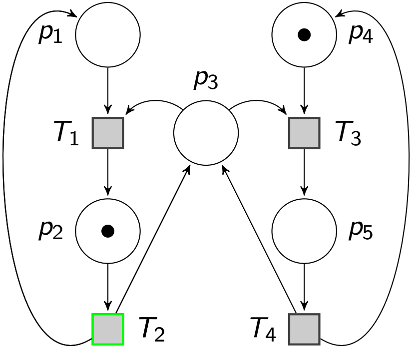
Two processes: I/O bound and CPU bound



Instituto de Computação uff

An example...

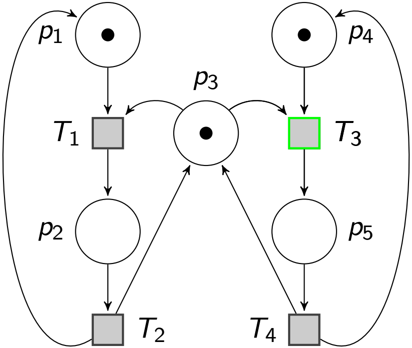
Two processes: I/O bound and CPU bound



Instituto de Computação

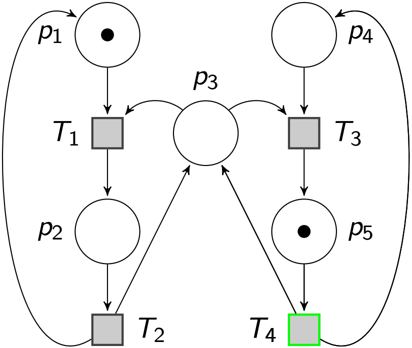
An example...

Two processes: I/O bound and CPU bound



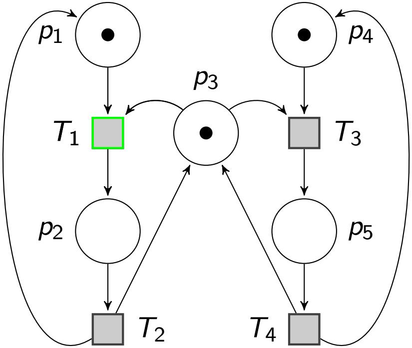
An example...

Two processes: I/O bound and CPU bound



An example...

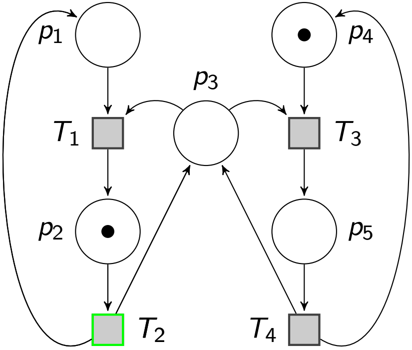
Two processes: I/O bound and CPU bound



Instituto de Computação uff

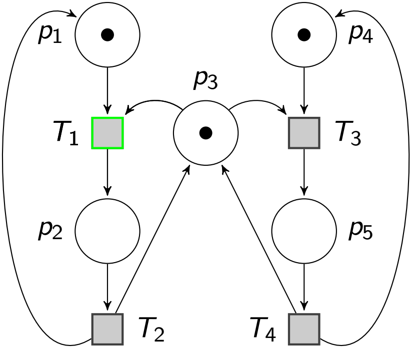
An example...

Two processes: I/O bound and CPU bound



An example...

Two processes: I/O bound and CPU bound



$\mathcal{DS}_3$ *model*

Frame: $\mathcal{F} = \langle W, R_\pi, M, (\Pi, \Lambda), \delta \rangle$

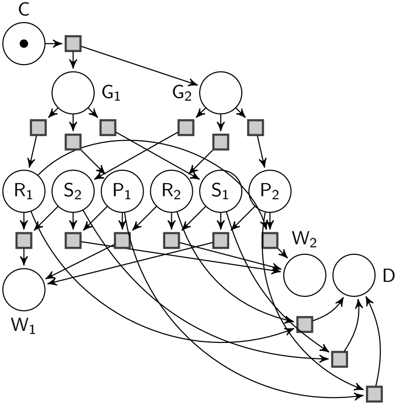
- W is a non-empty set of states
- R_π is a binary relation on W for each program π
- M is a function $M\colon W \to S$ that returns a sequence of names for each state
- Π a stochastic Petri net program
- Λ a function $\Lambda\colon \Pi \to \mathbb{R}^+$
- δ a delay function $\delta\colon W \times \Pi \to \vec{\mathbb{R}^+}$

Truth probability of a modality

$\mathcal{M}_3, w \Vdash \langle s, \pi_b \rangle \varphi$

$$\Pr(\mathcal{M}_3, w \Vdash \langle s, \pi_b \rangle \varphi \mid \delta(w, \Pi)) = \frac{\delta(w, \pi_b)}{\sum\limits_{\pi_b \in \Pi : f(s, \pi_b) \neq \epsilon} \delta(w, \pi_b)}$$

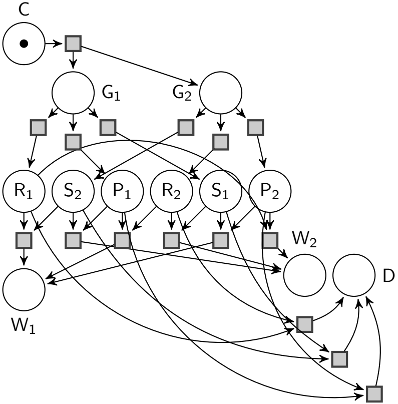
Rock-Paper-Scissors



Π:

$ct_3g_1g_2 \odot g_1t_1r_1 \odot g_1t_1s_1 \odot g_1t_1p_1 \odot g_2t_1r_2 \odot g_2t_1s_2 \odot g_2t_1p_2 \odot r_1s_2t_2w_1 \odot r_1p_2t_2w_2 \odot r_1r_2t_2d \odot s_1r_2t_2w_2 \odot s_1s_2t_2d \odot s_1p_2t_2w_1 \odot p_1r_2t_2w_1 \odot p_1s_2t_2w_2 \odot p_1p_2t_2d.$

Rock-Paper-Scissors



Π:
Does it always have a winner?

Petri-PDL model checker

```
mod PETRI-PDL is
  sort Place Places BasicProg Prog Net .
  subsort Place < Places .
  subsort BasicProg < Prog .

  op __ : Places Places -> Places [prec 20 assoc comm id: epsilon] .
  op _t1_ : Place Place -> BasicProg [prec 30] .
  op __t2_ : Place Place Place -> BasicProg [prec 30] .
  op _t3__ : Place Place Place -> BasicProg [prec 30] .
  op _+_ : Prog Prog -> Prog [assoc comm prec 40] .
  op _,_ : Places Prog -> Net .

  vars A B C : Place . var W : Places . var P : Prog .

  rl [t1] : A W , A t1 B => B W , A t1 B .
  rl [t2] : A B W , A B t2 C => C W , A B t2 C .
  rl [t3] : A W , A t3 B C => B C W , A t3 B C .

  rl [t1] : A W , A t1 B + P => B W , A t1 B + P .
  rl [t2] : A B W , A B t2 C + P => C W , A B t2 C + P .
  rl [t3] : A W , A t3 B C + P => B C W , A t3 B C + P .
endm
```

Model checking "Rock-Paper-Scissors"

```
mod VALUATION is
  inc PETRI−PDL−MODEL−CHECKER .
  ops c g1 g2 s1 s2 r1 r2 p1 p2 w1 w2 d : −> Place .
  ops p q : −> Formula .
  eq valuation(w1) = p . eq valuation(w2) = q . eq valuation(d) = ((¬ p) (¬ q)
        ) .
endm
```

```
reduce in VALUATION : modelCheck(¬ < c,(g1 t1 r1 + g1 t1 p1 + g1 t1 s1 +
      g2 t1 r2 +
 g2 t1 p2 + g2 t1 s2 + (((((((
    s1 s2 t2 d + s1 p2 t2 w1) + s1 r2 t2 w2) + p1 s2 t2 w2) + p1 p2 t2 d) +
          p1 r2 t2 w1) + r1 s2 t2 w1) + r1 p2
    t2 w2) + r1 r2 t2 d) + c t3 g1 g2 > (¬ (p \/ q)), 4, mt−placeslistset) .
rewrites: 1139 in 24ms cpu (25ms real) (45942 rewrites/second)
result PPDLModel: ppdlModel(false, c −> g1 g2 −> g1 s2 −> s1 s2 −> d)
```

Instituto de Computação uff

Model checking "Rock-Paper-Scissors"

```
mod VALUATION is
  inc PETRI−PDL−MODEL−CHECKER .
  ops c g1 g2 s1 s2 r1 r2 p1 p2 w1 w2 d : −> Place .
  ops p q : −> Formula .
  eq valuation(w1) = p . eq valuation(w2) = q . eq valuation(d) = ((¬ p) (¬ q) ) .
endm
```

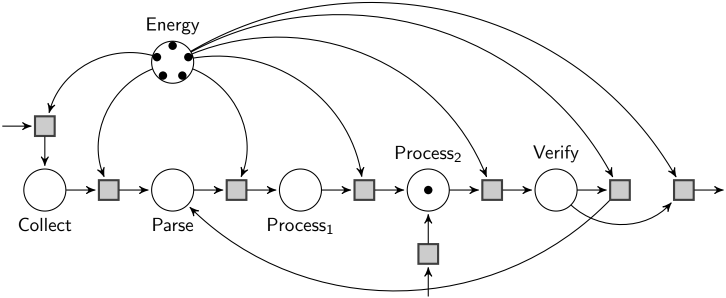
```
reduce in VALUATION : modelCheck(¬ < c,(g1 t1 r1 + g1 t1 p1 + g1 t1 s1 + g2 t1 r2 +
 g2 t1 p2 + g2 t1 s2 + (((((((
    s1 s2 t2 d + s1 p2 t2 w1) + s1 r2 t2 w2) + p1 s2 t2 w2) + p1 p2 t2 d) + p1 r2 t2 w1) + r1 s2 t2 w1) + r1 p2
    t2 w2) + r1 r2 t2 d) + c t3 g1 g2 > (¬ (p \/ q)), 4, mt−placeslistset) .
rewrites: 1139 in 24ms cpu (25ms real) (45942 rewrites/second)
result PPDLModel: ppdlModel(false, c −> g1 g2 −> g1 s2 −> s1 s2 −> d)
```

No!

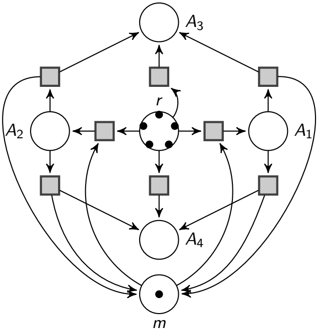
Gives a counterexample

Instituto de Computação

Example: a SPN model for agents

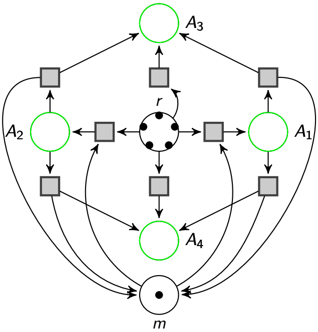


Example: a multi-agent scenario



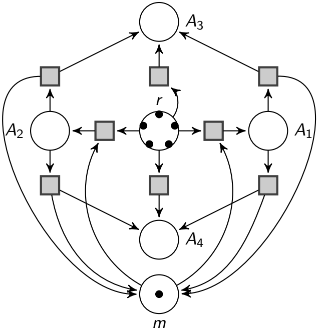
Scenario

Example: a multi-agent scenario



Scenario
A_1, A_2, A_3 and A_4 are agents that must collect and process some data from the resource centre r.

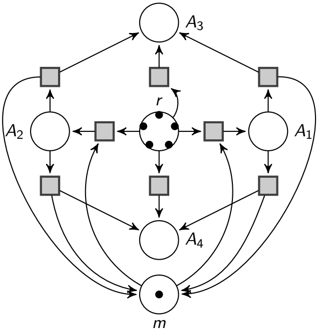
Example: a multi-agent scenario



Scenario

A_1 and A_2 can not make the full process and needs that A_3 or A_4 completes the computation.

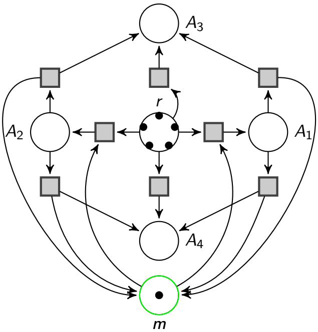
Example: a multi-agent scenario



Scenario

A_3 and A_4 have a faster processor then A_1 and A_2.

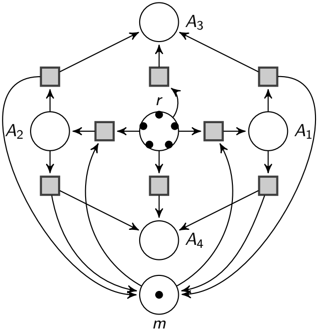
Example: a multi-agent scenario



Scenario

A_1 and A_2 are in a shared memory system, but the clock of the processor of A_1 is faster then A_2.

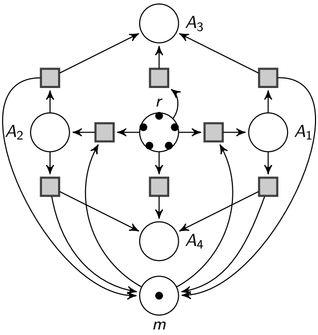
Example: a multi-agent scenario



Formalizing

Controlling the clock difference:

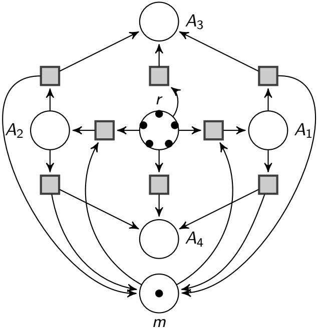
Example: a multi-agent scenario



Formalizing

Controlling the clock difference: *set adequate values to* λ.

Example: a multi-agent scenario

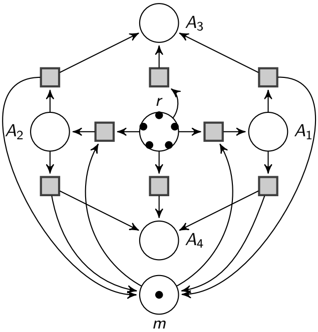


Formalizing

$\mathcal{DS}_3$ formula:

Instituto de Computação uff

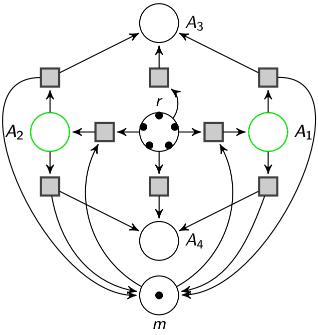
Example: a multi-agent scenario



Formalizing

$\mathcal{DS}_3$ formula: $\langle\{rrrrrm\}, rmt_2A_1 \odot rmt_2A_2 \odot rt_1A_3 \odot rt_1A_4 \odot A_1t_3A_3m \odot A_2t_3A_3m \odot A_1t_3A_4m \odot A_4t_3A_4m\rangle p$.

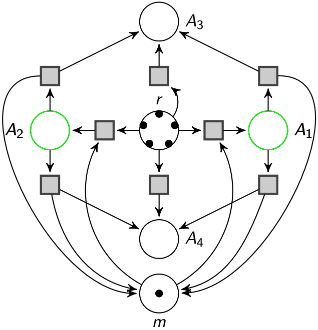
Example: a multi-agent scenario



Formalizing

Can A_1 and A_2 compute some data in parallel?

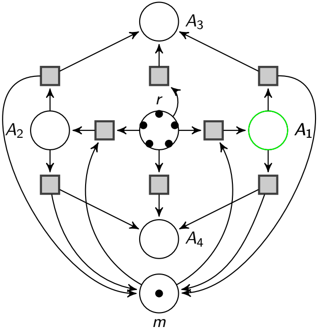
Example: a multi-agent scenario



Formalizing

Can A_1 and A_2 compute some data in parallel? *Look at the result of the firing function.*

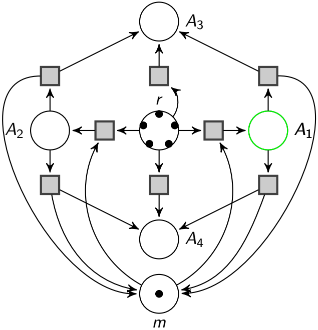
Example: a multi-agent scenario



Formalizing

From a world w is it possible that A_1 collect some data to process?

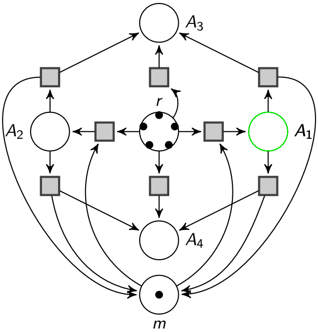
Example: a multi-agent scenario



Formalizing

From a world w is it possible that A_1 collect some data to process? Compute $Pr(\mathcal{M}, w \Vdash \langle s, rmt_2A_1 \rangle \top \mid \delta(w, rmt_2A_1 \odot rmt_2A_2 \odot rt_1A_3 \odot rt_1A_4 \odot A_1t_3A_3m \odot A_2t_3A_3m \odot A_1t_3A_4m \odot A_4t_3A_4m))$.

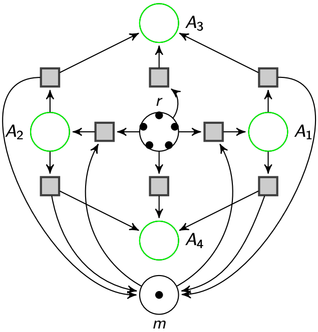
Example: a multi-agent scenario



Formalizing

From a world w is it possible that A_1 collect some data to process? Compute $\dfrac{\delta(w, rmt_2A_1)}{\displaystyle\sum_{\pi_b \in \Pi : f(s,\pi_b) \neq \epsilon} \delta(w, \pi_b)}$.

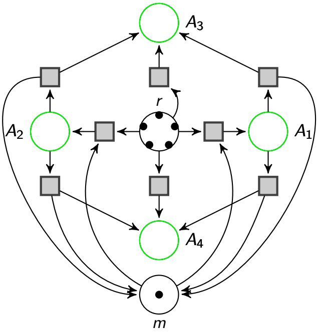
Example: a multi-agent scenario



Formalizing

Are agents A_1 and A_2 overheading agents A_3 and A_4?

Example: a multi-agent scenario



Formalizing

Are agents A_1 and A_2 overheading agents A_3 and A_4? Verify if $\sum_{\delta(v_i, A_1 t_1 A_3 \odot A_1 t_1 A_4 \odot A_2 t_1 A_3 \odot A_2 t_1 A_4)} 1 > \sum_{\delta(v_i, r t_1 A_3 \odot r t_1 A_4)} 1$.

Multi-agent Environment for Reasoning in Logic and Inferring Numerically (MERLIN)

```
a1 <− agent()
a2 <− agent()
a3 <− agent()
a4 <− agent()
setDataCenters(1)
send(a1, a3)
send(a1, a4)
send(a2, a3)
send(a2, a4)
collect(a1, freq=.5, shared=1)
collect(a2, freq=.5, shared=1)
collect(a3, freq=1)
collect(a4, freq=1)
```

http://github.com/blopesvieira/Merlin

Instituto de Computação UFF

Multi-agent Environment for Reasoning in Logic and Inferring Numerically (MERLIN)

```
> setResource(a1, 1)
> prToSend("a1", "a3")
[1] 0.4
```

Instituto de Computação uff

Ongoing...

- Automatic theorem prover
- Model checking framework for Dynamic Logics
- Studies on the computational complexity of the logics

Thank you!

Special thanks

Profa. Daniele Nantes

CNPq CAPES FAPERJ Fundação Carlos Chagas Filho de Amparo à Pesquisa do Estado do Rio de Janeiro

References I

Christiano Braga and Bruno Lopes.
Towards Reasoning in Dynamic Logics with Rewriting Logic: the Petri-PDL Case.
In Márcio Cornélio and Bill Roscoe, editors, *Formal Methods: Foundations and Applications*, Lecture Notes in Computer Science. Springer International Publishing, in press.

Bruno Lopes, Mario Benevides, and Edward Hermann Haeusler.
Extending propositional dynamic logic for Petri nets.
Electronic Notes in Theoretical Computer Science, 305(11):67–83, 2014.

References II

Bruno Lopes, Mario Benevides, and Edward Hermann Haeusler.
Propositional dynamic logic for Petri nets.
Logic Journal of the IGPL, 22(5), 2014.

Bruno Lopes, Mario Benevides, and Edward Hermann Haeusler.
Reasoning about Multi-Agent Systems Using Stochastic Petri Nets.
In Javier Bajo et al., editor, *Trends in Practical Applications of Agents, Multi-Agent Systems and Sustainability*, Advances in Intelligent Systems and Computing, pages 75–86. Springer International Publishing, 2015.

References III

Cláudia Nalon, Bruno Lopes, Edward Hermann Haeusler, and Gilles Dowek.
A calculus for automatic verification of Petri Nets based on Resolution and Dynamic Logics.
Electronic Notes in Theoretical Computer Science, 312:125–141, 2015.

Contact information

Bruno Lopes (*bruno@ic.uff.br*)

http://www.ic.uff.br/∼bruno

Instituto de Computação

Universidade Federal Fluminense

Niterói - RJ, Brazil