Rooted probabilistic branching bisimulation as a congruence

Matias D. Lee[1,2] Erik P. de Vink[3,4]

[1]FaMAF, Universidad Nacional de Córdoba, Argentina
[2]CONICET, Argentina
[3]TU/e, Eindhoven, The Netherlands
[4]CWI, Amsterdam, The Netherlands

QAPL 2015

Outline

Assumptions...I will assume that you know:

Probabilistic Transition Systems

- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification

- Probabilistic transition systems specification
- The Format

What is a signature

$$\Sigma = \{0, a.\_, b.\_, \_ + \_\}$$

What are rules...

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

What is a *transitions system specification* $(\Sigma, \mathcal{A}, R)$

- A signature Σ.

$$\Sigma = \{0, a.\_, b.\_, \_ + \_\}$$

- A *set of actions* $\mathcal{A}$.

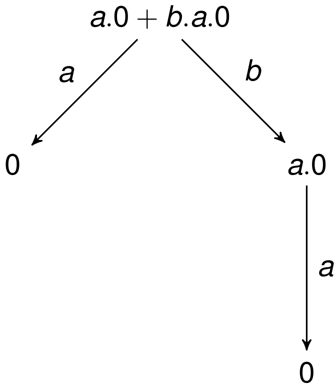
$$\mathcal{A} = \{a, b\}$$

- A set of rules R.

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$

Given a TSS, each term defines a transition system

$$\frac{}{a.x \xrightarrow{a} x} \qquad \frac{x \xrightarrow{a} x'}{x + y \xrightarrow{a} x'} \qquad \frac{y \xrightarrow{a} y'}{x + y \xrightarrow{a} y'}$$



Not every TSS defines compositional operators

- Signature: $\Sigma = \{a, a', f(\_)\}$
- Label: $\mathcal{A} = \{a\}$
- Rules R:

$$\frac{}{a \xrightarrow{a} a} \qquad \frac{}{a' \xrightarrow{a} a'} \qquad \frac{x \xrightarrow{a} a}{f(x) \xrightarrow{a} f(x)}$$

Not every TSS defines compositional operators

- Signature: $\Sigma = \{a, a', f(\_)\}$
- Label: $\mathcal{A} = \{a\}$
- Rules R:

$$\frac{}{a \xrightarrow{a} a} \qquad \frac{}{a' \xrightarrow{a} a'} \qquad \frac{x \xrightarrow{a} a}{f(x) \xrightarrow{a} f(x)}$$

A solution: Define a rule format (*rules with a particular shape*) that only allows to define good specification languages.

Our Goal

Define a specification format

– *a rule format with extra restrictions* –

to define languages to specify

probabilistic transition systems with internal actions.

Outline

Assumptions...I will assume that you know:

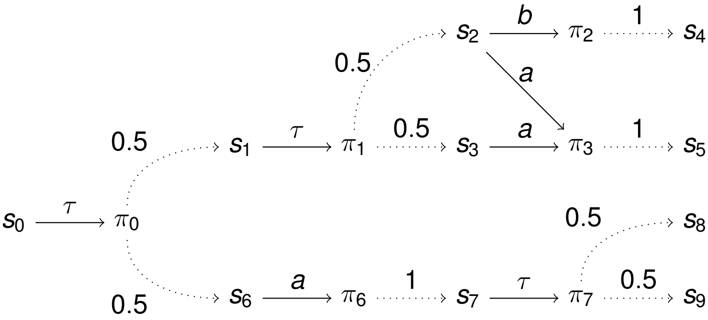
Probabilistic Transition Systems
- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification
- Probabilistic transition systems specification
- The Format

Probabilistic transition systems (PTS) with internal actions

$A = (S, \mathcal{A}, \rightarrow)$ is a PTS (or probabilistic automaton) where

- S is a set of states,
- $\mathcal{A}$ is a set of actions such that $\tau \in \mathcal{A}$ and
- $\rightarrow \subseteq S \times \mathcal{A} \times \mathcal{D}(A)$.



Outline

Assumptions...I will assume that you know:

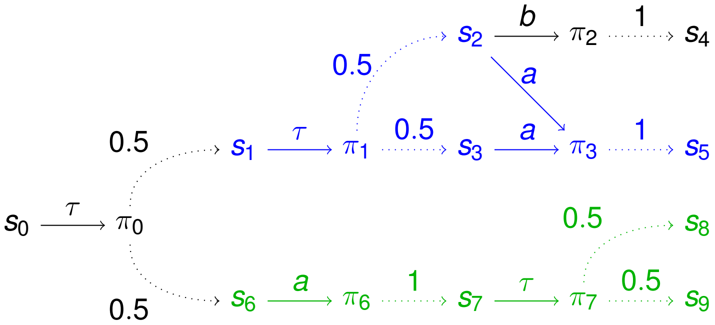
Probabilistic Transition Systems
- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification
- Probabilistic transition systems specification
- The Format

... I assume that you know what is a scheduler

A *scheduler* ς takes a traces α and returns a subdistributions over the transitions that can executes the last state of α.

Weak transitions

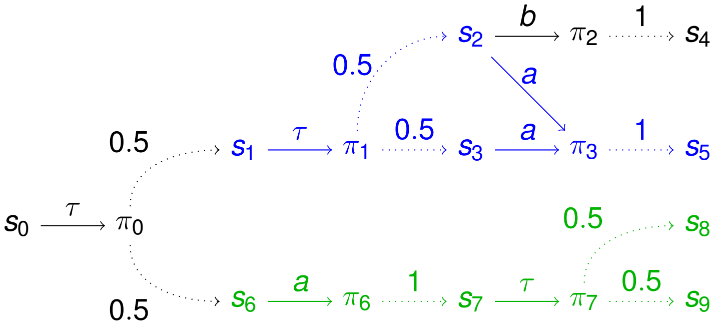


Because there are deterministic schedulers ς and ς we write

- $s_1 \stackrel{a}{\Longrightarrow} \delta(s_5)$
- $s_6 \stackrel{a}{\Longrightarrow}_\varsigma [0.5]\delta(s_8) + [0.5]\delta(s_9)$.
- Combining weak transition to get *weak hyper transitions*:

$$[0.5]\delta(s_1)+[0.5]\delta(s_6) \stackrel{a}{\Longrightarrow} [0.5]\delta(s_5)+[0.5]([0.5]\delta(s_8) + [0.5]\delta(s_9))$$

Weak transitions

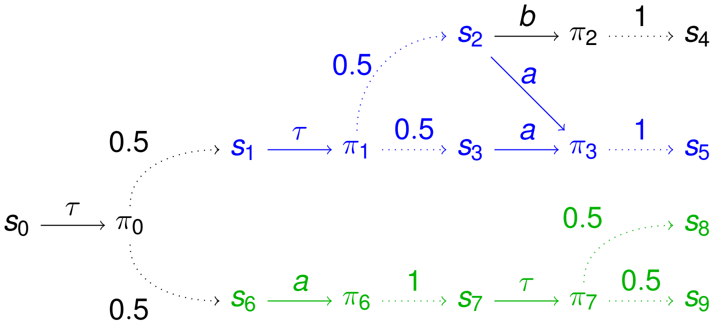


Because there are deterministic schedulers ς and ς we write

- $s_1 \stackrel{a}{\Longrightarrow} \delta(s_5)$
- $s_6 \stackrel{a}{\Longrightarrow}_{\varsigma} [0.5]\delta(s_8) + [0.5]\delta(s_9)$.
- Combining weak transition to get *weak hyper transitions*:

$$[0.5]\delta(s_1)+[0.5]\delta(s_6) \stackrel{a}{\Longrightarrow} [0.5]\delta(s_5)+[0.5]([0.5]\delta(s_8) + [0.5]\delta(s_9))$$

Weak transitions



Because there are deterministic schedulers ς and ς we write

- $s_1 \overset{a}{\Longrightarrow} \delta(s_5)$
- $s_6 \overset{a}{\Longrightarrow}_{\varsigma} [0.5]\delta(s_8) + [0.5]\delta(s_9)$.
- Combining weak transition to get *weak hyper transitions*:

$$[0.5]\delta(s_1)+[0.5]\delta(s_6) \overset{a}{\Longrightarrow} [0.5]\delta(s_5)+[0.5]([0.5]\delta(s_8) + [0.5]\delta(s_9))$$

Allowed weak transitions

We write $s \stackrel{a}{\Longrightarrow}_{\varsigma} \pi$ to denote that combined transiton $s \stackrel{a}{\Longrightarrow} \pi$ uses the scheduler ς.

Allowed weak transitions

We write $s \xRightarrow{a}_{\varsigma} \pi$ to denote that combined transiton $s \xRightarrow{a} \pi$ uses the scheduler ς.

$$s \xRightarrow{a,P}_{\varsigma} \pi \text{ if and only if } s \xRightarrow{a}_{\varsigma} \pi \text{ and } Im(\sigma) \subseteq P$$

Allowed weak transitions

We write $s \overset{a}{\Longrightarrow}_{\varsigma} \pi$ to denote that combined transiton $s \overset{a}{\Longrightarrow} \pi$ uses the scheduler ς.

$s \overset{a,P}{\Longrightarrow}_{\varsigma} \pi$ if and only if $s \overset{a}{\Longrightarrow}_{\varsigma} \pi$ and $Im(\sigma) \subseteq P$

Similar definition for $\pi \overset{a,P}{\Longrightarrow} \pi'$

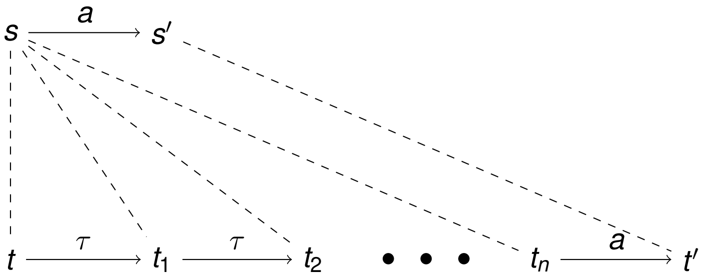
Outline

Assumptions...I will assume that you know:

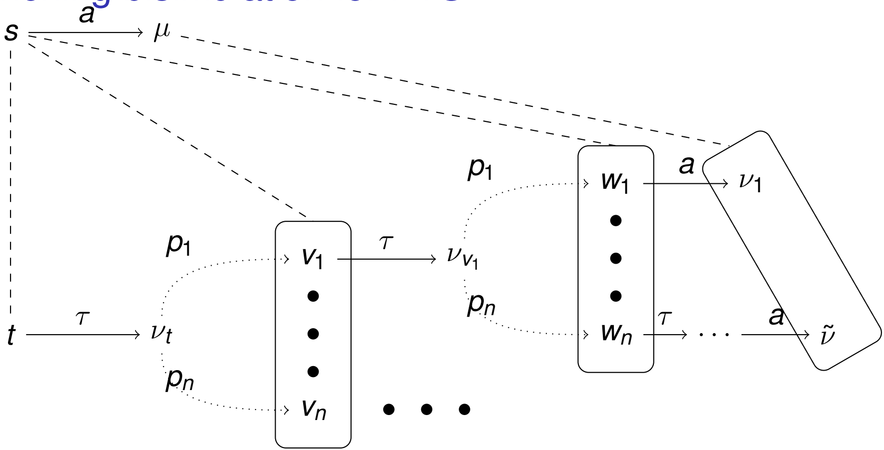
Probabilistic Transition Systems
- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification
- Probabilistic transition systems specification
- The Format

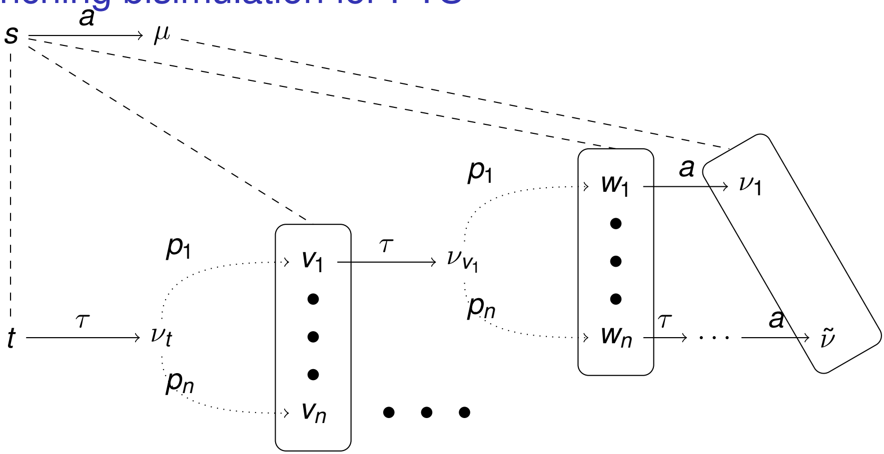
Branching bisimulation



Branching bisimulation for PTS



Branching bisimulation for PTS



Let $\mathcal{B}$ an equivalence relation over states and $\mathcal{L}(\mathcal{B})$ the lifting of the relation to distribution over states,

$$t \xrightarrow{\tau} \nu \text{ is } \textit{branching preserving} \text{ iff } \delta(t)\,\mathcal{L}(\mathcal{B})\,\nu$$

Branching bisimulation for PTS

Definition

An equivalence relation $\mathcal{B}$ on S is called *branching bisimulation* if there is a set $P \subseteq D(\tau)$ that is branching preserving w.r.t. $\mathcal{B}$ and whenever $s \,\mathcal{B}\, t$, if $s \xrightarrow{a} \mu$ then either

1. $a = \tau$ and $s \xrightarrow{a} \mu \in P$
2. $t \overset{\tau,P}{\Longrightarrow} \tilde{\nu} \xrightarrow{a} \nu$ and $\mu \, \mathcal{L}(\mathcal{B}) \, \nu$.

We write $s \approx_b t$ is there exists a probabilistic branching bisimulation relating s and t.

Outline

Assumptions...I will assume that you know:

Probabilistic Transition Systems
- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification
- Probabilistic transition systems specification
- The Format

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	
$a.\_ : d \to s$	
$0 : s$	
-	
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	
$a.\_ : d \to s$	
$0 : s$	
-	
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	$+ : dd \to d$
$a.\_ : d \to s$	
$0 : s$	
-	
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	$\boldsymbol{+} : dd \to d$
$a.\_ : d \to s$	$\boldsymbol{a}.\_ : d \to d$
$0 : s$	
-	
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	$\mathbf{+} : dd \to d$
$a.\_ : d \to s$	$\mathbf{a.\_} : d \to d$
$0 : s$	$\mathbf{0} : d$
-	
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	$\mathbf{+} : dd \to d$
$a.\_ : d \to s$	$\mathbf{a.}\_ : d \to d$
$0 : s$	$\mathbf{0} : d$
-	$\delta(\_) : s \to d$
-	

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \to s$	$\mathbf{+} : dd \to d$
$a.\_ : d \to s$	$\mathbf{a}.\_ : d \to d$
$0 : s$	$\mathbf{0} : d$
-	$\delta(\_) : s \to d$
-	$\bigoplus_I p_i \cdot \_ : d^{\|I\|} \to d$

Two sort signatures (Σ_s, Σ_d)

- Σ_s function names of sort s used to represent terms/processes.
- Σ_d function names of sort d used to represent distribution over terms/processes.

Σ_d is constructed based on Σ_s. Example:

Σ_s	Σ_d
$+ : ss \rightarrow s$	$\boldsymbol{+} : dd \rightarrow d$
$a.\_ : d \rightarrow s$	$\boldsymbol{a}.\_ : d \rightarrow d$
$0 : s$	$\mathbf{0} : d$
-	$\delta(\_) : s \rightarrow d$
-	$\bigoplus_I p_i \cdot \_ : d^{\lvert I \rvert} \rightarrow d$

We call (Σ_s, Σ_d) probabilistically lifted signature.

The interpretation of $T(\Sigma_d)$: $[\![\_]\!]$

- $[\![\delta(t)]\!] = \delta_t$ for all $t \in T(\Sigma_d)$
- $[\![\bigoplus_{i\in I}[p_i]\theta_i]\!] = \sum_{i\in I} p_i[\![\theta_i]\!]$ for $\{ \theta_i \mid i \in I \} \subseteq T(\Sigma_d)$
- $$[\![\boldsymbol{f}(\theta_1, \ldots, \theta_{rk(f)})]\!](f(\xi_1, \ldots, \xi_{rk(f)})) = \prod_{\sigma_i=1}^{rk(f)} [\![\theta_i]\!](\xi_i)$$
  assuming that the sort of f: $s^{rk(f)} \to s$

Example

- $[\![\theta_1 + \theta_2]\!](t_1 + t_2) = [\![\theta_1]\!](t_1) \cdot [\![\theta_2]\!](t_2)$
- $[\![\theta_1 + \theta_2]\!](a.(t_1 + t_2)) = 0$

The interpretation of $T(\Sigma_d)$: $[\![\_]\!]$

- $[\![\delta(t)]\!] = \delta_t$ for all $t \in T(\Sigma_d)$
- $[\![\bigoplus_{i\in I}[p_i]\theta_i]\!] = \sum_{i\in I} p_i[\![\theta_i]\!]$ for $\{\ \theta_i \mid i \in I\ \} \subseteq T(\Sigma_d)$
- $[\![\boldsymbol{f}(\theta_1, \ldots, \theta_{rk(f)})]\!](f(\xi_1, \ldots, \xi_{rk(f)})) = \prod_{\sigma_i=1}^{rk(f)} [\![\theta_i]\!](\xi_i)$
  assuming that the sort of f: $s^{rk(f)} \to s$

Example

- $[\![\theta_1 + \theta_2]\!](t_1 + t_2) = [\![\theta_1]\!](t_1) \cdot [\![\theta_2]\!](t_2)$
- $[\![\theta_1 + \theta_2]\!](a.(t_1 + t_2)) = 0$

Definition (PTSS)

A *probabilistic transition system specification* (PTSS) is a triple $P = (\Sigma, \mathcal{A}, R)$ where Σ is a probabilistically lifted signature, $\mathcal{A}$ is a set of labels, and $\mathcal{R}$ is a set of rules of the form:

$$\frac{\{\, t_k \xrightarrow{a_k} \theta_k \mid k \in K \,\} \;\cup\; \{\, t_l \stackrel{b_l}{\not\longrightarrow} \mid l \in L \,\}}{t \xrightarrow{a} \theta}$$

where K, L are index sets, $t, t_k, t_\ell \in \mathbb{T}(\Sigma_s)$, $a, a_k, b_\ell \in \mathcal{A}$, and $\theta_k, \theta \in \mathbb{T}(\Sigma_d)$.

Running Example: PTSS $\tilde{P}$

- Lifted signature (Σ_s, Σ_d) where

$$\Sigma_s = \{\_ + \_, a.\_, b.\_, 0\}$$

$$\Sigma_d = \{\_ \mathbf{+} \_, \mathbf{a.}\_, \mathbf{b.}\_, \mathbf{0}, \delta(\_), \} \cup \{\bigoplus_I p_i \mid I \text{ is an index set}\}$$

- $\mathcal{A} = \{a, b\}$
- Rules R:

$$\frac{}{a.\mu \xrightarrow{a} \mu} \qquad \frac{x \xrightarrow{a} \mu}{x + y \xrightarrow{a} \mu} \qquad \frac{y \xrightarrow{a} \mu}{x + y \xrightarrow{a} \mu}$$

Outline

Assumptions...I will assume that you know:

Probabilistic Transition Systems
- Probabilistic Transition Systems
- Weak transitions
- Branching Bisimulation

Probabilistic transition systems specification
- Probabilistic transition systems specification
- The Format

Standard restriction: rooted condition

Definition

A symmetric relation R $\subseteq T(\Sigma_s) \times T(\Sigma_s)$ is called a *rooted branching bisimulation* with respect to a PTS A $= (T(\Sigma_s), \mathcal{A}, \rightarrow)$ if, for all $s, t \in T(\Sigma_s)$ such that s R t:

- $s \xrightarrow{a} \theta_s$ imply $t \xrightarrow{a} \theta_t$ for some θ_t with $\theta_s \approx_b \theta_t$.

The idea behind the specification format

The idea behind the specification format

To avoid detecting internal transitions.

Quantitative premises and look-ahead

- Suppose the rules allow quantitative premises

$$\theta(Y) \trianglerighteq p$$

with $\theta \in \mathbb{T}(\Sigma_d)$, $Y \subseteq \mathcal{V}$, $\trianglerighteq \in \{>, \geq\}$ and $p \in [0, 1]$

- Add the operator f to $\tilde{P}$ defined by

$$\frac{x \xrightarrow{a} \mu \quad \mu(\{y\}) > 0 \quad y \xrightarrow{b} \nu}{f(x) \xrightarrow{a} \mathbf{0}} \tag{1}$$

- $a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})$ and
- $b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ but $\tau.\boldsymbol{b}.\mathbf{0} \stackrel{b}{\not\rightarrow}$ then
- $f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{a}$ but $f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \stackrel{a}{\not\rightarrow}$.

Quantitative premises and look-ahead

- Suppose the rules allow quantitative premises

$$\theta(Y) \trianglerighteq p$$

with $\theta \in \mathbb{T}(\Sigma_d)$, $Y \subseteq \mathcal{V}$, $\trianglerighteq \in \{>, \geq\}$ and $p \in [0,1]$

- Add the operator f to $\tilde{P}$ defined by

$$\frac{x \xrightarrow{a} \mu \quad \mu(\{y\}) > 0 \quad y \xrightarrow{b} \nu}{f(x) \xrightarrow{a} \mathbf{0}} \qquad (1)$$

- $a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\tau.\boldsymbol{b}.\mathbf{0})$ and
- $b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ but $\tau.\boldsymbol{b}.\mathbf{0} \not\xrightarrow{b}$ then
- $f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{a}$ but $f(a.(\tau.\boldsymbol{b}.\mathbf{0})) \not\xrightarrow{a}$.

Quantitative premises and look-ahead

- Suppose the rules allow quantitative premises

$$\theta(Y) \trianglerighteq p$$

with $\theta \in \mathbb{T}(\Sigma_d)$, $Y \subseteq \mathcal{V}$, $\trianglerighteq \in \{>, \geq\}$ and $p \in [0, 1]$

- Add the operator f to $\tilde{P}$ defined by

$$\frac{x \xrightarrow{a} \mu \quad \mu(\{y\}) > 0 \quad y \xrightarrow{b} \nu}{f(x) \xrightarrow{a} \mathbf{0}} \tag{1}$$

- $a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})$ and
- $b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ but $\tau.\boldsymbol{b}.\mathbf{0} \stackrel{b}{\not\rightarrow}$ then
- $f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{a}$ but $f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \stackrel{a}{\not\rightarrow}$.

Quantitative premises and look-ahead

- Suppose the rules allow quantitative premises

$$\theta(Y) \trianglerighteq p$$

with $\theta \in \mathbb{T}(\Sigma_d)$, $Y \subseteq \mathcal{V}$, $\trianglerighteq \in \{>, \geq\}$ and $p \in [0, 1]$

- Add the operator f to $\tilde{P}$ defined by

$$\frac{x \xrightarrow{a} \mu \quad \mu(\{y\}) > 0 \quad y \xrightarrow{b} \nu}{f(x) \xrightarrow{a} \mathbf{0}} \qquad (1)$$

- $a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})$ and
- $b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ but $\tau.\boldsymbol{b}.\mathbf{0} \not\xrightarrow{b}$ then
- $f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{a}$ but $f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \not\xrightarrow{a}$.

Quantitative premises and look-ahead

- Suppose the rules allow quantitative premises

$$\theta(Y) \trianglerighteq p$$

with $\theta \in \mathbb{T}(\Sigma_d)$, $Y \subseteq \mathcal{V}$, $\trianglerighteq \in \{>, \geq\}$ and $p \in [0,1]$

- Add the operator f to $\tilde{P}$ defined by

$$\frac{x \xrightarrow{a} \mu \quad \mu(\{y\}) > 0 \quad y \xrightarrow{b} \nu}{f(x) \xrightarrow{a} \mathbf{0}} \tag{1}$$

- $a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})$ and
- $b.\mathbf{0} \xrightarrow{b} \mathbf{0}$ but $\tau.\boldsymbol{b}.\mathbf{0} \stackrel{b}{\not\longrightarrow}$ then
- $f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{a}$ but $f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \stackrel{a}{\not\longrightarrow}$.

Look-ahead by operator nesting

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}} \tag{2}$$

$t = a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) = t'$ **...** but

Look-ahead by operator nesting

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}} \tag{2}$$

$t = a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) = t'$ **...** but

Look-ahead by operator nesting

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}} \tag{2}$$

$t = a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) = t'$ **...** but

$$g(t, b.0) \xrightarrow{\quad b \quad} \mathbf{0}$$

$$g(t', \tau.b.0)$$

Look-ahead by operator nesting

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}} \tag{2}$$

$t = a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) = t'$ **...** but

$$f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{\quad a \quad} \boldsymbol{g}(\delta(t), \boldsymbol{b}.\mathbf{0}) \qquad g(t, b.0) \xrightarrow{\quad b \quad} \mathbf{0}$$

$$f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \xrightarrow{a} \boldsymbol{g}(\delta(t'), \boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) \qquad g(t', \tau.b.0)$$

Look-ahead by operator nesting

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}} \tag{2}$$

$t = a.(\boldsymbol{b}.\mathbf{0}) \approx_b a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) = t'$ **...** but

$$f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{\quad a \quad} \boldsymbol{g}(\delta(t), \boldsymbol{b}.\mathbf{0}) \overset{1}{\dashrightarrow} g(t, b.0) \xrightarrow{\quad b \quad} \mathbf{0}$$

$$f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \xrightarrow{a} \boldsymbol{g}(\delta(t'), \boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) \overset{1}{\dashrightarrow} g(t', \tau.b.0)$$

Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)}$$

Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- Definition: When the target of a positive premise is used as an argument of the function in the target of the conclusion, we say that the position of the argument is *wild*. If the position is not wild, then it is *tame*.
- The second argument of g is *wild*.

Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- To "control" wild arguments we introduce *patience rules*.
- In this case, a patience rule for the second argument of g is defined by

$$\frac{x_2 \xrightarrow{\tau} \mu}{g(x_1, x_2) \xrightarrow{\tau} \boldsymbol{g}(\delta(x_1), \mu)} \tag{3}$$

Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- To "control" wild arguments we introduce *patience rules*.
- In this case, a patience rule for the second argument of g is defined by

$$\frac{x_2 \xrightarrow{\tau} \mu}{g(x_1, x_2) \xrightarrow{\tau} \boldsymbol{g}(\delta(x_1), \mu)} \tag{3}$$

$$g(t, b.0) \xrightarrow{b} \mathbf{0}$$

$$g(t', \tau.b.0)$$

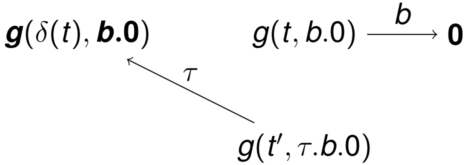
Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- To "control" wild arguments we introduce *patience rules*.
- In this case, a patience rule for the second argument of g is defined by

$$\frac{x_2 \xrightarrow{\tau} \mu}{g(x_1, x_2) \xrightarrow{\tau} \boldsymbol{g}(\delta(x_1), \mu)} \tag{3}$$



Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- To "control" wild arguments we introduce *patience rules*.
- In this case, a patience rule for the second argument of g is defined by

$$\frac{x_2 \xrightarrow{\tau} \mu}{g(x_1, x_2) \xrightarrow{\tau} \boldsymbol{g}(\delta(x_1), \mu)} \tag{3}$$

$$f(a.(\boldsymbol{b}.\mathbf{0})) \xrightarrow{\quad a \quad} \boldsymbol{g}(\delta(t), \boldsymbol{b}.\mathbf{0}) \qquad g(t, b.0) \xrightarrow{\quad b \quad} \mathbf{0}$$

$$g(t', \tau.b.0) \xrightarrow{\tau} \boldsymbol{g}(\delta(t), \boldsymbol{b}.\mathbf{0})$$

$$f(a.(\boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0})) \xrightarrow{a} \boldsymbol{g}(\delta(t'), \boldsymbol{\tau}.\boldsymbol{b}.\mathbf{0}) \qquad g(t', \tau.b.0)$$

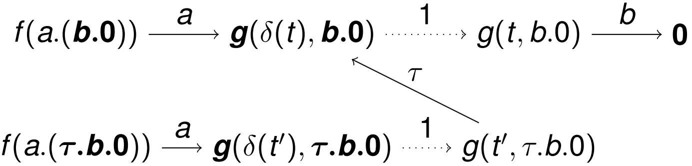
Wild arguments and patience rules

Add to $\tilde{P}$ operators f and g defined by the following rules

$$\frac{x \xrightarrow{a} \mu}{f(x) \xrightarrow{a} \boldsymbol{g}(\delta(x), \mu)} \qquad \frac{x_2 \xrightarrow{b} \mu}{g(x_1, x_2) \xrightarrow{b} \mathbf{0}}$$

- To "control" wild arguments we introduce *patience rules*.
- In this case, a patience rule for the second argument of g is defined by

$$\frac{x_2 \xrightarrow{\tau} \mu}{g(x_1, x_2) \xrightarrow{\tau} \boldsymbol{g}(\delta(x_1), \mu)} \tag{3}$$



Definition: RBB safe specification format (1/2)

Let $P = (\Sigma, \mathcal{A}, \mathcal{R})$ be a PTSS where each argument of $f \in \Sigma$ is defined as wild or tame with respect to $\mathcal{G}_P$. The PTSS P is in the *probabilistic RBB safe* specification format if for all $r \in \mathcal{R}$ one of the following conditions holds.

1. r is a patience rule for a wild argument of a function symbol in Σ.

Definition: RBB safe specification format (2/2)

2. r is a *RBB safe rule*, i.e. r has the following shape

$$\frac{\{\ t_m \xrightarrow{a_m} \mu_m \mid m \in M\ \} \qquad \{\ t_n \stackrel{b_n}{\not\longrightarrow} \mid n \in N\}}{f(\zeta_1, \ldots, \zeta_{rk(f)}) \xrightarrow{a} \theta}$$

where M and N are index sets, ζ_k, and μ_m, with $1 \leqslant k \leqslant rk(f)$ and $m \in M$, are all different variables, $f \in \Sigma_s$, $t_m, t_n \in \mathbb{T}(\Sigma_s)$ and $\theta \in \mathbb{T}(\Sigma_d)$, and the following conditions are met.

2.1 If the i-th argument of f is wild and has a patience rule in $\mathcal{R}$, then for all $\psi \in prem(r)$ such that $\zeta_i = x_i \in var(\psi)$, $\psi = x_i \xrightarrow{a_i} \mu_i$ with $a_i \neq \tau$.

2.2 If the i-th argument of f is wild and does not have a patience rule in $\mathcal{R}$, then ζ_i does not occur in the source of a premise of r.

2.3 Variables μ_m, for $m \in M$, and variables ζ_i, where i-th argument of f is wild, only occur at w-nested positions in θ.

2.4 $\mu_m \notin var(t_{m'})$ for all $m, m' \in M$.

How to prove the result (1/2)

- Branching bisimulation allows a characterization without schedulers.

How to prove the result (2/2)

- Follow the ideas for qualitative version of the format.

Corrigendum

- In the paper, we state that:

  a branching bisimulation that uses probabilistic schedulers

  has a characterization without scheduler.

Corrigendum

- In the paper, we state that:

  a branching bisimulation that uses probabilistic schedulers
  has a characterization without scheduler. ✗

Corrigendum

- In the paper, we state that:

  a branching bisimulation that uses probabilistic schedulers

  has a characterization without scheduler. ✗

- The result holds for branching bisimulation that use deterministic schedulers

Final comments

Combined transitions make the things really complex... but,

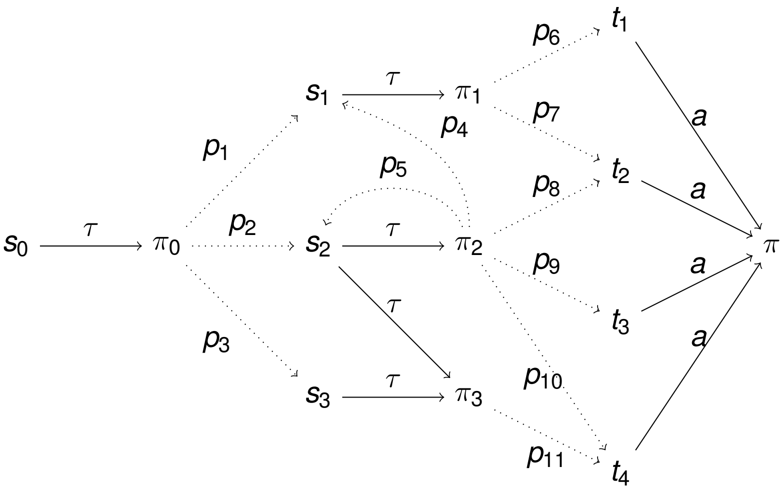
- Distributions terms and
- the characterization without schedulers of the branching bisimulation

have allowed to repeat the ideas used in the qualitative case.

The End

Questions?

Schedulers “are not” needed.



For all scheduler σ' such that $s \stackrel{a}{\Longrightarrow}_{\sigma'} \pi'$, $\pi' = \pi$.
(notice that all τ-transition are branching preserving)