Classification of Probabilistic Systems

Ana Sokolova

Computational Systems Group, University of Salzburg, Austria

Joint work with: Falk Bartels CWI,NL

Erik de Vink TU Eindhoven, NL

Bart Jacobs RU Nijmegen, NL

Outline

- probabilistic systems as coalgebras
- two strong semantics
  - * bisimilarity
  - * behaviour equivalence
- expressiveness comparison
- a hierarchy
- beyond discrete probabilities, beyond $\mathbf{Sets}$

Formal methods

In general:

- models - transition systems, automata, terms,...
  with a clear semantics
- analysis - model checking
  process algebra
  theorem proving...

In this talk

- models - probabilistic transition systems
- semantics - bisimilarity/behavior equivalence

In this talk

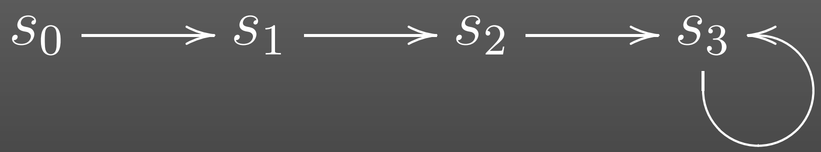
- models - probabilistic transition systems
- semantics - bisimilarity/behavior equivalence

Aim: expressiveness of many models

in a single framework

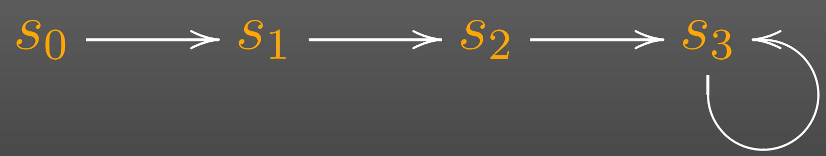
Example models

deterministic systems



Example models

deterministic systems



states $+$ transitions $\alpha : S \to S$

$$\alpha(s_0) = s_1,\ \alpha(s_1) = s_2,\ \ldots$$

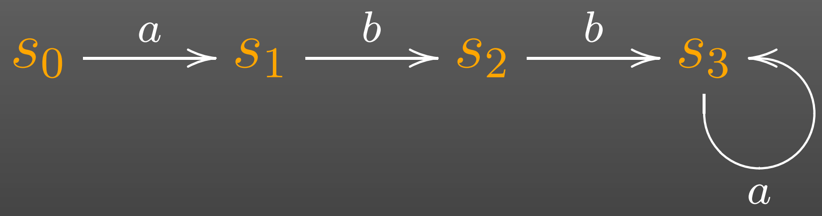
Example models

labelled deterministic systems A - labels

$$s_0 \xrightarrow{a} s_1 \xrightarrow{b} s_2 \xrightarrow{b} s_3 \circlearrowleft a$$

Example models

labelled deterministic systems $\quad A$ - labels

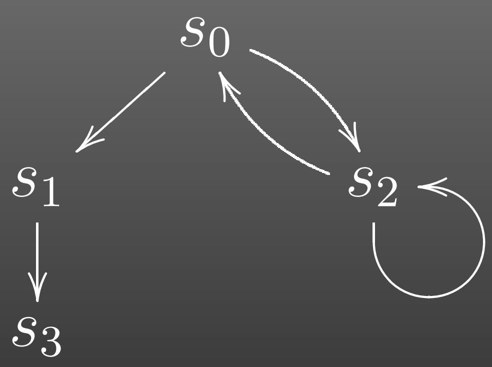


states $+$ transitions $\alpha : S \to A \times S$

$$\alpha(s_0) = \langle a, s_1 \rangle,\ \alpha(s_1) = \langle b, s_2 \rangle,\ \ldots$$

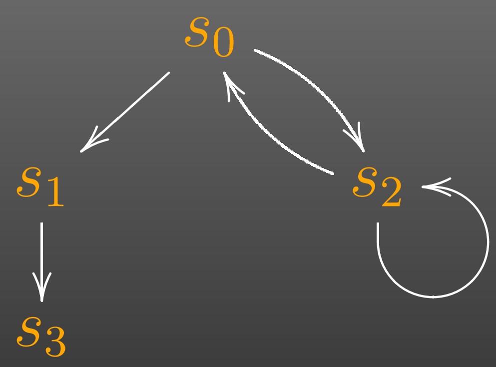
Example models

transition systems



Example models

transition systems

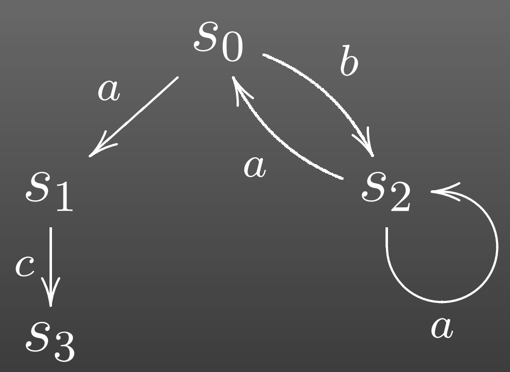


states $+$ transitions $\alpha : S \to \mathcal{P}(S)$

$$\alpha(s_0) = \{s_1, s_2\},\ \alpha(s_1) = \{s_3\},\ \ldots$$

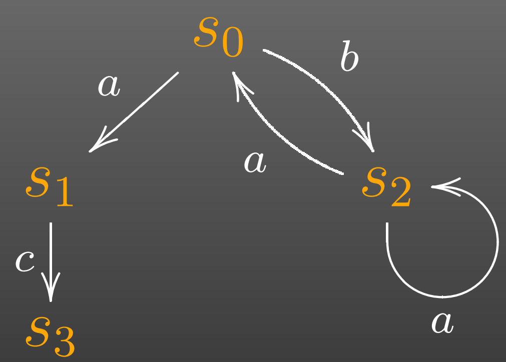
Example models

labelled transition systems A - labels



Example models

labelled transition systems A - labels



states $+$ transitions $\alpha : S \to \mathcal{P}(A \times S)$

$$\alpha(s_0) = \{\langle a, s_1\rangle, \langle b, s_2\rangle\},\ \alpha(s_1) = \{\langle c, s_3\rangle\},\ \ldots$$

Coalgebras

are an elegant generalization of transition systems with

states + transitions

Coalgebras

are an elegant generalization of transition systems with

states + transitions

as pairs

$\langle S, \alpha : S \to \mathcal{F}S \rangle$, for $\mathcal{F}$ a functor

Coalgebras

are an elegant generalization of transition systems with

states + transitions

as pairs

$$\langle S, \alpha : S \to \mathcal{F}S \rangle, \text{ for } \mathcal{F} \text{ a functor}$$

- rich mathematical structure
- a uniform way for treating transition systems
- general notions and results, generic notion of bisimulation

Probabilistic systems

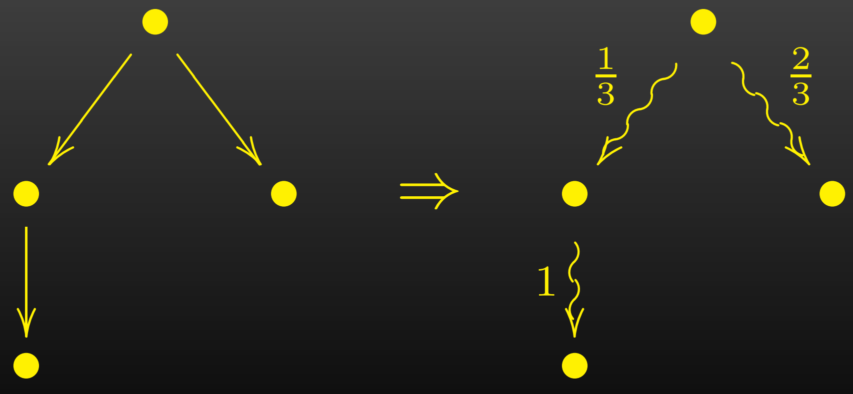
arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

There are many ways to do it:

Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

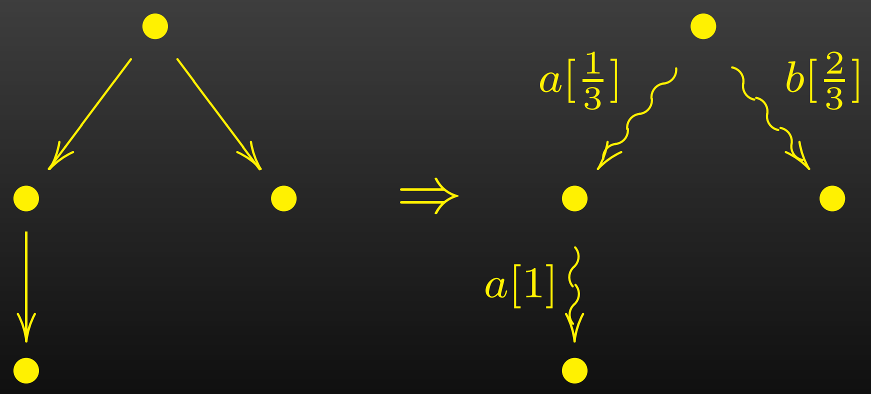
There are many ways to do it:



Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

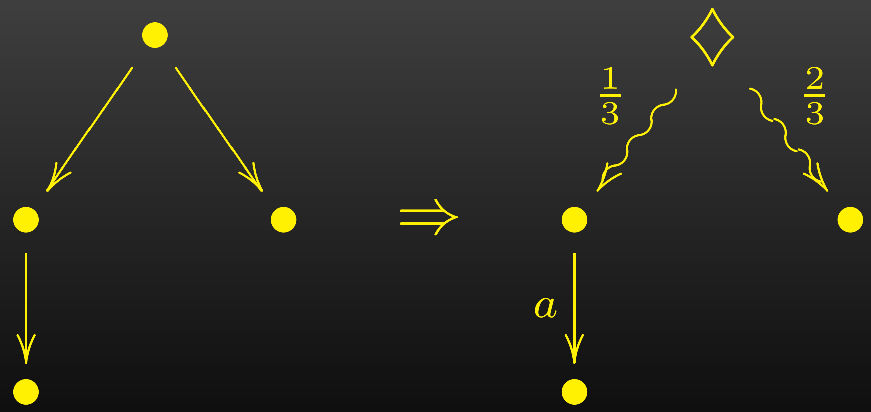
There are many ways to do it:



Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

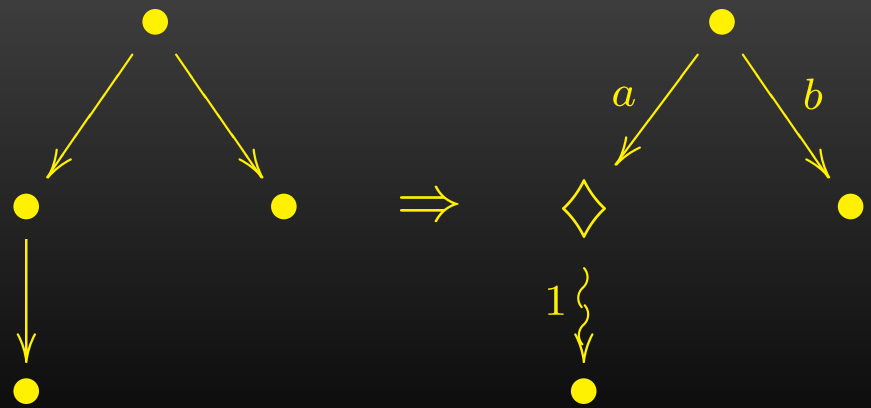
There are many ways to do it:



Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

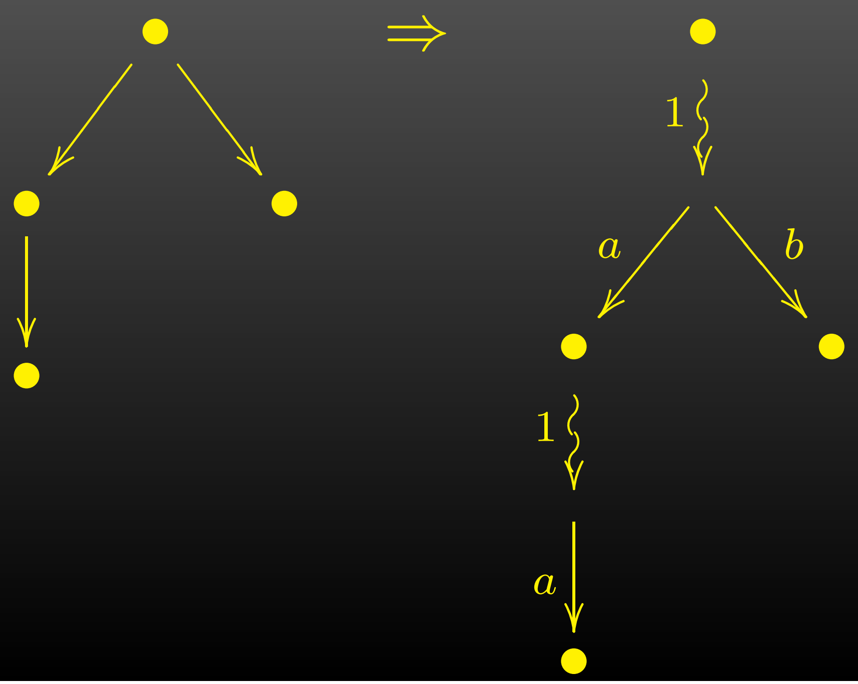
There are many ways to do it:



Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

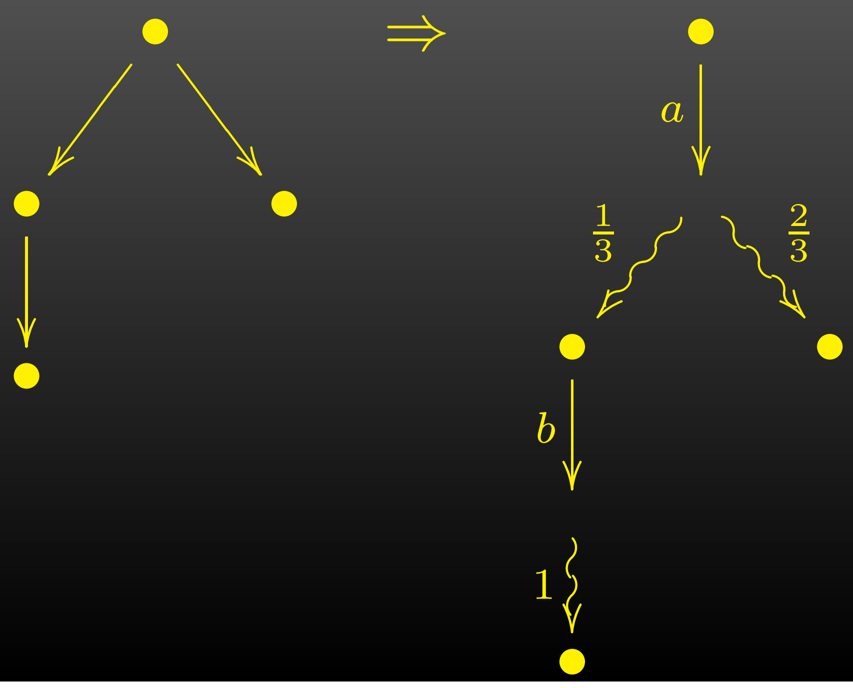
There are many ways to do it:



Probabilistic systems

arise by enriching transition systems with (discrete) probabilities as labels on the transitions.

There are many ways to do it:



Probabilistic systems

Thanks to the probability distribution functor $\mathcal{D}$

Probabilistic systems

Thanks to the probability distribution functor $\mathcal{D}$

$$\mathcal{D}S = \begin{array}{l}\text{the set of all discrete}\\ \text{probability distributions on } S\end{array}$$

Probabilistic systems

Thanks to the probability distribution functor $\mathcal{D}$

$$\mathcal{D}S = \{\mu : S \to [0,1], \mu[S] = 1\}, \quad \mu[X] = \sum_{s \in X} \mu(x)$$

$$\mathcal{D}f : \mathcal{D}S \to \mathcal{D}T, \;\; \mathcal{D}f(\mu)(t) = \mu[f^{-1}(\{t\})]$$

Probabilistic systems

Thanks to the probability distribution functor $\mathcal{D}$

$$\mathcal{D}S = \{\mu : S \to [0,1], \mu[S] = 1\}, \quad \mu[X] = \sum_{s \in X} \mu(x)$$

$$\mathcal{D}f : \mathcal{D}S \to \mathcal{D}T, \;\; \mathcal{D}f(\mu)(t) = \mu[f^{-1}(\{t\})]$$

the probabilistic systems are also coalgebras

Probabilistic systems

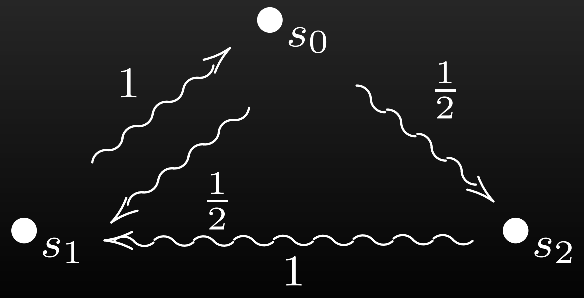
Thanks to the probability distribution functor $\mathcal{D}$

$$\mathcal{D}S = \{\mu : S \to [0,1], \mu[S] = 1\}, \quad \mu[X] = \sum_{s \in X} \mu(x)$$

$$\mathcal{D}f : \mathcal{D}S \to \mathcal{D}T, \ \ \mathcal{D}f(\mu)(t) = \mu[f^{-1}(\{t\})]$$

the probabilistic systems are also coalgebras

Example: $\alpha : S \to \mathcal{D}S$



Probabilistic systems

Thanks to the probability distribution functor $\mathcal{D}$

$$\mathcal{D}S = \{\mu : S \to [0,1], \mu[S] = 1\}, \quad \mu[X] = \sum_{s \in X} \mu(x)$$

$$\mathcal{D}f : \mathcal{D}S \to \mathcal{D}T, \;\; \mathcal{D}f(\mu)(t) = \mu[f^{-1}(\{t\})]$$

the probabilistic systems are also coalgebras
... of functors built by the following syntax

$$\mathcal{F} ::= \_ \mid A \mid \mathcal{P} \mid \mathcal{D} \mid \mathcal{G} + \mathcal{H} \mid \mathcal{G} \times \mathcal{H} \mid \mathcal{G}^A \mid \mathcal{G} \circ \mathcal{H}$$

reactive, generative

evolve from LTS - functor $\textcircled{\mathcal{P}}(A \times \_) \cong \textcircled{\mathcal{P}}^A$

reactive, generative

evolve from LTS - functor $\mathcal{P}(A \times \_) \cong \mathcal{P}^A$

reactive systems:

functor $(\mathcal{D} + 1)^A$

reactive, generative

evolve from LTS - functor $\mathcal{P}(A \times \_) \cong \mathcal{P}^A$

reactive systems:

functor $(\mathcal{D} + 1)^A$

generative systems:

functor $(\mathcal{D} + 1)(A \times \_) = \mathcal{D}(A \times \_) + 1$

reactive, generative

evolve from LTS - functor $\mathcal{P}(A \times \_) \cong \mathcal{P}^A$

reactive systems:

functor $(\mathcal{D} + 1)^A$

generative systems:

functor $(\mathcal{D} + 1)(A \times \_) = \mathcal{D}(A \times \_) + 1$

note: in the probabilistic case

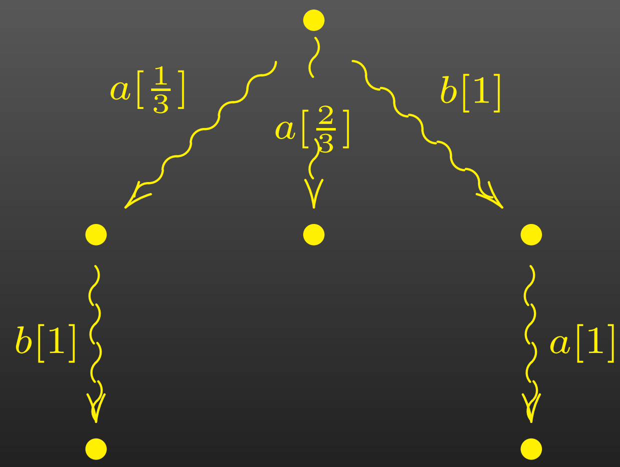
$$(\mathcal{D} + 1)^A \ncong \mathcal{D}(A \times \_) + 1$$

Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...

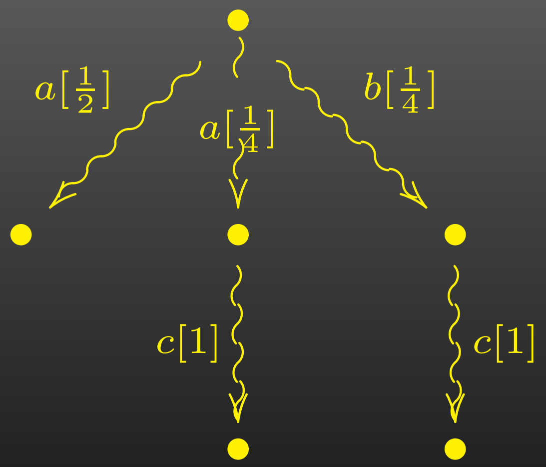
Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...



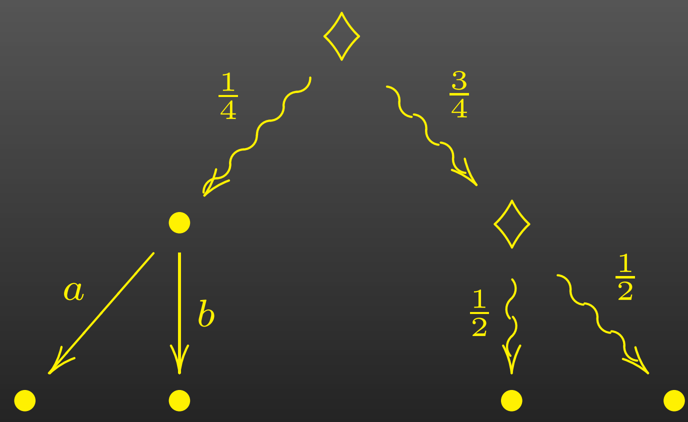
Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...



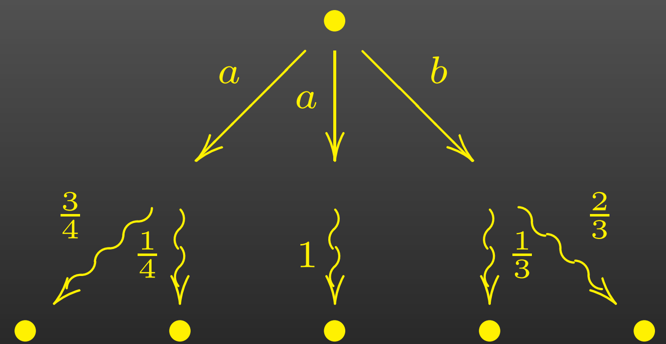
Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...



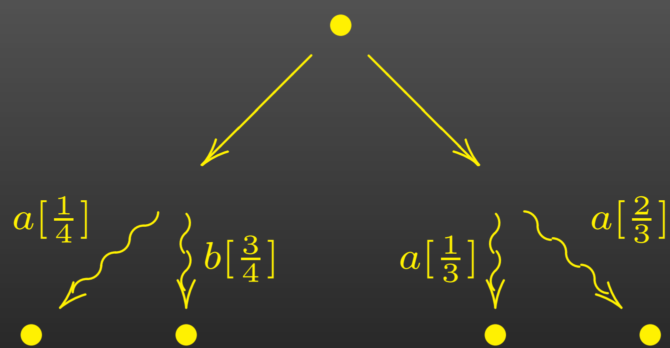
Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...



Probabilistic system types

MC	$\mathcal{D}$
DLTS	$(\_ + 1)^A$
LTS	$\mathcal{P}(A \times \_) \cong \mathcal{P}^A$
React	$(\mathcal{D} + 1)^A$
Gen	$\mathcal{D}(A \times \_) + 1$
Str	$\mathcal{D} + (A \times \_) + 1$
Alt	$\mathcal{D} + \mathcal{P}(A \times \_)$
Var	$\mathcal{D}(A \times \_) + \mathcal{P}(A \times \_)$
SSeg	$\mathcal{P}(A \times \mathcal{D})$
Seg	$\mathcal{P}\mathcal{D}(A \times \_)$
...	...



Bisimulation - LTS

R - equivalence on states, is a bisimulation if

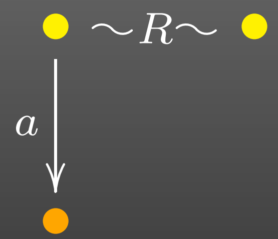
Bisimulation - LTS

R - equivalence on states, is a bisimulation if

$$\bullet \sim R \sim \bullet$$

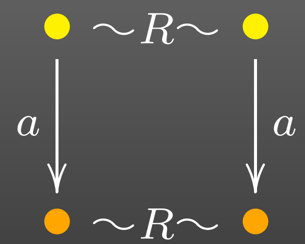
Bisimulation - LTS

R - equivalence on states, is a bisimulation if



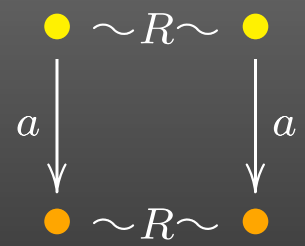
Bisimulation - LTS

R - equivalence on states, is a bisimulation if



Bisimulation - LTS

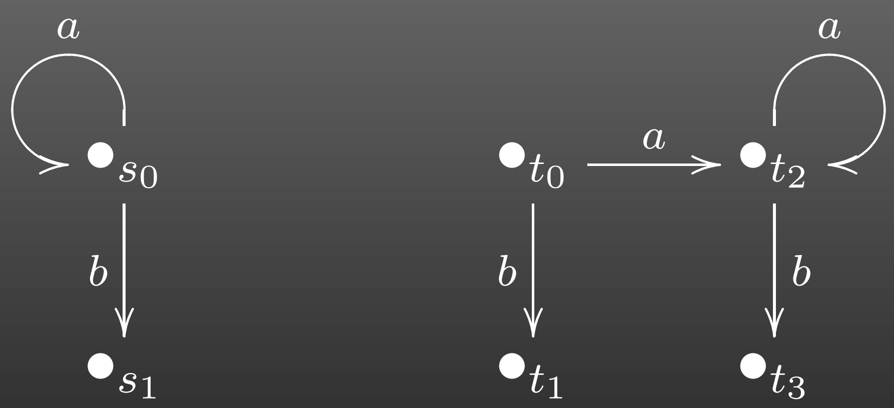
R - equivalence on states, is a bisimulation if



... two states are bisimilar if they are related by some bisimulation

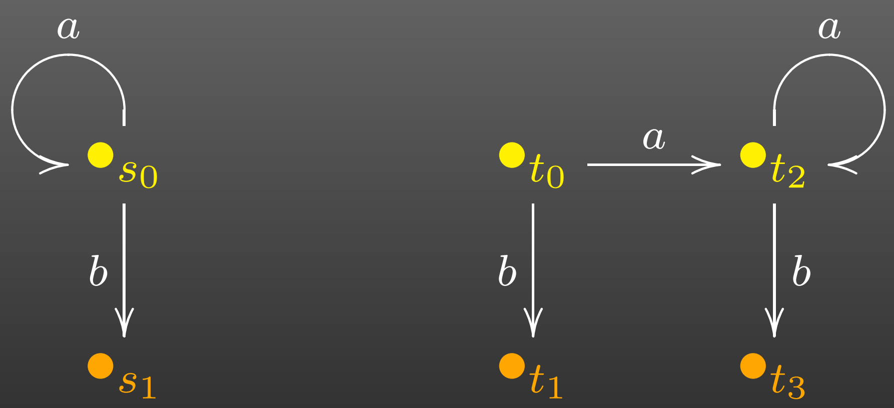
Bisimulation - LTS

Example: Consider the LTS



Bisimulation - LTS

Example: Consider the LTS



the coloring is a bisimulation, so s_0 and t_0 are bisimilar

Bisimulation - generative

R - equivalence on states, is a bisimulation if

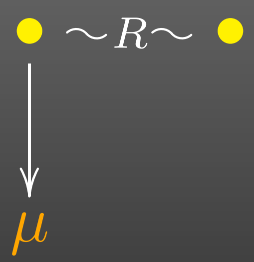
Bisimulation - generative

R - equivalence on states, is a bisimulation if

- $\sim R \sim$

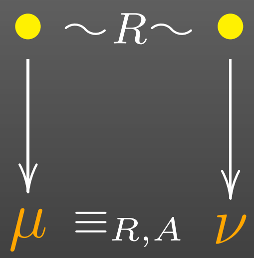
Bisimulation - generative

R - equivalence on states, is a bisimulation if



Bisimulation - generative

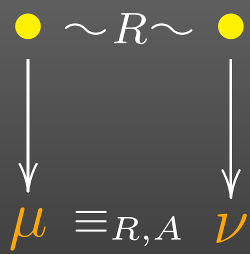
R - equivalence on states, is a bisimulation if



$\equiv_{R,A}$ relates distributions that assign the same probability to each label and each R-class

Bisimulation - generative

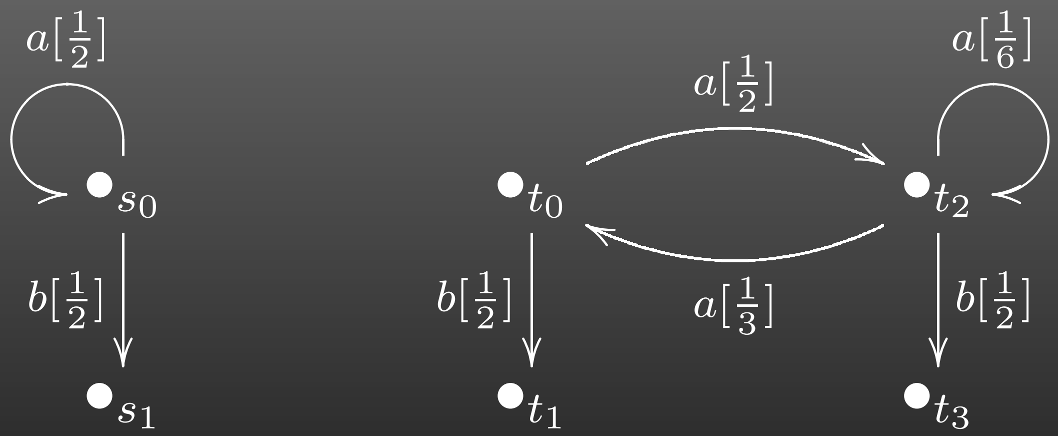
R - equivalence on states, is a bisimulation if



... two states are bisimilar if they are related by some bisimulation

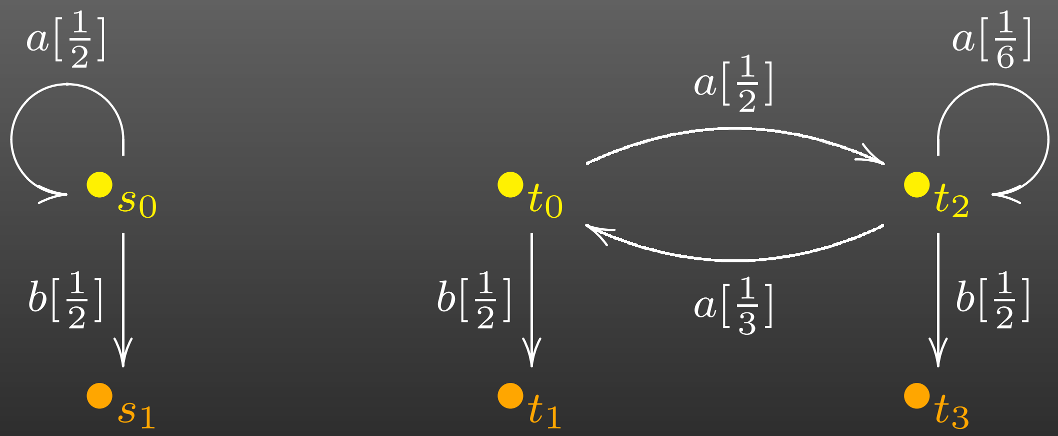
Bisimulation - generative

Consider the generative systems



Bisimulation - generative

Example: Consider the generative systems



the coloring is a bisimulation, so s_0 and t_0 are bisimilar

Bisimulation - simple Segala

R - equivalence on states, is a bisimulation if

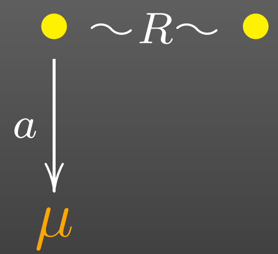
Bisimulation - simple Segala

R - equivalence on states, is a bisimulation if

$$\bullet \sim R \sim \bullet$$

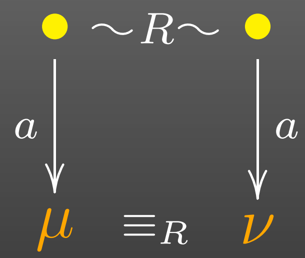
Bisimulation - simple Segala

R - equivalence on states, is a bisimulation if



Bisimulation - simple Segala

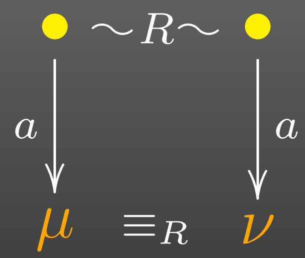
R - equivalence on states, is a bisimulation if



$\equiv_R$ relates distributions that assign the same probability to each R-class

Bisimulation - simple Segala

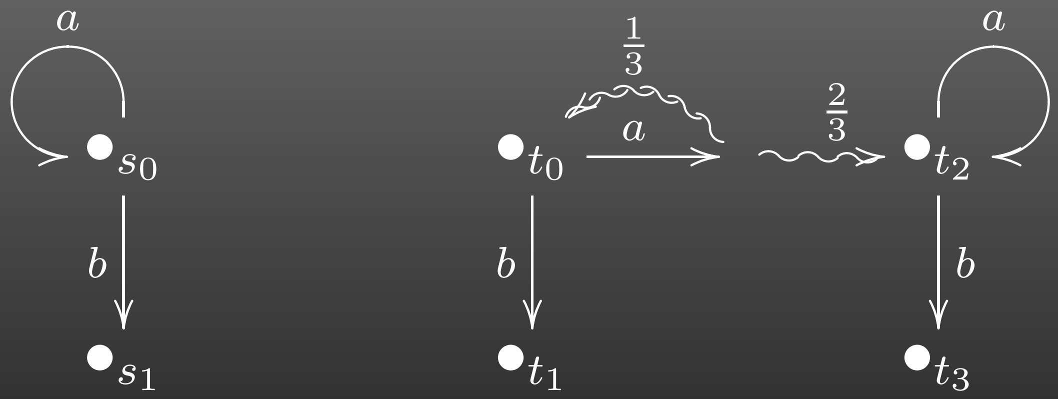
R - equivalence on states, is a bisimulation if



... two states are bisimilar if they are related by some bisimulation

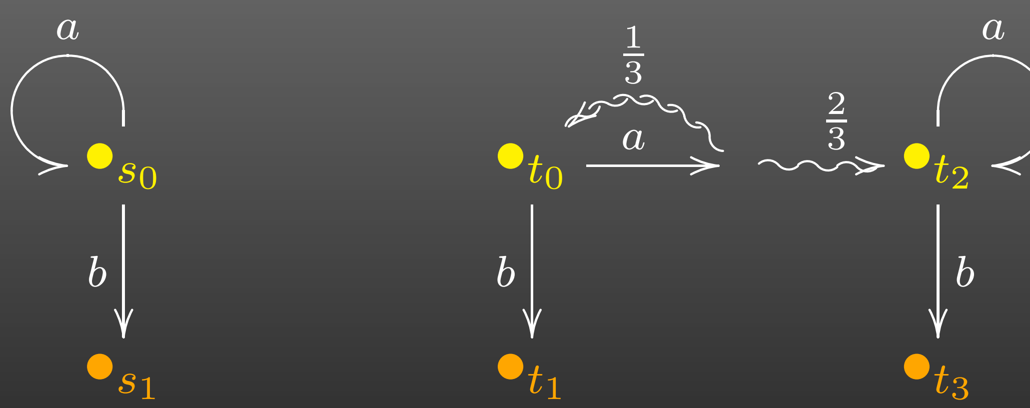
Bisimulation - simple Segala

Example: Consider the simple Segala systems



Bisimulation - simple Segala

Example: Consider the simple Segala systems



the coloring is a bisimulation, so s_0 and t_0 are bisimilar

Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that

Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that γ exists:

$$\begin{array}{ccccc} S & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & S \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \alpha} \\ \mathcal{F}S & \xleftarrow[\mathcal{F}\pi_1]{} & \mathcal{F}R & \xrightarrow[\mathcal{F}\pi_2]{} & \mathcal{F}S \end{array}$$

Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that

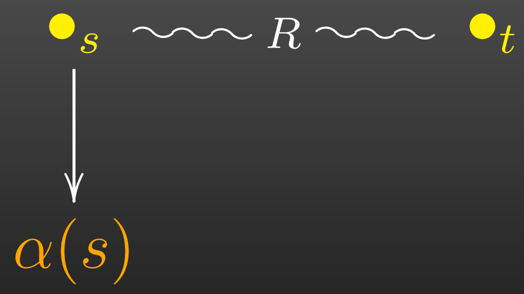
$$\bullet_s \rightsquigarrow R \rightsquigarrow \bullet_t$$

Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that

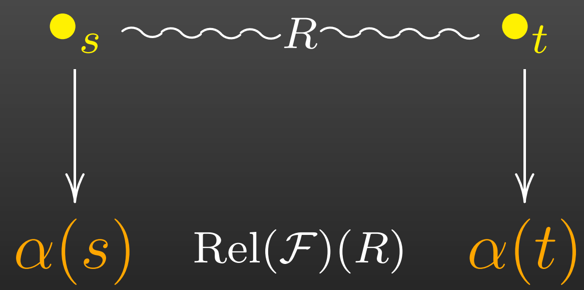


Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that



Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that

$$\begin{array}{ccc} \bullet_s & \sim\!\sim\!\sim\!\sim R \sim\!\sim\!\sim\!\sim & \bullet_t \\ \downarrow & & \downarrow \\ \alpha(s) & \mathrm{Rel}(\mathcal{F})(R) & \alpha(t) \end{array}$$

... two states are bisimilar if they are related by some bisimulation

Coalgebraic bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ such that

$$\begin{array}{ccc} \bullet_s & \sim\sim\sim R \sim\sim\sim & \bullet_t \\ \downarrow & & \downarrow \\ \alpha(s) & \mathrm{Rel}(\mathcal{F})(R) & \alpha(t) \end{array}$$

Theorem: Coalgebraic and concrete bisimilarity coincide for all probabilistic transition systems!

Expressiveness

When do we consider one type of system more expressive than another?

Expressiveness

When do we consider one type of system more expressive than another?

Example:

LTS $\mathcal{P}(A \times \_)$

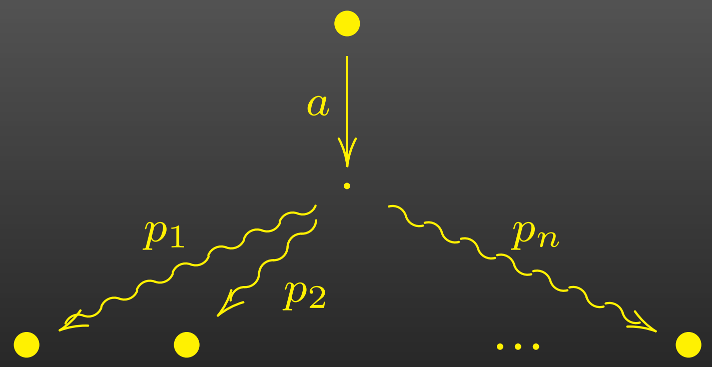
are clearly not more expressive than

Alternating Systems $\mathcal{D} + \mathcal{P}(A \times \_)$

Expressiveness

simple Segala system $\longrightarrow$ Segala system

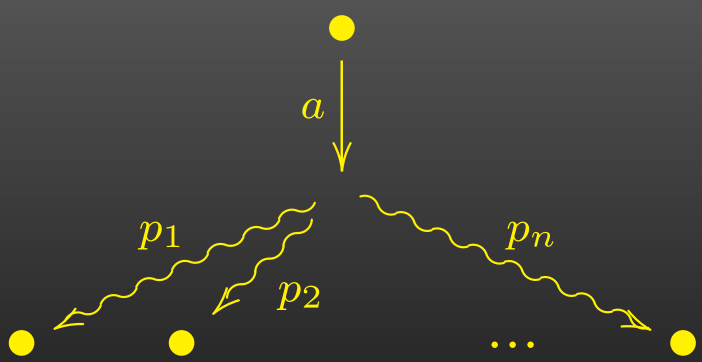
$\mathcal{P}(A \times \mathcal{D})$ $\quad\quad$ $\mathcal{PD}(A \times \_)$

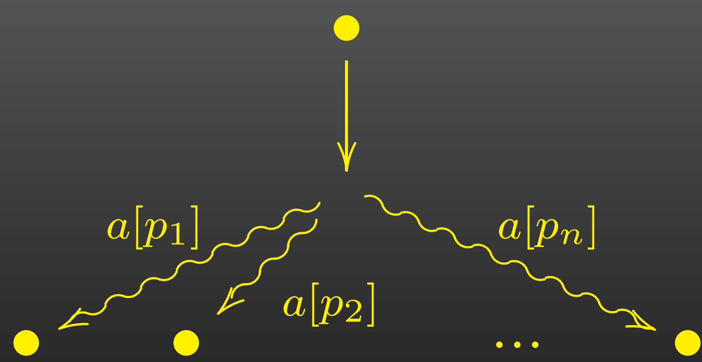


Expressiveness

simple Segala system $\rightarrow$ Segala system

$\mathcal{P}(A \times \mathcal{D})$ $\qquad$ $\mathcal{PD}(A \times \_)$

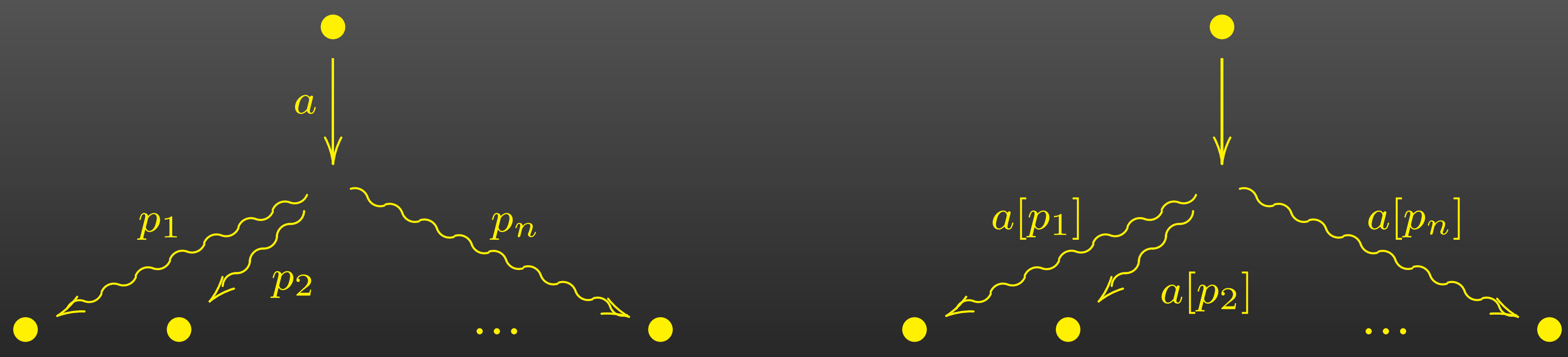




Expressiveness

simple Segala system $\longrightarrow$ Segala system

$\mathcal{P}(A \times \mathcal{D})$ $\qquad$ $\mathcal{PD}(A \times \_)$



When do we consider one type of systems more expressive than another?

Our expressiveness criterion

$\mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$

if there is a mapping $\langle S, \alpha : S \to \mathcal{F}S \rangle \overset{\mathcal{T}}{\mapsto} \langle S, \tilde{\alpha} : S \to \mathcal{G}S \rangle$ that preserves and reflects bisimilarity

Our expressiveness criterion

$\mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$

if there is a mapping $\langle S, \alpha : S \to \mathcal{F}S \rangle \stackrel{\mathcal{T}}{\mapsto} \langle S, \tilde{\alpha} : S \to \mathcal{G}S \rangle$ that preserves and reflects bisimilarity

$$s_{\langle S,\alpha \rangle} \sim t_{\langle T,\beta \rangle} \iff s_{\mathcal{T}\langle S,\alpha \rangle} \sim t_{\mathcal{T}\langle T,\beta \rangle}$$

Our expressiveness criterion

$$\mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$$

if there is a mapping $\langle S, \alpha : S \to \mathcal{F}S \rangle \stackrel{\mathcal{T}}{\mapsto} \langle S, \tilde{\alpha} : S \to \mathcal{G}S \rangle$ that preserves and reflects bisimilarity

$$s_{\langle S,\alpha \rangle} \sim t_{\langle T,\beta \rangle} \iff s_{\mathcal{T}\langle S,\alpha \rangle} \sim t_{\mathcal{T}\langle T,\beta \rangle}$$

Theorem: An injective natural transformation $\mathcal{F} \Rightarrow \mathcal{G}$ is sufficient for $\mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$

Proof idea

Generally, a natural transformation $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ gives rise to a translation of coalgebras $\mathcal{T}_\tau : \mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$ as follows:

$$\begin{array}{ccc} S & & S \\ & & \downarrow{\scriptstyle \alpha} \\ {\scriptstyle \alpha}\Big\downarrow & \overset{\mathcal{T}_\tau}{\Longrightarrow} & \mathcal{F}S \\ & & \downarrow{\scriptstyle \tau_S} \\ \mathcal{F}S & & \mathcal{G}S \end{array}$$

Proof idea

Generally, a natural transformation $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ gives rise to a translation of coalgebras $\mathcal{T}_\tau : \mathsf{Coalg}_{\mathcal{F}} \to \mathsf{Coalg}_{\mathcal{G}}$ as follows:

$$\begin{array}{c} S \\ \downarrow{\scriptstyle \alpha} \\ \mathcal{F}S \end{array} \quad \overset{\mathcal{T}_\tau}{\Longrightarrow} \quad \begin{array}{c} S \\ \downarrow{\scriptstyle \alpha} \\ \mathcal{F}S \\ \downarrow{\scriptstyle \tau_S} \\ \mathcal{G}S \end{array}$$

this translation we use in the proof!

Preservation - proof

The translation $\mathcal{T}_\tau$ preserves bisimulations:

A bisimulation $R \subseteq S \times T$ between $\langle S, \alpha \rangle$ and $\langle T, \beta \rangle$

$$\begin{array}{ccccc} S & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & T \\ \downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \beta} \\ \mathcal{F}S & \xleftarrow{\mathcal{F}\pi_1} & \mathcal{F}R & \xrightarrow{\mathcal{F}\pi_2} & \mathcal{F}T \end{array}$$

Preservation - proof

The translation $\mathcal{T}_\tau$ preserves bisimulations:

A bisimulation $R \subseteq S \times T$ between $\langle S, \alpha \rangle$ and $\langle T, \beta \rangle$

$$
\begin{array}{ccccc}
S & \xleftarrow{\pi_1} & R & \xrightarrow{\pi_2} & T \\
{\scriptstyle \alpha}\downarrow & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \beta} \\
\mathcal{F}S & \xleftarrow{\mathcal{F}\pi_1} & \mathcal{F}R & \xrightarrow{\mathcal{F}\pi_2} & \mathcal{F}T \\
{\scriptstyle \tau_S}\downarrow & \text{nat. } \tau & \downarrow{\scriptstyle \tau_R} & \text{nat. } \tau & \downarrow{\scriptstyle \tau_T} \\
\mathcal{G}S & \xleftarrow[\mathcal{G}\pi_1]{} & \mathcal{G}R & \xrightarrow[\mathcal{G}\pi_2]{} & \mathcal{G}T
\end{array}
$$

is a bisimulation between $\mathcal{T}_\tau \langle S, \alpha \rangle$ and $\mathcal{T}_\tau \langle T, \beta \rangle$

Reflection?

But $\mathcal{T}_\tau$ need not reflect bisimilarity.

Example: The natural transformation

$$\widetilde{\mathsf{supp}} : \mathcal{D} + 1 \Rightarrow \mathcal{P}$$

that forgets the probabilities does not reflect.

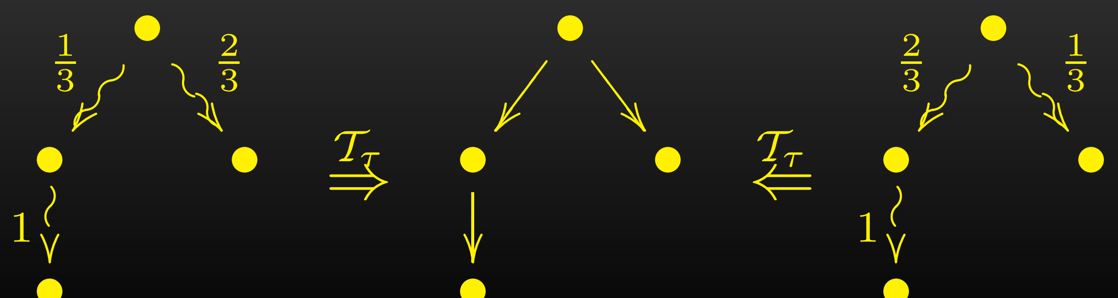
Reflection?

But $\mathcal{T}_\tau$ need not reflect bisimilarity.

Example: The natural transformation

$$\widetilde{\mathsf{supp}} : \mathcal{D} + 1 \Rightarrow \mathcal{P}$$

that forgets the probabilities does not reflect.



Reflection?

But $\mathcal{T}_\tau$ need not reflect bisimilarity.

Example: The natural transformation

$$\widetilde{\mathsf{supp}} : \mathcal{D} + 1 \Rightarrow \mathcal{P}$$

that forgets the probabilities does not reflect.

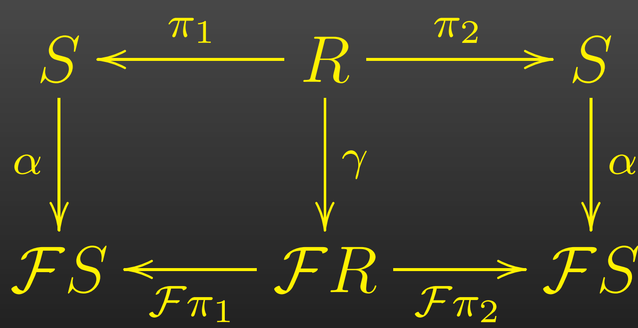
Injectivity turns out to be sufficient for reflection via cocongruences - behavioural equivalence

Recall bisimulation

A bisimulation on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $R \subseteq S \times S$ or (R, π_1, π_2) such that γ exists:



... two states are bisimilar if they are related by some bisimulation

Cocongruence

A cocongruence on

$$\langle S, \alpha : S \longrightarrow \mathcal{F}S \rangle$$

is $(Q, q_1 : S \longrightarrow Q, q_2 : S \longrightarrow Q)$ such that

Cocongruence

A cocongruence on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $(Q, q_1 : S \to Q, q_2 : S \to Q)$ such that γ exists:

$$\begin{array}{ccccc}
S & \xrightarrow{q_1} & Q & \xleftarrow{q_2} & S \\
\downarrow{\scriptstyle \alpha} & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \beta} \\
\mathcal{F}S & \xrightarrow[\mathcal{F}q_1]{} & \mathcal{F}Q & \xleftarrow[\mathcal{F}q_2]{} & \mathcal{F}S
\end{array}$$

Cocongruence

A cocongruence on

$$\langle S, \alpha : S \to \mathcal{F}S \rangle$$

is $(Q, q_1 : S \to Q, q_2 : S \to Q)$ such that γ exists:

$$\begin{array}{ccccc} S & \xrightarrow{q_1} & Q & \xleftarrow{q_2} & S \\ {\scriptstyle \alpha}\downarrow & & \downarrow{\scriptstyle \gamma} & & \downarrow{\scriptstyle \beta} \\ \mathcal{F}S & \xrightarrow[\mathcal{F}q_1]{} & \mathcal{F}Q & \xleftarrow[\mathcal{F}q_2]{} & \mathcal{F}S \end{array}$$

... two states are behaviourly equivalent if they are related by some cocongruence $s \approx t \iff q_1(s) = q_2(t)$

Results

- The translation $\mathcal{T}_\tau$ preserves behavioural equivalence just like it preserves bisimilarity

Results

- The translation $\mathcal{T}_\tau$ preserves behavioural equivalence just like it preserves bisimilarity
- If $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ has injective components, then $\mathcal{T}_\tau$ reflects behavioural equivalence

Results

- The translation $\mathcal{T}_\tau$ preserves behavioural equivalence just like it preserves bisimilarity
- If $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ has injective components, then $\mathcal{T}_\tau$ reflects behavioural equivalence

* The proof uses that $\mathbf{Sets}$ has a factorization system with a diagonal fill-in

The semantics

- bisimilarity $\subseteq$ behavioural equivalence
  in $\mathbb{S}\mathrm{ets}$ since pushouts exist

The semantics

- bisimilarity $\subseteq$ behavioural equivalence
  in Sets since pushouts exist

- behavioural equivalence $\subseteq$ bisimilarity
  if $\mathcal{F}$ preserves weak pullbacks

The semantics

- bisimilarity $\subseteq$ behavioural equivalence
  in $\mathbb{S}\mathrm{ets}$ since pushouts exist

- behavioural equivalence $\subseteq$ bisimilarity
  if $\mathcal{F}$ preserves weak pullbacks

Hence, the theorem holds:

If $\mathcal{F}$ preserves w.p. and $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ is injective, then $\mathcal{T}_\tau$ preserves and reflects bisimilarity.

The semantics

- bisimilarity $\subseteq$ behavioural equivalence
  in $\mathbb{S}\mathrm{ets}$ since pushouts exist

- behavioural equivalence $\subseteq$ bisimilarity
  if $\mathcal{F}$ preserves weak pullbacks

Hence, the theorem holds:

If $\mathcal{F}$ preserves w.p. and $\tau : \mathcal{F} \Rightarrow \mathcal{G}$ is injective, then $\mathcal{T}_\tau$ preserves and reflects bisimilarity.

All our functors preserve w.p.

Some basic transformations

Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,

Some basic transformations

Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,

Some basic transformations

Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,
- $\delta : \_ \Rightarrow \mathcal{D}$ with $\delta_X(x) := \delta_x$ (*Dirac*),

Some basic transformations

Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,
- $\delta : \_ \Rightarrow \mathcal{D}$ with $\delta_X(x) := \delta_x$ (*Dirac*),
- $\iota_l : \mathcal{F} \Rightarrow \mathcal{F} + \mathcal{G}$ and $\iota_r : \mathcal{G} \Rightarrow \mathcal{F} + \mathcal{G}$,

Some basic transformations

Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,
- $\delta : \_ \Rightarrow \mathcal{D}$ with $\delta_X(x) := \delta_x$ (*Dirac*),
- $\iota_l : \mathcal{F} \Rightarrow \mathcal{F} + \mathcal{G}$ and $\iota_r : \mathcal{G} \Rightarrow \mathcal{F} + \mathcal{G}$,
- $\phi + \psi : \mathcal{F} + \mathcal{G} \Rightarrow \mathcal{F}' + \mathcal{G}'$ for
  $\phi : \mathcal{F} \Rightarrow \mathcal{F}'$ and $\psi : \mathcal{G} \Rightarrow \mathcal{G}'$ (both with i.c.),

Some basic transformations

Examples of injective natural transformations:

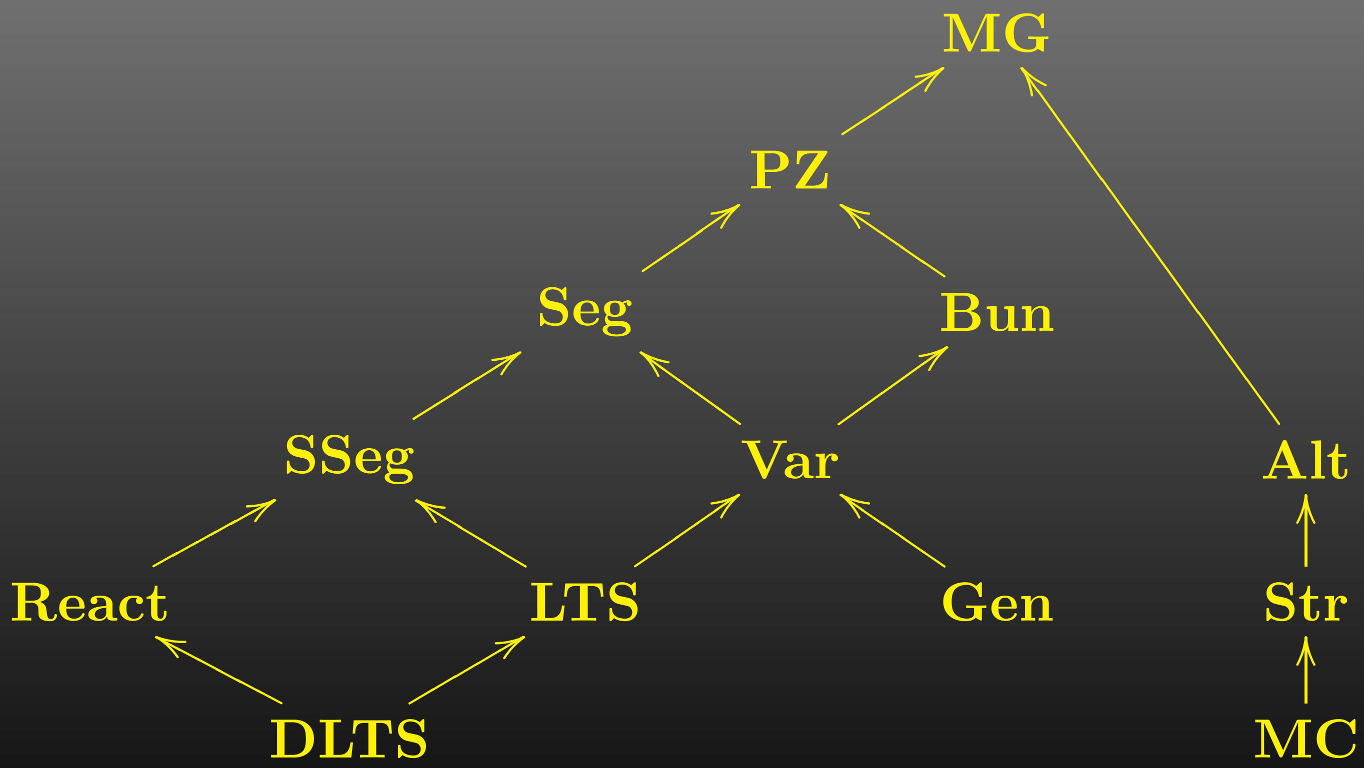
- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,
- $\delta : \_ \Rightarrow \mathcal{D}$ with $\delta_X(x) := \delta_x$ (*Dirac*),
- $\iota_l : \mathcal{F} \Rightarrow \mathcal{F} + \mathcal{G}$ and $\iota_r : \mathcal{G} \Rightarrow \mathcal{F} + \mathcal{G}$,
- $\phi + \psi : \mathcal{F} + \mathcal{G} \Rightarrow \mathcal{F}' + \mathcal{G}'$ for
  $\phi : \mathcal{F} \Rightarrow \mathcal{F}'$ and $\psi : \mathcal{G} \Rightarrow \mathcal{G}'$ (both with i.c.),
- $\kappa : \mathrm{A} \times \mathcal{P} \Rightarrow \mathcal{P}(\mathrm{A} \times \_)$ with $\kappa_X(a, M) := \{\langle a, x\rangle \mid x \in M\}$,

Some basic transformations

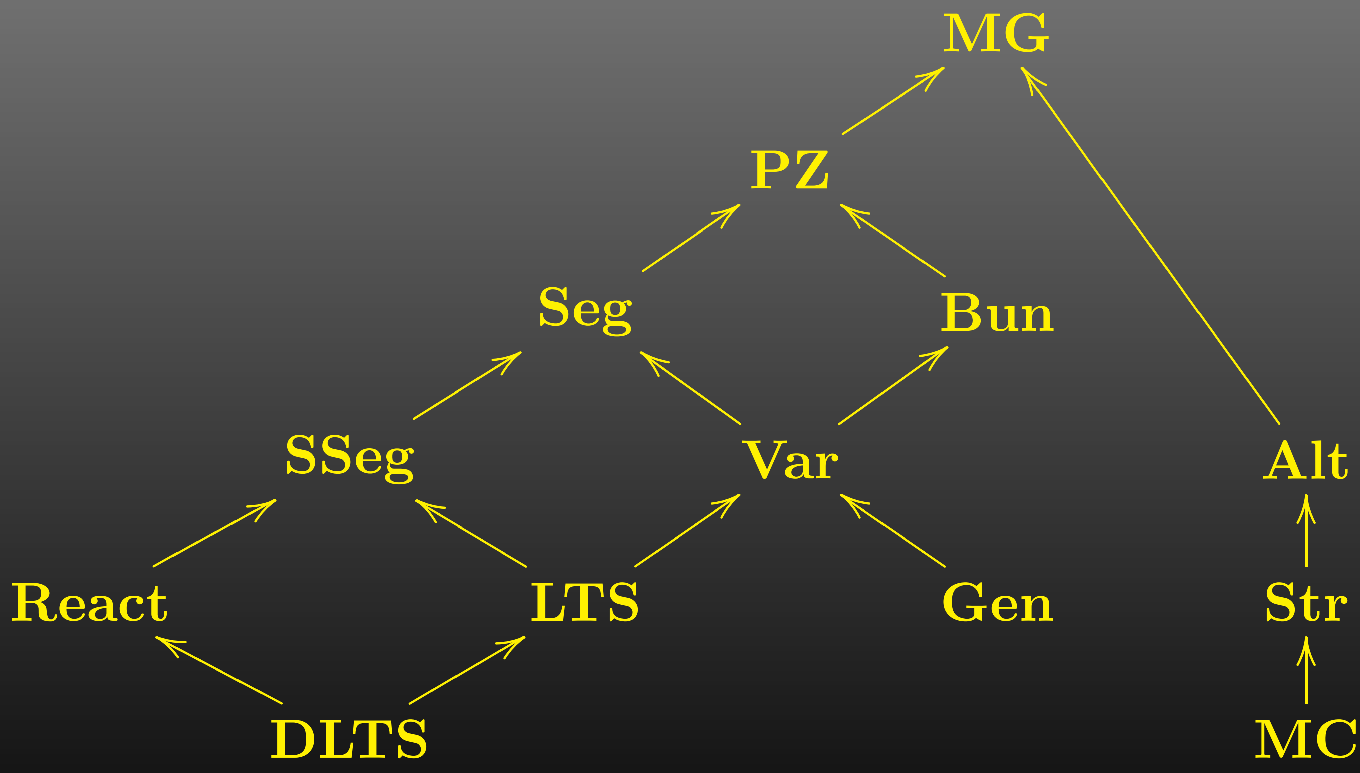
Examples of injective natural transformations:

- $\eta : 1 \Rightarrow \mathcal{P}$ with $\eta_X(*) := \emptyset$,
- $\sigma : \_ \Rightarrow \mathcal{P}$ with $\sigma_X(x) := \{x\}$,
- $\delta : \_ \Rightarrow \mathcal{D}$ with $\delta_X(x) := \delta_x$ (*Dirac*),
- $\iota_l : \mathcal{F} \Rightarrow \mathcal{F} + \mathcal{G}$ and $\iota_r : \mathcal{G} \Rightarrow \mathcal{F} + \mathcal{G}$,
- $\phi + \psi : \mathcal{F} + \mathcal{G} \Rightarrow \mathcal{F}' + \mathcal{G}'$ for
  $\phi : \mathcal{F} \Rightarrow \mathcal{F}'$ and $\psi : \mathcal{G} \Rightarrow \mathcal{G}'$ (both with i.c.),
- $\kappa : \mathrm{A} \times \mathcal{P} \Rightarrow \mathcal{P}(\mathrm{A} \times \_)$ with $\kappa_X(a, M) := \{\langle a, x\rangle \mid x \in M\}$,
- $\ldots$

The hierarchy



The hierarchy



* Falk Bartels, Ana Sokolova, Erik de Vink, TCS 327

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

objects in $\mathbf{Meas}$: measure spaces (X, S_X)

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

objects in $\mathbf{Meas}$: measure spaces (X, S_X)

with S_X a σ-algebra

closed w.r.t. $\emptyset$, complements, countable unions

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

objects in $\mathbf{Meas}$: measure spaces (X, S_X)

with S_X a σ-algebra

closed w.r.t. $\emptyset$, complements, countable unions

arrows in $\mathbf{Meas}$: measurable functions

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

objects in $\mathbf{Meas}$: measure spaces (X, S_X)

with S_X a σ-algebra

closed w.r.t. $\emptyset$, complements, countable unions

arrows in $\mathbf{Meas}$: measurable functions

$f : X \to Y$ with $f^{-1}(S_Y) \subseteq S_X$

Markov processes

are beyond discrete probabilities, live in $\mathbf{Meas}$ beyond $\mathbf{Sets}$

objects in $\mathbf{Meas}$: measure spaces (X, S_X)

with S_X a σ-algebra

closed w.r.t. $\emptyset$, complements, countable unions

arrows in $\mathbf{Meas}$: measurable functions

$f : X \to Y$ with $f^{-1}(S_Y) \subseteq S_X$

Markov processes are Giry-coalgebras in Meas!

Markov and Giry

The Giry functor (monad) on $\mathbf{Meas}$ is given on objects by

$$\mathcal{G}(X, S_X) = (\mathcal{G}X, S_{\mathcal{G}X})$$

Markov and Giry

The Giry functor (monad) on **Meas** is given on objects by

$$\mathcal{G}(X, S_X) = (\mathcal{G}X, S_{\mathcal{G}X})$$

with $\mathcal{G}X$ containing all subprobability measures

Markov and Giry

The Giry functor (monad) on $\mathbf{Meas}$ is given on objects by

$$\mathcal{G}(X, S_X) = (\mathcal{G}X, S_{\mathcal{G}X})$$

with $\mathcal{G}X$ containing all subprobability measures

and the smallest σ-algebra making the evaluation maps

$ev_M : \mathcal{G}X \to [0,1], \quad ev_M(\phi) = \phi(M)$ measurable

Markov and Giry

The Giry functor (monad) on **Meas** is given on objects by

$$\mathcal{G}(X, S_X) = (\mathcal{G}X, S_{\mathcal{G}X})$$

with $\mathcal{G}X$ containing all subprobability measures

and the smallest σ-algebra making the evaluation maps

$ev_M : \mathcal{G}X \to [0,1], \quad ev_M(\phi) = \phi(M)$ measurable

and on arrows

$$\mathcal{G}(f)\Big(S_X \xrightarrow{\varphi} [0,1]\Big) \quad = \quad \Big(S_Y \xrightarrow{f^{-1}} S_X \xrightarrow{\varphi} [0,1]\Big)$$

Chains vs. processes

The situation is

$$\mathcal{G} \circlearrowright \mathbf{Meas} \underset{D}{\overset{U}{\rightleftarrows}} \mathbf{Sets} \circlearrowleft \mathcal{D} \qquad \text{with} \quad D \dashv U$$

with an obvious natural transformation $\rho : \mathcal{D}U \Rightarrow U\mathcal{G}$

Chains vs. processes

The situation is

$$\mathcal{G} \circlearrowright \mathbf{Meas} \underset{D}{\overset{U}{\rightleftarrows}} \mathbf{Sets} \circlearrowleft \mathcal{D} \qquad \text{with} \quad D \dashv U$$

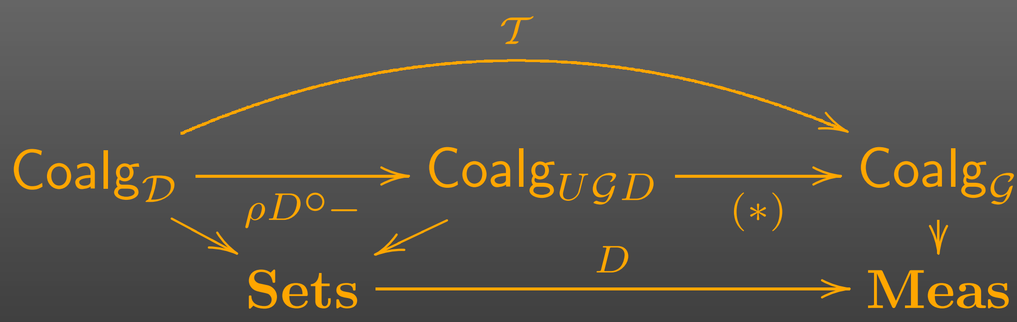
with an obvious natural transformation $\rho : \mathcal{D}U \Rightarrow U\mathcal{G}$

and we can translate chains into processes

$$\begin{array}{ccccc}
 & & \overset{\mathcal{T}}{\frown} & & \\
\mathsf{Coalg}_{\mathcal{D}} & \xrightarrow[\rho D \circ -]{} & \mathsf{Coalg}_{U\mathcal{G}D} & \xrightarrow[(*)]{} & \mathsf{Coalg}_{\mathcal{G}} \\
\searrow & & \swarrow & & \downarrow \\
 & \mathbf{Sets} & & \xrightarrow{D} & \mathbf{Meas}
\end{array}$$

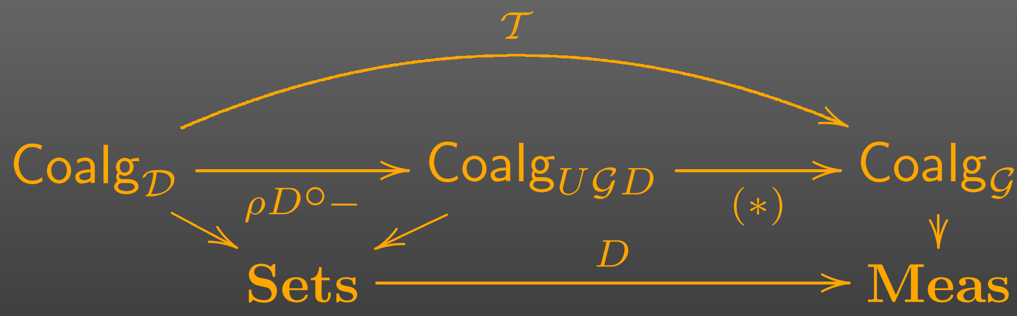
Chains vs. processes

So we can translate chains into processes



Chains vs. processes

So we can translate chains into processes



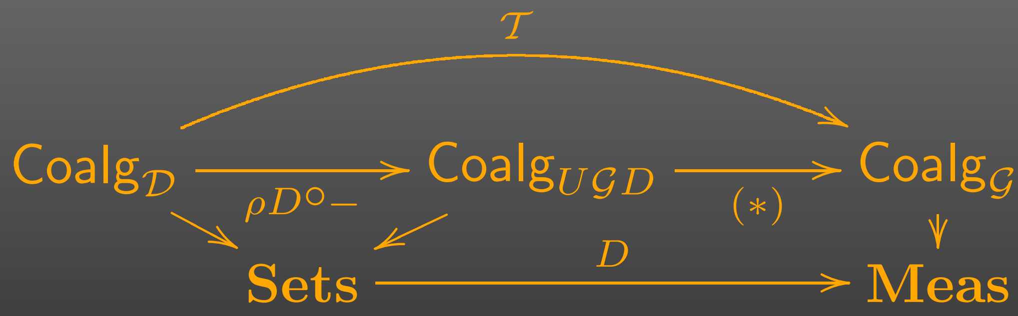
with $\left(X \xrightarrow{c} \mathcal{D}(X) = \mathcal{D}UD(X)\right) \longmapsto \left(X \xrightarrow{c} \mathcal{D}UD(X) \xrightarrow{\rho_{DX}} U\mathcal{G}D(X)\right)$

and $(*)$ from

$$\frac{X \longrightarrow U\mathcal{G}D(X) \quad \text{in } \mathbf{Sets}}{D(X) \longrightarrow \mathcal{G}D(X) \quad \text{in } \mathbf{Meas}}$$

Chains vs. processes

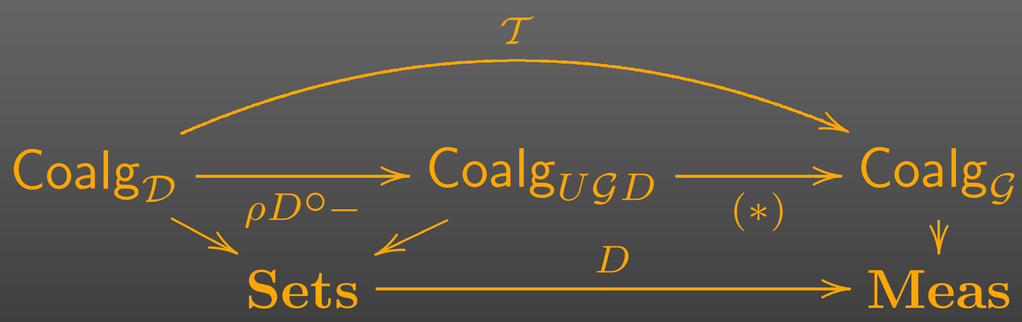
So we can translate chains into processes



Theorem: The translation $\mathcal{T}$ preserves and reflects behavioral equivalence (bisimilarity does not work here)

Chains vs. processes

So we can translate chains into processes



Theorem: The translation $\mathcal{T}$ preserves and reflects behavioral equivalence (bisimilarity does not work here)

Hence: $\mathbf{MC} = \mathsf{Coalg}_{\mathcal{D}}^{\mathbf{Sets}} \longrightarrow \mathsf{Coalg}_{\mathcal{G}}^{\mathbf{Meas}} = \mathbf{MP}$

Conclusions

- probabilistic models are coalgebras

Conclusions

- probabilistic models are coalgebras
- coalgebras are beautiful :-)

Conclusions

- probabilistic models are coalgebras
- coalgebras are beautiful :-)
- an expressiveness comparison can be made

Conclusions

- probabilistic models are coalgebras
- coalgebras are beautiful :-)
- an expressiveness comparison can be made
- expressiveness hierarchy w.r.t bisimilarity/beh.equivalence follows

Conclusions

- probabilistic models are coalgebras
- coalgebras are beautiful :-)
- an expressiveness comparison can be made
- expressiveness hierarchy w.r.t bisimilarity/beh.equivalence follows
- it also works beyond $\mathbf{Sets}$

Conclusions

- probabilistic models are coalgebras
- coalgebras are beautiful :-)
- an expressiveness comparison can be made
- expressiveness hierarchy w.r.t bisimilarity/beh.equivalence follows
- it also works beyond $\mathbf{Sets}$
- future work: build another floor in $\mathbf{Meas}$