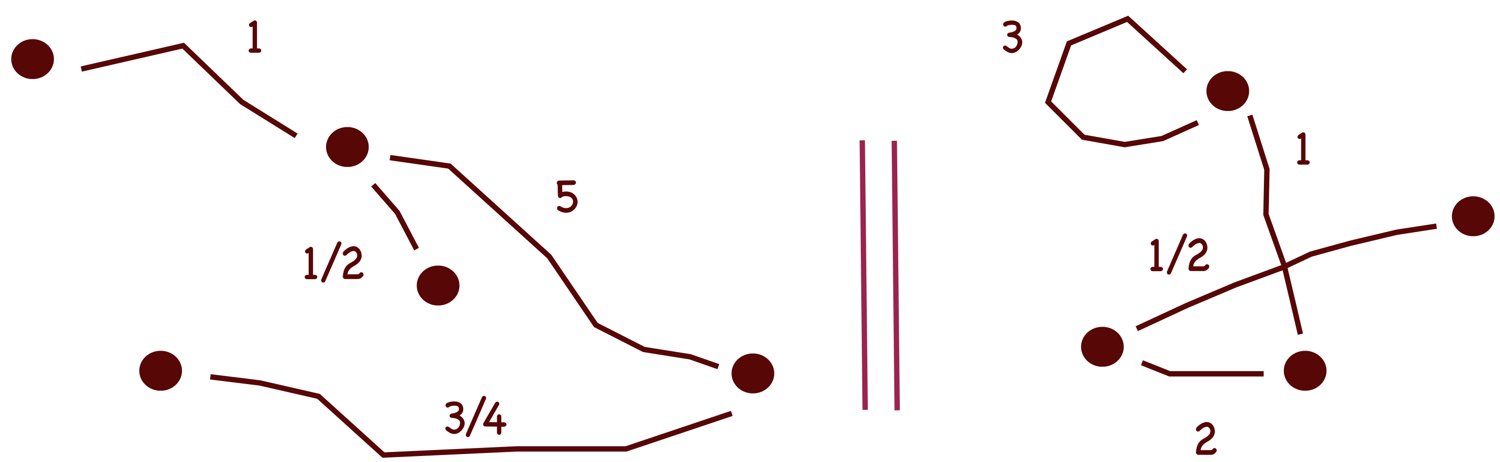
Probabilistic Systems Semantics via Coalgebra

Ana Sokolova UNIVERSITY of SALZBURG

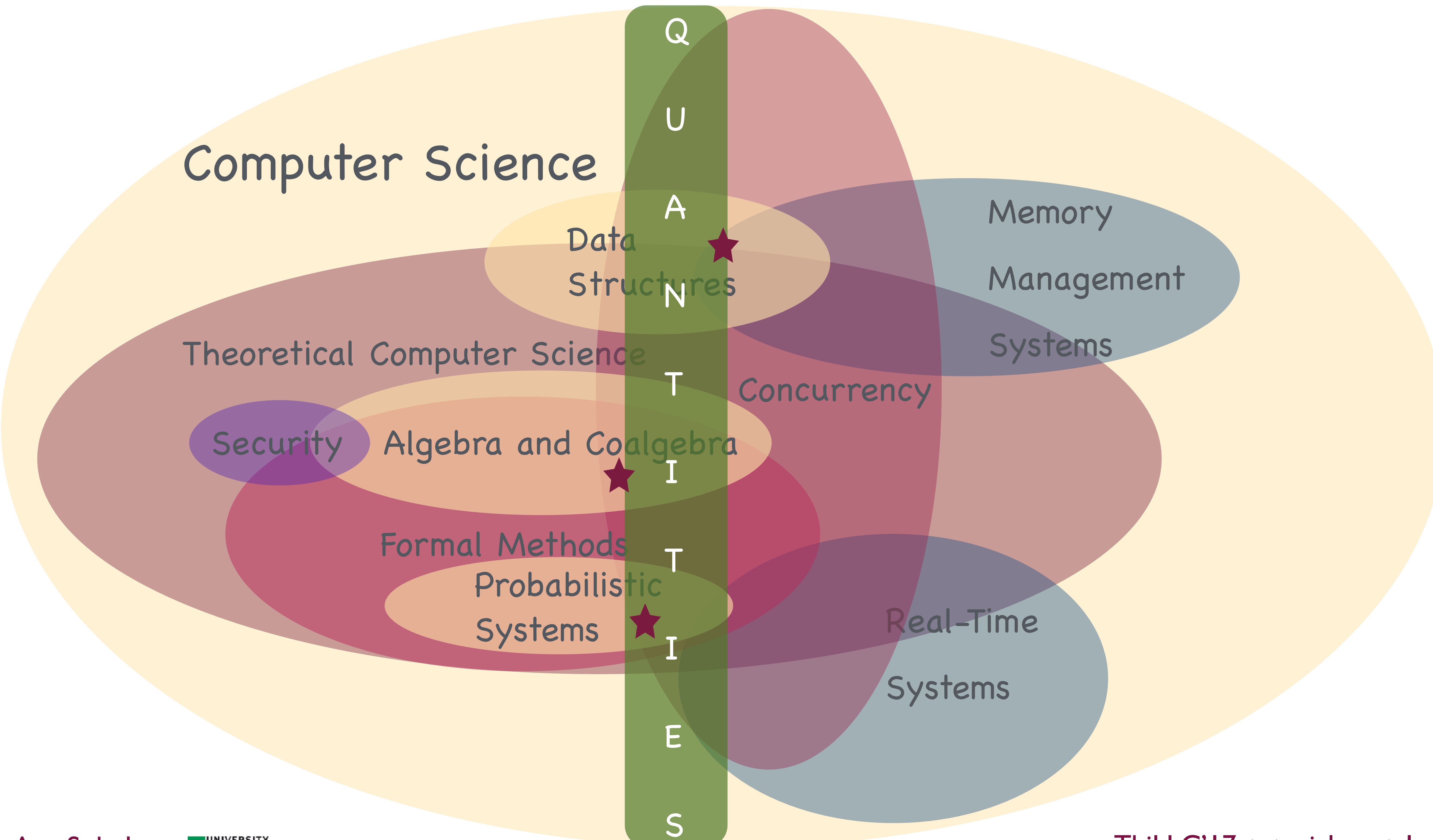


TbiLLC'17 tutorial, part 1

Rigorous methods
for
engineering of
and
reasoning about
reactive systems

probabilistic

My background



In this tutorial:

probabilistic systems semantics
using (co)algebra

In this tutorial:

models

probabilistic systems semantics
using (co)algebra

state-based
automata-like

In this tutorial:

models

probabilistic systems semantics
using (co)algebra

In this tutorial:

state-based automata-like

models

relations on states

probabilistic systems semantics using (co)algebra

In this tutorial:

state-based automata-like

models

behaviour semantics

relations on states

probabilistic systems semantics using (co)algebra

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

to identify states with the same behaviour

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

to identify states with the same behaviour

to order states according to behaviour

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

to identify states with the same behaviour

to order states according to behaviour

there are many of them: bisimilarity, trace,…

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

to identify states with the same behaviour

to order states according to behaviour

there are many of them: bisimilarity, trace,…

$$\text{Sys} \overset{?}{\approx} \text{Spec}$$

Behaviour semantics are used for verification:

- Behaviour equivalence $\approx$
- Behaviour preorder $\sqsubseteq$

to identify states with the same behaviour

to order states according to behaviour

there are many of them: bisimilarity, trace,…

Sys $\overset{?}{\approx}$ Spec

Sys $\overset{?}{\sqsubseteq}$ Spec

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

Part 2. Traces, linear-time semantics

Part 3. Belief-state-transformer semantics via convexity

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

bisimilarity

Part 2. Traces, linear-time semantics

Part 3. Belief-state-transformer semantics via convexity

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

bisimilarity

Part 2. Traces, linear-time semantics

trace equivalence

Part 3. Belief-state-transformer semantics via convexity

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

bisimilarity

Part 2. Traces, linear-time semantics

trace equivalence

Part 3. Belief-state-transformer semantics via convexity

distribution bisimilarity

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

bisimilarity

Part 2. Traces, linear-time semantics

trace equivalence

Part 3. Belief-state-transformer semantics via convexity

distribution bisimilarity

all with help of coalgebra

Plan:

Part 1. Modelling probabilistic systems for branching-time semantics

bisimilarity

Part 2. Traces, linear-time semantics

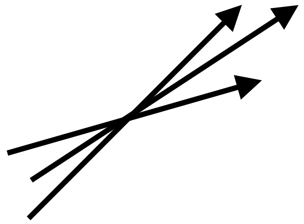
trace equivalence

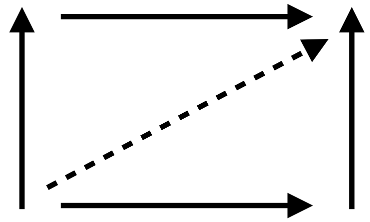
Part 3. Belief-state-transformer semantics via convexity

distribution bisimilarity

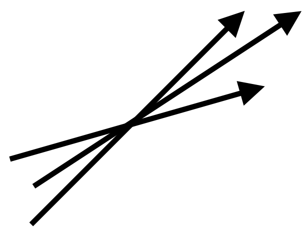
Mathematical framework based on category theory for state-based systems semantics

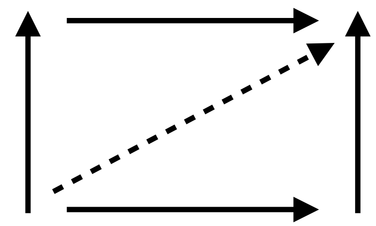
all with help of coalgebra





Highlights





Highlights

F. Bartels, E. de Vink, A. S. A hierarchy of probabilistic system types TCS'04, CMCS'03

A.S. Coalgebraic analysis of probabilistic systems PhD thesis, TU Eindhoven'05

A. S. Probabilistic systems coalgebraically TCS'11

B. Jacobs, I. Hasuo, A. S. Generic trace semantics via coinduction LMCS'07

B. Jacobs, I. Hasuo, A. S. The microcosm principle and concurrency in coalgebra FoSSaCS'08

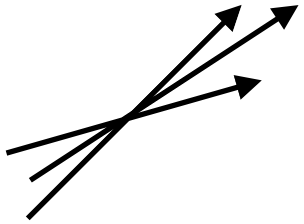
A. Silva, A. S. Sound and complete axiomatisation of trace semantics for probabilistic systems MFPS'11

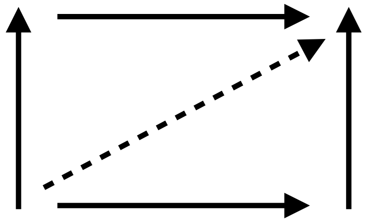
B. Jacobs, A. Silva, A. S. Trace semantics via determinization JSS'15

A. S., H. Woracek Congruences of convex algebras JPAA'15

A. S., H. Woracek Termination in convex sets of distributions CALCO'17

F. Bonchi, A. Silva, A. S. The power of convex algebras CONCUR'17





Source Wikipedia, by Diacritica - Own work

Highlights

F. Bartels, E. de Vink, A. S. A hierarchy of probabilistic system types TCS'04, CMCS'03

A.S. Coalgebraic analysis of probabilistic systems PhD thesis, TU Eindhoven'05

A. S. Probabilistic systems coalgebraically TCS'11

B. Jacobs, I. Hasuo, A. S. Generic trace semantics via coinduction LMCS'07

B. Jacobs, I. Hasuo, A. S. The microcosm principle and concurrency in coalgebra FoSSaCS'08

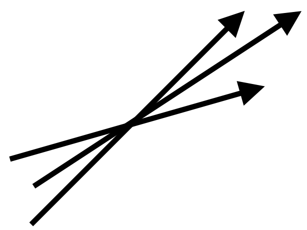
A. Silva, A. S. Sound and complete axiomatisation of trace semantics for probabilistic systems MFPS'11

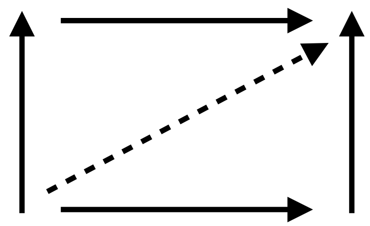
B. Jacobs, A. Silva, A. S. Trace semantics via determinization JSS'15

A. S., H. Woracek Congruences of convex algebras JPAA'15

A. S., H. Woracek Termination in convex sets of distributions CALCO'17

F. Bonchi, A. Silva, A. S. The power of convex algebras CONCUR'17





Source Wikipedia, by Diacritica - Own work

Highlights

F. Bartels, E. de Vink, A. S. A hierarchy of probabilistic system types TCS'04, CMCS'03

A.S. Coalgebraic analysis of probabilistic systems PhD thesis, TU Eindhoven'05

A. S. Probabilistic systems coalgebraically TCS'11

B. Jacobs, I. Hasuo, A. S. Generic trace semantics via coinduction LMCS'07

B. Jacobs, I. Hasuo, A. S. The microcosm principle and concurrency in coalgebra FoSSaCS'08

A. Silva, A. S. Sound and complete axiomatisation of trace semantics for probabilistic systems MFPS'11

B. Jacobs, A. Silva, A. S. Trace semantics via determinization JSS'15

A. S., H. Woracek Congruences of convex algebras JPAA'15

A. S., H. Woracek Termination in convex sets of distributions CALCO'17

F. Bonchi, A. Silva, A. S. The power of convex algebras CONCUR'17

and some very new work

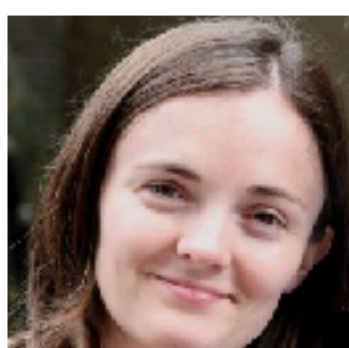
Joint work with

Erik de Vink **TU/e**

Bart Jacobs
Radboud University

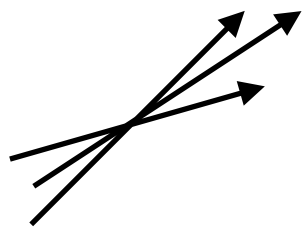
Ichiro Hasuo
THE UNIVERSITY OF TOKYO

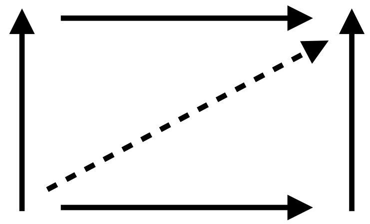
Harald Woracek TU WIEN



Alexandra Silva UCL

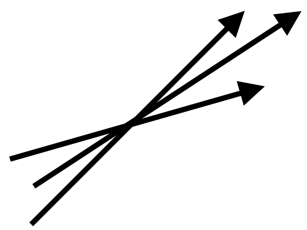
Filippo Bonchi ENS DE LYON

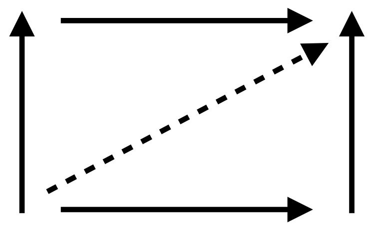




Part I

Modelling probabilistic systems for branching-time semantics





Part I

Modelling probabilistic systems
for
branching-time semantics

coalgebraically

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

states

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

states

object in the base category ***C***

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

states

behaviour type

object in the base category ***C***

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

states

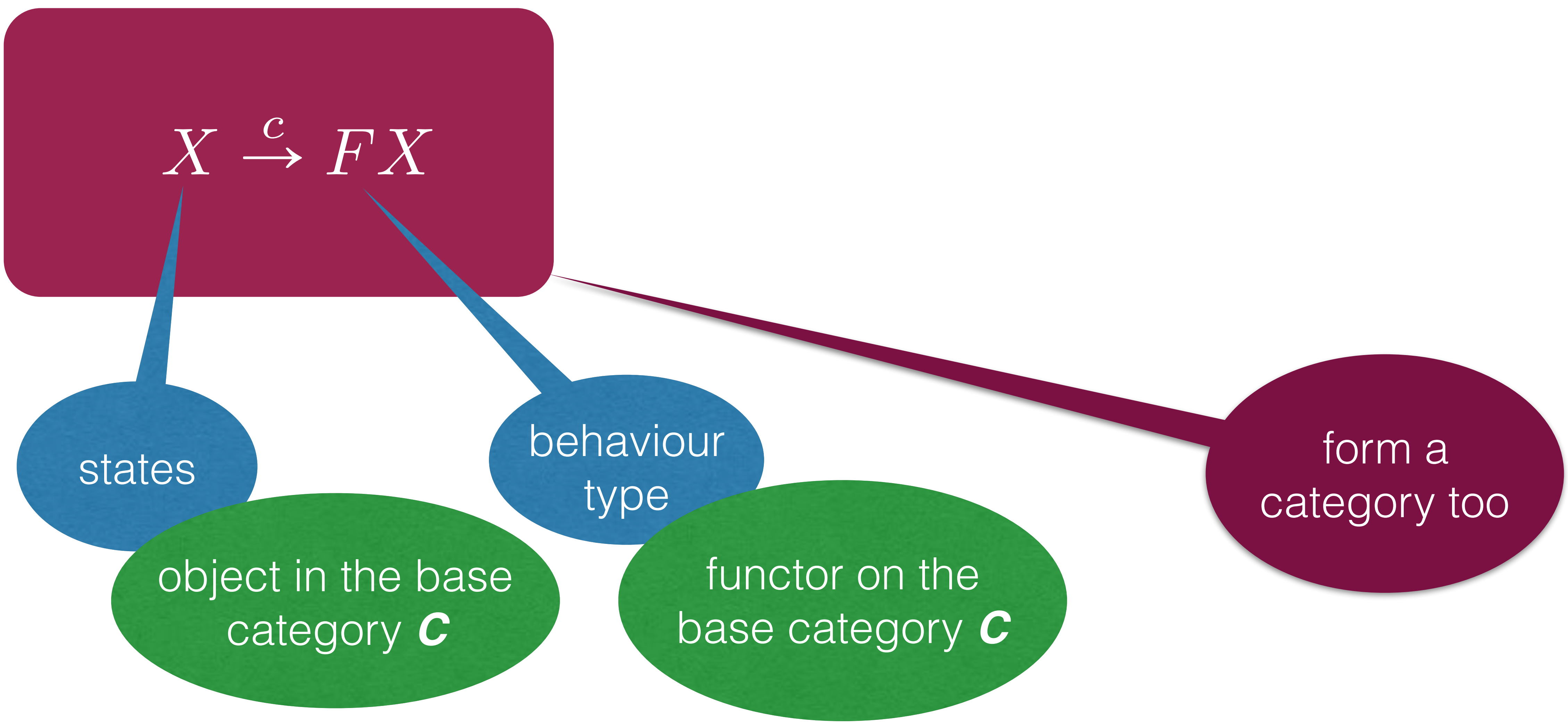
object in the base category ***C***

behaviour type

functor on the base category ***C***

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.



Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \xrightarrow{c} FX$$

states

object in the base category ***C***

behaviour type

functor on the base category ***C***

form a category too

$$\mathbf{CoAlg_C}(F)$$

Coalgebras

Uniform framework for dynamic transition systems, based on category theory.

$$X \stackrel{c}{\to} FX$$

states

object in the base category ***C***

behaviour type

functor on the base category ***C***

generic notion of behavioural equivalence

$\approx$

form a category too

$\mathbf{CoAlg}_{\mathbf{C}}(F)$

The category of F-coalgebras

$\mathbf{CoAlg_C}(F)$

Objects = coalgebras

Arrows = coalgebra homomorphisms

The category of F-coalgebras

$\mathbf{CoAlg_C}(F)$

Objects = coalgebras

$X \xrightarrow{c} FX$

Arrows = coalgebra homomorphisms

The category of F-coalgebras

$\mathbf{CoAlg_C}(F)$

Objects = coalgebras

Arrows = coalgebra homomorphisms

$$X \xrightarrow{c} FX$$

states

behaviour type

object in the base category ***C***

functor on the base category ***C***

The category of F-coalgebras

$\mathbf{CoAlg_C}(F)$

Objects = coalgebras

$$X \xrightarrow{c} FX$$

states

behaviour type

object in the base category ***C***

functor on the base category ***C***

Arrows = coalgebra homomorphisms

$$h \colon X \to Y$$

$$\begin{tikzcd} X \arrow[r, "h"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Fh"'] & FY \end{tikzcd}$$

The category of F-coalgebras

$\mathbf{CoAlg}_{\mathbf{C}}(F)$

Objects = coalgebras

$$X \xrightarrow{c} FX$$

states

behaviour type

object in the base category ***C***

functor on the base category ***C***

Arrows = coalgebra homomorphisms

$$h \colon X \to Y$$

$$\begin{tikzcd} X \arrow[r, "h"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Fh"'] & FY \end{tikzcd}$$

behaviour-preserving maps

Modelling discrete probabilistic systems

Almost all known probabilistic systems can be modelled as coalgebras on $\mathbf{Sets}$ for functors given by the following grammar:

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

Modelling discrete probabilistic systems

Almost all known probabilistic systems can be modelled as coalgebras on $\mathbf{Sets}$ for functors given by the following grammar:

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

$$X \xrightarrow{c} FX$$

Modelling discrete probabilistic systems

Almost all known probabilistic systems can be modelled as coalgebras on $\mathbf{Sets}$ for functors given by the following grammar:

probability distribution functor

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

$$X \xrightarrow{c} FX$$

Modelling discrete probabilistic systems

Almost all known probabilistic systems can be modelled as coalgebras on $\mathbf{Sets}$ for functors given by the following grammar:

probability distribution functor

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

$$X \xrightarrow{c} FX$$

in all cases concrete and coalgebraic bisimilarity (and behavioural equivalence) coincide

Example

Example

NFA

$2 \times (\mathcal{P}(-))^A \cong \mathcal{P}(1 + A \times (-))$

$$\begin{tikzcd} & x_1 \arrow[dl, "a"'] \arrow[dr, "a"] & \\ x_2 \arrow[d] & & x_3 \arrow[loop right, "b"] \\ * & & \end{tikzcd}$$

Modelling discrete probabilistic systems

Probability distribution functor on $\mathbf{Sets}$

$$\mathcal{D}X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1\}$$

for $f\colon X \to Y$ we have $\mathcal{D}f\colon \mathcal{D}X \to \mathcal{D}Y$ by

$$\mathcal{D}f(\mu)(y) = \sum_{x \in f^{-1}(y)} \mu(x) = \mu(f^{-1}(y))$$

Modelling discrete probabilistic systems

Probability distribution functor on $\mathbf{Sets}$

$$\mathcal{D}X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1\}$$

and its variants

$$\mathcal{D}_{\leqslant 1}X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) \leqslant 1\}$$

$$\mathcal{D}_f X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1, \mathrm{supp}(\mu) \text{ is finite}\}$$

Modelling discrete probabilistic systems

Probability distribution functor on $\mathbf{Sets}$

$$\mathcal{D}X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1\}$$

and its variants

$$\mathcal{D}_{\leqslant 1}X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) \leqslant 1\}$$

$\{x \in X \mid \mu(x) \neq 0\}$

$$\mathcal{D}_f X = \{\mu\colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1, \mathrm{supp}(\mu) \text{ is finite}\}$$

Modelling discrete probabilistic systems

Probability distribution functor on $\mathbf{Sets}$

$$\mathcal{D}X = \{\mu \colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1\}$$

the support
is
always
discrete

and its variants

$$\mathcal{D}_{\leqslant 1}X = \{\mu \colon X \to [0,1] \mid \sum_{x \in X} \mu(x) \leqslant 1\}$$

$\{x \in X \mid \mu(x) \neq 0\}$

$$\mathcal{D}_f X = \{\mu \colon X \to [0,1] \mid \sum_{x \in X} \mu(x) = 1, \mathrm{supp}(\mu) \text{ is finite}\}$$

Examples

Examples

NFA

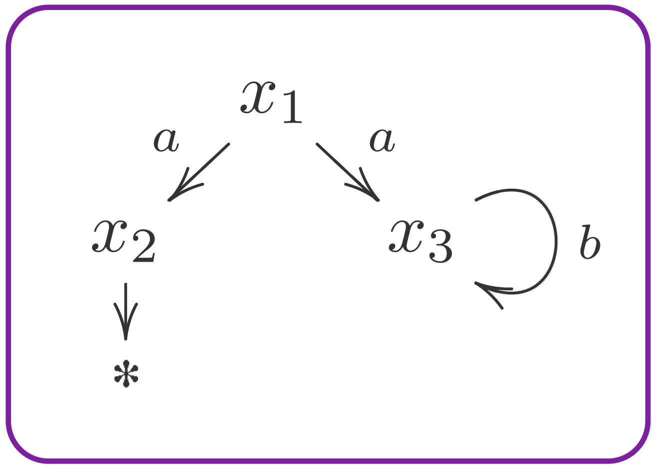
$$2 \times (\mathcal{P}(-))^A \;\cong\; \mathcal{P}(1 + A \times (-))$$

$$\begin{tikzcd}
& x_1 \arrow[dl, "a"'] \arrow[dr, "a"] & \\
x_2 \arrow[d] & & x_3 \arrow[loop right, "b"] \\
* & &
\end{tikzcd}$$

Examples

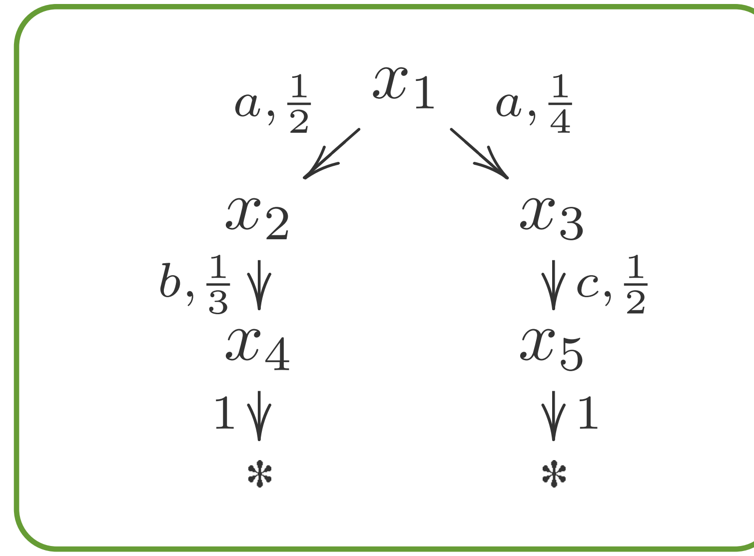
NFA

$2 \times (\mathcal{P}(-))^A \cong \mathcal{P}(1 + A \times (-))$



Generative PTS

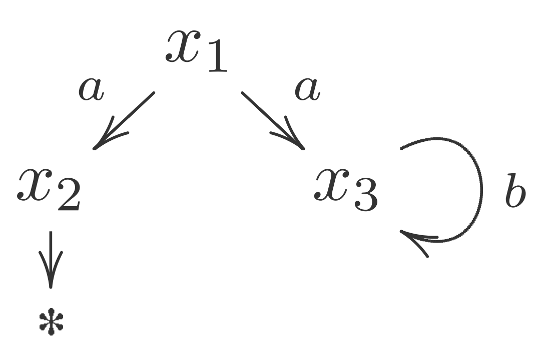
$\mathcal{D}_{\leq 1}(1 + A \times (-))$



Examples

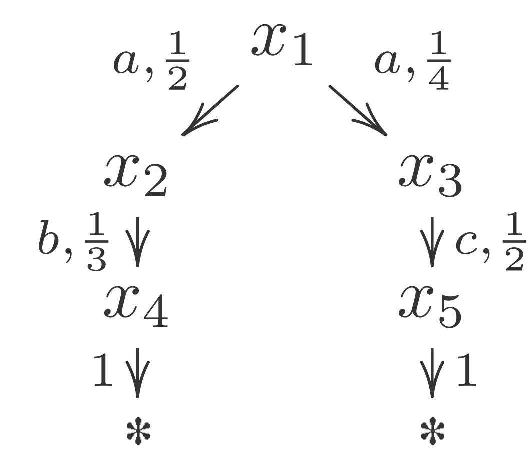
NFA

$2 \times (\mathcal{P}(-))^A \cong \mathcal{P}(1 + A \times (-))$



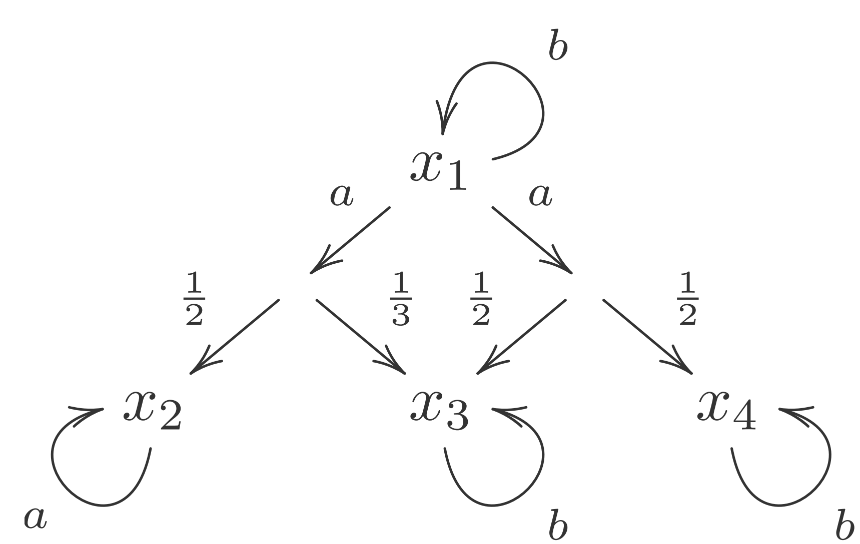
Generative PTS

$\mathcal{D}_{\leq 1}(1 + A \times (-))$



Simple PA

$\mathcal{P}(A \times \mathcal{D}_{\leq 1}(-))$



Expressiveness hierarchy

F.Bartels, A.S., E. de Vink '03/'04

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

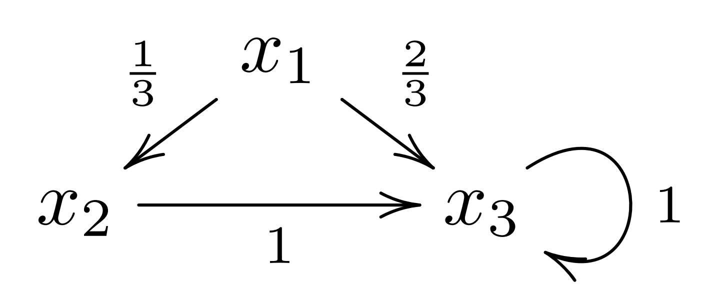
Expressiveness hierarchy

F.Bartels, A.S., E. de Vink '03/'04

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

Markov chain

$X \to \mathcal{D}X$



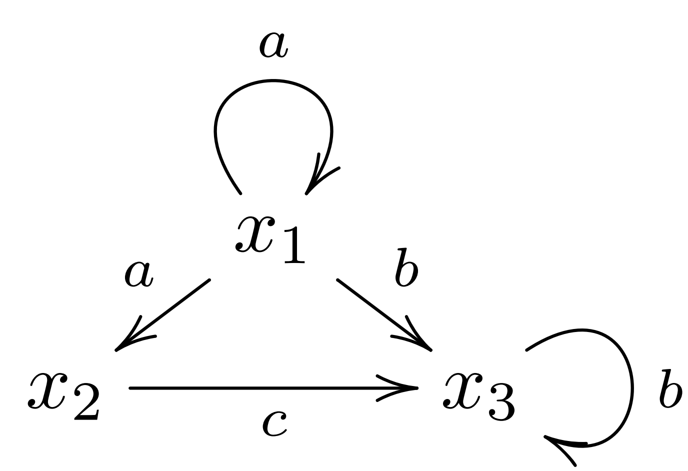
Expressiveness hierarchy

F.Bartels, A.S., E. de Vink ’03/‘04

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

LTS

$X \to \mathcal{P}(A \times X)$



Expressiveness hierarchy

F.Bartels, A.S., E. de Vink '03/'04

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

Generative system

$X \to \mathcal{D}(A \times X) + 1$

$$\begin{tikzcd} & x_1 \arrow[dl, "{a,\frac{3}{4}}"'] \arrow[dr, "{a,\frac{1}{4}}"] & \\ x_2 \arrow[d, "{b,1}"'] & & x_3 \arrow[d, "{c,1}"] \\ x_4 \arrow[d] & & x_5 \arrow[d] \\ * & & * \end{tikzcd}$$

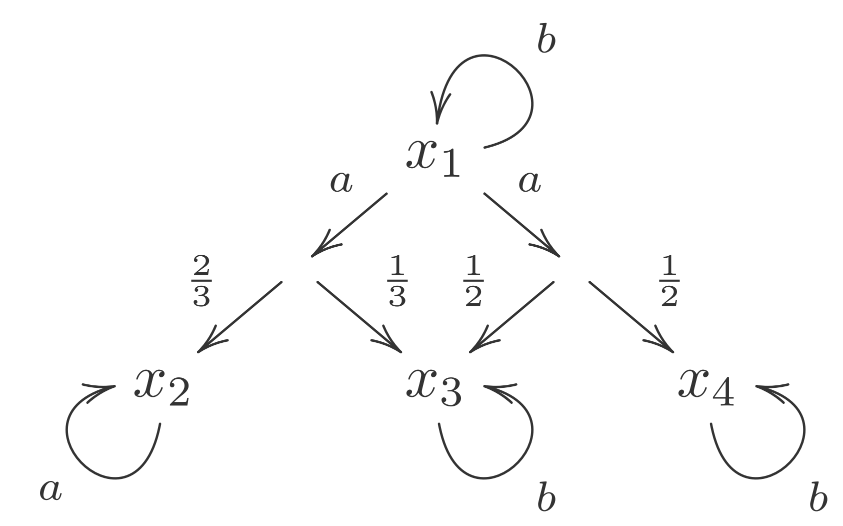
Expressiveness hierarchy

F.Bartels, A.S., E. de Vink ’03/‘04

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

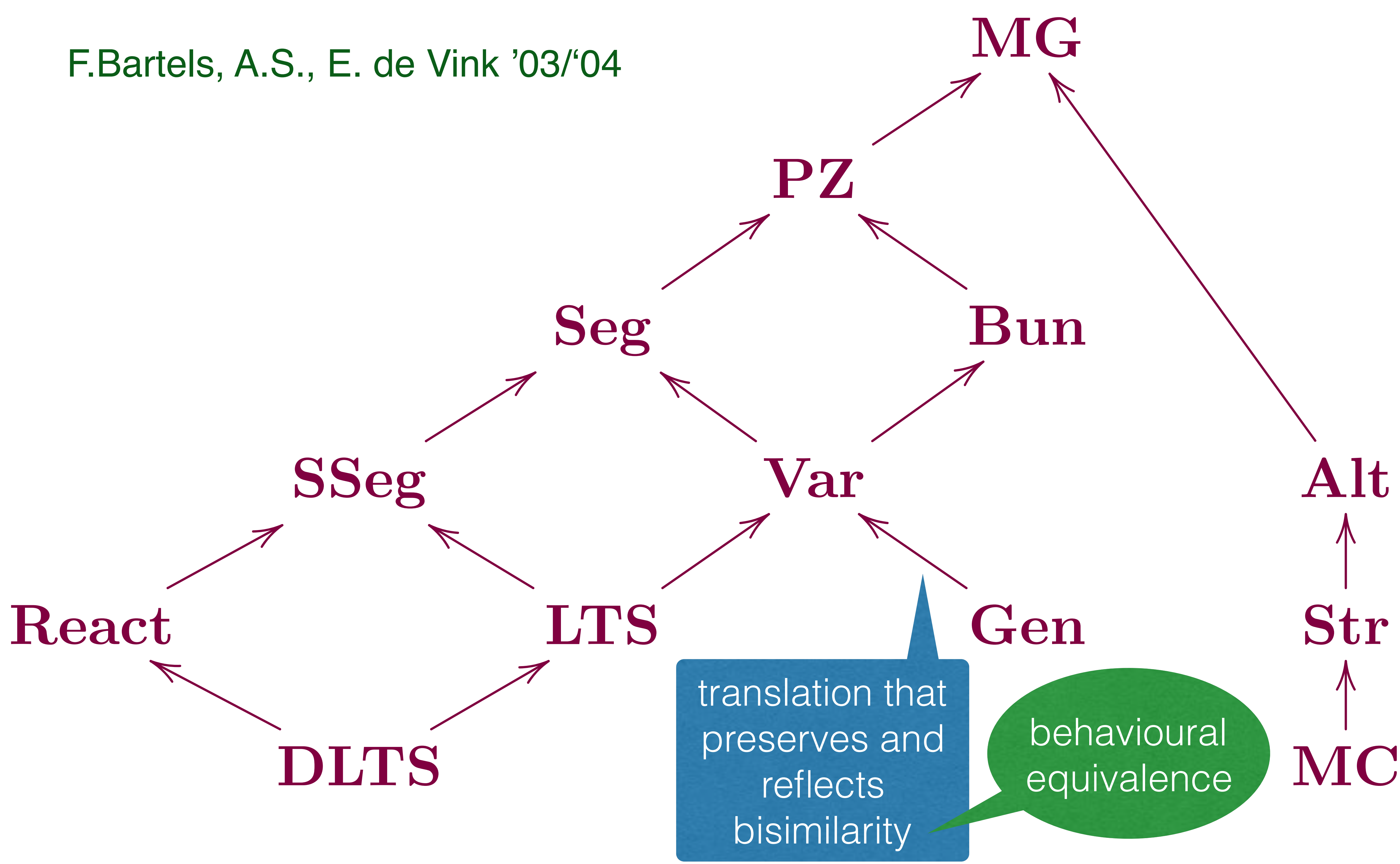
Simple Segala system (PA)

$X \to \mathcal{P}(A \times \mathcal{D}X)$



Expressiveness hierarchy

F.Bartels, A.S., E. de Vink '03/'04



Behavioural equivalence

$\mathbf{CoAlg}_{\mathbf{C}}(F)$

$X \xrightarrow{c} FX$

$h \colon X \to Y$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Behavioural equivalence

generic notion
$\approx$

$\mathbf{CoAlg}_{\mathbf{C}}(F)$

$X \xrightarrow{c} FX$

$h \colon X \to Y$

$$
\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\
FX \arrow[r, "Ff"'] & FY
\end{tikzcd}
$$

Behavioural equivalence

generic notion
$\approx$

branching-time semantics

$\mathbf{CoAlg_C}(F)$

$X \xrightarrow{c} FX$

$h \colon X \to Y$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Behavioural equivalence

generic notion $\approx$

branching-time semantics

$\mathbf{CoAlg_C}(F)$

$$X \xrightarrow{c} FX$$

$$h\colon X \to Y$$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Kernel bisimulation = kernel of a coalgebra homomorphism

Behavioural equivalence

generic notion
$\approx$

branching-time semantics

$\mathbf{CoAlg_C}(F)$

$$X \xrightarrow{c} FX$$

$$h\colon X \to Y$$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Kernel bisimulation = kernel of a coalgebra homomorphism

$$\ker(h) = \{(x, y) \mid h(x) = h(y)\}$$

Behavioural equivalence

generic notion $\approx$

branching-time semantics

$\mathbf{CoAlg_C}(F)$

$X \xrightarrow{c} FX$

$h \colon X \to Y$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Kernel bisimulation = kernel of a coalgebra homomorphism

$\ker(h) = \{(x, y) \mid h(x) = h(y)\}$

Behaviour equivalence = union of all kernel bisimulations

Behavioural equivalence

generic notion
$\approx$

branching-time semantics

$\mathbf{CoAlg_C}(F)$

$X \xrightarrow{c} FX$

$h\colon X \to Y$

$$\begin{tikzcd} X \arrow[r, "f"] \arrow[d, "c_X"'] & Y \arrow[d, "c_Y"] \\ FX \arrow[r, "Ff"'] & FY \end{tikzcd}$$

Kernel bisimulation = kernel of a coalgebra homomorphism

$\ker(h) = \{(x, y) \mid h(x) = h(y)\}$

Behaviour equivalence = union of all kernel bisimulations

coincides with "concrete" bisimilarity

Bisimilarity

LTS

Bisimilarity

LTS

bisimulation

R

Bisimilarity

LTS

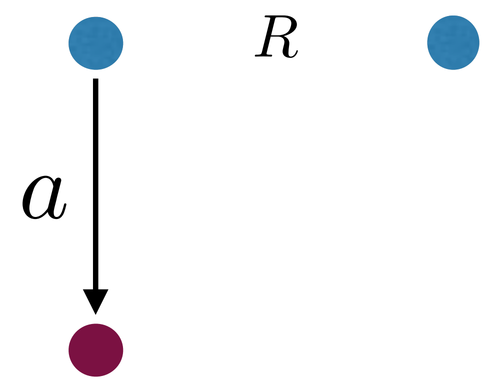
bisimulation

$\bullet \quad R \quad \bullet$

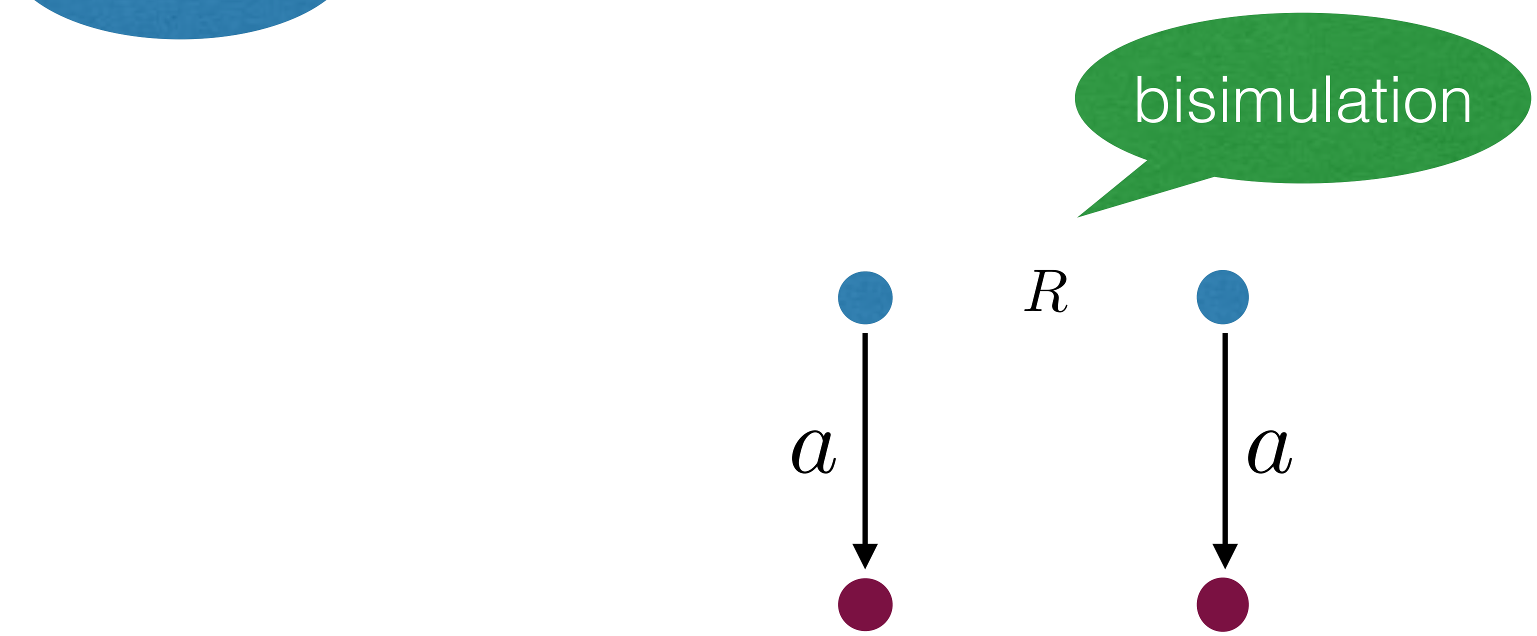
Bisimilarity

LTS

bisimulation

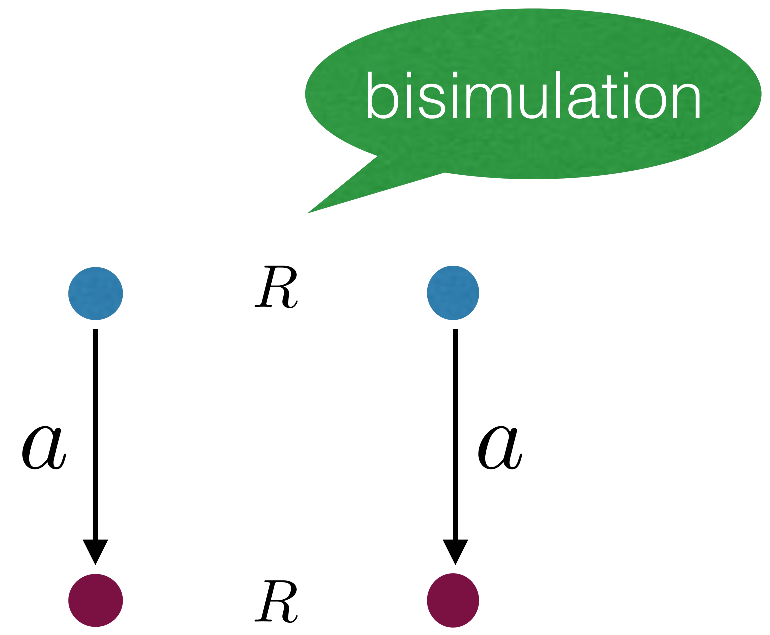


Bisimilarity



Bisimilarity

LTS

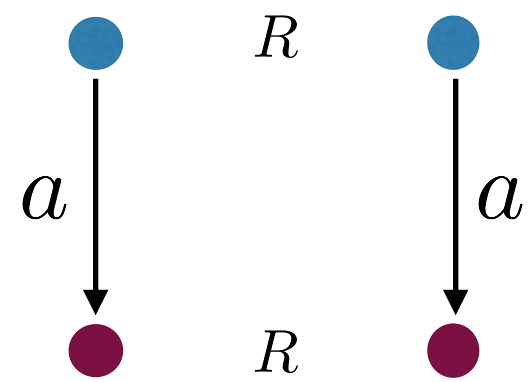


Bisimilarity

$\sim$ largest bisimulation

LTS

bisimulation



Bisimilarity

LTS

$\sim$ largest bisimulation

bisimulation

$$\begin{tikzcd} \bullet \arrow[r, phantom, "R"] \arrow[d, "a"'] & \bullet \arrow[d, "a"] \\ \bullet \arrow[r, phantom, "R"] & \bullet \end{tikzcd}$$

transfer condition

$$x \; R \; y \Rightarrow$$

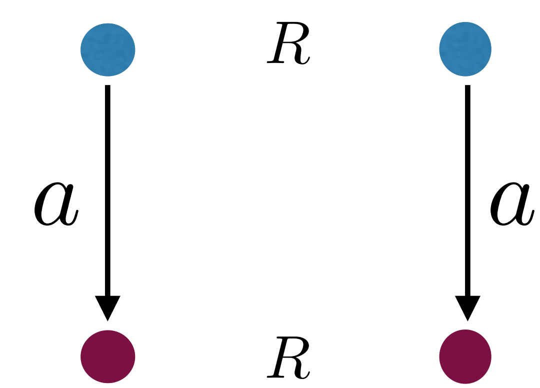
$$x \xrightarrow{a} x' \Rightarrow \exists y'.\, y \xrightarrow{a} y' \wedge x' \; R \; y'$$

Bisimilarity

LTS

$\sim$ largest bisimulation

bisimulation



transfer condition

$$x \; R \; y \Rightarrow$$

$$x \xrightarrow{a} x' \Rightarrow \exists y'.\, y \xrightarrow{a} y' \wedge x' \; R \; y'$$

coincides with behavioural equivalence

Bisimilarity

Markov chains

Bisimilarity

Markov chains

bisimulation

R

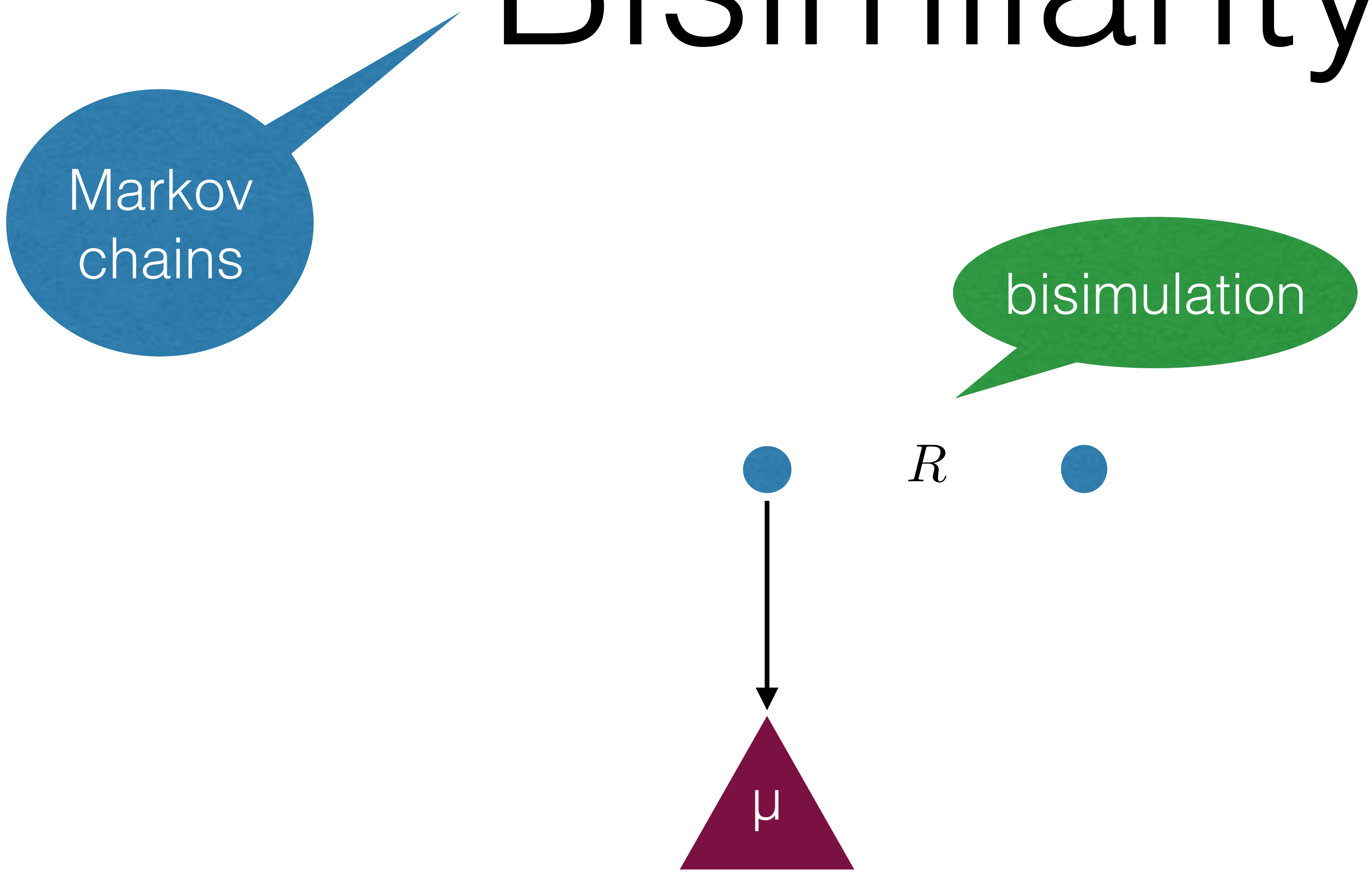
Bisimilarity

Markov chains

bisimulation

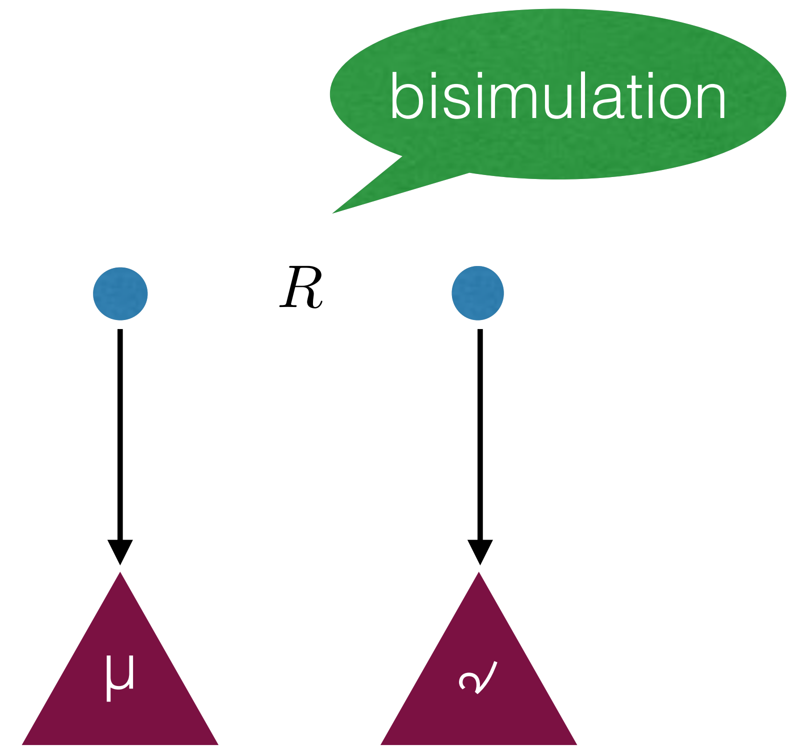
$\bullet \quad R \quad \bullet$

Bisimilarity

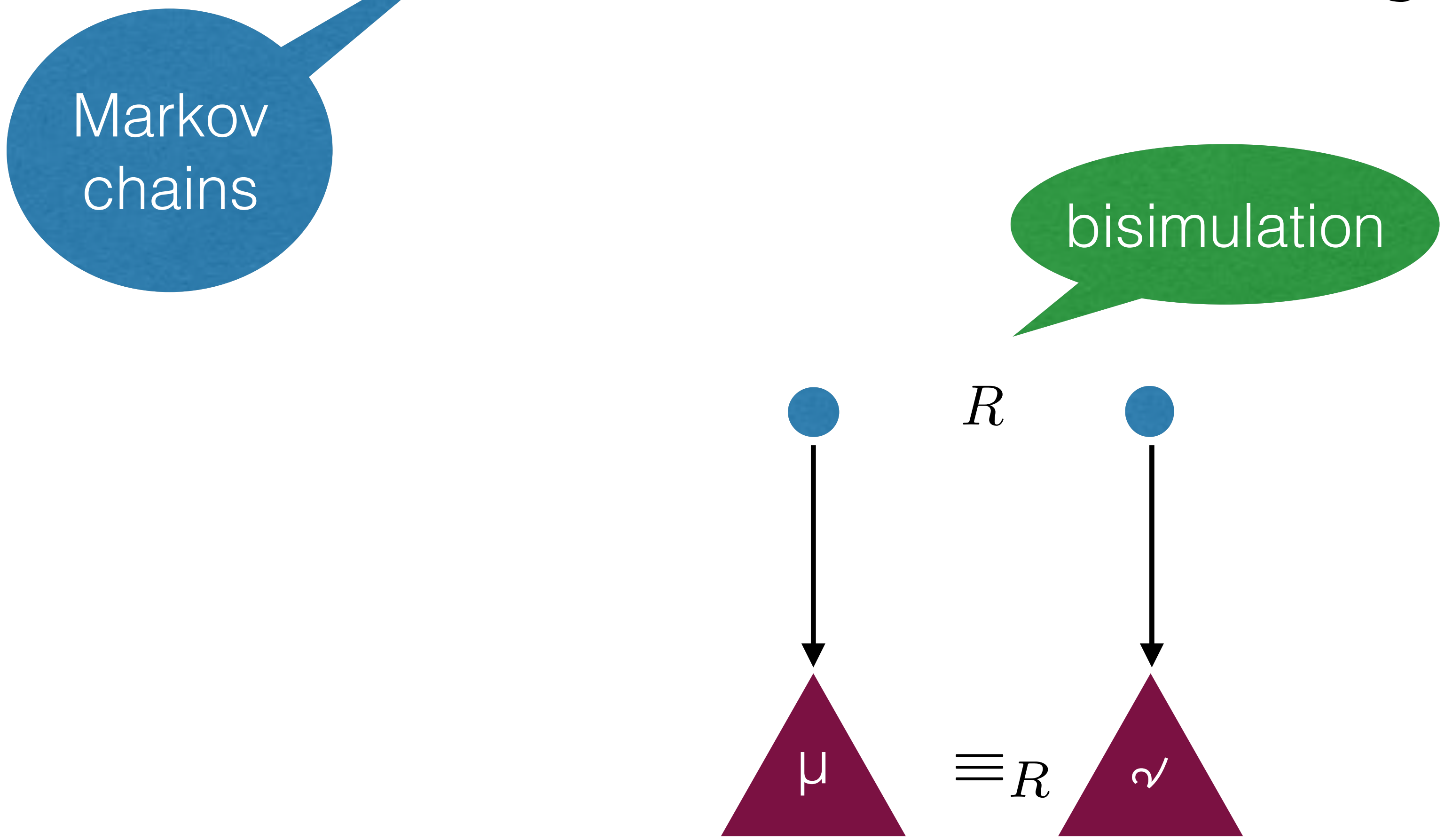


Bisimilarity

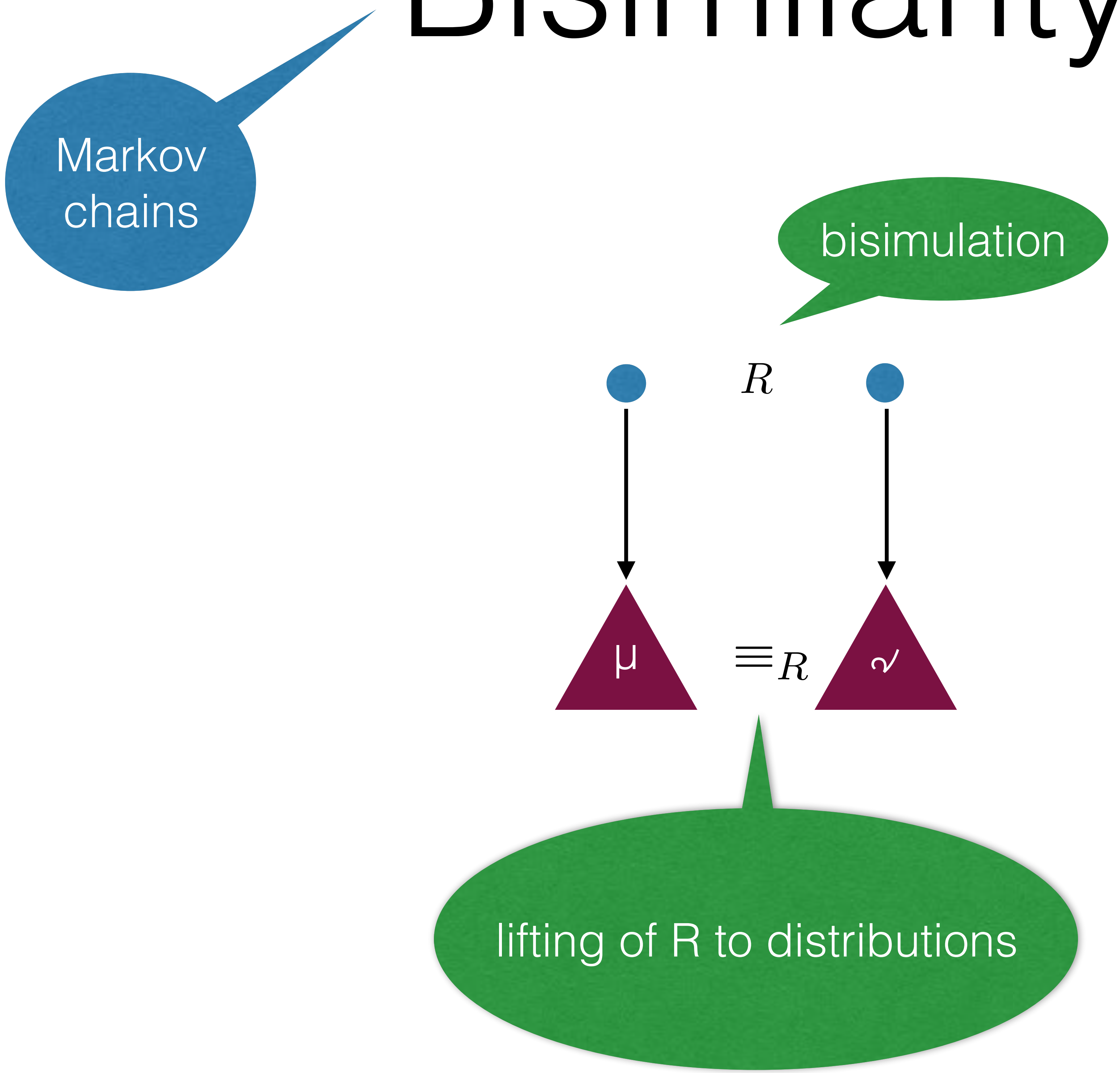
Markov chains



Bisimilarity



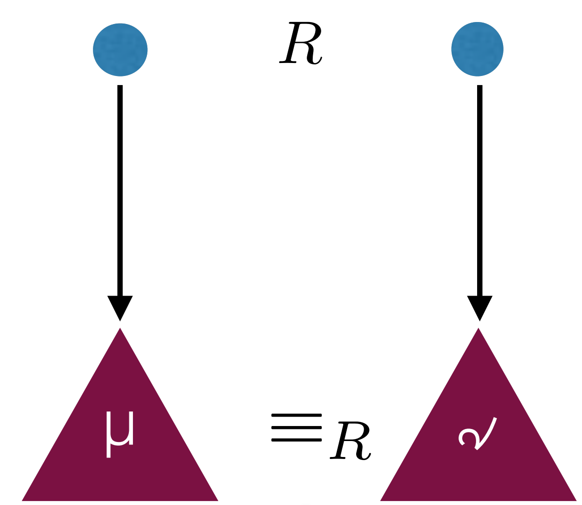
Bisimilarity



Bisimilarity

Markov chains

bisimulation



lifting of R to distributions

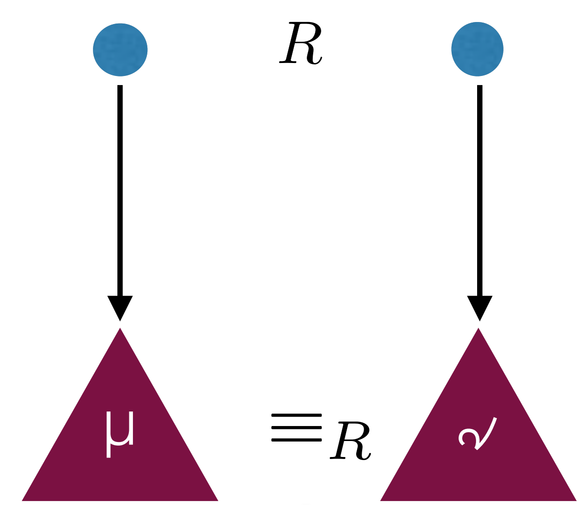
assign the same probability
to “R-classes”

Bisimilarity

$\sim$ largest bisimulation

Markov chains

bisimulation



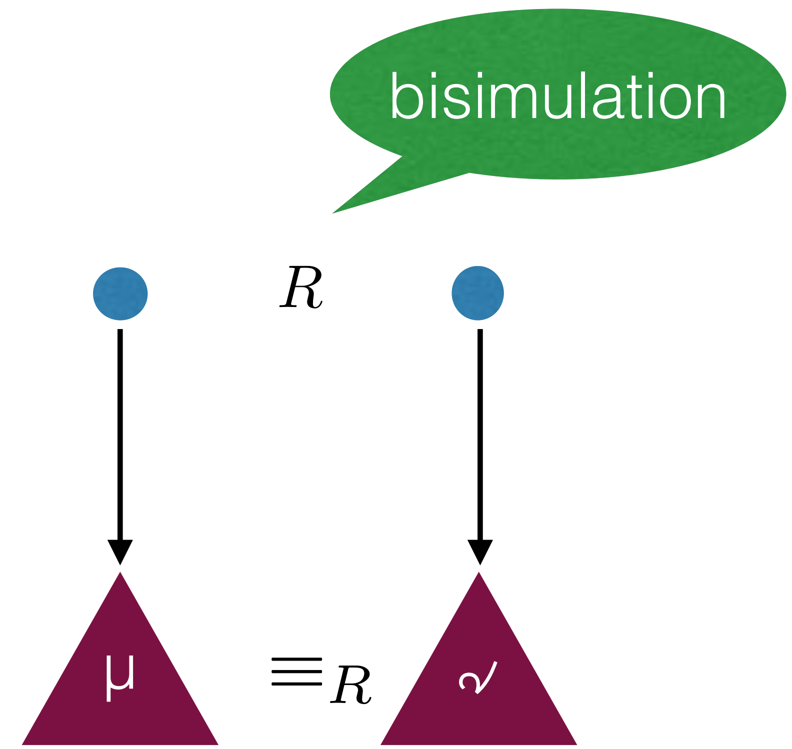
lifting of R to distributions

assign the same probability to “R-classes”

Bisimilarity

Markov chains

$\sim$ largest bisimulation



transfer condition

$$x \; R \; y \Rightarrow$$

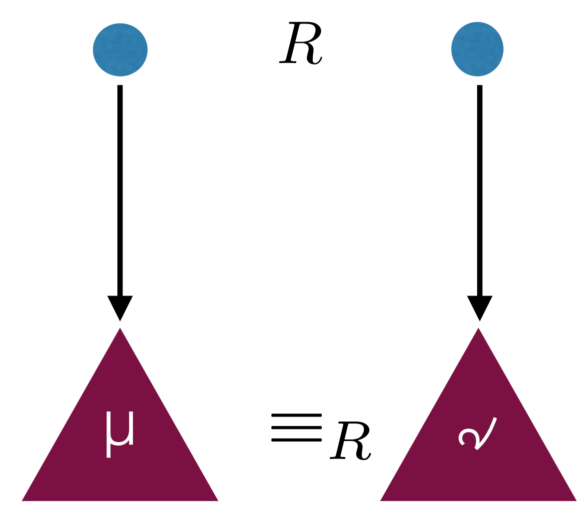
$$x \to \mu \Rightarrow \; y \to \nu \wedge \mu \; \equiv_R \; \nu$$

Bisimilarity

$\sim$ largest bisimulation

Markov chains

bisimulation



transfer condition

$$x \; R \; y \Rightarrow$$

$$x \to \mu \Rightarrow \; y \to \nu \wedge \mu \; \equiv_R \; \nu$$

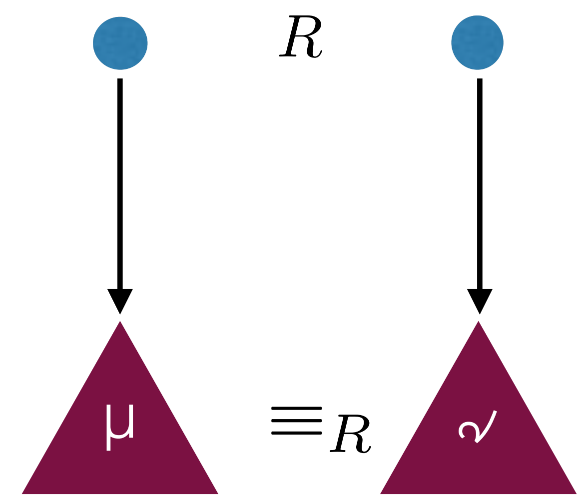
coincides with behavioural equivalence

Bisimilarity

$\sim$ largest bisimulation

Markov chains

bisimulation



transfer condition

$$x \; R \; y \Rightarrow$$

$$x \rightarrow \mu \Rightarrow \; y \rightarrow \nu \wedge \mu \; \equiv_R \; \nu$$

coincides with behavioural equivalence

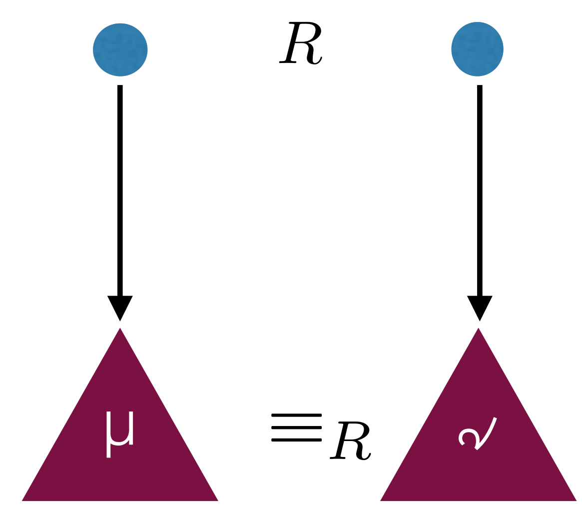
but is trivial

Bisimilarity

Markov chains

$\sim$ largest bisimulation

bisimulation



for non-trivial behaviour we need labels / termination

transfer condition

$x \; R \; y \Rightarrow$

$$x \to \mu \Rightarrow \; y \to \nu \wedge \mu \; \equiv_R \; \nu$$

coincides with behavioural equivalence

but is trivial

Bisimilarity

Simple
Segala systems /
simple PA

Bisimilarity

Simple
Segala systems /
simple PA

bisimulation

R

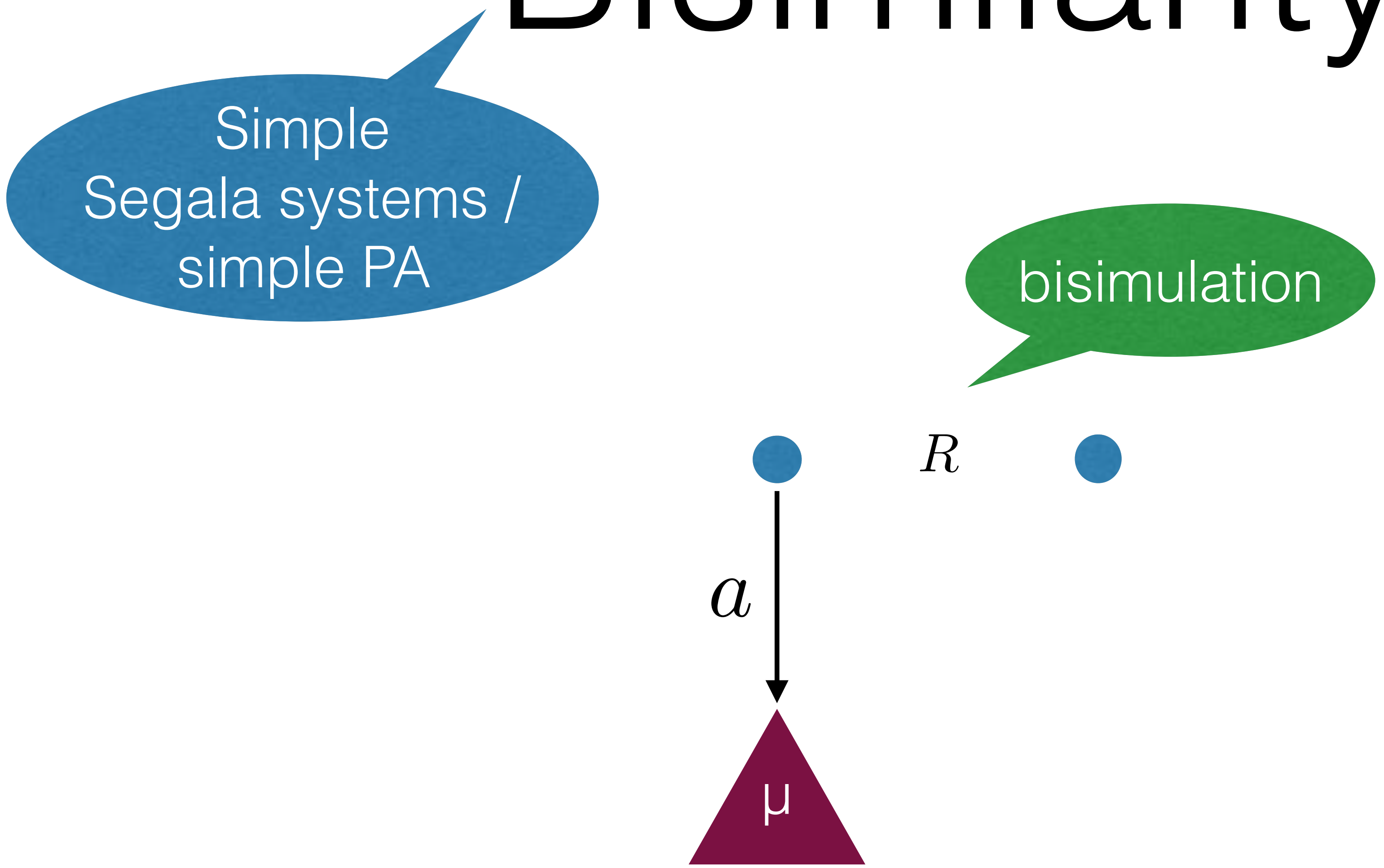
Bisimilarity

Simple
Segala systems /
simple PA

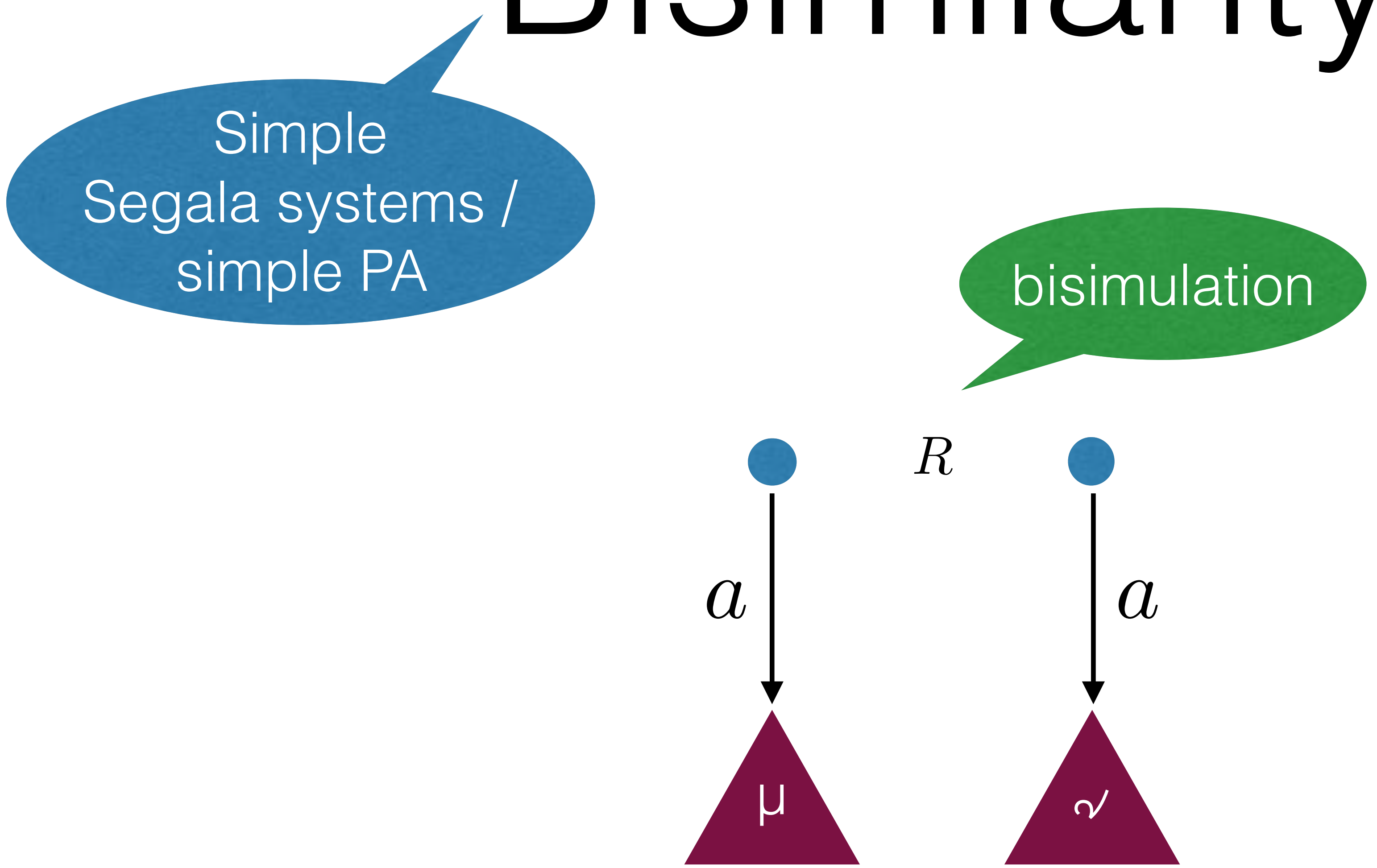
bisimulation

R

Bisimilarity



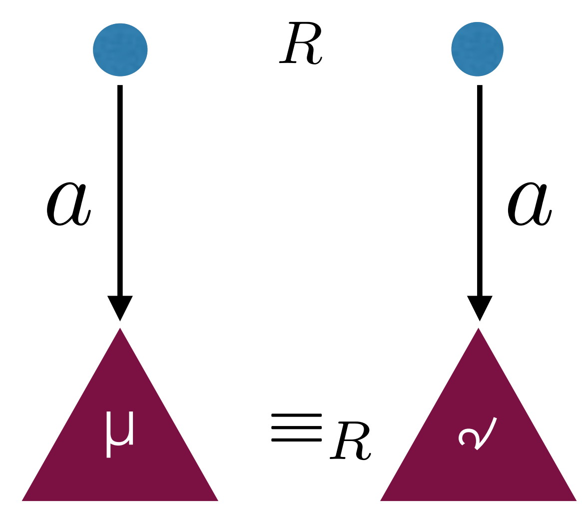
Bisimilarity



Bisimilarity

Simple Segala systems / simple PA

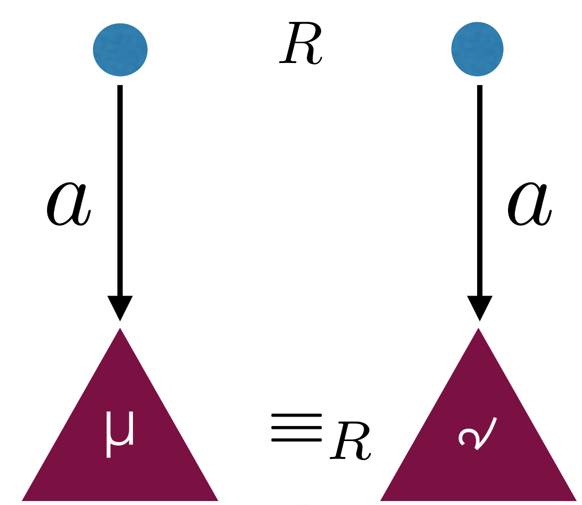
bisimulation



Bisimilarity

Simple
Segala systems /
simple PA

bisimulation

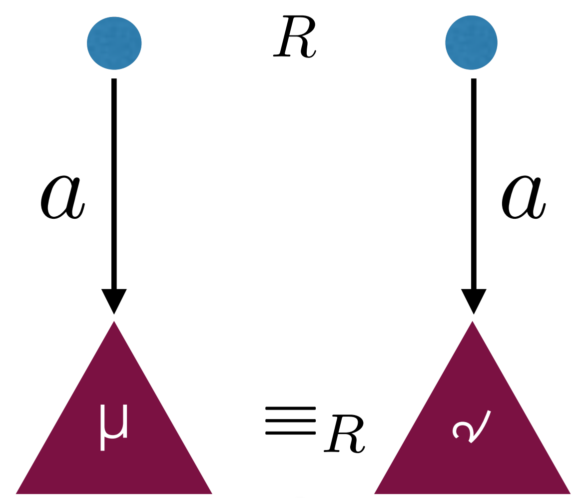


lifting of R to distributions

Bisimilarity

Simple
Segala systems /
simple PA

bisimulation



lifting of R to distributions

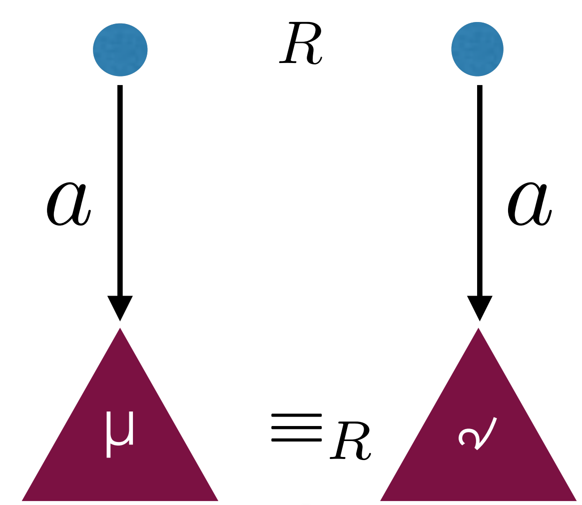
assign the same probability
to “R-classes”

Bisimilarity

$\sim$ largest bisimulation

Simple Segala systems / simple PA

bisimulation



lifting of R to distributions

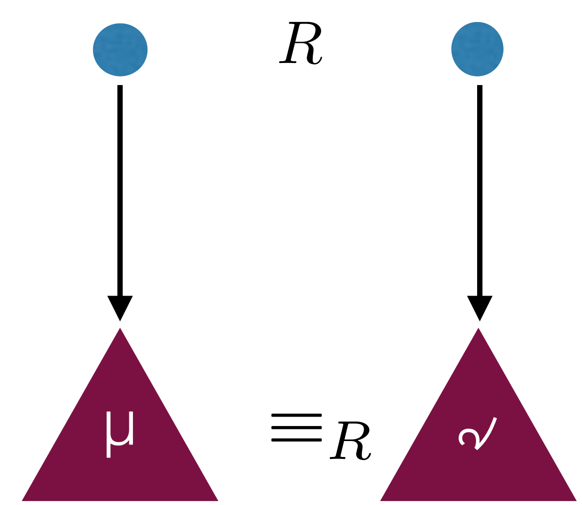
assign the same probability to “R-classes”

Bisimilarity

$\sim$ largest bisimulation

Simple Segala systems / simple PA

bisimulation



transfer condition

$$x \; R \; y \Rightarrow$$

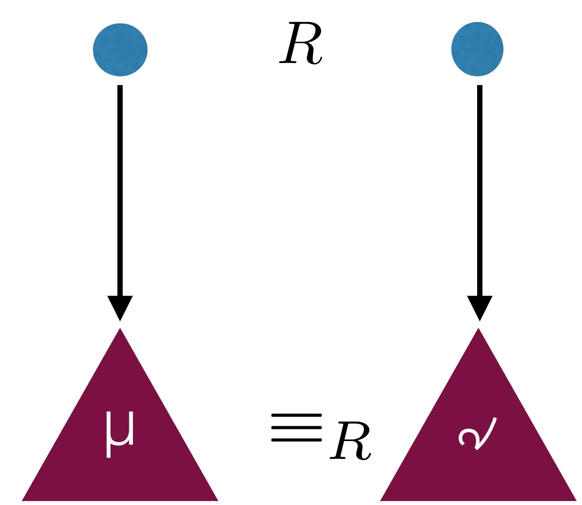
$$x \xrightarrow{a} \mu \Rightarrow \exists \nu.\ y \xrightarrow{a} \nu \wedge \mu \ \equiv_R \ \nu$$

Bisimilarity

$\sim$ largest bisimulation

Simple Segala systems / simple PA

bisimulation



coincides with behavioural equivalence

transfer condition

$$x\ R\ y \Rightarrow$$

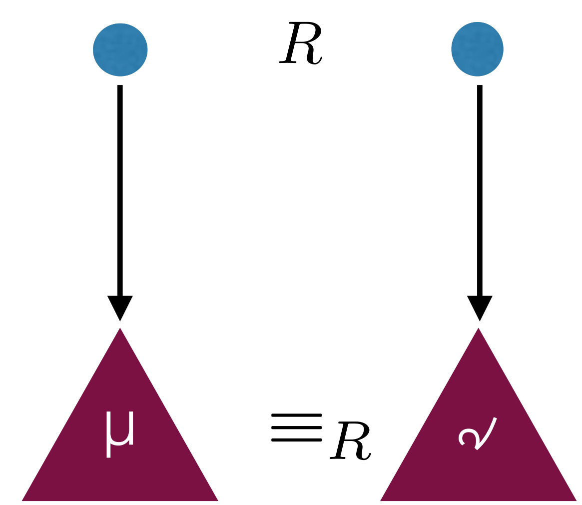
$$x \xrightarrow{a} \mu \Rightarrow \exists \nu.\ y \xrightarrow{a} \nu \wedge \mu \equiv_R \nu$$

Bisimilarity

$\sim$ largest bisimulation

Simple Segala systems / simple PA

bisimulation



all concrete bisimilarity notions coincide with behavioural equivalence

transfer condition

$$x \; R \; y \Rightarrow$$

$$x \xrightarrow{a} \mu \Rightarrow \exists \nu.\; y \xrightarrow{a} \nu \wedge \mu \; \equiv_R \; \nu$$

coincides with behavioural equivalence

Bisimilarity

F - coalgebras

Bisimilarity

F - coalgebras

bisimulation

R

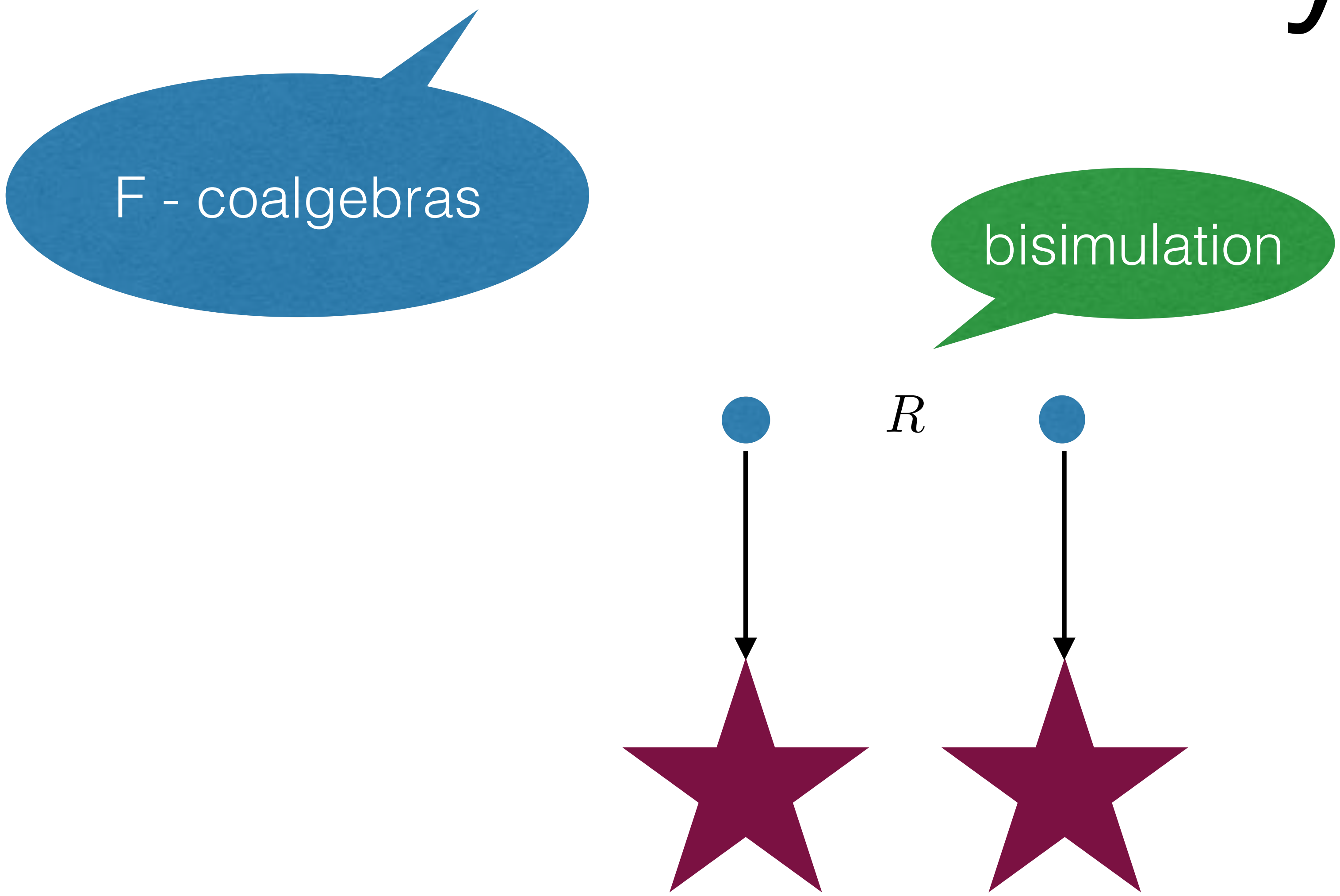
Bisimilarity

F - coalgebras

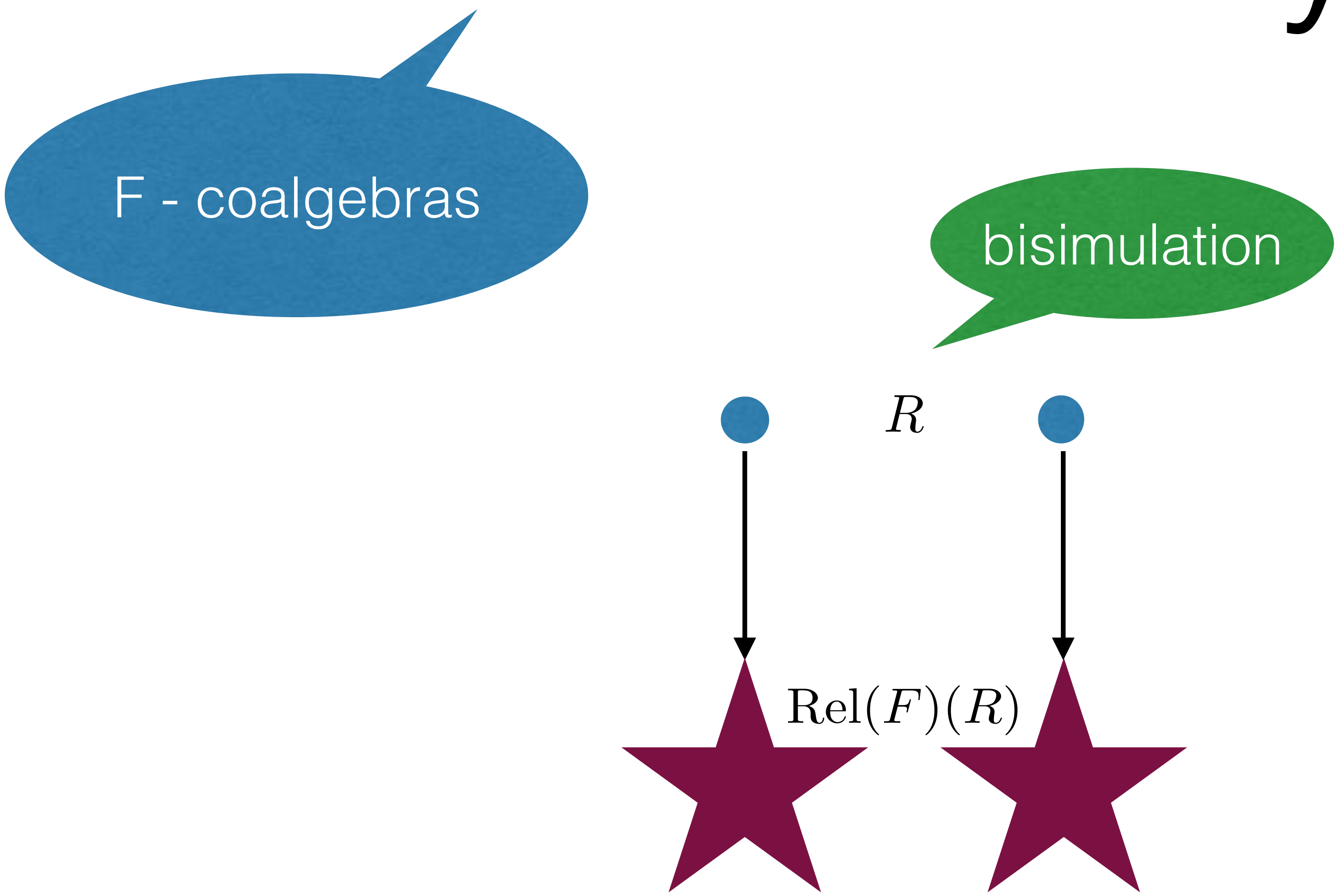
bisimulation

R

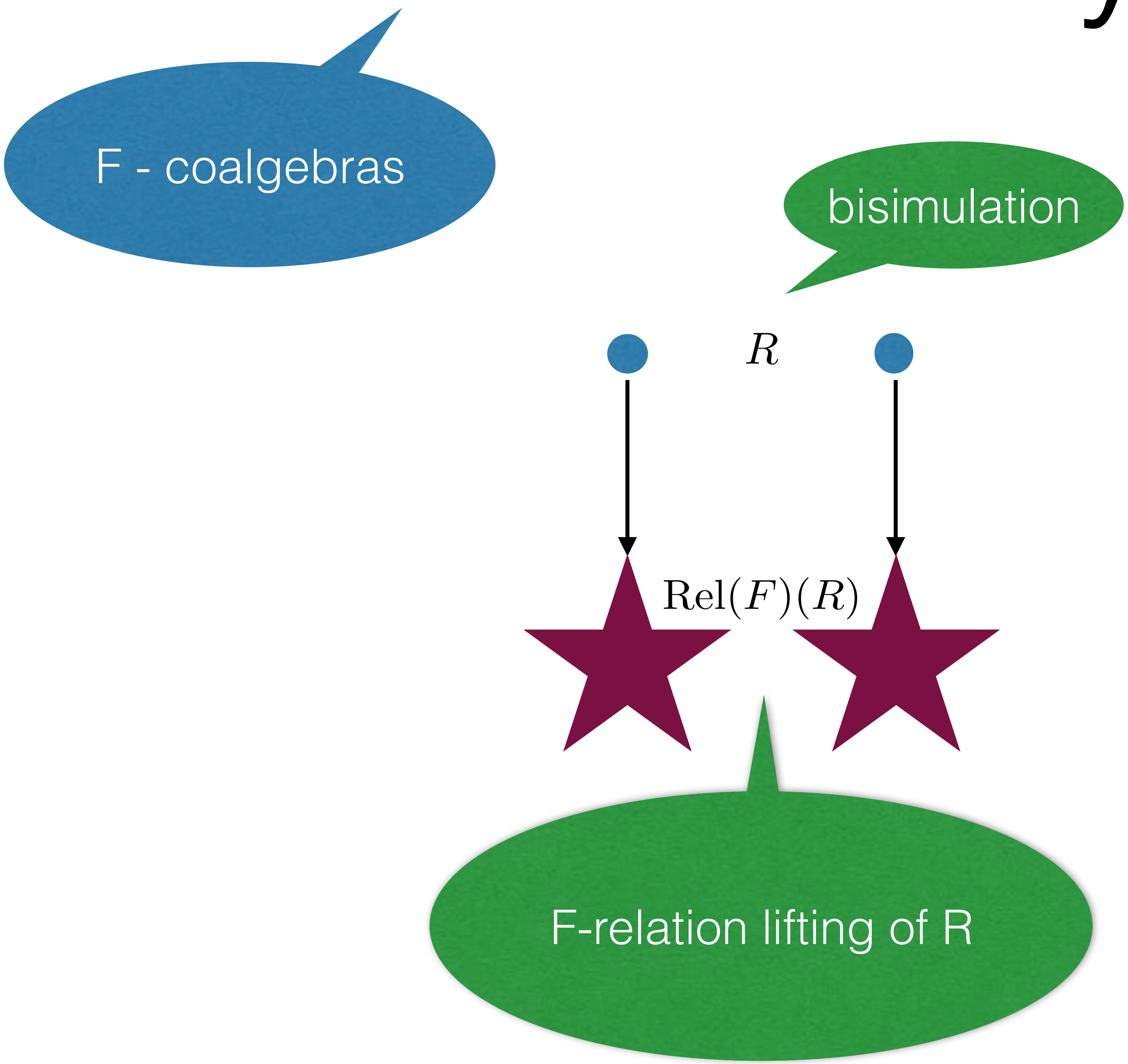
Bisimilarity



Bisimilarity



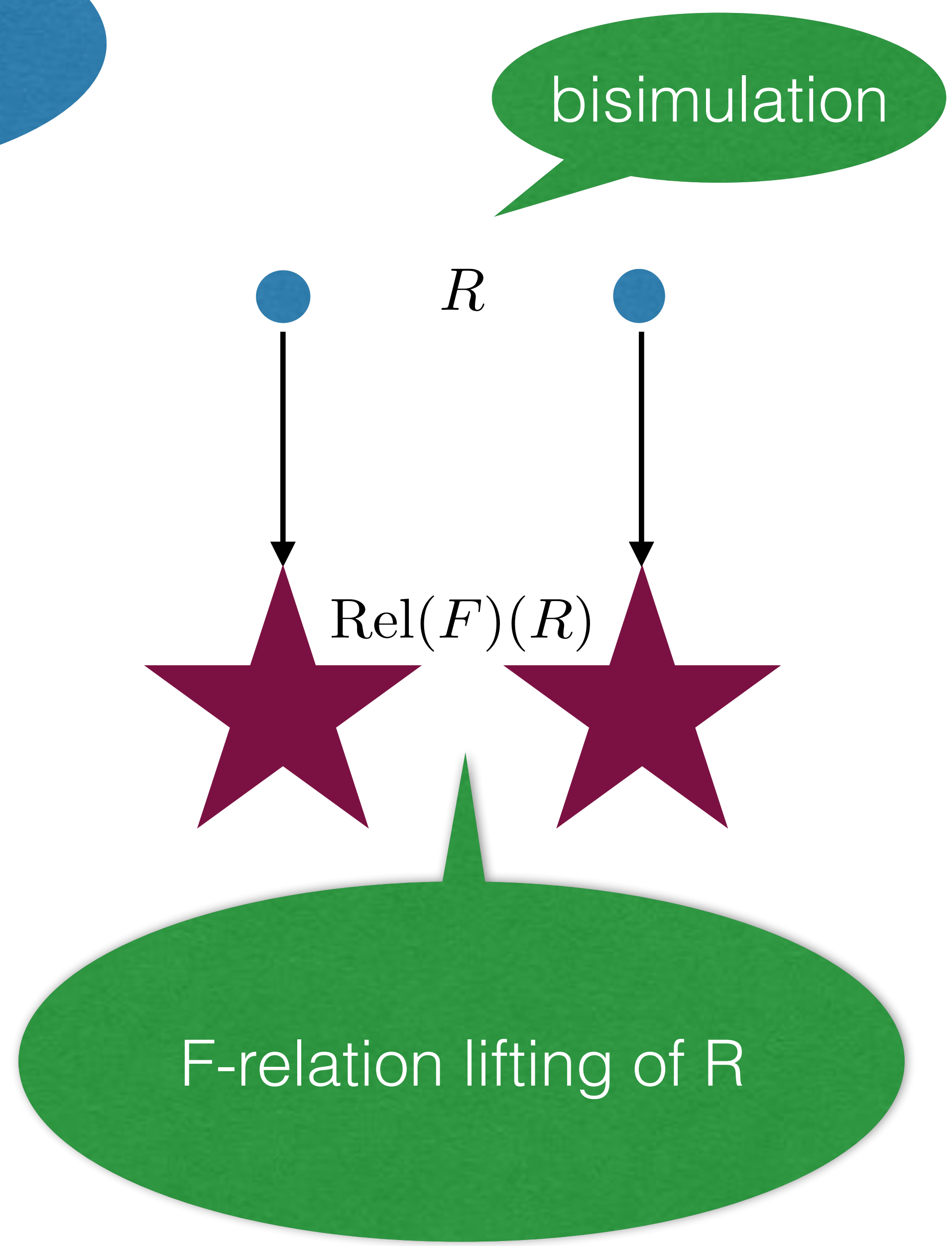
Bisimilarity



Bisimilarity

$\sim$ largest bisimulation

F - coalgebras

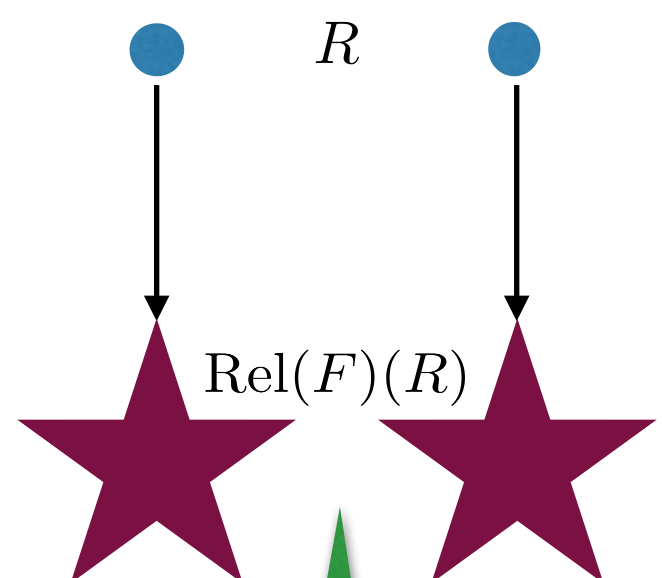


Bisimilarity

$\sim$ largest bisimulation

our class of F-coalgebras

bisimulation



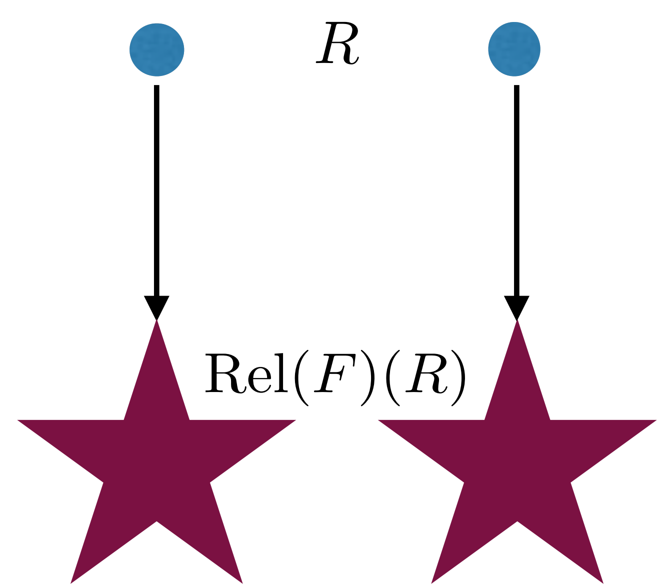
F-relation lifting of R

Bisimilarity

$\sim$ largest bisimulation

our class of F-coalgebras

bisimulation



transfer condition

$x\ R\ y \Rightarrow$

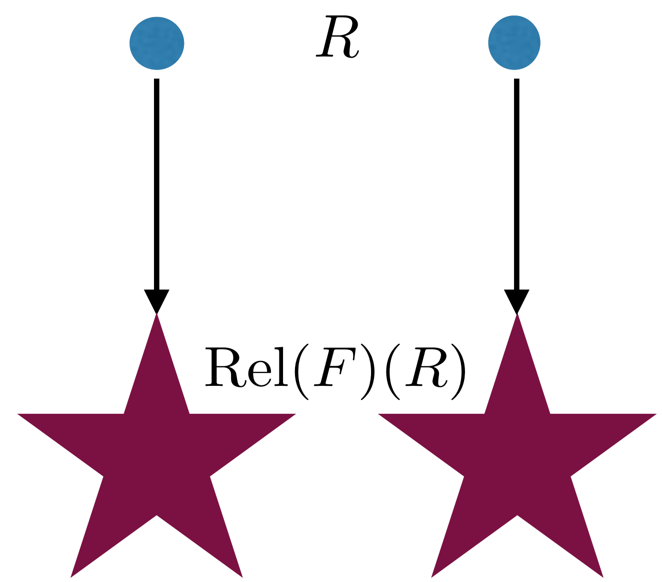
$$c(x)\ \ \mathrm{Rel}(F)(R)\ \ c(y)$$

Bisimilarity

$\sim$ largest bisimulation

our class of F-coalgebras

bisimulation



transfer condition

$x \; R \; y \Rightarrow$

$$c(x) \;\; \mathrm{Rel}(F)(R) \;\; c(y)$$

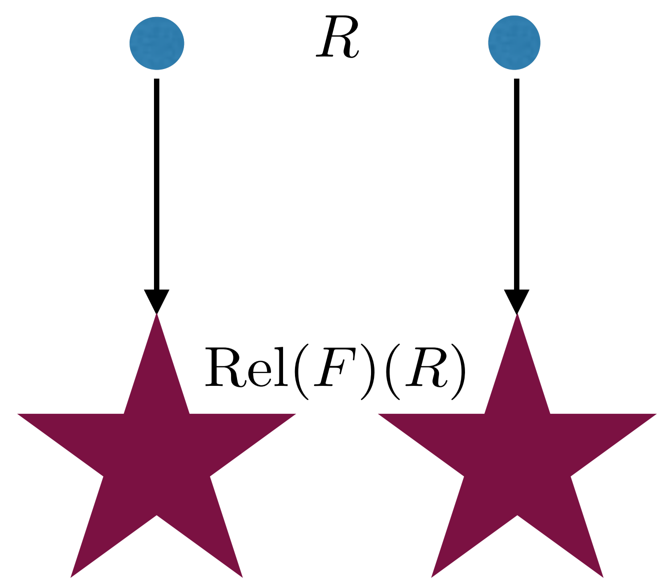
coincides with behavioural equivalence

Bisimilarity

$\sim$ largest bisimulation

our class of F-coalgebras

bisimulation



provides a modular proof of coincidence

transfer condition

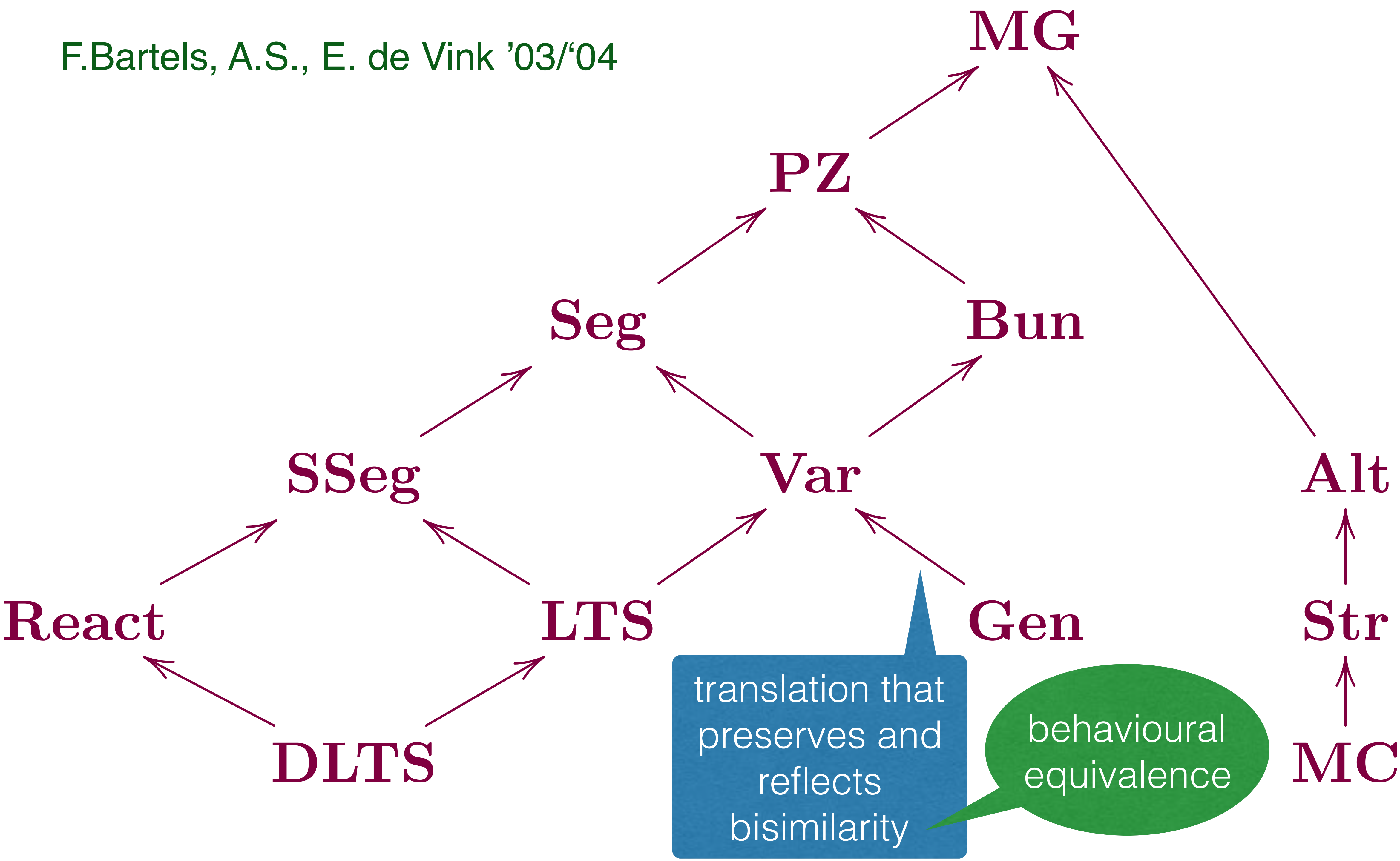
$$x \; R \; y \Rightarrow$$

$$c(x) \;\; \mathrm{Rel}(F)(R) \;\; c(y)$$

coincides with behavioural equivalence

Expressiveness hierarchy

F.Bartels, A.S., E. de Vink ’03/‘04



The translation

The translation

that preserves
and reflects
bisimilarity

The translation

that preserves and reflects bisimilarity

behavioural equivalence

The translation

that preserves and reflects bisimilarity

behavioural equivalence

Theorem

For F-coalgebras → G-coalgebras, it suffices to give an injective natural transformation from F to G.

The translation

that preserves and reflects bisimilarity

behavioural equivalence

Theorem

For F-coalgebras → G-coalgebras, it suffices to give an injective natural transformation from F to G.

behavioural equivalence is preserved and reflected

The translation

that preserves and reflects bisimilarity

behavioural equivalence

Theorem

For F-coalgebras → G-coalgebras, it suffices to give an injective natural transformation from F to G.

behavioural equivalence is preserved and reflected

If F preserves weak pullbacks then behavioural equivalence coincides with coalgebraic bisimilarity (and so bisimilarity is preserved and reflected)

Example translation

Example translation

that preserves and reflects bisimilarity

Example translation

that preserves and reflects bisimilarity

behavioural equivalence

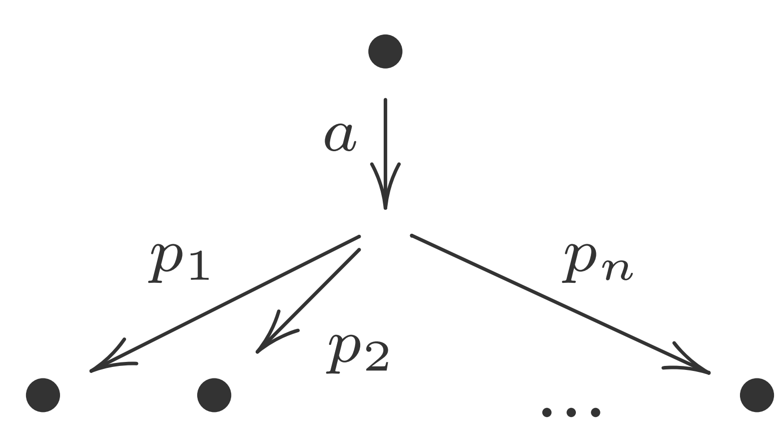
Example translation

that preserves and reflects bisimilarity

behavioural equivalence

Simple Segala system (PA)

$X \to \mathcal{P}(A \times \mathcal{D}X)$



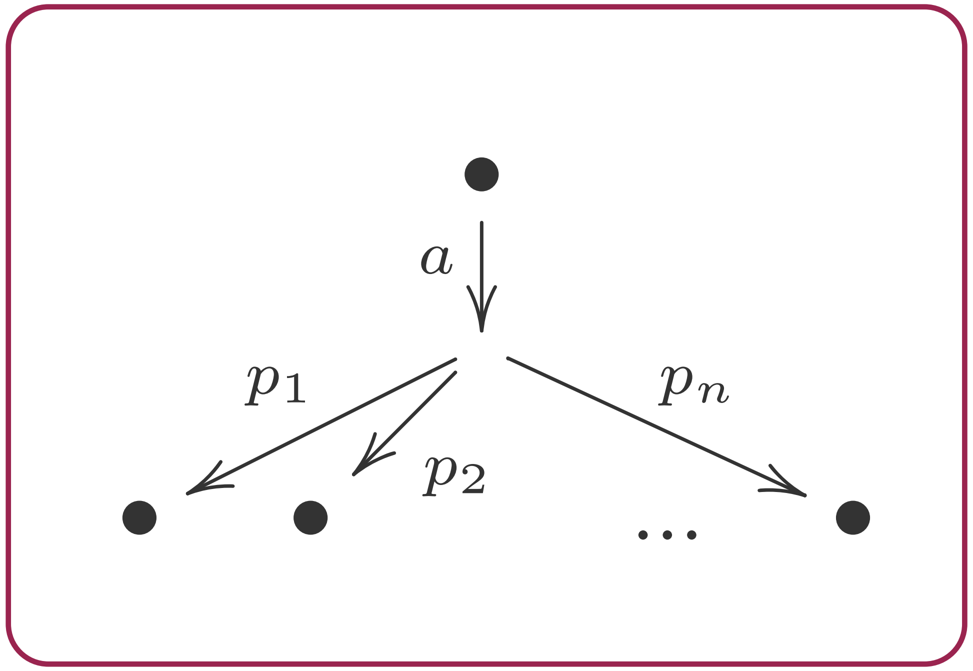
Example translation

that preserves and reflects bisimilarity

behavioural equivalence

Simple Segala system (PA)

$X \to \mathcal{P}(A \times \mathcal{D}X)$



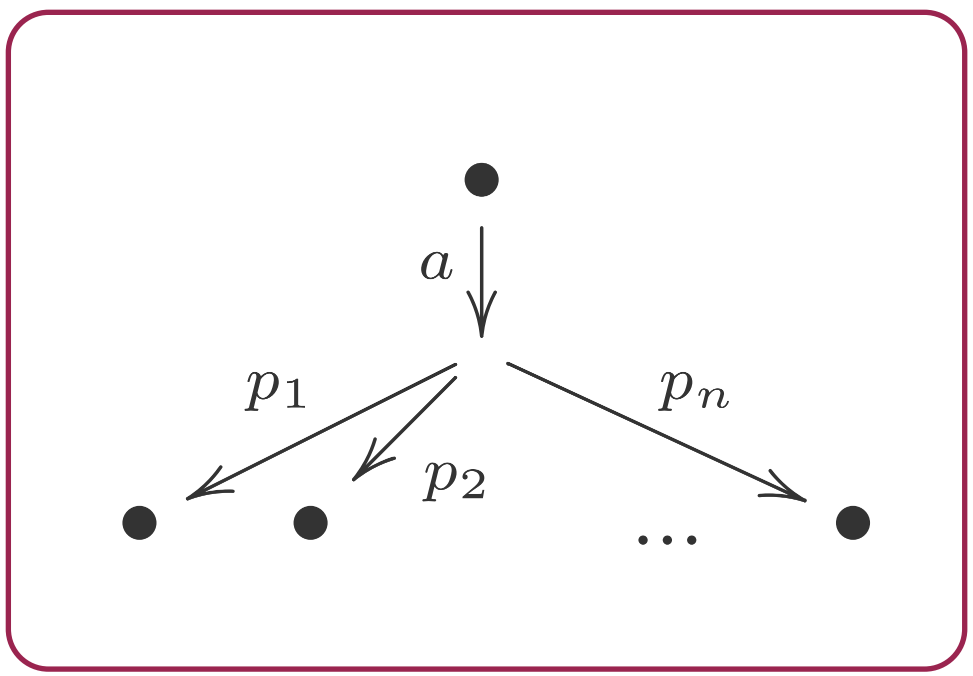
Example translation

that preserves and reflects bisimilarity

behavioural equivalence

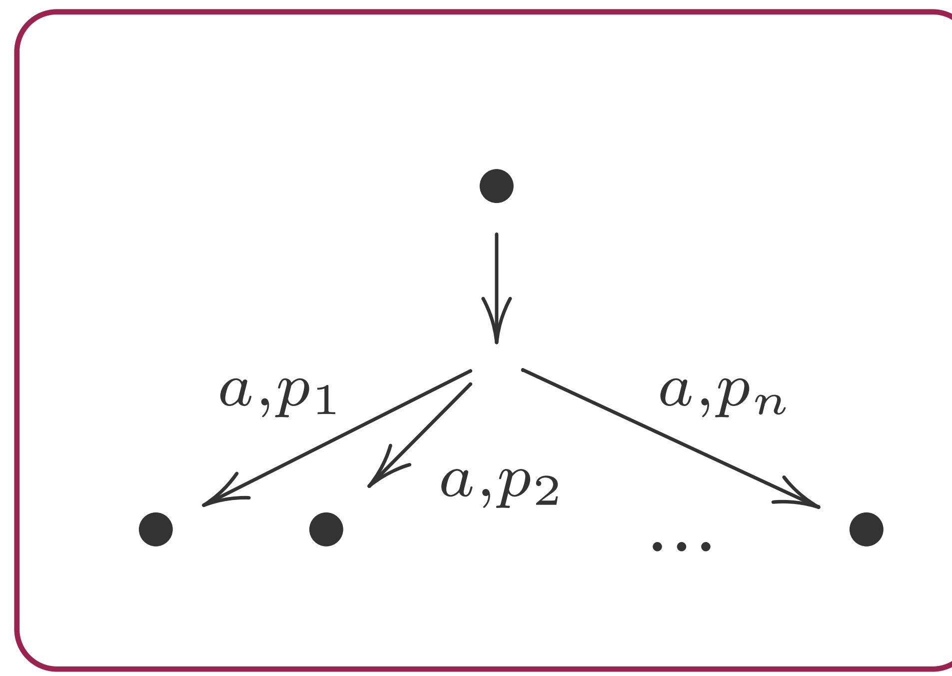
Simple Segala system (PA)

$X \to \mathcal{P}(A \times \mathcal{D}X)$



General Segala system (PA)

$X \to \mathcal{P}\mathcal{D}(A \times X)$



Probabilities are not that special…

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \mathrm{supp}(x) \text{ is finite and } \sum_{x \in X} \varphi(x) \in S\}$$

Probabilities are not that special…

M-valued valuations

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \mathrm{supp}(x) \text{ is finite and } \sum_{x\in X} \varphi(x) \in S\}$$

Probabilities are not that special…

M-valued valuations

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \mathrm{supp}(x) \text{ is finite and } \sum_{x \in X} \varphi(x) \in S\}$$

$$V_S = \mathcal{P}_f$$
$$M = (\{0,1\}, \vee, 0)$$
$$S = M$$

Probabilities are not that special…

M-valued valuations

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \text{supp}(x) \text{ is finite and } \sum_{x\in X} \varphi(x) \in S\}$$

$$V_S = \mathcal{P}_f \quad M = (\{0,1\}, \vee, 0) \quad S = M$$

$$V_S = \mathcal{M}_f \quad M = (\mathbb{N}, +, 0) \quad S = M$$

Probabilities are not that special…

M-valued valuations

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \mathrm{supp}(x) \text{ is finite and } \sum_{x \in X} \varphi(x) \in S\}$$

$$V_S = \mathcal{P}_f \quad M = (\{0,1\}, \vee, 0) \quad S = M$$

$$V_S = \mathcal{M}_f \quad M = (\mathbb{N}, +, 0) \quad S = M$$

$$V_S = \mathcal{D}_f \quad M = (\mathbb{R}^+, +, 0) \quad S = [0,1]$$

Probabilities are not that special…

M-valued valuations

- Subsets, multisets, distributions,.. are all instances of the same functor
- For a monoid (M, +, 0) and a subset S ⊆ M

$$V_S(X) = \{\varphi\colon X \to M \mid \mathrm{supp}(x) \text{ is finite and } \sum_{x\in X} \varphi(x) \in S\}$$

$$V_S = \mathcal{P}_f \quad M = (\{0,1\}, \vee, 0) \quad S = M$$

$$V_S = \mathcal{M}_f \quad M = (\mathbb{N}, +, 0) \quad S = M$$

$$V_S = \mathcal{D}_f \quad M = (\mathbb{R}^+, +, 0) \quad S = [0,1]$$

additional structure on *M* adds structure to V_S