Introduction

Modelling parallel systems

Linear Time Properties

Regular Properties

Linear Temporal Logic (LTL)

Computation-Tree Logic

Equivalences and Abstraction

- bisimulation
- CTL, CTL*-equivalence
- computing the bisimulation quotient
- abstraction stutter steps
- simulation relations ⟵

Classification of implementation relations

Classification of implementation relations

- linear vs. branching time
  - ∗ linear time: trace relations
  - ∗ branching time: (bi)simulation relations
- (nonsymmetric) preorders vs. equivalences:
  - ∗ preorders: trace inclusion, simulation
  - ∗ equivalences: trace equivalence, bisimulation
- strong vs. weak relations
  - ∗ strong: reasoning about all transitions
  - ∗ weak: abstraction from stutter steps

Classification of implementation relations

- linear vs. branching time
  - linear time: trace relations
  - branching time: (bi)simulation relations
- (nonsymmetric) preorders vs. equivalences:
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation
- strong vs. weak relations
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps

Classification of implementation relations

- linear vs. branching time
  - ∗ linear time: trace relations
  - ∗ branching time: (bi)simulation relations
- (nonsymmetric) preorders vs. equivalences:
  - ∗ preorders: trace inclusion, simulation
  - ∗ equivalences: trace equivalence, bisimulation
- strong vs. weak relations
  - ∗ strong: reasoning about all transitions
  - ∗ weak: abstraction from stutter steps

The simulation preorder

The simulation preorder

is a nonsymmetric branching time relation

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- "unidirected" version of bisimulation:

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- "unidirected" version of bisimulation:

if $\mathcal{T}_1$ is simulated by $\mathcal{T}_2$ then $\mathcal{T}_2$ can mimick all steps of $\mathcal{T}_1$, but possibly has more behaviors

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- "unidirected" version of bisimulation:

> if $\mathcal{T}_1$ is simulated by $\mathcal{T}_2$ then $\mathcal{T}_2$ can mimick all steps of $\mathcal{T}_1$, but possibly has more behaviors

- relies on a coinductive definition (as bisimulation equivalence)

The simulation preorder

is a nonsymmetric branching time relation

- plays of central role for abstraction
- the BT-analogue to trace inclusion
- "unidirected" version of bisimulation:

if $\mathcal{T}_1$ is simulated by $\mathcal{T}_2$ then $\mathcal{T}_2$ can mimick all steps of $\mathcal{T}_1$, but possibly has more behaviors

- relies on a coinductive definition (as bisimulation equivalence)

here: just strong simulation, i.e., no abstraction from stutter steps

Simulation for two TS

Simulation for two TS

$$\begin{aligned} \text{let} \quad \mathcal{T}_1 &= (S_1, Act_1, \rightarrow_1, S_{0,1}, AP, L_1) \\ \mathcal{T}_2 &= (S_2, Act_2, \rightarrow_2, S_{0,2}, AP, L_2) \end{aligned}$$

be two transition systems

Simulation for two TS

let $\mathcal{T}_1 = (S_1, Act_1, \rightarrow_1, S_{0,1}, AP, L_1)$

$\mathcal{T}_2 = (S_2, Act_2, \rightarrow_2, S_{0,2}, AP, L_2)$

be two transition systems

- over the same set ***AP*** of atomic propositions
- possibly with terminal states

Simulation for a pair of TS

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

Simulation for a pair of TS

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$

Simulation for a pair of TS

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$

(2) for all $(s_1, s_2) \in \mathcal{R}$:

$$\forall s_1' \in Post(s_1)\ \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$

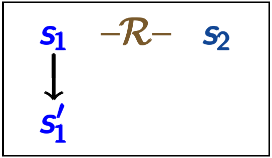
Simulation for a pair of TS

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

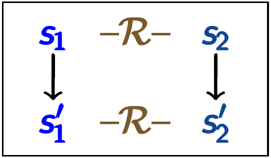
(1) if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$

(2) for all $(s_1, s_2) \in \mathcal{R}$:

$$\forall s_1' \in Post(s_1)\ \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$



can be completed to



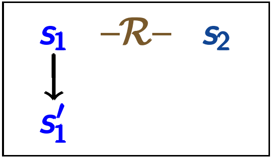
Simulation for a pair of TS

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

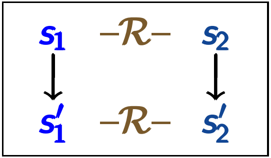
(1) if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$

(2) for all $(s_1, s_2) \in \mathcal{R}$:

$$\forall s_1' \in Post(s_1)\ \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$



can be completed to



(I) for all initial states s_1 of $\mathcal{T}_1$
there is an initial state s_2 of $\mathcal{T}_2$ with $(s_1, s_2) \in \mathcal{R}$

Simulation preorder $\preceq$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) labeling condition

(2) stepwise simulation condition

(I) initial condition

simulation preorder $\preceq$ for TS:

$$\mathcal{T}_1 \preceq \mathcal{T}_2$$

Simulation preorder $\preceq$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) labeling condition
(2) stepwise simulation condition
(I) initial condition

simulation preorder $\preceq$ for TS:

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \text{iff} \quad \left\{ \begin{array}{c} \text{there exists a simulation } \mathcal{R} \\ \text{for } (\mathcal{T}_1, \mathcal{T}_2) \end{array} \right.$$

Simulation preorder $\preceq$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) labeling condition

(2) stepwise simulation condition

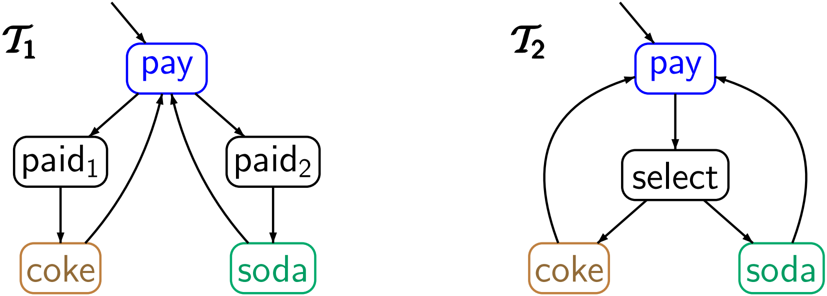
(I) initial condition

> simulation preorder $\preceq$ for TS:
>
> $$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \text{iff} \quad \left\{ \begin{array}{c} \text{there exists a simulation } \mathcal{R} \\ \text{for } (\mathcal{T}_1, \mathcal{T}_2) \end{array} \right.$$

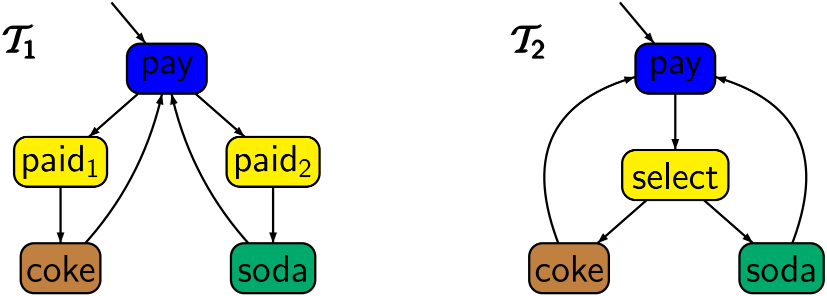
If s_1 is a state of $\mathcal{T}_1$ and s_2 a state of $\mathcal{T}_2$ then

> $$s_1 \preceq s_2 \quad \text{iff} \quad \text{there exists a simulation } \mathcal{R} \text{ for } (\mathcal{T}_1, \mathcal{T}_2) \text{ such that } (s_1, s_2) \in \mathcal{R}$$

Two beverage machines

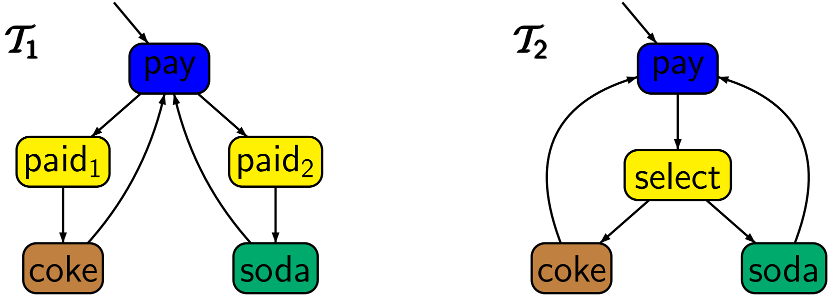


Two beverage machines



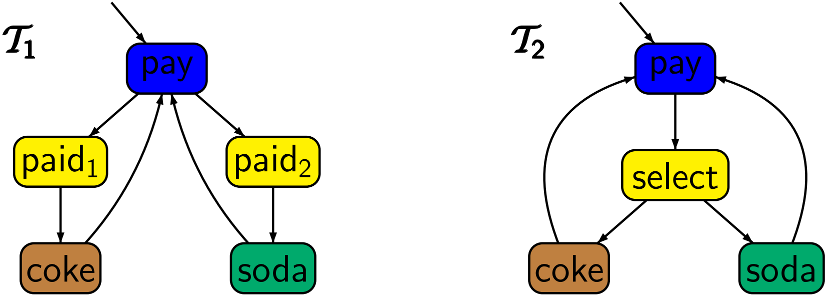
for $AP = \{pay, coke, soda\}$:

Two beverage machines



for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$

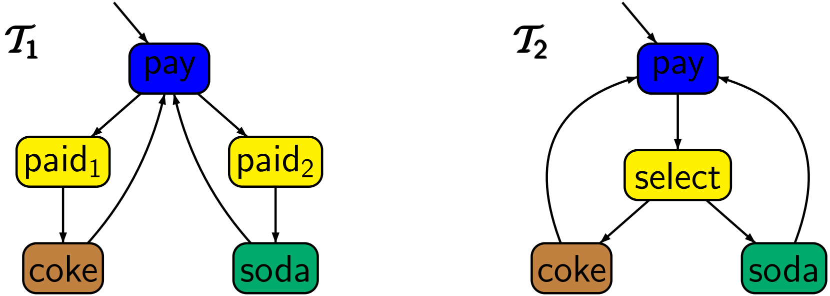
Two beverage machines



for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

Two beverage machines

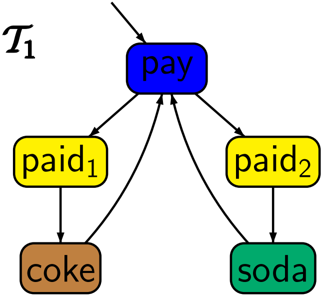


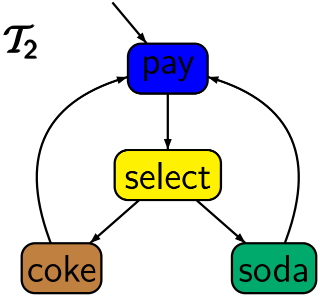
for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$$\{ \ (pay, pay), \ (paid_1, select), \ (paid_2, select), \ (coke, coke), \ (soda, soda) \ \}$$

Two beverage machines



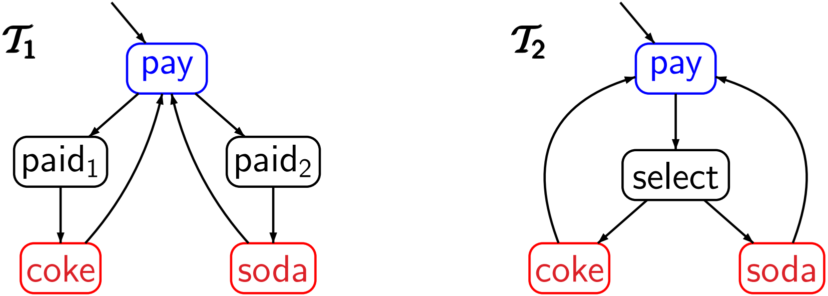


for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$$\{ (pay, pay), (paid_1, select), (paid_2, select), (coke, coke), (soda, soda) \}$$

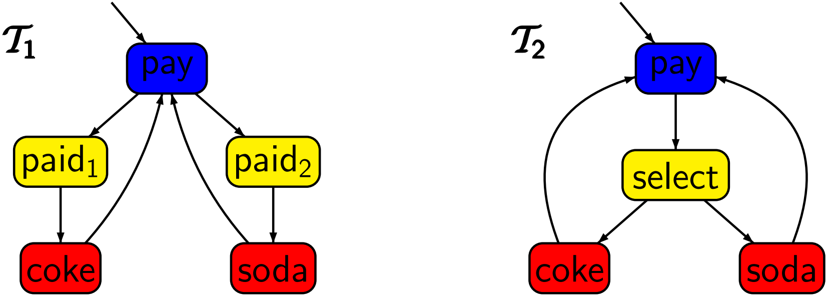
Two beverage machines



for $AP = \{\textit{pay}, \textit{coke}, \textit{soda}\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{\textit{pay}, \textit{drink}\}$:

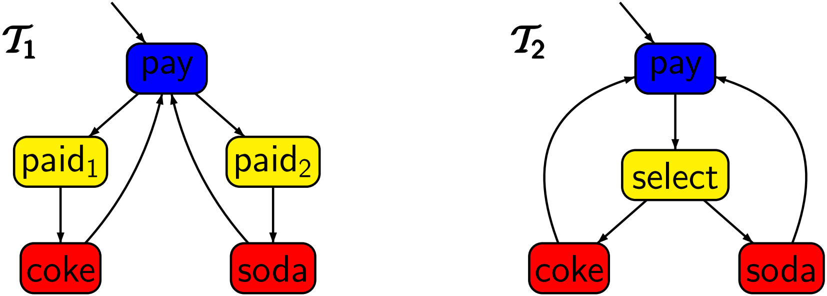
Two beverage machines



for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{pay, drink\}$:

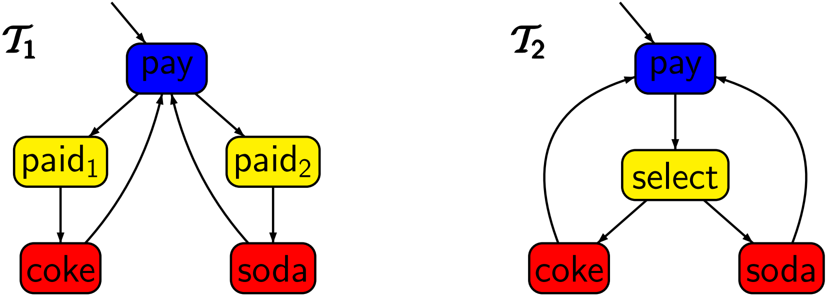
Two beverage machines



for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{pay, drink\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, and $\mathcal{T}_2 \preceq \mathcal{T}_1$

Two beverage machines

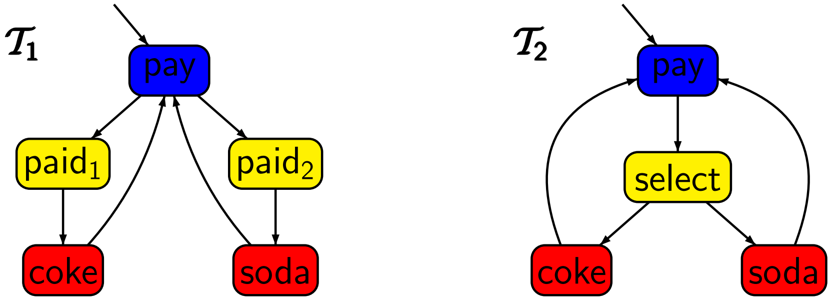


for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{pay, drink\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, and $\mathcal{T}_2 \preceq \mathcal{T}_1$

simulation for $(\mathcal{T}_1, \mathcal{T}_2)$: as before

Two beverage machines



for $AP = \{pay, coke, soda\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, but $\mathcal{T}_2 \not\preceq \mathcal{T}_1$

for $AP = \{pay, drink\}$: $\mathcal{T}_1 \preceq \mathcal{T}_2$, and $\mathcal{T}_2 \preceq \mathcal{T}_1$

simulation for $(\mathcal{T}_2, \mathcal{T}_1)$:

$$\{(pay, pay), (select, paid_1), (select, paid_2), (coke, coke), (soda, soda)\}$$

Simulation condition

$$\begin{array}{ccc} s_1 & -\mathcal{R}- & s_2 \\ \downarrow & & \\ s_1' & & \end{array} \quad \text{can be completed to} \quad \begin{array}{ccc} s_1 & -\mathcal{R}- & s_2 \\ \downarrow & & \downarrow \\ s_1' & -\mathcal{R}- & s_2' \end{array}$$

Path fragment lifting for simulation $\mathcal{R}$

$$s_1 \quad -\mathcal{R}- \quad s_2$$

$$\downarrow$$

$$s_{1,1}$$

$$\downarrow$$

$$s_{1,2}$$

$$\downarrow$$

$$s_{1,3}$$

$$\downarrow$$

$$\vdots$$

$$\downarrow$$

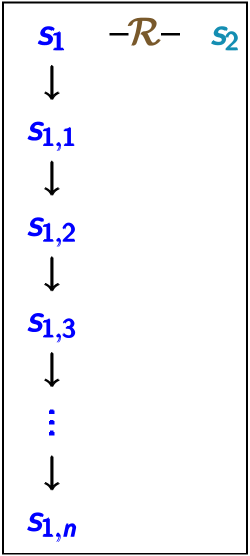
$$s_{1,n}$$

Path fragment lifting for simulation $\mathcal{R}$

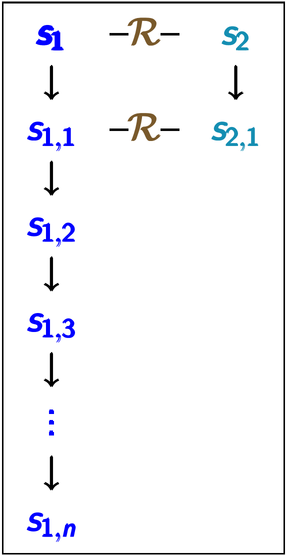
$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
\vdots & & \\
\downarrow & & \\
s_{1,n} & &
\end{array}
$$

can be completed to

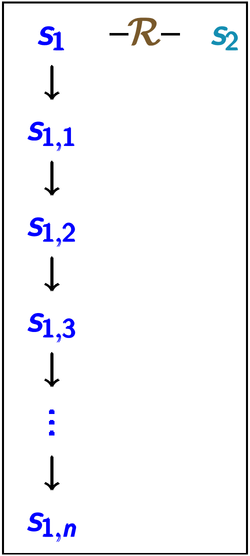
Path fragment lifting for simulation $\mathcal{R}$



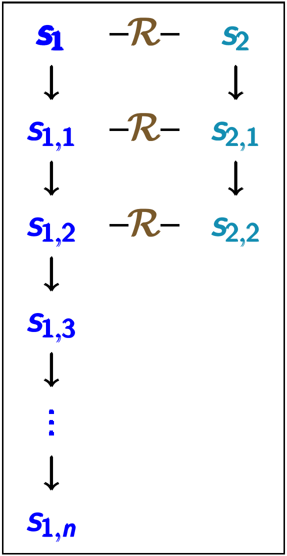
can be completed to



Path fragment lifting for simulation $\mathcal{R}$



can be completed to



Path fragment lifting for simulation $\mathcal{R}$

BSEQOR5.1-9

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
\vdots & & \\
\downarrow & & \\
s_{1,n} & &
\end{array}
$$

can be completed to

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & -\mathcal{R}- & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & -\mathcal{R}- & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & -\mathcal{R}- & s_{2,3} \\
\downarrow & & \\
\vdots & & \\
\downarrow & & \\
s_{1,n} & &
\end{array}
$$

Path fragment lifting for simulation $\mathcal{R}$

BSEQOR5.1-9

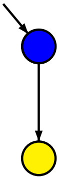
$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
\vdots & & \\
\downarrow & & \\
s_{1,n} & &
\end{array}
$$

can be completed to

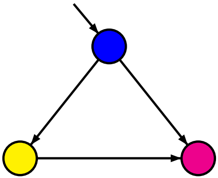
$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & -\mathcal{R}- & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & -\mathcal{R}- & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & -\mathcal{R}- & s_{2,3} \\
\downarrow & & \downarrow \\
\vdots & & \vdots \\
\downarrow & & \downarrow \\
s_{1,n} & -\mathcal{R}- & s_{2,n}
\end{array}
$$

Correct or wrong?

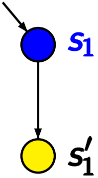
BSEQOR5.1-12



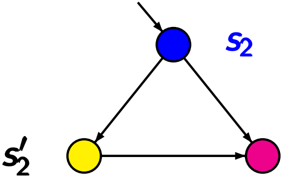
$\preceq$



Correct or wrong?

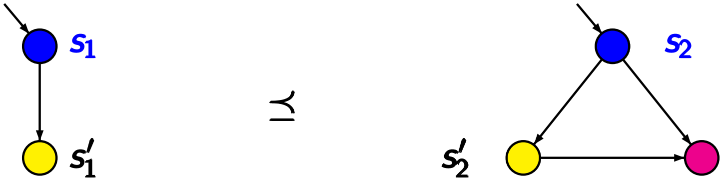


$\preceq$



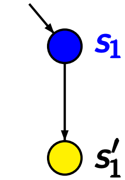
correct.

Correct or wrong?

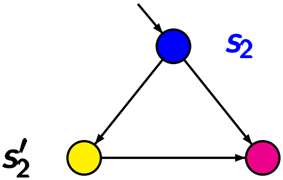


correct. simulation: $\{(s_1, s_2), (s'_1, s'_2)\}$

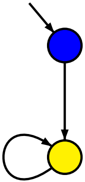
Correct or wrong?



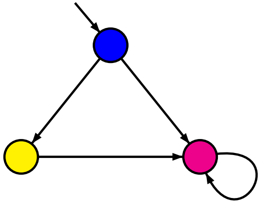
$\preceq$



correct. simulation: $\{(s_1, s_2), (s_1', s_2')\}$

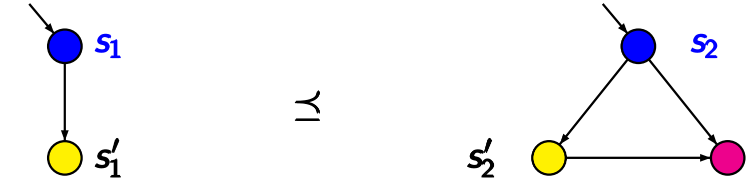


$\preceq$

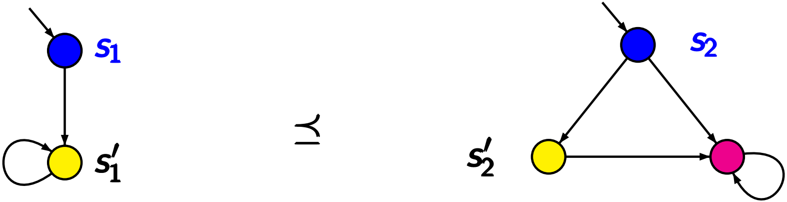


Correct or wrong?

BSEQOR5.1-12



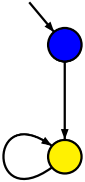
correct. simulation: $\{(s_1, s_2), (s_1', s_2')\}$



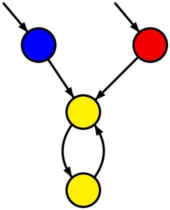
wrong. there is <u>no</u> path fragment in $\mathcal{T}_2$ corresponding to the path fragment $s_1\, s_1'\, s_1'$

Correct or wrong?

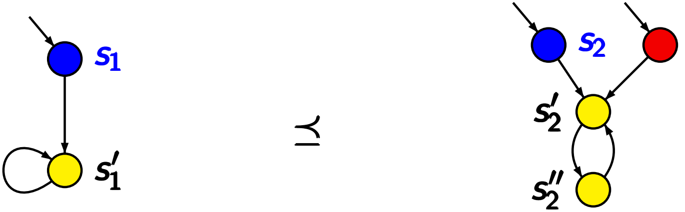
BSEQOR5.1-13



$\preceq$

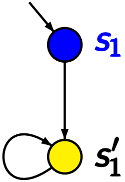


Correct or wrong?

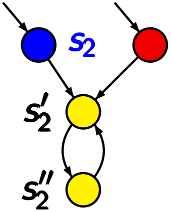


correct. simulation: $\{(s_1, s_2), (s'_1, s'_2), (s'_1, s''_2)\}$

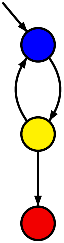
Correct or wrong?



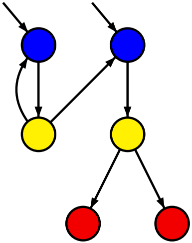
$\preceq$



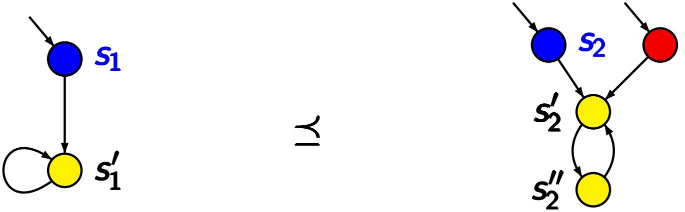
correct. simulation: $\{(s_1, s_2), (s_1', s_2'), (s_1', s_2'')\}$



$\preceq$



Correct or wrong?



correct. simulation: $\{(s_1, s_2), (s_1', s_2'), (s_1', s_2'')\}$

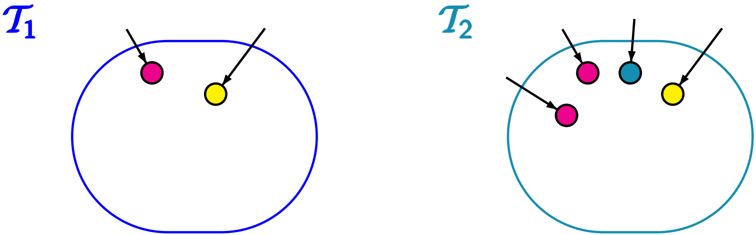
wrong. $s_1' \not\preceq s_2'$ and $s_1' \not\preceq t_2'$

Simulation preorder ...

- as a relation that compares two transition systems

Simulation preorder ...

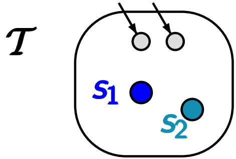
- as a relation that compares two transition systems



Simulation preorder ...

- as a relation that compares two transition systems
- as a relation on the states of one transition system

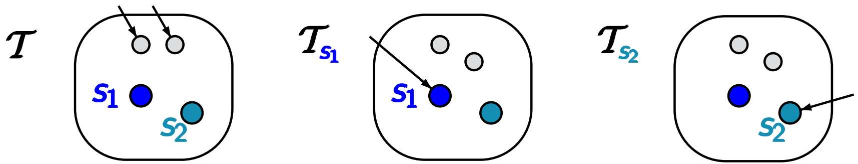
- as a relation that compares two transition systems
- as a relation on the states of one transition system



$s_1 \preceq_{\mathcal{T}} s_2$ iff ?

Simulation preorder ...

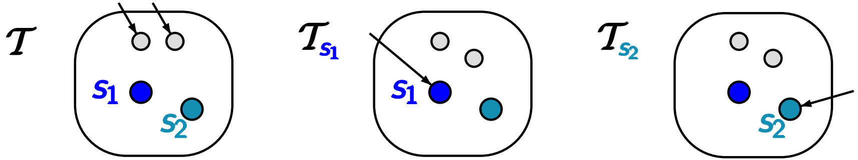
- as a relation that compares two transition systems
- as a relation on the states of one transition system



$s_1 \preceq_{\mathcal{T}} s_2$ iff $\mathcal{T}_{s_1} \preceq \mathcal{T}_{s_2}$

Simulation preorder ...

- as a relation that compares two transition systems
- as a relation on the states of one transition system



$s_1 \preceq_{\mathcal{T}} s_2$ iff $\mathcal{T}_{s_1} \preceq \mathcal{T}_{s_2}$

iff there exists a simulation $\mathcal{R}$ for $\mathcal{T}$ with $(s_1, s_2) \in \mathcal{R}$

Simulation preorder for a single TS

Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on S such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on S such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

(1) $L(s_1) = L(s_2)$

(2) $\ldots$

Simulation preorder for a single TS

Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on S such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

(1) $L(s_1) = L(s_2)$

(2) each transition of s_1 can be mimicked by a transition of s_2

Simulation preorder for a single TS

Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on S such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

(1) $L(s_1) = L(s_2)$

(2) each transition of s_1 can be mimicked by a transition of s_2

$$\begin{array}{ccc} s_1 & \preceq_{\mathcal{T}} & s_2 \\ \downarrow & & \\ s_1' & & \end{array} \qquad \text{can be completed to} \qquad \begin{array}{ccc} s_1 & \preceq_{\mathcal{T}} & s_2 \\ \downarrow & & \downarrow \\ s_1' & \preceq_{\mathcal{T}} & s_2' \end{array}$$

Simulation preorder for a single TS

Let $\mathcal{T} = (S, Act, \rightarrow, \ldots)$ be a transition system.

The simulation preorder $\preceq_{\mathcal{T}}$ is the coarsest relation on S such that for all states $s_1, s_2 \in S$ with $s_1 \preceq_{\mathcal{T}} s_2$:

(1) $L(s_1) = L(s_2)$

(2) each transition of s_1 can be mimicked by a transition of s_2

$\preceq_{\mathcal{T}}$ is a preorder, i.e., transitive and reflexive.

Simulation for a TS

Let $\mathcal{T}$ be a transition system with state space S.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

Simulation for a TS

Let $\mathcal{T}$ be a transition system with state space S.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L(s_1) = L(s_2)$

(2) $\ldots$

Let $\mathcal{T}$ be a transition system with state space S.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L(s_1) = L(s_2)$

(2) for all $(s_1, s_2) \in \mathcal{R}$:

$$\forall s_1' \in Post(s_1)\, \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$

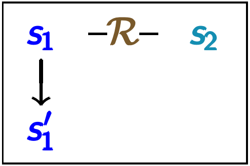
Let $\mathcal{T}$ be a transition system with state space S.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

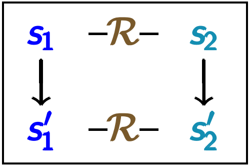
(1) if $(s_1, s_2) \in \mathcal{R}$ then $L(s_1) = L(s_2)$

(2) for all $(s_1, s_2) \in \mathcal{R}$:

$$\forall s_1' \in Post(s_1)\, \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$



can be completed to



Let $\mathcal{T}$ be a transition system with state space S.

A simulation for $\mathcal{T}$ is a binary relation $\mathcal{R} \subseteq S \times S$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L(s_1) = L(s_2)$

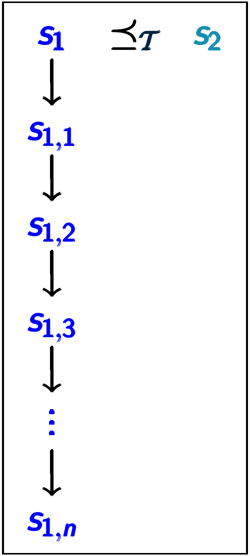
(2) for all $(s_1, s_2) \in \mathcal{R}$:

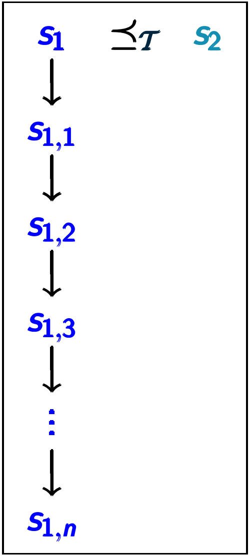
$$\forall s_1' \in Post(s_1)\, \exists s_2' \in Post(s_2) \text{ s.t. } (s_1', s_2') \in \mathcal{R}$$

simulation preorder $\preceq_{\mathcal{T}}$:

$s_1 \preceq_{\mathcal{T}} s_2$ iff there exists a simulation $\mathcal{R}$ for $\mathcal{T}$ s.t. $(s_1, s_2) \in \mathcal{R}$

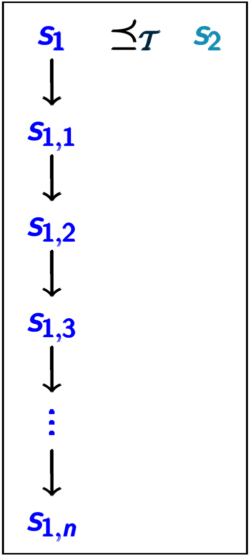
Path fragment lifting for $\preceq_{\mathcal{T}}$



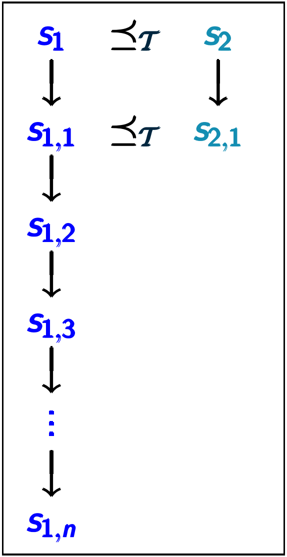


can be completed to

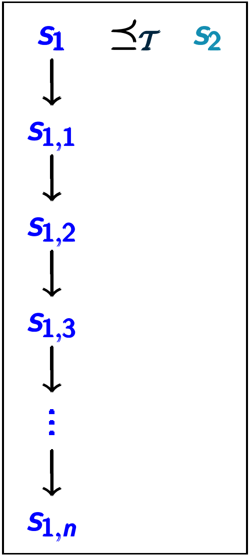
Path fragment lifting for $\preceq_{\mathcal{T}}$



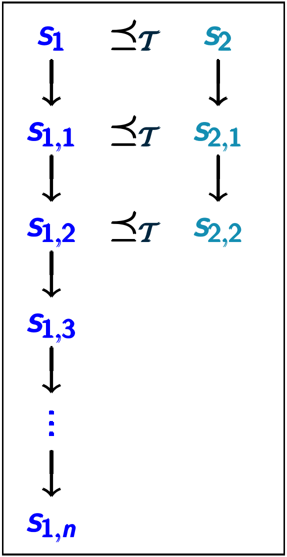
can be completed to



Path fragment lifting for $\preceq_{\mathcal{T}}$



can be completed to



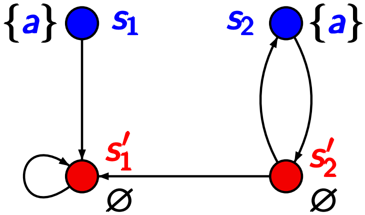
Path fragment lifting for $\preceq_{\mathcal{T}}$

$$\begin{array}{ccc}
s_1 & \preceq_{\mathcal{T}} & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
\vdots & & \\
\downarrow & & \\
s_{1,n} & &
\end{array}$$

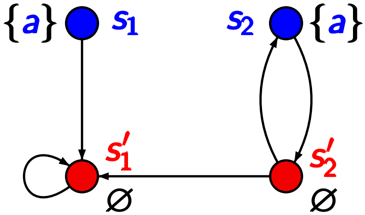
can be completed to

$$\begin{array}{ccc}
s_1 & \preceq_{\mathcal{T}} & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & \preceq_{\mathcal{T}} & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & \preceq_{\mathcal{T}} & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & \preceq_{\mathcal{T}} & s_{2,3} \\
\downarrow & & \downarrow \\
\vdots & & \vdots \\
\downarrow & & \downarrow \\
s_{1,n} & \preceq_{\mathcal{T}} & s_{2,n}
\end{array}$$

Example: simulation preorder $\preceq_{\mathcal{T}}$



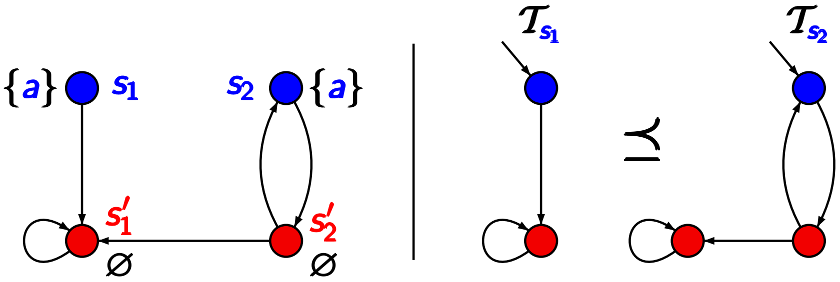
$s_1 \preceq_{\mathcal{T}} s_2$



$s_1 \preceq_{\mathcal{T}} s_2$ as

$\{(s_1, s_2), (s_1', s_2'), (s_1', s_1')\}$ is a simulation for $\mathcal{T}$

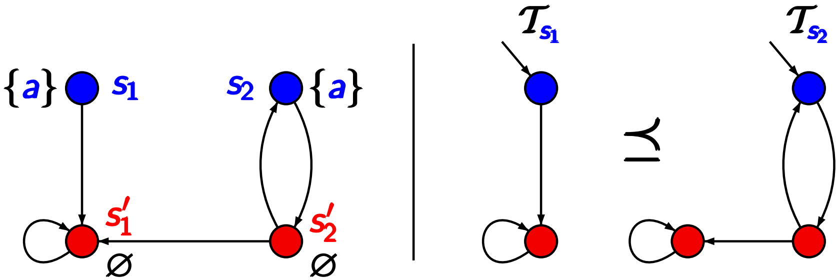
Example: simulation preorder $\preceq_{\mathcal{T}}$



$s_1 \preceq_{\mathcal{T}} s_2$ as

$\{(s_1, s_2), (s_1', s_2'), (s_1', s_1')\}$ is a simulation for $\mathcal{T}$

Example: simulation preorder $\preceq_{\mathcal{T}}$



$s_1 \preceq_{\mathcal{T}} s_2$ as

$\{(s_1, s_2), (s'_1, s'_2), (s'_1, s'_1)\}$ is a simulation for $\mathcal{T}$

$s_1 \rightarrow s'_1 \rightarrow s'_1 \rightarrow s'_1 \rightarrow \ldots$

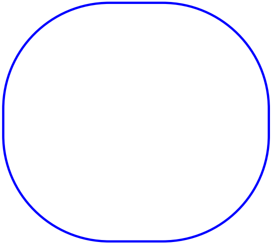
is simulated by

$s_2 \rightarrow s'_2 \rightarrow s'_1 \rightarrow s'_1 \rightarrow \ldots$

Abstraction and simulation

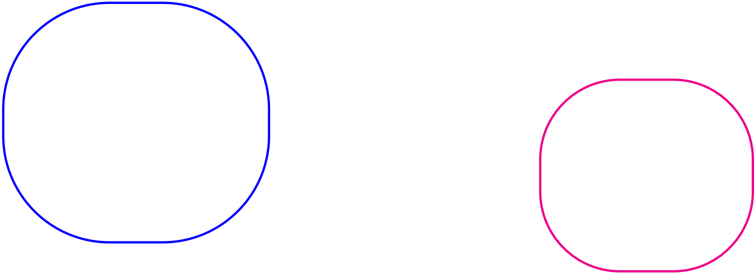
GRM5.5-6

Abstraction and simulation



transition system $\mathcal{T}$
with state space S

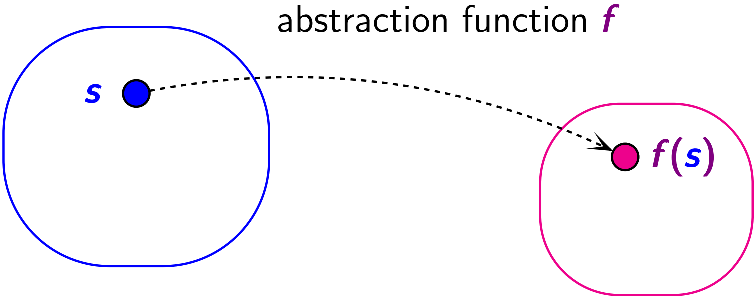
Abstraction and simulation



transition system $\mathcal{T}$
with state space S

"small" abstract
state space S'

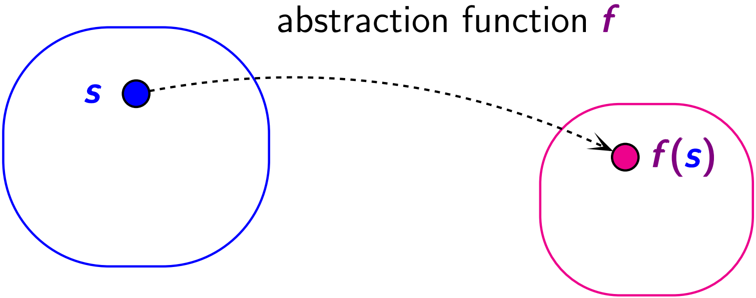
Abstraction and simulation



transition system $\mathcal{T}$ with state space S

abstract transition system $\mathcal{T}_f$ with state space S'

Abstraction and simulation



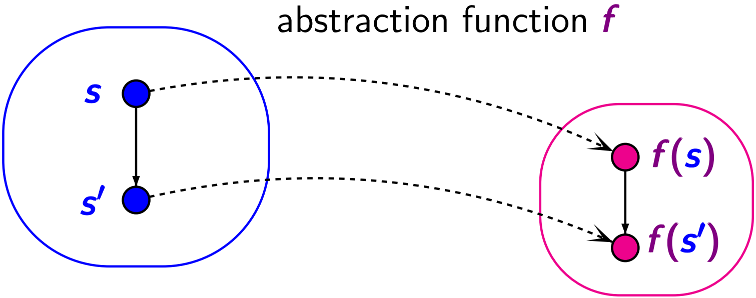
transition system $\mathcal{T}$ with state space S

abstract transition system $\mathcal{T}_f$ with state space S'

lifting of transitions:

$$\frac{s \longrightarrow s'}{f(s) \longrightarrow f(s')}$$

Abstraction and simulation



transition system $\mathcal{T}$ with state space S

abstract transition system $\mathcal{T}_f$ with state space S'

lifting of transitions:

$$\frac{s \longrightarrow s'}{f(s) \longrightarrow f(s')}$$

Abstraction and simulation

given: transition system $\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$

set S' and abstraction function $f : S \to S'$

s.t. $L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

given: transition system $\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$

set S' and abstraction function $f : S \to S'$

s.t. $L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

goal: define abstract transition system $\mathcal{T}_f$

with state space S' s.t. $\mathcal{T} \preceq \mathcal{T}_f$

Abstraction and simulation

abstraction function $f : S \rightarrow S'$ s.t.
$L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

transition system
$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$
$$\Downarrow$$
abstract transition system
$$\mathcal{T}_f = (S', Act', \longrightarrow_f, S'_0, AP, L')$$

Abstraction and simulation

abstraction function $f : S \to S'$ s.t.

$L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

transition system

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

$$\Downarrow$$

abstract transition system

$$\mathcal{T}_f = (S', Act', \longrightarrow_f, S'_0, AP, L')$$

where $S'_0 = \{f(s_0) : s_0 \in S_0\}$ and $L'(f(s)) = L(s)$

$$\frac{s \longrightarrow s'}{f(s) \longrightarrow_f f(s')}$$

abstraction function $f : S \to S'$ s.t.

$L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

transition system

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

$$\Downarrow$$

abstract transition system

$$\mathcal{T}_f = (S', Act', \longrightarrow_f, S'_0, AP, L')$$

Then $\mathcal{T} \preceq \mathcal{T}_f$

Abstraction and simulation

abstraction function $f : S \to S'$ s.t.

$L(s) = L(t)$ if $f(s) = f(t)$ for all $s, t \in S$

transition system

$$\mathcal{T} = (S, Act, \longrightarrow, S_0, AP, L)$$

$$\Downarrow$$

abstract transition system

$$\mathcal{T}_f = (S', Act', \longrightarrow_f, S'_0, AP, L')$$

Then $\mathcal{T} \preceq \mathcal{T}_f$ $\longleftarrow$ $\mathcal{R} = \{\langle s, f(s)\rangle : s \in S\}$ is a simulation for $(\mathcal{T}, \mathcal{T}_f)$

Data abstraction


```
WHILE x > 0 DO
   x := x−1;
   y := y+1
OD
IF even(y)
  THEN return "1"
  ELSE return "0"
FI
```

$x \in \mathbb{N}$

$y \in \mathbb{N}$

Data abstraction


```
WHILE x > 0 DO
    x := x-1;
    y := y+1
OD
IF even(y)
  THEN return "1"
  ELSE return "0"
FI
```

data abstr. $\longrightarrow$

$x \in \mathbb{N} \quad \longrightarrow \quad x \in \{gzero, zero\}$

$y \in \mathbb{N} \quad \longrightarrow \quad y \in \{even, odd\}$

Data abstraction


```
WHILE x > 0 DO
    x := x−1;
    y := y+1
OD
IF even(y)
  THEN return "1"
  ELSE return "0"
FI
```

data abstr. ⟶

```
WHILE x = gzero DO
    x := gzero or x := zero
    IF y = even
       THEN y := odd
       ELSE y := even
    FI
OD
IF y = even
    THEN return "1"
    ELSE return "0"
FI
```

$x \in \mathbb{N} \longrightarrow x \in \{gzero, zero\}$

$y \in \mathbb{N} \longrightarrow y \in \{even, odd\}$

Data abstraction


```
WHILE x > 0 DO
   x := x-1;
   y := y+1
OD
IF even(y)
  THEN return "1"
  ELSE return "0"
FI
```

data abstr. ⟶

```
WHILE x = gzero DO
   x := gzero or x := zero
   IF y = even
      THEN y := odd
      ELSE y := even
   FI
OD
IF y = even
   THEN return "1"
   ELSE return "0"
FI
```

concrete operation ⇝ abstract operation

Data abstraction


```
WHILE x > 0 DO
    x := x-1;
    y := y+1
OD
IF even(y)
  THEN return "1"
  ELSE return "0"
FI
```

data abstr. $\longrightarrow$

```
WHILE x = gzero DO
    x := gzero or x := zero
    IF y = even
       THEN y := odd
       ELSE y := even
    FI
OD
IF y = even
    THEN return "1"
    ELSE return "0"
FI
```

concrete operation $\rightsquigarrow$ abstract operation, e.g.,

$x := x-1$ $\qquad$ *gzero* $\mapsto$ *gzero* or *zero*

Abstraction and simulation

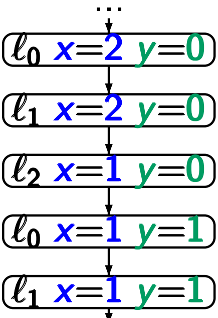
abstract TS simulates the concrete one

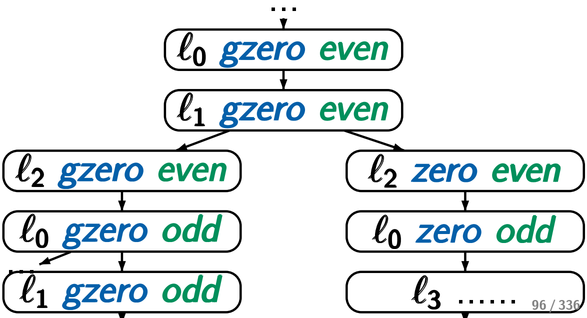
```
WHILE x > 0 DO
   x := x−1
   y := y+1
OD

IF even(y)
 THEN return 1
 ELSE return 0
```

```
WHILE x = gzero DO
   x := gzero or x := zero
   IF y = even
        THEN y := odd
        ELSE y := even
   FI
OD
IF y = even
     THEN return 1
     ELSE return 0  FI
```

```
ℓ0  WHILE  x > 0 DO
ℓ1     x := x−1
ℓ2     y := y+1
    OD

ℓ3  IF  even(y)
ℓ4   THEN  return 1
ℓ5   ELSE  return 0
```

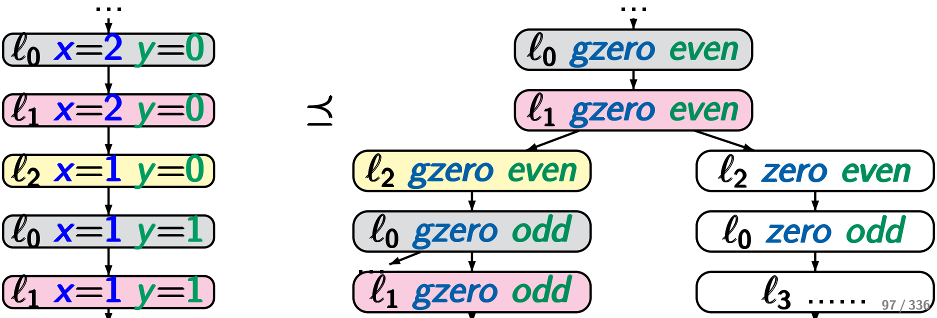
```
ℓ0  WHILE  x = gzero DO
ℓ1     x := gzero or x := zero
ℓ2     IF  y = even
              THEN  y := odd
              ELSE  y := even
       FI
    OD
ℓ3  IF  y = even
ℓ4        THEN  return 1
ℓ5        ELSE  return 0   FI
```





```
ℓ0  WHILE x > 0 DO
ℓ1      x := x−1
ℓ2      y := y+1
    OD

ℓ3  IF even(y)
ℓ4   THEN return 1
ℓ5   ELSE return 0
```

```
ℓ0  WHILE x = gzero DO
ℓ1      x := gzero or x := zero
ℓ2      IF y = even
              THEN y := odd
              ELSE y := even
        FI
    OD
ℓ3  IF y = even
ℓ4        THEN return 1
ℓ5        ELSE return 0   FI
```



Simulation preorder vs. and trace inclusion

Simulation preorder vs. and trace inclusion

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \Longrightarrow \quad Tracesfin(\mathcal{T}_1) \subseteq Tracesfin(\mathcal{T}_2)$$

Simulation preorder vs. and trace inclusion

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \Longrightarrow \quad \mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$$

reason: path fragment lifting for $\preceq$

Simulation preorder vs. and trace inclusion

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies \mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$$

if $\mathcal{T}_1$ does not have terminal states, then:

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies \mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$$

Simulation preorder vs. and trace inclusion

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies Tracesfin(\mathcal{T}_1) \subseteq Tracesfin(\mathcal{T}_2)$$

if $\mathcal{T}_1$ does not have terminal states, then:

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$$

... does not hold if $\mathcal{T}_1$ has terminal states ...

Simulation preorder vs. and trace inclusion

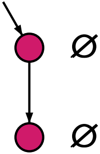
$$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \Longrightarrow \quad Tracesfin(\mathcal{T}_1) \subseteq Tracesfin(\mathcal{T}_2)$$

if $\mathcal{T}_1$ does not have terminal states, then:

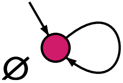
$$\mathcal{T}_1 \preceq \mathcal{T}_2 \quad \Longrightarrow \quad Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$$

... does not hold if $\mathcal{T}_1$ has terminal states ...

$\mathcal{T}_1$



$\mathcal{T}_2$



$\mathcal{T}_1 \preceq \mathcal{T}_2$

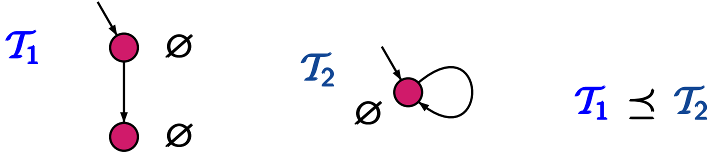
Simulation preorder vs. and trace inclusion

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies Tracesfin(\mathcal{T}_1) \subseteq Tracesfin(\mathcal{T}_2)$$

if $\mathcal{T}_1$ does not have terminal states, then:

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \implies Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$$

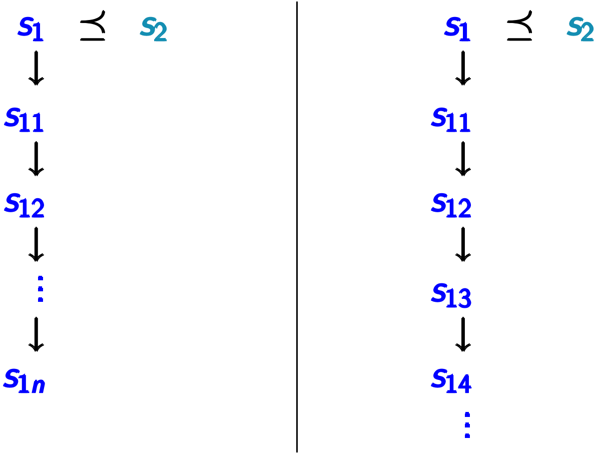
... does not hold if $\mathcal{T}_1$ has terminal states ...



$$Traces(\mathcal{T}_1) = \{ \varnothing\varnothing \} \neq \{ \varnothing^\omega \} = Traces(\mathcal{T}_2)$$

Path fragment lifting

simulation preorder



terminal

finite
path

infinite
path

simulation preorder

$$
\begin{array}{ccc}
s_1 & \preceq & s_2 \\
\downarrow & & \downarrow \\
s_{11} & \preceq & s_{21} \\
\downarrow & & \downarrow \\
s_{12} & \preceq & s_{22} \\
\downarrow & & \downarrow \\
\vdots & & \vdots \\
\downarrow & & \downarrow \\
s_{1n} & \preceq & s_{2n}
\end{array}
\qquad\Bigg|\qquad
\begin{array}{ccc}
s_1 & \preceq & s_2 \\
\downarrow & & \downarrow \\
s_{11} & \preceq & s_{21} \\
\downarrow & & \downarrow \\
s_{12} & \preceq & s_{22} \\
\downarrow & & \downarrow \\
s_{13} & \preceq & s_{23} \\
\downarrow & & \downarrow \\
s_{14} & \preceq & s_{24} \\
\vdots & & \vdots
\end{array}
$$

terminal finite path

not necessarily terminal

infinite path

infinite path

simulation preorder

$$\begin{array}{ccc} s_1 & \preceq & s_2 \\ \downarrow & & \downarrow \\ s_{11} & \preceq & s_{21} \\ \downarrow & & \downarrow \\ s_{12} & \preceq & s_{22} \\ \downarrow & & \downarrow \\ \vdots & & \vdots \\ \downarrow & & \downarrow \\ s_{1n} & \preceq & s_{2n} \end{array}$$

terminal finite path

not necessarily terminal

bisimulation

$$\begin{array}{ccc} s_1 & \sim & s_2 \\ \downarrow & & \downarrow \\ s_{11} & \sim & s_{21} \\ \downarrow & & \downarrow \\ s_{12} & \sim & s_{22} \\ \downarrow & & \downarrow \\ \vdots & & \vdots \\ \downarrow & & \downarrow \\ s_{1n} & \sim & s_{2n} \end{array}$$

terminal finite path

terminal finite path

Simulation equivalence $\simeq$

Simulation equivalence $\simeq$

kernel of the simulation preorder, i.e.,

$$\simeq \; = \; \preceq \cap \preceq^{-1}$$

kernel of the simulation preorder, i.e.,

$$\simeq \; = \; \preceq \cap \preceq^{-1}$$

For TS $\mathcal{T}_1$ and $\mathcal{T}_2$ over the same set of atomic propositions:

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \text{iff} \quad \mathcal{T}_1 \preceq \mathcal{T}_2 \;\text{ and }\; \mathcal{T}_2 \preceq \mathcal{T}_1$$

Simulation equivalence $\simeq_{\mathcal{T}}$

kernel of the simulation preorder, i.e.,

$$\simeq \; = \; \preceq \cap \preceq^{-1}$$

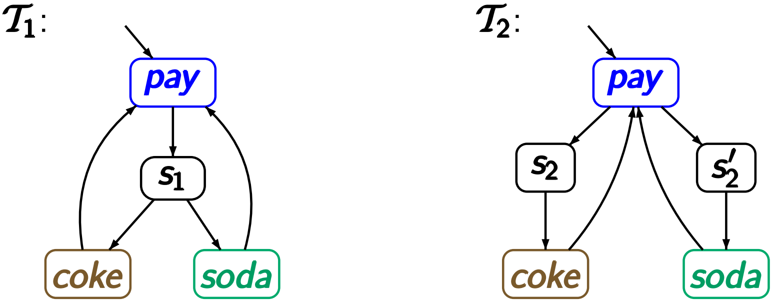
For TS $\mathcal{T}_1$ and $\mathcal{T}_2$ over the same set of atomic propositions:

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \text{iff} \quad \mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \preceq \mathcal{T}_1$$

for states s_1 and s_2 of a TS $\mathcal{T}$:

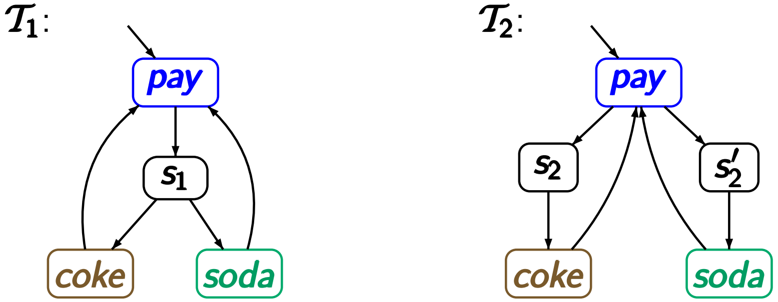
$$s_1 \simeq_{\mathcal{T}} s_2 \quad \text{iff} \quad s_1 \preceq_{\mathcal{T}} s_2 \text{ and } s_2 \preceq_{\mathcal{T}} s_1$$

Two beverage machines



for $AP = \{pay, coke, soda\}$

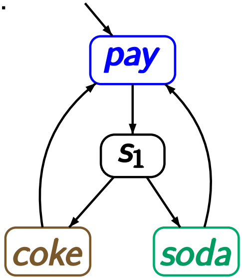
Two beverage machines

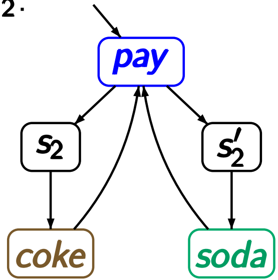


for $AP = \{pay, coke, soda\}$

$\mathcal{T}_2 \preceq \mathcal{T}_1$, but $\mathcal{T}_1 \not\preceq \mathcal{T}_2$

Two beverage machines

$\mathcal{T}_1$: 

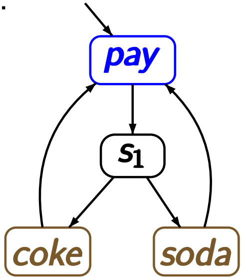
$\mathcal{T}_2$: 

for $AP = \{pay, coke, soda\}$

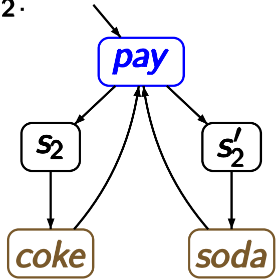
$\mathcal{T}_2 \preceq \mathcal{T}_1$, but $\mathcal{T}_1 \not\simeq \mathcal{T}_2$ $\longleftarrow$ since $\mathcal{T}_1 \not\preceq \mathcal{T}_2$

Two beverage machines

$\mathcal{T}_1$:



$\mathcal{T}_2$:

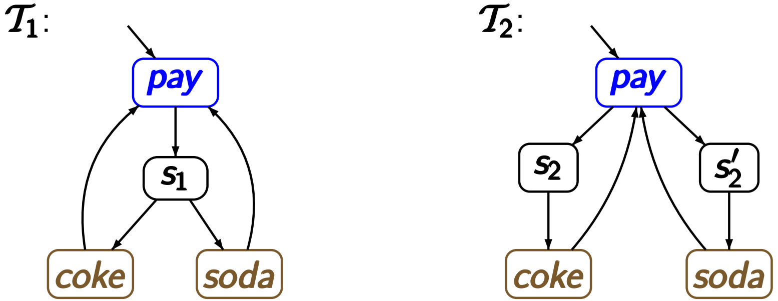


for $AP = \{pay, coke, soda\}$

$\mathcal{T}_2 \preceq \mathcal{T}_1$, but $\mathcal{T}_1 \not\simeq \mathcal{T}_2$ $\longleftarrow$ since $\mathcal{T}_1 \not\preceq \mathcal{T}_2$

for $AP = \{pay, drink\}$:

Two beverage machines

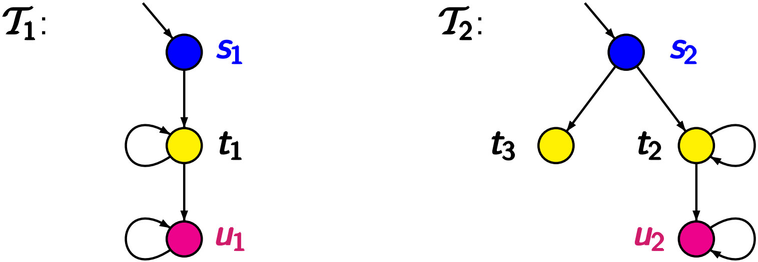


for $AP = \{pay, coke, soda\}$

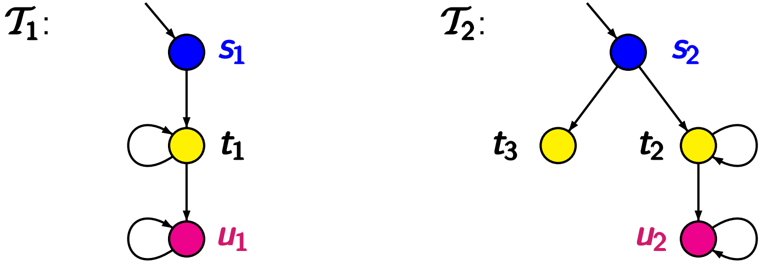
$\mathcal{T}_2 \preceq \mathcal{T}_1$, but $\mathcal{T}_1 \not\simeq \mathcal{T}_2$ ⟵ since $\mathcal{T}_1 \not\preceq \mathcal{T}_2$

for $AP = \{pay, drink\}$: $\mathcal{T}_1 \simeq \mathcal{T}_2$

Example: simulation equivalent TS



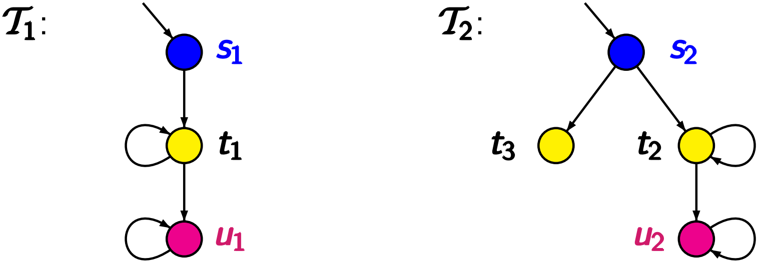
Example: simulation equivalent TS



simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$$\{(s_1, s_2), (t_1, t_2), (u_1, u_2)\}$$

Example: simulation equivalent TS



simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$$\{(s_1, s_2), (t_1, t_2), (u_1, u_2)\}$$

simulation for $(\mathcal{T}_2, \mathcal{T}_1)$:

$$\{(s_2, s_1), (t_2, t_1), (t_3, t_1), (u_2, u_1)\}$$

Bisimulation vs. simulation equivalence

Bisimulation vs. simulation equivalence

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

Bisimulation vs. simulation equivalence

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

   Proof: Let $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$.

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

Bisimulation vs. simulation equivalence

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

   Proof: Let $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$.

   - $\mathcal{R}$ is a simulation for $(\mathcal{T}_1, \mathcal{T}_2)$

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

   Proof: Let $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$.

   - $\mathcal{R}$ is a simulation for $(\mathcal{T}_1, \mathcal{T}_2) \implies \mathcal{T}_1 \preceq \mathcal{T}_2$

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

That is:

1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

   Proof: Let $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$.

   - $\mathcal{R}$ is a simulation for $(\mathcal{T}_1, \mathcal{T}_2)$ $\Longrightarrow$ $\mathcal{T}_1 \preceq \mathcal{T}_2$
   - $\mathcal{R}^{-1}$ is a simulation for $(\mathcal{T}_2, \mathcal{T}_1)$

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

> Bisimulation equivalence $\sim$ is strictly finer than simulation equivalence $\simeq$

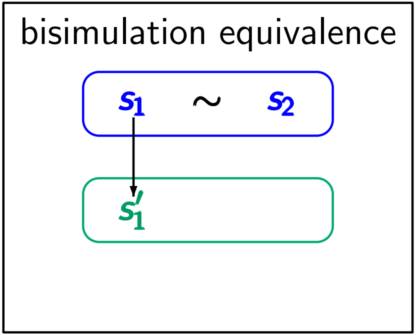
That is:

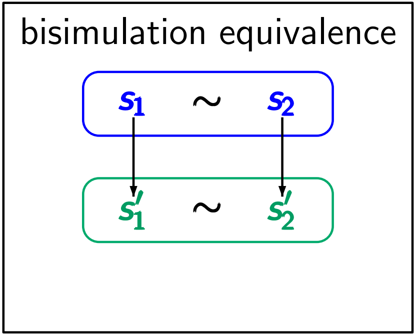
1. $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \simeq \mathcal{T}_2$

   Proof: Let $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$.

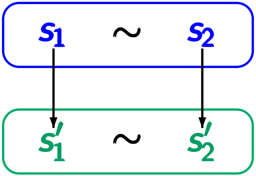
   - $\mathcal{R}$ is a simulation for $(\mathcal{T}_1, \mathcal{T}_2) \implies \mathcal{T}_1 \preceq \mathcal{T}_2$
   - $\mathcal{R}^{-1}$ is a simulation for $(\mathcal{T}_2, \mathcal{T}_1) \implies \mathcal{T}_2 \preceq \mathcal{T}_1$

2. there exist TS $\mathcal{T}_1$ and $\mathcal{T}_2$ s.t. $\mathcal{T}_1 \simeq \mathcal{T}_2$ and $\mathcal{T}_1 \not\sim \mathcal{T}_2$

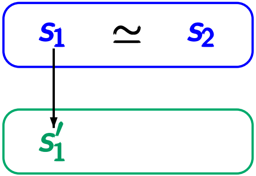

bisimulation equivalence
s_1 $\sim$ s_2
s_1'

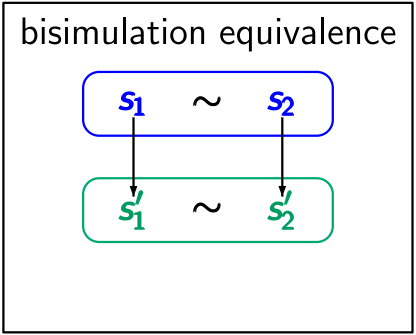

bisimulation equivalence
s_1 ∼ s_2
s_1' ∼ s_2'

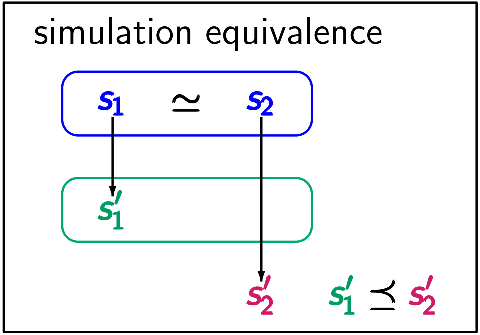
bisimulation equivalence

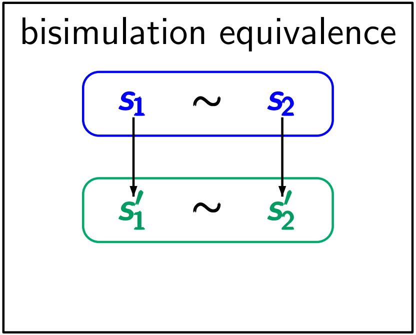


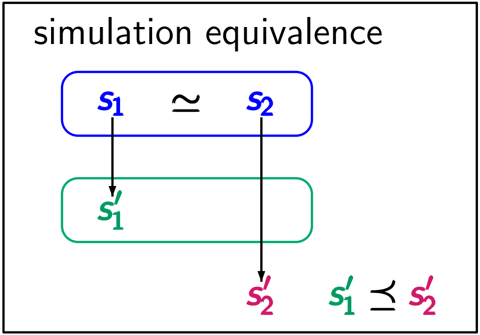
simulation equivalence

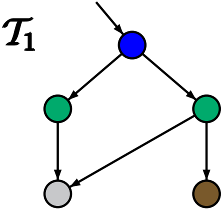


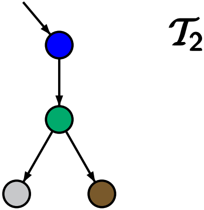

bisimulation equivalence
s_1 $\sim$ s_2
s_1' $\sim$ s_2'

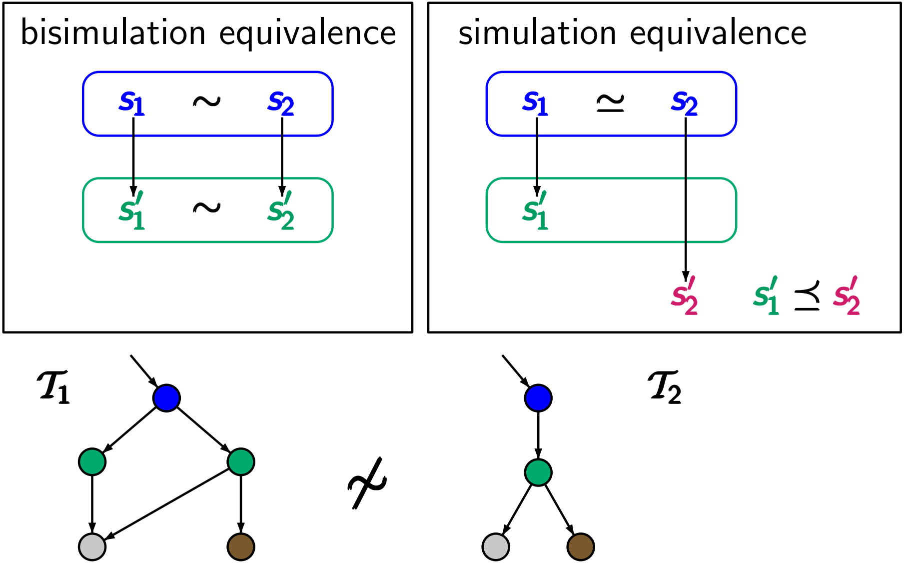

simulation equivalence
s_1 $\simeq$ s_2
s_1'
s_2'
$s_1' \preceq s_2'$

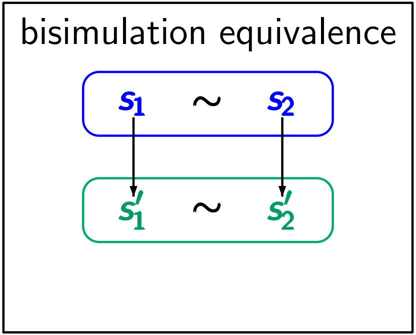

bisimulation equivalence
$s_1 \sim s_2$
$s_1' \sim s_2'$

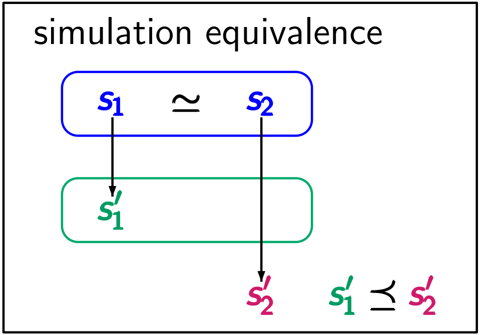

simulation equivalence
$s_1 \simeq s_2$
s_1'
s_2'
$s_1' \preceq s_2'$

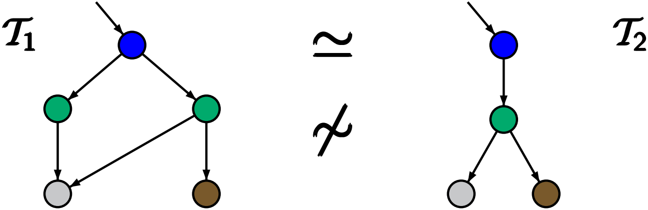

$\mathcal{T}_1$

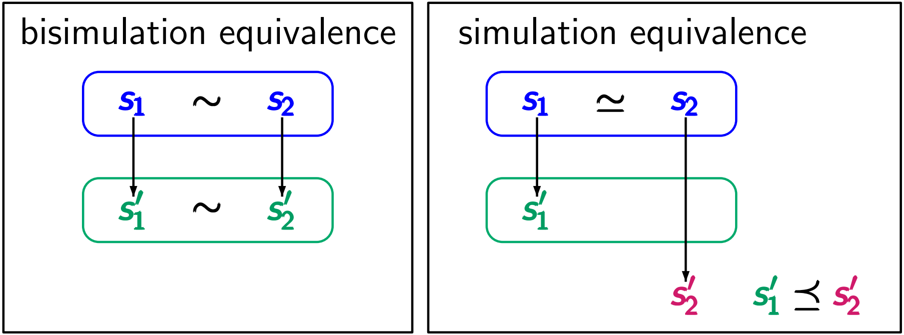

$\mathcal{T}_2$

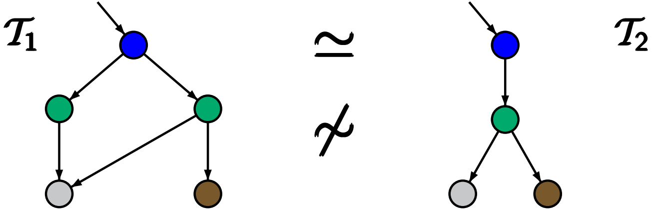

bisimulation equivalence
$s_1 \sim s_2$
$s_1' \sim s_2'$
simulation equivalence
$s_1 \simeq s_2$
s_1'
s_2'
$s_1' \preceq s_2'$
$\mathcal{T}_1$
$\not\sim$
$\mathcal{T}_2$


bisimulation equivalence
$s_1 \sim s_2$
$s_1' \sim s_2'$

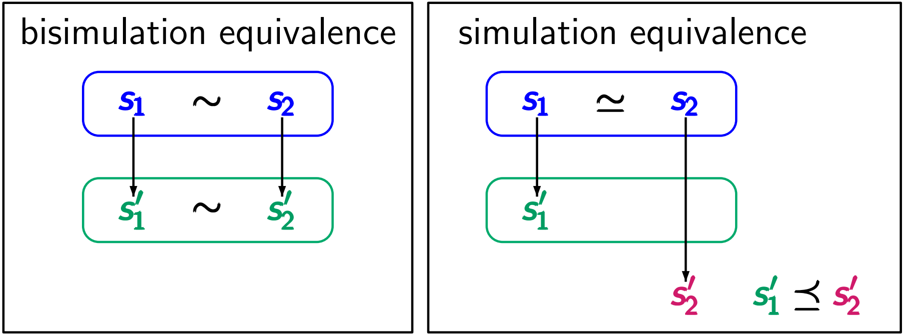

simulation equivalence
$s_1 \simeq s_2$
s_1'
s_2'
$s_1' \preceq s_2'$

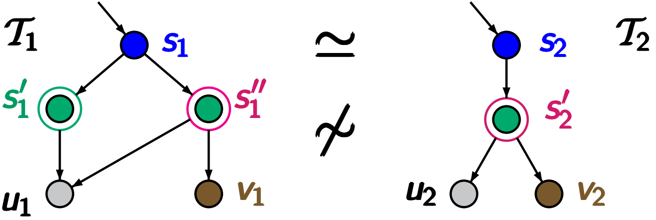

$\mathcal{T}_1$
$\simeq$
$\not\sim$
$\mathcal{T}_2$





$\mathcal{T}_2 \preceq \mathcal{T}_1$, as $\mathcal{T}_2$ is a "subsystem" of $\mathcal{T}_1$





simulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$$\{(s_1, s_2), (s_1', s_2'), (s_1'', s_2'), (u_1, u_2), (v_1, v_2)\}$$

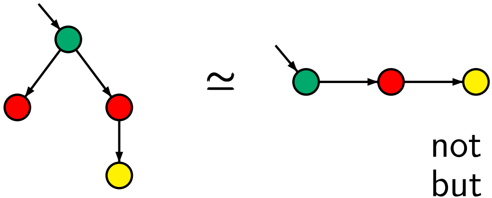
Simulation vs trace equivalence

Simulation vs trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

Simulation vs trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$$

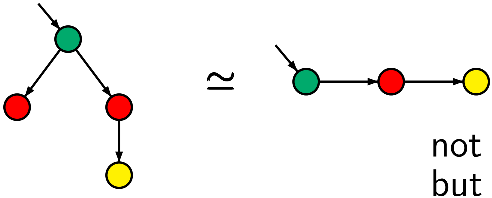


not trace equivalent
but simulation equivalent

Simulation vs trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$$

$$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2) \quad \not\Longrightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$

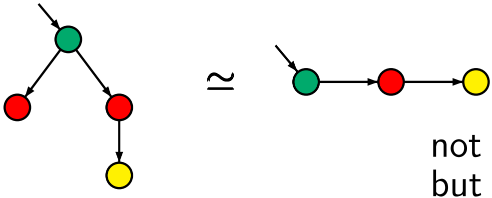


not trace equivalent
but simulation equivalent

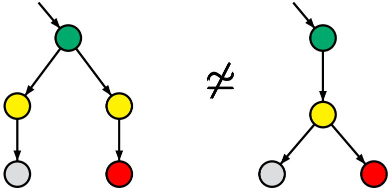
Simulation vs trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

$$\mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2) \quad \not\Longrightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$



not trace equivalent
but simulation equivalent

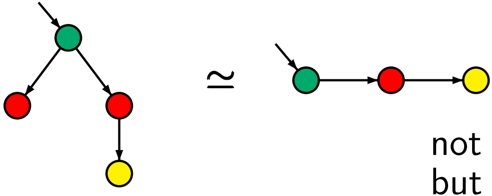


trace equivalent
not simulation equivalent

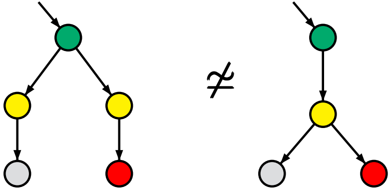
Simulation vs trace equivalence $\longleftarrow$ incomparable

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Rightarrow \quad Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$$

$$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2) \quad \not\Rightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$



not trace equivalent
but simulation equivalent



trace equivalent
not simulation equivalent

Simulation vs. finite trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$$

$$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2) \quad \not\Longrightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \Longrightarrow \quad Tracesfin(\mathcal{T}_1) = Tracesfin(\mathcal{T}_2)$$

Simulation vs. finite trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Longrightarrow \quad \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

$$\mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2) \quad \not\Longrightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \Longrightarrow \quad \mathit{Tracesfin}(\mathcal{T}_1) = \mathit{Tracesfin}(\mathcal{T}_2)$$

while "$\Longleftarrow$" does not hold

Simulation vs. finite trace equivalence

$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \not\Rightarrow \quad \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

$$\mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2) \quad \not\Rightarrow \quad \mathcal{T}_1 \simeq \mathcal{T}_2$$

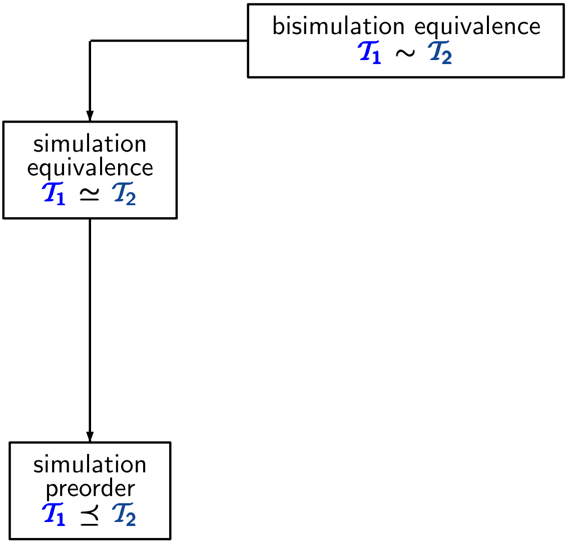
$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \Longrightarrow \quad \mathit{Tracesfin}(\mathcal{T}_1) = \mathit{Tracesfin}(\mathcal{T}_2)$$

while "$\Longleftarrow$" does not hold

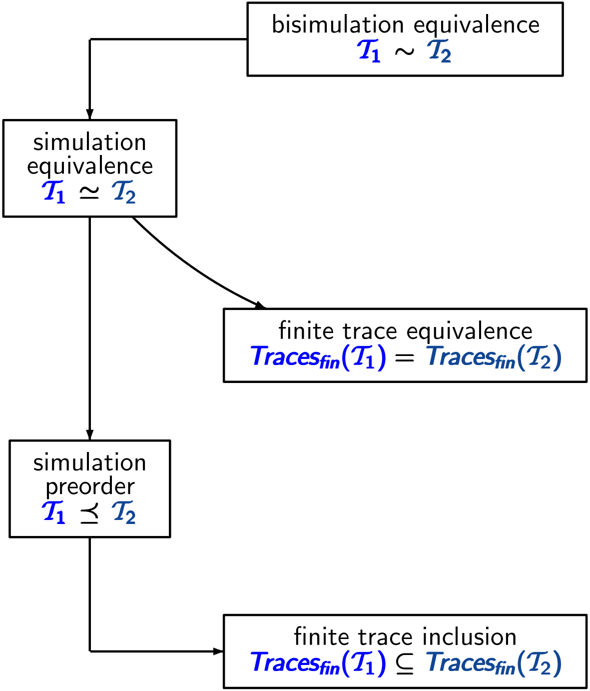
If $\mathcal{T}_1$, $\mathcal{T}_2$ do not have terminal states then:

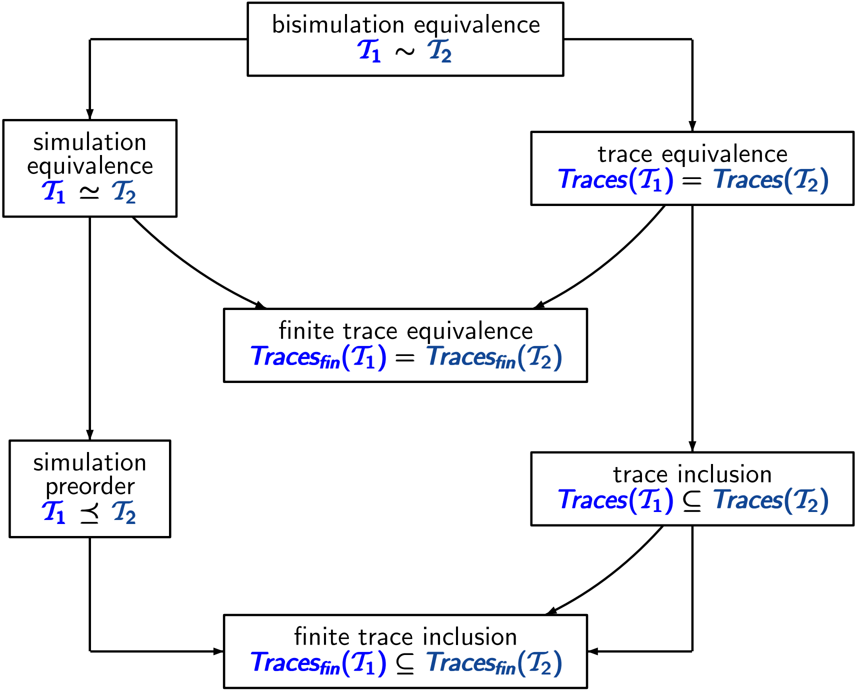
$$\mathcal{T}_1 \simeq \mathcal{T}_2 \quad \Longrightarrow \quad \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

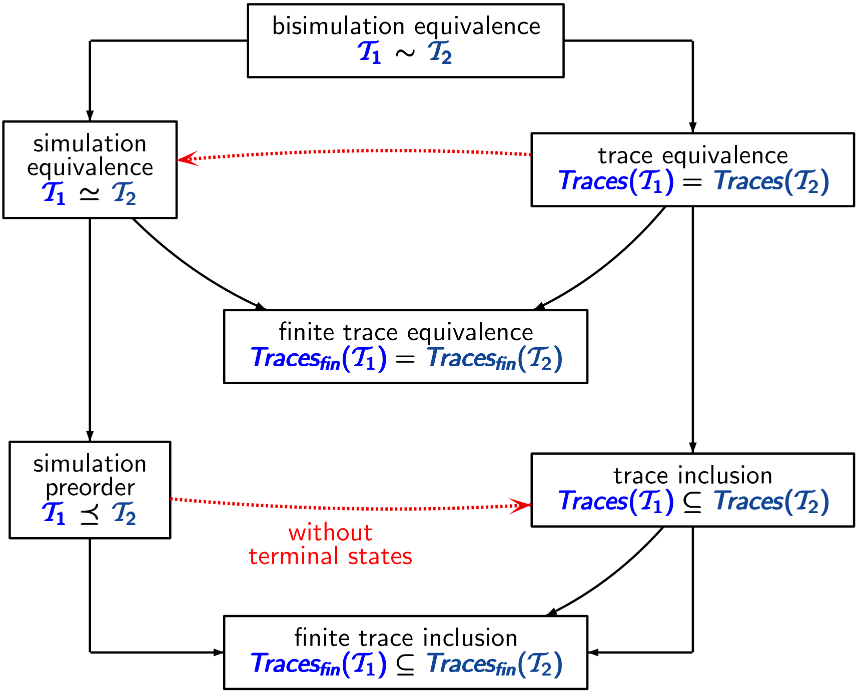
Summary: trace and (bi)simulation relations



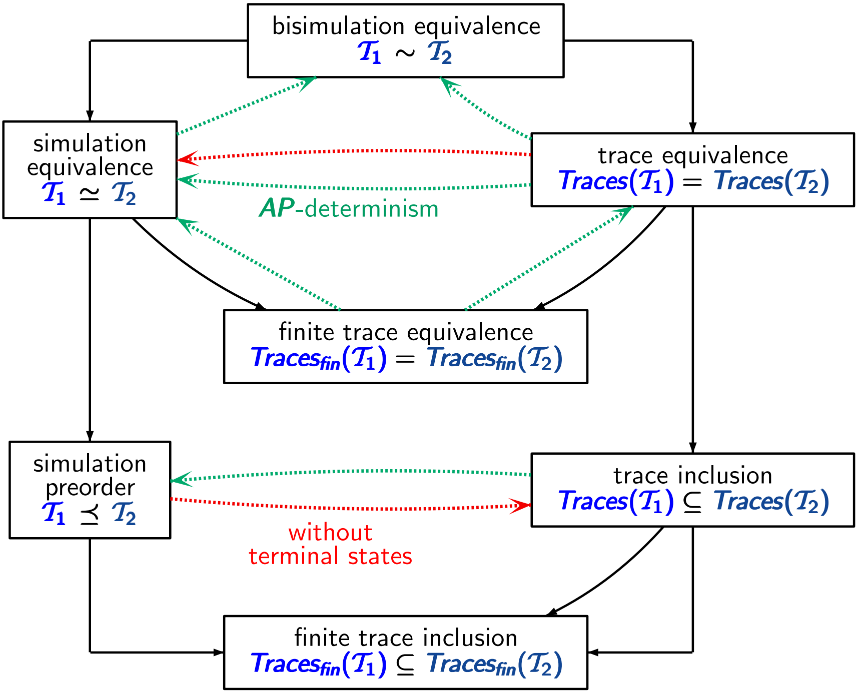
bisimulation equivalence
$\mathcal{T}_1 \sim \mathcal{T}_2$
simulation
equivalence
$\mathcal{T}_1 \simeq \mathcal{T}_2$
simulation
preorder
$\mathcal{T}_1 \preceq \mathcal{T}_2$


bisimulation equivalence
$\mathcal{T}_1 \sim \mathcal{T}_2$
simulation equivalence
$\mathcal{T}_1 \simeq \mathcal{T}_2$
finite trace equivalence
$Traces_{fin}(\mathcal{T}_1) = Traces_{fin}(\mathcal{T}_2)$
simulation preorder
$\mathcal{T}_1 \preceq \mathcal{T}_2$
finite trace inclusion
$Traces_{fin}(\mathcal{T}_1) \subseteq Traces_{fin}(\mathcal{T}_2)$


bisimulation equivalence
$\mathcal{T}_1 \sim \mathcal{T}_2$
simulation equivalence
$\mathcal{T}_1 \simeq \mathcal{T}_2$
trace equivalence
$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$
finite trace equivalence
$Traces_{fin}(\mathcal{T}_1) = Traces_{fin}(\mathcal{T}_2)$
simulation preorder
$\mathcal{T}_1 \preceq \mathcal{T}_2$
trace inclusion
$Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$
finite trace inclusion
$Traces_{fin}(\mathcal{T}_1) \subseteq Traces_{fin}(\mathcal{T}_2)$



bisimulation equivalence
$\mathcal{T}_1 \sim \mathcal{T}_2$
simulation equivalence
$\mathcal{T}_1 \simeq \mathcal{T}_2$
trace equivalence
$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$
finite trace equivalence
$Traces_{fin}(\mathcal{T}_1) = Traces_{fin}(\mathcal{T}_2)$
simulation preorder
$\mathcal{T}_1 \preceq \mathcal{T}_2$
trace inclusion
$Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$
without terminal states
finite trace inclusion
$Traces_{fin}(\mathcal{T}_1) \subseteq Traces_{fin}(\mathcal{T}_2)$


bisimulation equivalence
$\mathcal{T}_1 \sim \mathcal{T}_2$
simulation equivalence
$\mathcal{T}_1 \simeq \mathcal{T}_2$
trace equivalence
$Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$
AP-determinism
finite trace equivalence
$Traces_{fin}(\mathcal{T}_1) = Traces_{fin}(\mathcal{T}_2)$
simulation preorder
$\mathcal{T}_1 \preceq \mathcal{T}_2$
trace inclusion
$Traces(\mathcal{T}_1) \subseteq Traces(\mathcal{T}_2)$
without terminal states
finite trace inclusion
$Traces_{fin}(\mathcal{T}_1) \subseteq Traces_{fin}(\mathcal{T}_2)$

AP-determinism

AP-determinism

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS.

AP-determinism

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS.

$\mathcal{T}$ is called AP-deterministic iff

(1) for all states s and all subsets A of AP:

$$\big|\{\, t \in S : s \rightarrow t \wedge L(t) = A \,\}\big| \leqslant 1$$

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS.

$\mathcal{T}$ is called AP-deterministic iff

(1) for all states s and all subsets A of AP:

$$\big|\{\, t \in S : s \rightarrow t \wedge L(t) = A \,\}\big| \leqslant 1$$

(2) for all subsets A of AP:

$$\big|\{\, s_0 \in S_0 : L(s_0) = A \,\}\big| \leqslant 1$$

Trace relations in AP-deterministic TS

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) = Traces_{fin}(s_2)$ then

$$Traces(s_1) = Traces(s_2)$$

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) = Traces_{fin}(s_2)$ then $Traces(s_1) = Traces(s_2)$

mainly because:

Trace relations in AP-deterministic TS

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

> If $Traces_{fin}(s_1) = Traces_{fin}(s_2)$ then
> $$Traces(s_1) = Traces(s_2)$$

mainly because:

- each (finite or infinite) word σ_1 over 2^{AP} is induced by at most one path fragment starting in s_1 or s_2, respectively

Trace relations in AP-deterministic TS

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) = Traces_{fin}(s_2)$ then

$$Traces(s_1) = Traces(s_2)$$

mainly because:

- each (finite or infinite) word σ_1 over 2^{AP} is induced by at most one path fragment starting in s_1 or s_2, respectively
- if $\sigma = A_0 A_1 \ldots A_i A_{i+1} \ldots \in Traces(s_1)$ then there is no proper prefix $A_0 A_1 \ldots A_i$ of σ belongs to $Traces(s_1)$

  + analogous statement for s_2

Correct or wrong?

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$ then

$$Traces(s_1) \subseteq Traces(s_2)$$

Correct or wrong?

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$ then

$$Traces(s_1) \subseteq Traces(s_2)$$

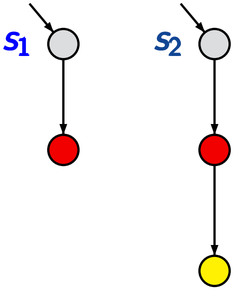
wrong.

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$ then

$$Traces(s_1) \subseteq Traces(s_2)$$

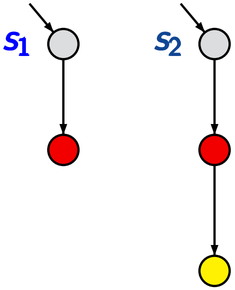
wrong.



Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

> If $Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$ then
>
> $Traces(s_1) \subseteq Traces(s_2)$

wrong.

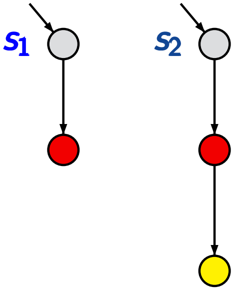


$Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.

If $Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$ then $Traces(s_1) \subseteq Traces(s_2)$

wrong.



$Traces_{fin}(s_1) \subseteq Traces_{fin}(s_2)$

 $\in Traces(s_1) \setminus Traces(s_2)$

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be *AP*-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1) $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1) $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)

(2) $s_1 \simeq_{\mathcal{T}} s_2$ (simulation equivalence)

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1) $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)

(2) $s_1 \simeq_{\mathcal{T}} s_2$ (simulation equivalence)

(3) $Traces_{fin}(s_1) = Traces_{fin}(s_2)$

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1) $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)

(2) $s_1 \simeq_{\mathcal{T}} s_2$ (simulation equivalence)

(3) $Traces_{fin}(s_1) = Traces_{fin}(s_2)$

(4) $Traces(s_1) = Traces(s_2)$

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1)	$s_1 \sim_{\mathcal{T}} s_2$	(bisimulation equivalence)
(2)	$s_1 \simeq_{\mathcal{T}} s_2$	(simulation equivalence)
(3)	$Traces_{fin}(s_1) = Traces_{fin}(s_2)$	
(4)	$Traces(s_1) = Traces(s_2)$	

$(1) \Longrightarrow (2)$: $\surd$

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1)	$s_1 \sim_{\mathcal{T}} s_2$	(bisimulation equivalence)
(2)	$s_1 \simeq_{\mathcal{T}} s_2$	(simulation equivalence)
(3)	$Traces_{fin}(s_1) = Traces_{fin}(s_2)$	
(4)	$Traces(s_1) = Traces(s_2)$	

(1) $\Longrightarrow$ (2): $\checkmark$

(2) $\Longrightarrow$ (3): ... path fragment lifting ...

(Bi)simulation and trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1) $s_1 \sim_{\mathcal{T}} s_2$ (bisimulation equivalence)

(2) $s_1 \simeq_{\mathcal{T}} s_2$ (simulation equivalence)

(3) $Traces_{fin}(s_1) = Traces_{fin}(s_2)$

(4) $Traces(s_1) = Traces(s_2)$

$(1) \Longrightarrow (2)$: $\surd$

$(2) \Longrightarrow (3)$: ... path fragment lifting ...

$(3) \Longrightarrow (4)$: just shown

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$.
Then the following statements are equivalent:

(1)	$s_1 \sim_{\mathcal{T}} s_2$	(bisimulation equivalence)
(2)	$s_1 \simeq_{\mathcal{T}} s_2$	(simulation equivalence)
(3)	$Traces_{fin}(s_1) = Traces_{fin}(s_2)$	
(4)	$Traces(s_1) = Traces(s_2)$	

$(1) \Longrightarrow (2)$: $\checkmark$

$(2) \Longrightarrow (3)$: ... path fragment lifting ...

$(3) \Longrightarrow (4)$: just shown

$(4) \Longrightarrow (1)$: ...

Bisimulation and trace equivalence

Let $\mathcal{T}$ be ***AP***-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces(s_1) = Traces(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Let $\mathcal{T}$ be ***AP***-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces(s_1) = Traces(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Proof: show that

$$\mathcal{R} = \{(s_1, s_2) : Traces(s_1) = Traces(s_2)\}$$

is a bisimulation.

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces(s_1) = Traces(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Proof: show that

$$\mathcal{R} = \{(s_1, s_2) : Traces(s_1) = Traces(s_2)\}$$

is a bisimulation.

Note that if $s \rightarrow t$ then

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces(s_1) = Traces(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Proof: show that

$$\mathcal{R} = \{(s_1, s_2) : Traces(s_1) = Traces(s_2)\}$$

is a bisimulation.

Note that if $s \to t$ then

$$Traces(t) = \{L(t)B_1B_2B_3\ldots \in (2^{AP})^+ \cup (2^{AP})^\omega : L(s)L(t)B_1B_2B_3\ldots \in Traces(s)\}$$

Bisimulation & finite trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces_{fin}(s_1) = Traces_{fin}(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Bisimulation & finite trace equivalence

Let $\mathcal{T}$ be AP-deterministic and s_1, s_2 states in $\mathcal{T}$. Then:

$$Traces_{fin}(s_1) = Traces_{fin}(s_2) \text{ implies } s_1 \sim_{\mathcal{T}} s_2$$

Proof: show that

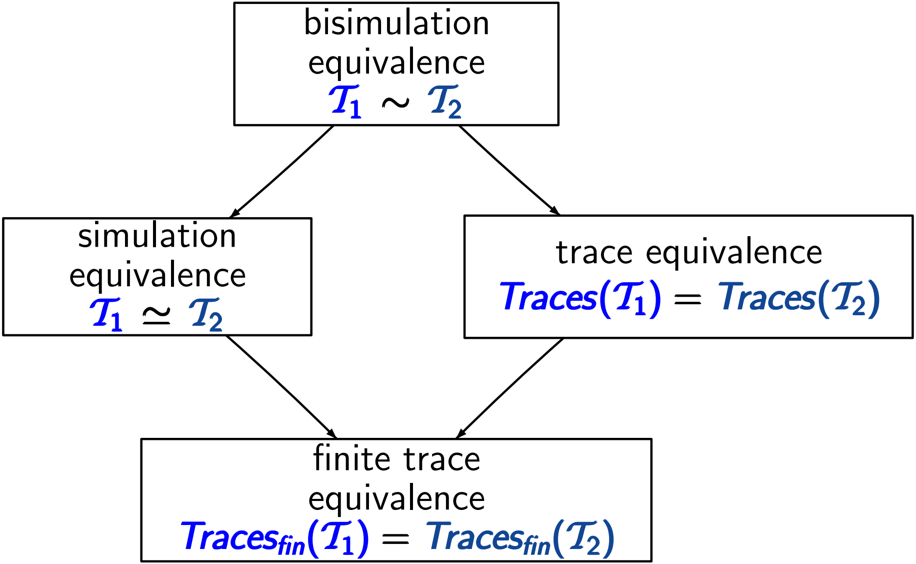
$$\mathcal{R} = \{(s_1, s_2) : Traces_{fin}(s_1) = Traces_{fin}(s_2)\}$$

is a bisimulation.

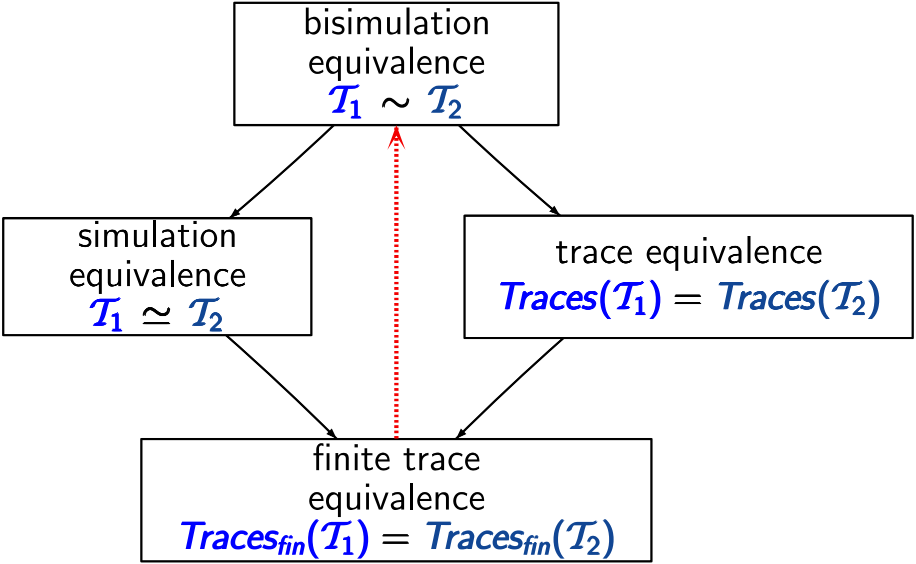
Note that if $s \rightarrow t$ then

$$Traces_{fin}(t) = \{L(t)B_1B_2 \ldots B_n \in (2^{AP})^+ : L(s)L(t)B_1B_2 \ldots B_n \in Traces_{fin}(s)\}$$

Trace and (bi)simulation equivalence



For AP-deterministic TS



For AP-deterministic TS