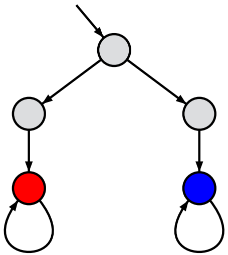
Overview

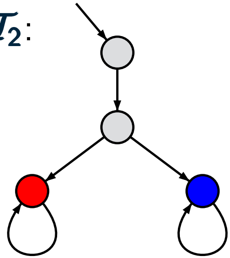
- Introduction
- Modelling parallel systems
- Linear Time Properties
- Regular Properties
- Linear Temporal Logic (LTL)
- Computation-Tree Logic
- **Equivalences and Abstraction**

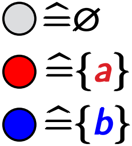
Trace equivalence

$\mathcal{T}_1$:



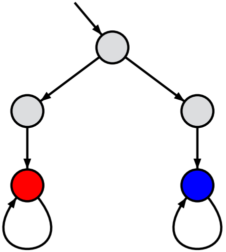
$\mathcal{T}_2$:



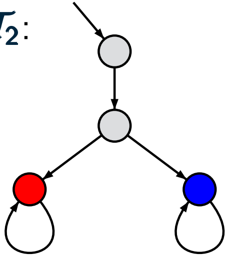


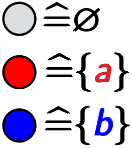
Trace equivalence

$\mathcal{T}_1$:



$\mathcal{T}_2$:

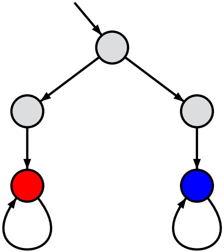




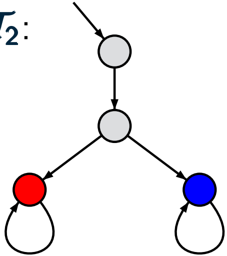
$$Traces(\mathcal{T}_1) = \{ \varnothing \varnothing a^\omega, \varnothing \varnothing b^\omega \} = Traces(\mathcal{T}_1)$$

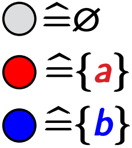
Trace equivalence

$\mathcal{T}_1$:



$\mathcal{T}_2$:

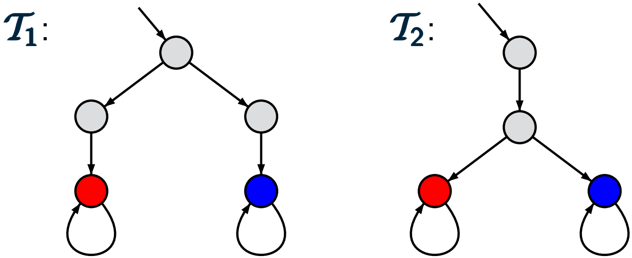


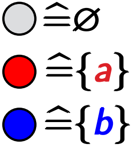


$$Traces(\mathcal{T}_1) = \{\ \varnothing\,\varnothing\, a^\omega,\ \varnothing\,\varnothing\, b^\omega\ \} = Traces(\mathcal{T}_1)$$

CTL-formula $\Phi = \exists\bigcirc(\exists\bigcirc a \wedge \exists\bigcirc b)$

Trace equivalence



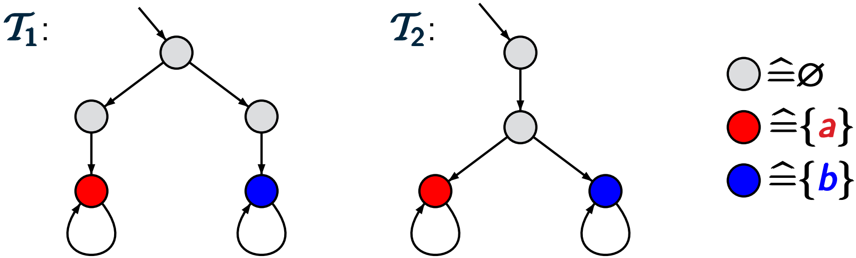


$$Traces(\mathcal{T}_1) = \{ \varnothing \varnothing a^\omega, \ \varnothing \varnothing b^\omega \} = Traces(\mathcal{T}_1)$$

CTL-formula $\Phi = \exists \bigcirc (\exists \bigcirc a \wedge \exists \bigcirc b)$

$\mathcal{T}_1 \not\models \Phi$ and $\mathcal{T}_2 \models \Phi$

Trace equivalence is not compatible with CTL



$$Traces(\mathcal{T}_1) = \{ \varnothing \varnothing a^\omega, \varnothing \varnothing b^\omega \} = Traces(\mathcal{T}_1)$$

CTL-formula $\Phi = \exists\bigcirc(\exists\bigcirc a \wedge \exists\bigcirc b)$

$\mathcal{T}_1 \not\models \Phi$ and $\mathcal{T}_2 \models \Phi$

Implementation relations

- for the design of complex systems
  - $\rightsquigarrow$ comparison of **2** transition systems

Implementation relations

- for the design of complex systems
  - $\rightsquigarrow$ comparison of **2** transition systems
- for the analysis of complex systems

Implementation relations

- for the design of complex systems
  - $\leadsto$ comparison of $\mathbf{2}$ transition systems
- for the analysis of complex systems
  - $\leadsto$ homogeneous model checking approach

Implementation relations

- for the design of complex systems
  - $\rightsquigarrow$ comparison of 2 transition systems
- for the analysis of complex systems
  - $\rightsquigarrow$ homogeneous model checking approach
  - $\rightsquigarrow$ graph minimization

Implementation relations

- for the design of complex systems
  - $\leadsto$ comparison of **2** transition systems
- for the analysis of complex systems
  - $\leadsto$ homogeneous model checking approach
  - $\leadsto$ graph minimization

> use equivalence relation $\sim$ for the states of a single transition system $\mathcal{T}$ and analyze the quotient $\mathcal{T}/\sim$

Implementation relations

- for the design of complex systems
  - $\leadsto$ comparison of **2** transition systems
- for the analysis of complex systems
  - $\leadsto$ homogeneous model checking approach
  - $\leadsto$ graph minimization

> use equivalence relation $\sim$ for the states of a single transition system $\mathcal{T}$ and analyze the quotient $\mathcal{T}/\sim$

goal: define the equivalence $\sim$ in such a way that

$$\mathcal{T} \models \Phi \quad \text{iff} \quad \mathcal{T}/\sim\, \models \Phi$$

for all "relevant" properties Φ

Linear-time implementation relations

Linear-time implementation relations

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

trace inclusion and trace equivalence:

e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

Linear-time implementation relations

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

preserves all linear-time **safety** properties

trace inclusion and trace equivalence:

e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

Linear-time implementation relations

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

preserves all linear-time **safety** properties

trace inclusion and trace equivalence:

e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

preserves all **LTL** properties

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

preserves all linear-time **safety** properties

trace inclusion and trace equivalence:

e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

preserves all **LTL** properties

* none of the LT relations is compatible with **CTL**

Linear-time implementation relations

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

preserves all linear-time **safety** properties

trace inclusion and trace equivalence:

e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

preserves all **LTL** properties

* none of the LT relations is compatible with **CTL**
* checking LT relations is computationally hard

finite trace inclusion and equivalence:

e.g., $\mathit{Tracesfin}(\mathcal{T}_1) \subseteq \mathit{Tracesfin}(\mathcal{T}_2)$

preserves all linear-time **safety** properties

trace inclusion and trace equivalence:

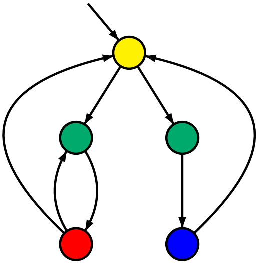
e.g., $\mathit{Traces}(\mathcal{T}_1) \subseteq \mathit{Traces}(\mathcal{T}_2)$

preserves all **LTL** properties

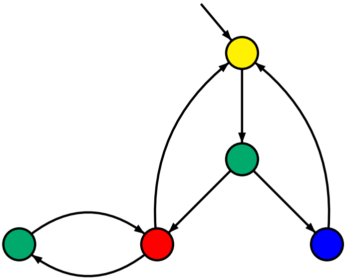
* none of the LT relations is compatible with **CTL**
* checking LT relations is computationally hard
* **minimization ???**

Minimization w.r.t. trace equivalence?

$\mathcal{T}_1$:

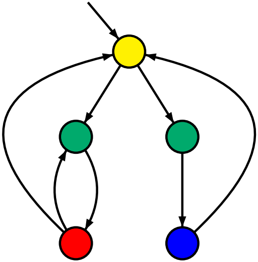


$\mathcal{T}_2$:

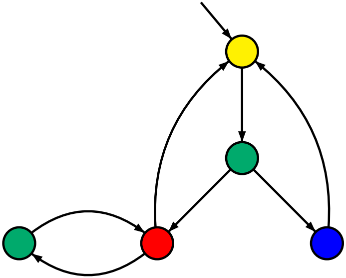


Minimization w.r.t. trace equivalence?

$\mathcal{T}_1$:



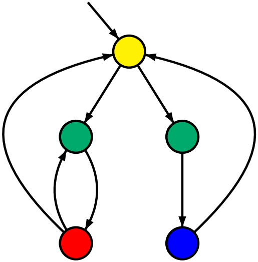
$\mathcal{T}_2$:



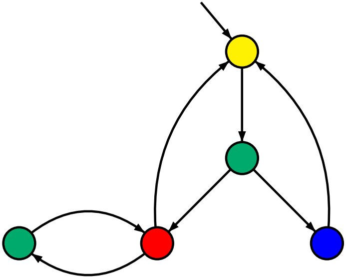
- $Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$

Minimization w.r.t. trace equivalence?

$\mathcal{T}_1$:



$\mathcal{T}_2$:

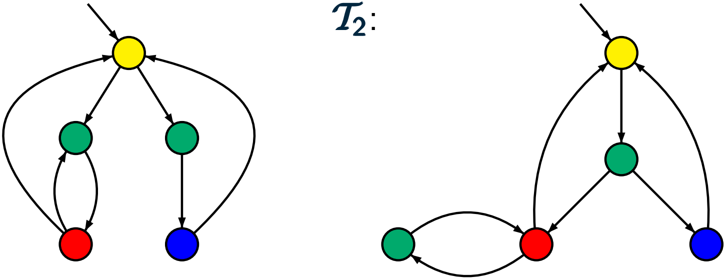


- $\mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$

  but $\mathcal{T}_1$ and $\mathcal{T}_2$ are not isomorphic

Minimization w.r.t. trace equivalence?

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



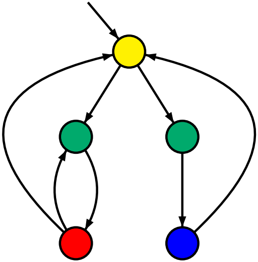
- $\mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$

  but $\mathcal{T}_1$ and $\mathcal{T}_2$ are not isomorphic

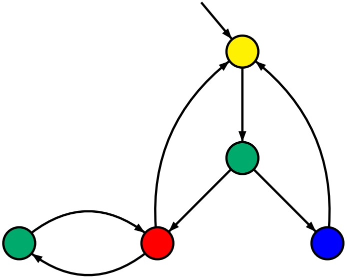
- $\mathcal{T}_1$, $\mathcal{T}_2$ have **5** states and **7** transitions each

Minimization w.r.t. trace equivalence?

$\mathcal{T}_1$:



$\mathcal{T}_2$:



- $Traces(\mathcal{T}_1) = Traces(\mathcal{T}_2)$

  but $\mathcal{T}_1$ and $\mathcal{T}_2$ are not isomorphic
- $\mathcal{T}_1$, $\mathcal{T}_2$ have **5** states and **7** transitions each
- there is no smaller TS that is trace-equivalent to $\mathcal{T}_i$

Classification of implementation relations

Classification of implementation relations

- linear vs. branching time
  - * linear time: trace relations
  - * branching time: (bi)simulation 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

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

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

Bisimulation for two transition systems

Bisimulation for two transition systems

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

Bisimulation for two transition systems

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

- with the same set AP

Bisimulation for two transition systems

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

- with the same set ***AP***
- possibly containing terminal states

Bisimulation for two transition systems

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

- with the same set AP
- possibly containing terminal states

Bisimulation equivalence of $\mathcal{T}_1$ and $\mathcal{T}_2$ requires that $\mathcal{T}_1$ and $\mathcal{T}_2$ can simulate each other in a stepwise manner.

Bisimulation for two transition systems

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

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

be two transition systems

- with the same set AP ⟵ observables
- possibly containing terminal states

Bisimulation equivalence of $\mathcal{T}_1$ and $\mathcal{T}_2$ requires that $\mathcal{T}_1$ and $\mathcal{T}_2$ can simulate each other in a stepwise manner.

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

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

binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

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

binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

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

binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

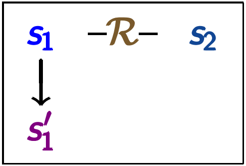
(2) $\forall s_1' \in Post(s_1)\, \exists s_2' \in Post(s_2)$ s.t. $(s_1', s_2') \in \mathcal{R}$

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

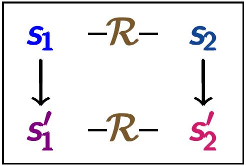
binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

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



can be completed to

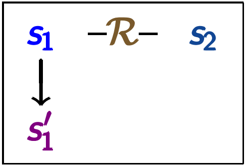


Bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$

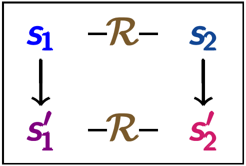
binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

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



can be completed to



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

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

binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

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

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

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

and such that the following initial condition holds:

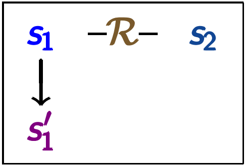
(I) $\forall s_{0,1} \in S_{0,1} \, \exists s_{0,2} \in S_{0,2}$ s.t. $(s_{0,1}, s_{0,2}) \in \mathcal{R}$

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

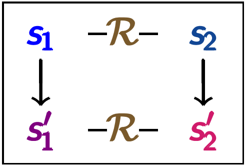
binary relation $\mathcal{R} \subseteq S_1 \times S_2$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(1) $L_1(s_1) = L_2(s_2)$

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



can be completed to



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

and such that the following initial condition holds:

(I) $\forall s_{0,1} \in S_{0,1}\, \exists s_{0,2} \in S_{0,2}$ s.t. $(s_{0,1}, s_{0,2}) \in \mathcal{R}$

$\forall s_{0,2} \in S_{0,2}\, \exists s_{0,1} \in S_{0,1}$ s.t. $(s_{0,1}, s_{0,2}) \in \mathcal{R}$

Bisimulation equivalence $\sim$

Bisimulation equivalence $\sim$

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

for all $(s_1, s_2) \in \mathcal{R}$: (1) labeling condition

(2) } mutual stepwise
(3) } simulation

and initial condition (I)

Bisimulation equivalence $\sim$

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

for all $(s_1, s_2) \in \mathcal{R}$: (1) labeling condition

(2) } mutual stepwise simulation
(3)

and initial condition (I)

bisimulation equivalence $\sim$ for TS:

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

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

(1) labeling condition

(2), (3) } mutual stepwise simulation

and initial condition (I)

> bisimulation equivalence $\sim$ for TS:
>
> $\mathcal{T}_1 \sim \mathcal{T}_2$ iff there is a bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$

Bisimulation equivalence $\sim$

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

for all $(s_1, s_2) \in \mathcal{R}$: (1) labeling condition

(2) } mutual stepwise simulation
(3)

and initial condition (I)

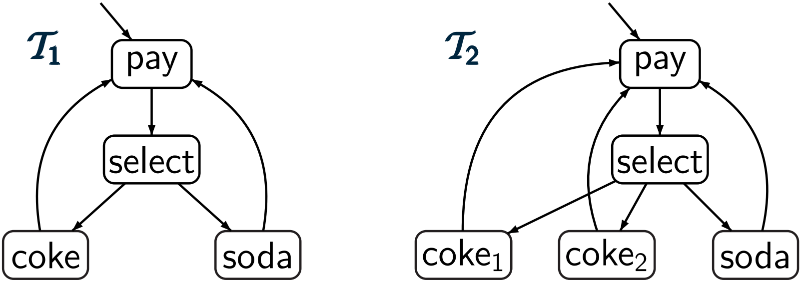
bisimulation equivalence $\sim$ for TS:

$\mathcal{T}_1 \sim \mathcal{T}_2$ iff there is a bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$

for state s_1 of $\mathcal{T}_1$ and state s_2 of $\mathcal{T}_2$:

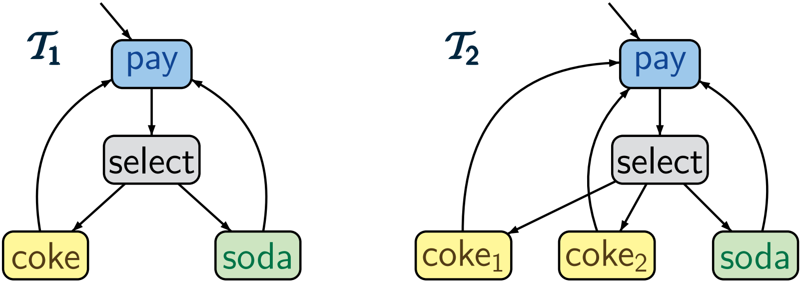
$s_1 \sim s_2$ iff there exists a bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$ such that $(s_1, s_2) \in \mathcal{R}$

Two beverage machines



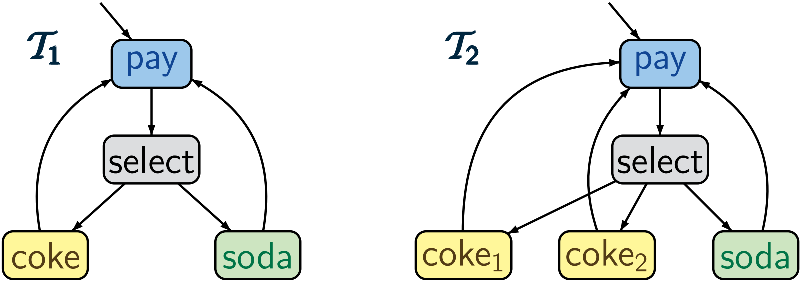
$$AP = \{pay, coke, soda\}$$

Two beverage machines



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

Two beverage machines

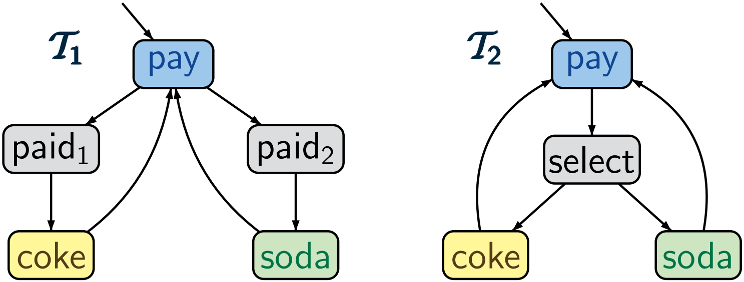


$AP = \{\mathit{pay}, \mathit{coke}, \mathit{soda}\}$

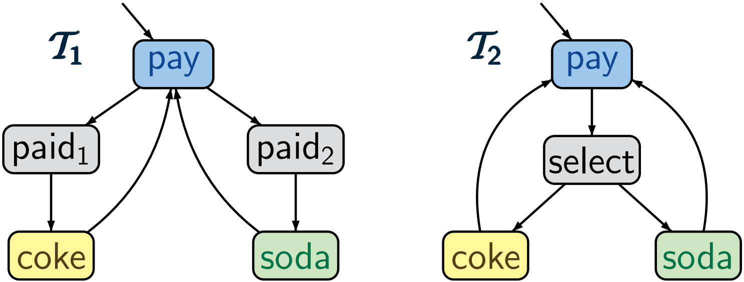
$\mathcal{T}_1 \sim \mathcal{T}_2$ as there is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$:

$\{$ (pay,pay), (select,select), (soda,soda)
(coke,$coke_1$), (coke,$coke_2$) $\}$

Two beverage machines



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



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

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

because there is no state in $\mathcal{T}_1$ that has both

- a successor labeled with coke and
- a successor labeled with soda

Simulation condition of bisimulations

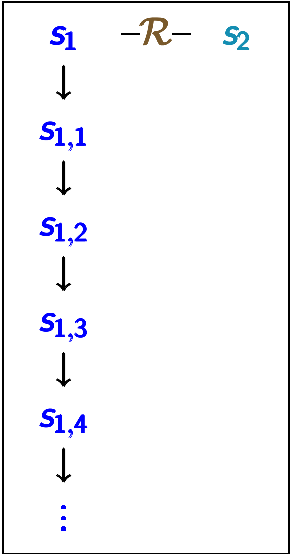
BSEQOR5.1-9-BIS

$$\begin{array}{ccc} s_1 & -\mathcal{R}- & s_2 \\ \downarrow & & \\ s_1' & & \end{array}$$

can be completed to

$$\begin{array}{ccc} s_1 & -\mathcal{R}- & s_2 \\ \downarrow & & \downarrow \\ s_1' & -\mathcal{R}- & s_2' \end{array}$$

Path lifting for bisimulation $\mathcal{R}$

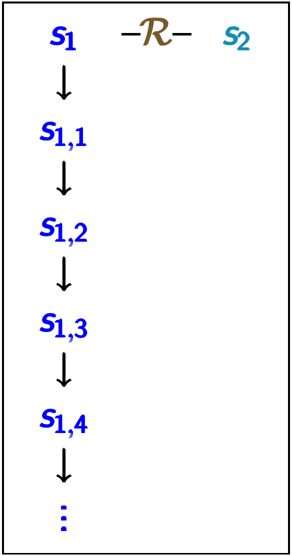


Path lifting for bisimulation $\mathcal{R}$

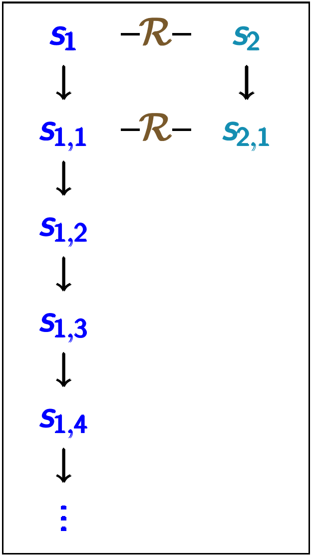
$$\begin{array}{ccc} s_1 & -\mathcal{R}- & s_2 \\ \downarrow & & \\ s_{1,1} & & \\ \downarrow & & \\ s_{1,2} & & \\ \downarrow & & \\ s_{1,3} & & \\ \downarrow & & \\ s_{1,4} & & \\ \downarrow & & \\ \vdots & & \end{array}$$

can be
completed to

Path lifting for bisimulation $\mathcal{R}$



can be completed to



Path lifting for bisimulation $\mathcal{R}$

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
s_{1,4} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

can be completed to

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & -\mathcal{R}- & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & -\mathcal{R}- & s_{2,2} \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
s_{1,4} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

Path lifting for bisimulation $\mathcal{R}$

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
s_{1,4} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

can be completed to

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & -\mathcal{R}- & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & -\mathcal{R}- & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & -\mathcal{R}- & s_{2,3} \\
\downarrow & & \\
s_{1,4} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

Path lifting for bisimulation $\mathcal{R}$

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \\
s_{1,1} & & \\
\downarrow & & \\
s_{1,2} & & \\
\downarrow & & \\
s_{1,3} & & \\
\downarrow & & \\
s_{1,4} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

can be completed to

$$
\begin{array}{ccc}
s_1 & -\mathcal{R}- & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & -\mathcal{R}- & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & -\mathcal{R}- & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & -\mathcal{R}- & s_{2,3} \\
\downarrow & & \downarrow \\
s_{1,4} & -\mathcal{R}- & s_{2,4} \\
\downarrow & & \downarrow \\
\vdots & & \vdots
\end{array}
$$

Properties of bisimulation equivalence

Properties of bisimulation equivalence

$\sim$ is an equivalence

Properties of bisimulation equivalence

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$

Properties of bisimulation equivalence

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$

↑

If S is the state space of $\mathcal{T}$ then

$$\mathcal{R} = \{(s, s) : s \in S\}$$

is a bisimulation for $(\mathcal{T}, \mathcal{T})$

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$
- symmetry: $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_2 \sim \mathcal{T}_1$

Properties of bisimulation equivalence

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$
- symmetry: $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_2 \sim \mathcal{T}_1$

$\uparrow$

If $\mathcal{R}$ is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$ then

$$\mathcal{R}^{-1} = \{(s_2, s_1) : (s_1, s_2) \in \mathcal{R}\}$$

is a bisimulation for $(\mathcal{T}_2, \mathcal{T}_1)$

Properties of bisimulation equivalence

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$
- symmetry: $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_2 \sim \mathcal{T}_1$
- transitivity: if $\mathcal{T}_1 \sim \mathcal{T}_2$ and $\mathcal{T}_2 \sim \mathcal{T}_3$ then $\mathcal{T}_1 \sim \mathcal{T}_3$

$\sim$ is an equivalence, i.e.,

- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$
- symmetry: $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_2 \sim \mathcal{T}_1$
- transitivity: if $\mathcal{T}_1 \sim \mathcal{T}_2$ and $\mathcal{T}_2 \sim \mathcal{T}_3$ then $\mathcal{T}_1 \sim \mathcal{T}_3$

↑

Let $\mathcal{R}_{1,2}$ be a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$,
$\mathcal{R}_{2,3}$ be a bisimulation for $(\mathcal{T}_2, \mathcal{T}_3)$.

Properties of bisimulation equivalence

$\sim$ is an equivalence, i.e.,

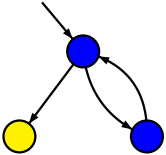
- reflexivity: $\mathcal{T} \sim \mathcal{T}$ for all transition systems $\mathcal{T}$
- symmetry: $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_2 \sim \mathcal{T}_1$
- transitivity: if $\mathcal{T}_1 \sim \mathcal{T}_2$ and $\mathcal{T}_2 \sim \mathcal{T}_3$ then $\mathcal{T}_1 \sim \mathcal{T}_3$

Let $\mathcal{R}_{1,2}$ be a bisimulation for $(\mathcal{T}_1, \mathcal{T}_2)$,
$\mathcal{R}_{2,3}$ be a bisimulation for $(\mathcal{T}_2, \mathcal{T}_3)$.

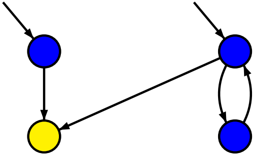
$$\mathcal{R} \stackrel{\text{def}}{=} \{ (s_1, s_3) : \exists s_2 \text{ s.t. } (s_1, s_2) \in \mathcal{R}_{1,2} \text{ and } (s_2, s_3) \in \mathcal{R}_{2,3} \}$$

is a bisimulation for $(\mathcal{T}_1, \mathcal{T}_3)$

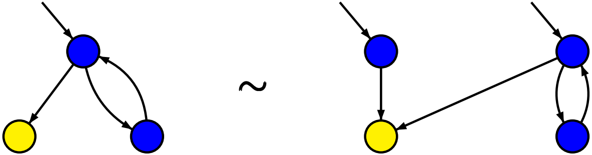
Correct or wrong?



$\sim$

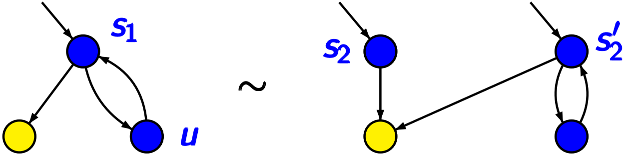


Correct or wrong?



wrong

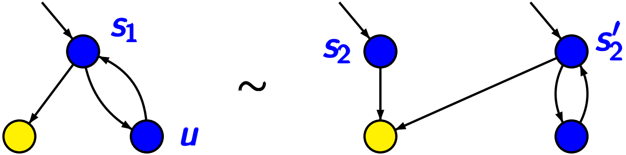
Correct or wrong?



wrong

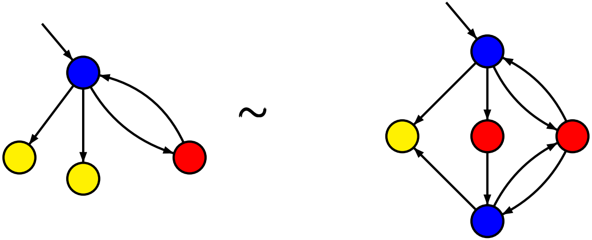
$s_1 \longrightarrow u$, but $s_2 \not\longrightarrow$ *blue* (thus $s_1 \not\sim s_2$)

Correct or wrong?

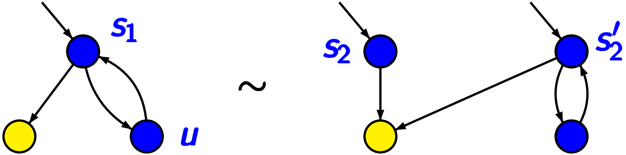


wrong

$s_1 \longrightarrow u$, but $s_2 \not\longrightarrow$ ***blue*** (thus $s_1 \not\sim s_2$)

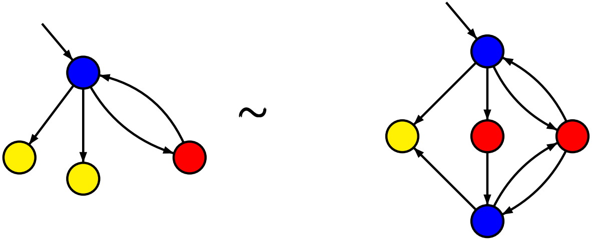


Correct or wrong?



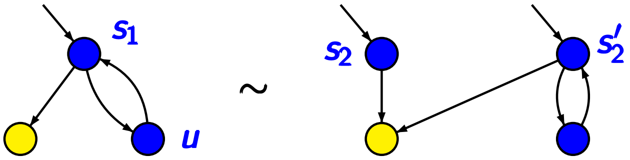
wrong

$s_1 \longrightarrow u$, but $s_2 \not\longrightarrow \textit{blue}$ (thus $s_1 \not\sim s_2$)



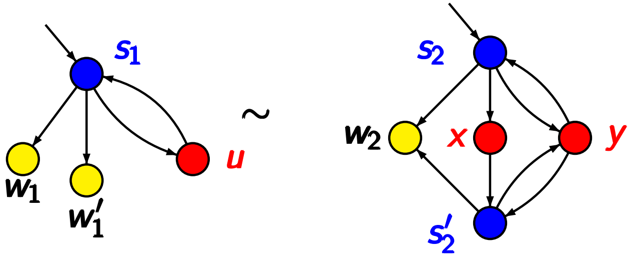
correct

Correct or wrong?



wrong

$s_1 \longrightarrow u$, but $s_2 \not\longrightarrow blue$ (thus $s_1 \not\sim s_2$)

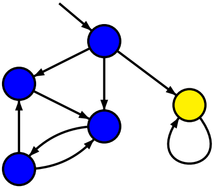


correct

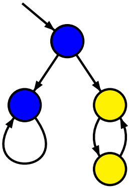
bisimulation:

$$\{(w_1, w_2), (w_1', w_2), (s_1, s_2), (s_1, s_2'), (u, x), (u, y)\}$$

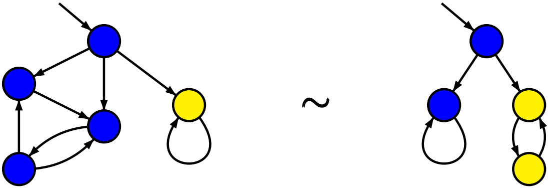
Correct or wrong?



$\sim$

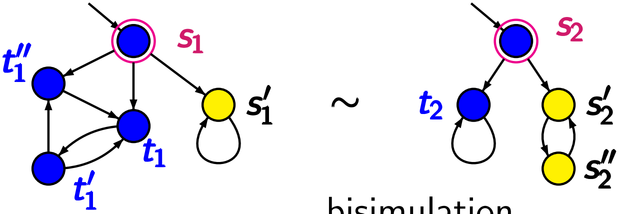


Correct or wrong?



correct

Correct or wrong?

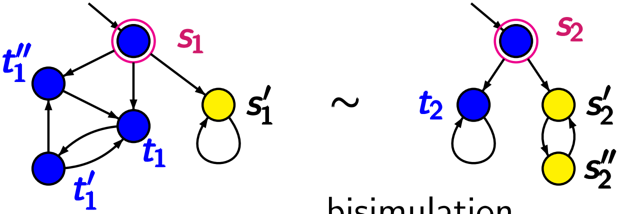


correct

bisimulation

$$\{(s_1, s_2), (s_1', s_2'), (s_1', s_2''), (t_1, t_2), (t_1', t_2), (t_1'', t_2)\}$$

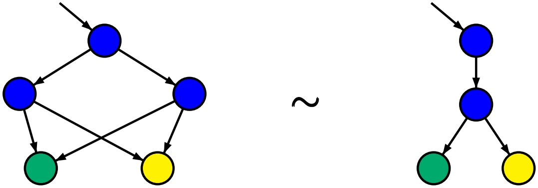
Correct or wrong?



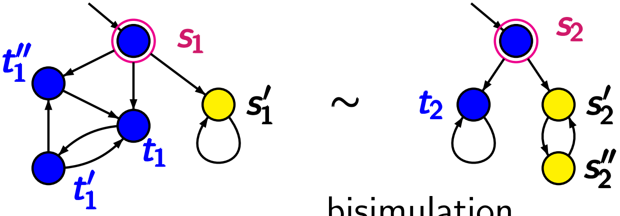
correct

bisimulation

$$\{(s_1, s_2), (s_1', s_2'), (s_1', s_2''), (t_1, t_2), (t_1', t_2), (t_1'', t_2)\}$$



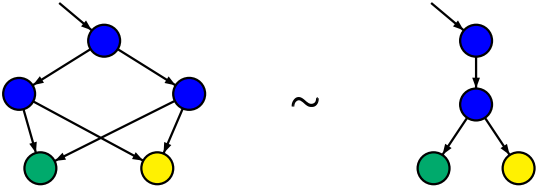
Correct or wrong?



correct

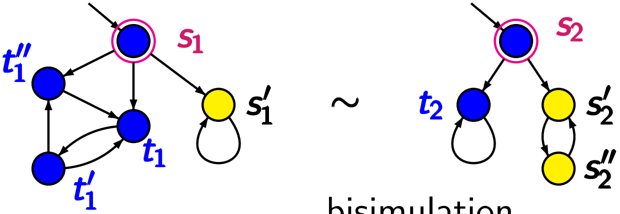
bisimulation

$$\{(s_1, s_2), (s_1', s_2'), (s_1', s_2''), (t_1, t_2), (t_1', t_2), (t_1'', t_2)\}$$



correct

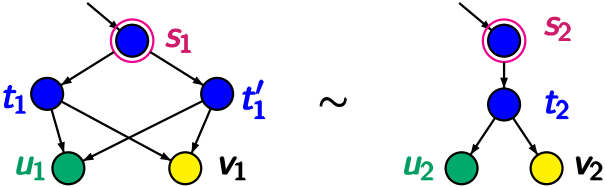
Correct or wrong?



correct

bisimulation

$$\{(s_1, s_2), (s_1', s_2'), (s_1', s_2''), (t_1, t_2), (t_1', t_2), (t_1'', t_2)\}$$



correct

bisimulation: $\{(s_1, s_2), (t_1, t_2), (t_1', t_2), (u_1, u_2), (v_1, v_2)\}$

Bisimulation vs. trace equivalence

Bisimulation vs. trace equivalence

$$\mathcal{T}_1 \sim \mathcal{T}_2 \implies \mathit{Traces}(\mathcal{T}_1) = \mathit{Traces}(\mathcal{T}_2)$$

Bisimulation vs. trace equivalence

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

proof: ... path fragment lifting ...

Bisimulation vs. trace equivalence

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

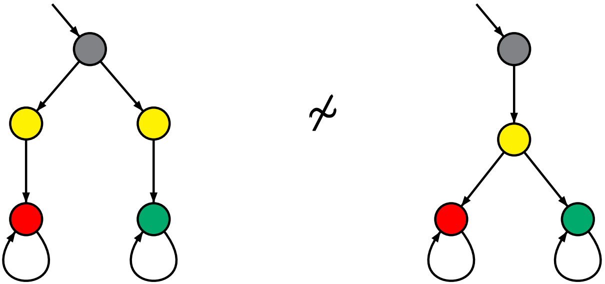
proof: ... path fragment lifting ...

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

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

proof: ... path fragment lifting ...

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



trace equivalent, but not bisimulation equivalent

Bisimulation vs. trace equivalence

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

proof: ... path fragment lifting ...

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

Trace equivalence is strictly coarser than bisimulation equivalence.

Bisimulation vs. trace equivalence

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

proof: ... path fragment lifting ...

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

Trace equivalence is strictly coarser than bisimulation equivalence.

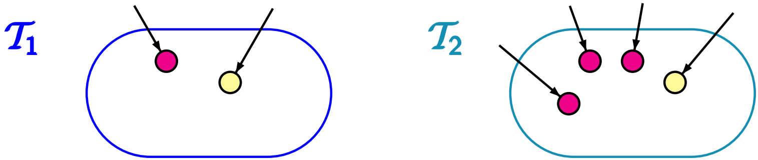
Bisimulation equivalent transition systems satisfy the same **LT properties** (e.g., **LTL** formulas).

Bisimulation equivalence ...

- as a relation that compares 2 transition systems

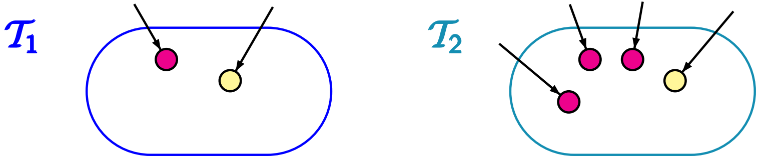
Bisimulation equivalence ...

- as a relation that compares **2** transition systems



Bisimulation equivalence ...

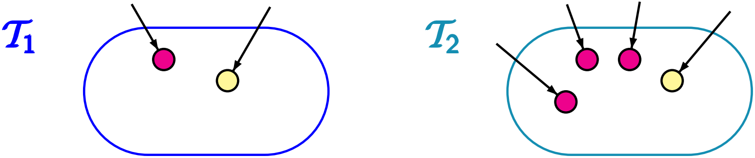
- as a relation that compares **2** transition systems



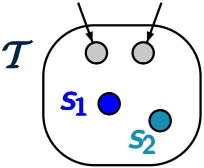
- as a relation on the states of **1** transition system

Bisimulation equivalence ...

- as a relation that compares **2** transition systems

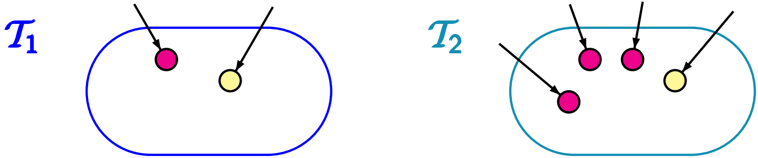


- as a relation on the states of **1** transition system

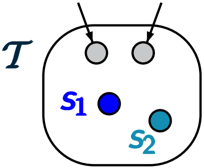


Bisimulation equivalence ...

- as a relation that compares **2** transition systems



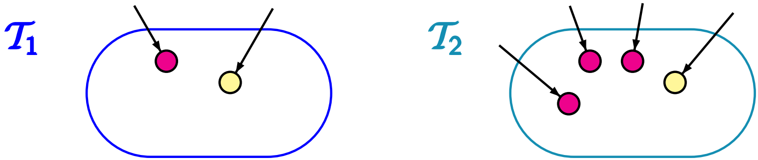
- as a relation on the states of **1** transition system



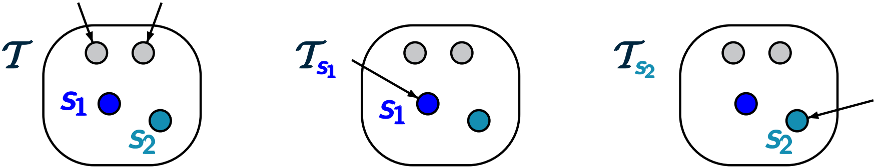
$s_1 \sim s_2$ iff $\mathcal{T}_{s_1} \sim \mathcal{T}_{s_2}$

Bisimulation equivalence ...

- as a relation that compares **2** transition systems



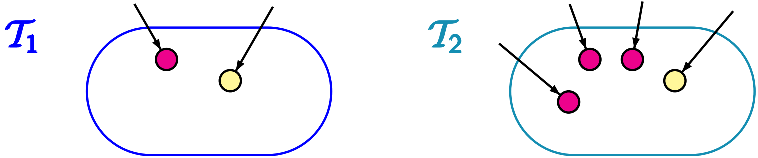
- as a relation on the states of **1** transition system



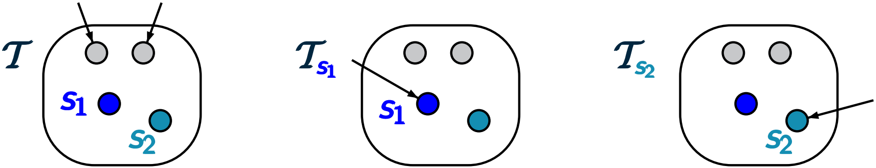
$s_1 \sim s_2$ iff $\mathcal{T}_{s_1} \sim \mathcal{T}_{s_2}$

Bisimulation equivalence ...

- as a relation that compares **2** transition systems



- as a relation on the states of **1** transition system



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

Bisimulations on a single TS

Bisimulations on a single TS

Let $\mathcal{T}$ be a TS with proposition set AP.

Let $\mathcal{T}$ be a TS with proposition set AP.

A bisimulation for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on the state space of $\mathcal{T}$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

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

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

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

Bisimulation equivalence $\sim_{\mathcal{T}}$ on a single TS

Let $\mathcal{T}$ be a TS with proposition set AP.

A bisimulation for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on the state space of $\mathcal{T}$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

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

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

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

bisimulation equivalence $\sim_{\mathcal{T}}$:

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

Bisimulation equivalence

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

Bisimulation equivalence $\sim_{\mathcal{T}}$ is

- the coarsest bisimulation on $\mathcal{T}$
- and an equivalence on S

Bisimulation equivalence

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

Bisimulation equivalence $\sim_{\mathcal{T}}$ is the coarsest equivalence on S s.t. for all states $s_1, s_2 \in S$ with $s_1 \sim_{\mathcal{T}} s_2$:

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

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

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

Two variants of bisimulation equivalence

$\sim$ relation that compares **2** transition systems

$\sim_{\mathcal{T}}$ equivalence on the state space of a single TS $\mathcal{T}$

Two variants of bisimulation equivalence

$\sim$ relation that compares **2** transition systems

$\sim_{\mathcal{T}}$ equivalence on the state space of a single TS $\mathcal{T}$

1. $\sim_{\mathcal{T}}$ can be derived from $\sim$

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

$$s_1 \sim_{\mathcal{T}} s_2 \quad \text{iff} \quad \mathcal{T}_{s_1} \sim \mathcal{T}_{s_1}$$

where $\mathcal{T}_s$ agrees with $\mathcal{T}$, except that state s is declared to be the unique initial state

Two variants of bisimulation equivalence

$\sim$ relation that compares **2** transition systems

$\sim_{\mathcal{T}}$ equivalence on the state space of a single TS $\mathcal{T}$

1. $\sim_{\mathcal{T}}$ can be derived from $\sim$

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

$$s_1 \sim_{\mathcal{T}} s_2 \quad \text{iff} \quad \mathcal{T}_{s_1} \sim \mathcal{T}_{s_1}$$

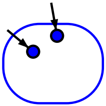
where $\mathcal{T}_s$ agrees with $\mathcal{T}$, except that state s is declared to be the unique initial state

2. $\sim$ can be derived from $\sim_{\mathcal{T}}$

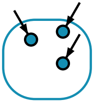
Derivation of $\sim$ from $\sim_{\mathcal{T}}$

given two transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$

$\mathcal{T}_1$ with state space S_1



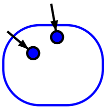
$\mathcal{T}_2$ with state space S_2



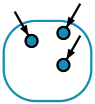
Derivation of $\sim$ from $\sim_{\mathcal{T}}$

given two transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$

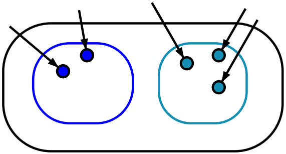
$\mathcal{T}_1$ with state space S_1



$\mathcal{T}_2$ with state space S_2



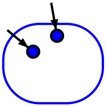
consider $\mathcal{T} = \mathcal{T}_1 \uplus \mathcal{T}_2$
(state space $S_1 \uplus S_2$)



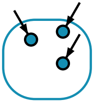
Derivation of $\sim$ from $\sim_{\mathcal{T}}$

given two transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$

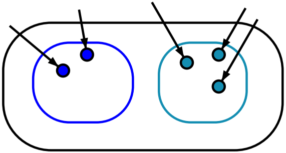
$\mathcal{T}_1$ with state space S_1



$\mathcal{T}_2$ with state space S_2



consider $\mathcal{T} = \mathcal{T}_1 \uplus \mathcal{T}_2$
(state space $S_1 \uplus S_2$)

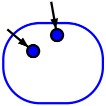


$\mathcal{T}_1 \sim \mathcal{T}_2$ iff $\forall$ initial states s_1 of $\mathcal{T}_1$
$\exists$ initial state s_2 of $\mathcal{T}_2$ s.t. $s_1 \sim_{\mathcal{T}} s_2$,

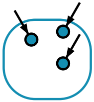
Derivation of $\sim$ from $\sim_{\mathcal{T}}$

given two transition systems $\mathcal{T}_1$ and $\mathcal{T}_2$

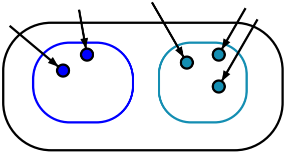
$\mathcal{T}_1$ with state space S_1



$\mathcal{T}_2$ with state space S_2



consider $\mathcal{T} = \mathcal{T}_1 \uplus \mathcal{T}_2$
(state space $S_1 \uplus S_2$)



$\mathcal{T}_1 \sim \mathcal{T}_2$ iff $\forall$ initial states s_1 of $\mathcal{T}_1$
$\exists$ initial state s_2 of $\mathcal{T}_2$ s.t. $s_1 \sim_{\mathcal{T}} s_2$,
and vice versa

Bisimulation quotient

Bisimulation quotient

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

Bisimulation quotient

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

bisimulation quotient $\mathcal{T}/\sim$ arises from $\mathcal{T}$
by collapsing bisimulation equivalent states

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\sim \, = (S', Act', \rightarrow', S'_0, AP, L')$$

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', Act', \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$

set of bisimulation equivalence classes

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', Act', \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S'_0 = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', Act', \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S'_0 = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', Act', \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S'_0 = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$

↑

well-defined

by the labeling condition of bisimulations

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\, = (S', Act', \rightarrow', S_0', AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S_0' = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$
- transition relation:

$$\frac{s \longrightarrow s'}{[s]_{\sim_{\mathcal{T}}} \longrightarrow [s']_{\sim_{\mathcal{T}}}}$$

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\sim = (S', Act', \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\sim_{\mathcal{T}}$
- set of initial states: $S'_0 = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$
- transition relation:

$$\frac{s \longrightarrow s'}{[s]_{\sim_{\mathcal{T}}} \longrightarrow [s']_{\sim_{\mathcal{T}}}}$$

action labels irrelevant

Bisimulation quotient

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', \{\tau\}, \rightarrow', S'_0, AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S'_0 = \{[s]_{\sim_{\mathcal{T}}} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$
- transition relation:

$$\frac{s \xrightarrow{\alpha} s'}{[s]_{\sim_{\mathcal{T}}} \xrightarrow{\tau} [s']_{\sim_{\mathcal{T}}}}$$

action labels irrelevant

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

bisimulation quotient:

$$\mathcal{T}/\!\sim\; = (S', \{\tau\}, \to', S_0', AP, L')$$

- state space: $S' = S/\!\sim_{\mathcal{T}}$
- set of initial states: $S_0' = \big\{ [s]_{\sim_{\mathcal{T}}} : s \in S_0 \big\}$
- labeling function: $L'([s]_{\sim_{\mathcal{T}}}) = L(s)$
- transition relation:

$$\frac{s \xrightarrow{\alpha} s'}{[s]_{\sim_{\mathcal{T}}} \xrightarrow{\tau} [s']_{\sim_{\mathcal{T}}}}$$

$$\mathcal{T} \sim \mathcal{T}/\!\sim$$

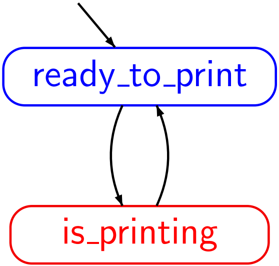
Example: interleaving of n printers

parallel system $\mathcal{T} = \underbrace{\mathit{Printer} \mathbin{|||} \mathit{Printer} \mathbin{|||} \ldots \mathbin{|||} \mathit{Printer}}_{n \text{ printer}}$

Example: interleaving of n printers

parallel system $\mathcal{T} = \underbrace{\textit{Printer} \,|||\, \textit{Printer} \,|||\, \ldots \,|||\, \textit{Printer}}_{n \text{ printer}}$

transition system
for each printer

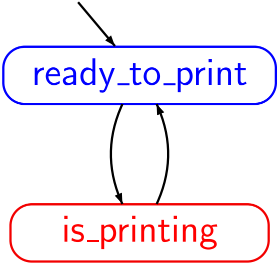


Example: interleaving of n printers

parallel system $\mathcal{T} = \underbrace{\mathit{Printer} \,|||\, \mathit{Printer} \,|||\, \ldots \,|||\, \mathit{Printer}}_{n \text{ printer}}$

$AP = \{0, 1, \ldots, n\}$ "number of available printers"

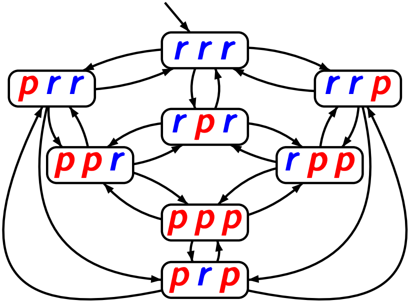
transition system
for each printer



Example: n=3 printers

parallel system $\mathcal{T} = \underbrace{Printer \,|||\, Printer \,|||\, \ldots \,|||\, Printer}_{n \text{ printer}}$

$AP = \{0, 1, 2, 3\}$



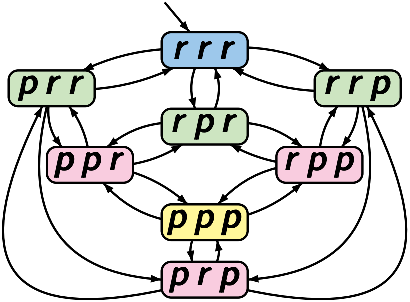
p: is printing

r: ready to print

Example: n=3 printers

parallel system $\mathcal{T} = \underbrace{\mathit{Printer} \,|||\, \mathit{Printer} \,|||\, \ldots \,|||\, \mathit{Printer}}_{n \text{ printer}}$

$AP = \{0, 1, 2, 3\}$



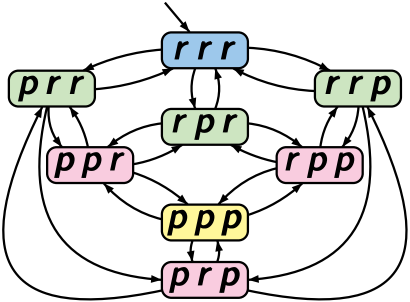
p: is printing

r: ready to print

Example: n=3 printers

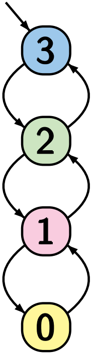
parallel system $\mathcal{T} = \underbrace{\mathit{Printer} \,|||\, \mathit{Printer} \,|||\, \ldots \,|||\, \mathit{Printer}}_{n \text{ printer}}$

$AP = \{0, 1, 2, 3\}$



p: is printing

r: ready to print

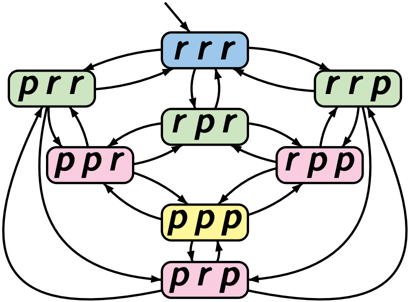


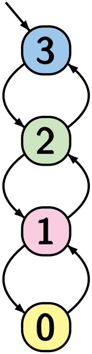
bisimulation quotient

Example: n=3 printers

parallel system $\mathcal{T} = \underbrace{Printer \parallel\!\mid Printer \parallel\!\mid \ldots \parallel\!\mid Printer}_{n \text{ printer}}$

$AP = \{0, 1, 2, 3\}$





2^n states

$n+1$ states

Mutual exclusion

Mutual exclusion

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

↑

given two concurrent processes P_1 and P_2

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

↑

given two concurrent processes P_1 and P_2

- two additional shared variables: $x_1, x_2 \in \mathbb{N}$

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

↑

given two concurrent processes P_1 and P_2

- two additional shared variables: $x_1, x_2 \in \mathbb{N}$
- if P_1 and P_2 are waiting then:

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

↑

given two concurrent processes P_1 and P_2

- two additional shared variables: $x_1, x_2 \in \mathbb{N}$
- if P_1 and P_2 are waiting then:

  if $x_1 < x_2$ then P_1 enters its critical section

  if $x_2 < x_1$ then P_2 enters its critical section

Mutual exclusion: Bakery algorithm

solutions for mutual exclusion problems:

- semaphore
- Peterson's algorithm
- Bakery algorithm

↑

given two concurrent processes P_1 and P_2

- two additional shared variables: $x_1, x_2 \in \mathbb{N}$
- if P_1 and P_2 are waiting then:

  if $x_1 < x_2$ then P_1 enters its critical section

  if $x_2 < x_1$ then P_2 enters its critical section

  $x_1 = x_2$: cannot happen

Bakery algorithm

protocol for P_1:

```
LOOP FOREVER
    noncritical actions
    x1 := x2+1
    AWAIT (x1 < x2) ∨ (x2=0);
    critical section;
    x1 := 0
END LOOP
```

symmetric protocol for P_2

protocol for P_1:

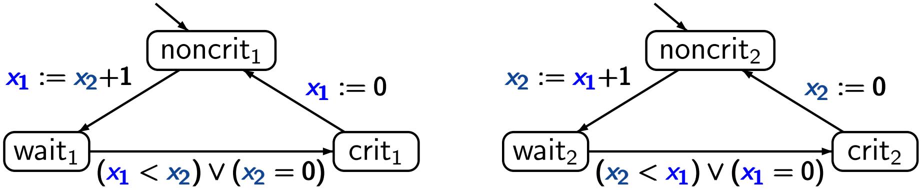
```
LOOP FOREVER
    noncritical actions
    x1 := x2+1
    AWAIT (x1 < x2) ∨ (x2=0);
    critical section;
    x1 := 0
END LOOP
```

initially:

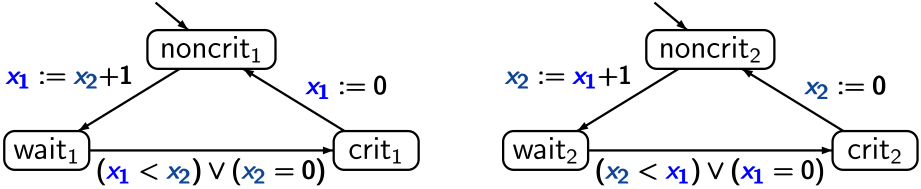
$x_1 = x_2 = 0$

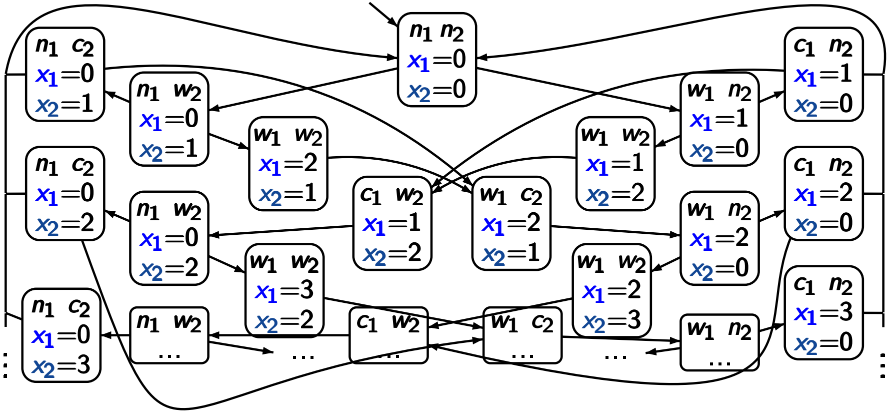
symmetric protocol for P_2

Program graphs for the Bakery algorithm

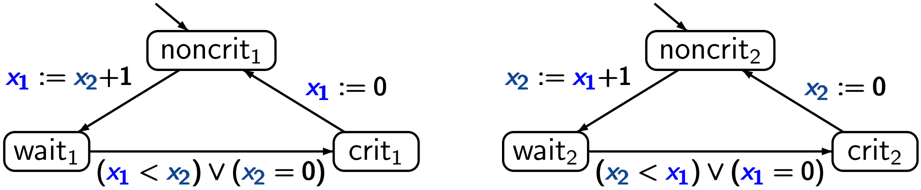


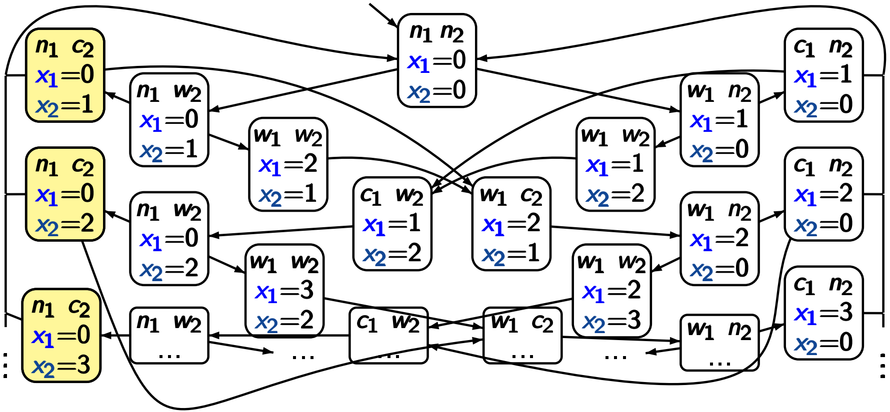
Transition system for the Bakery algorithm



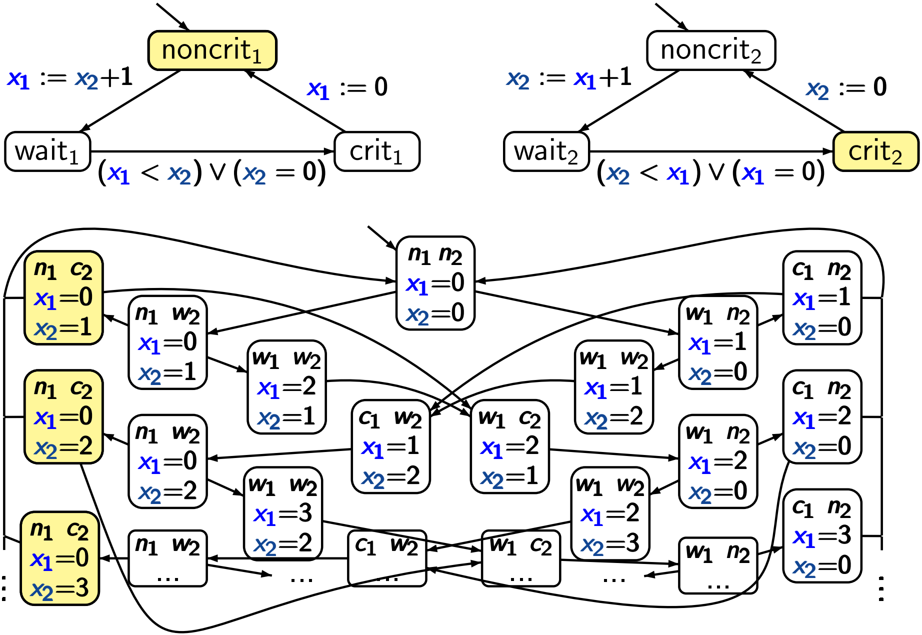


Transition system for the Bakery algorithm

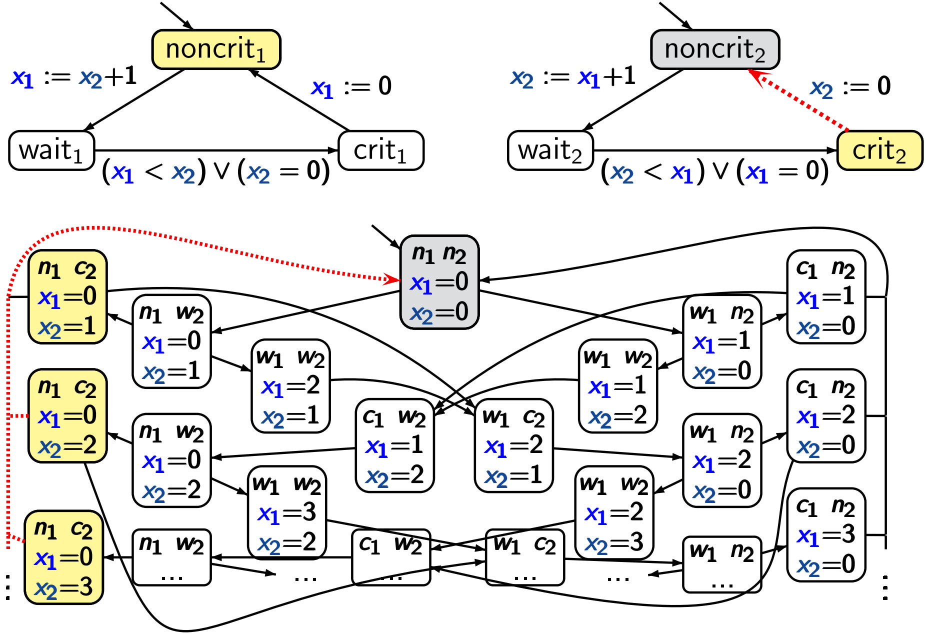




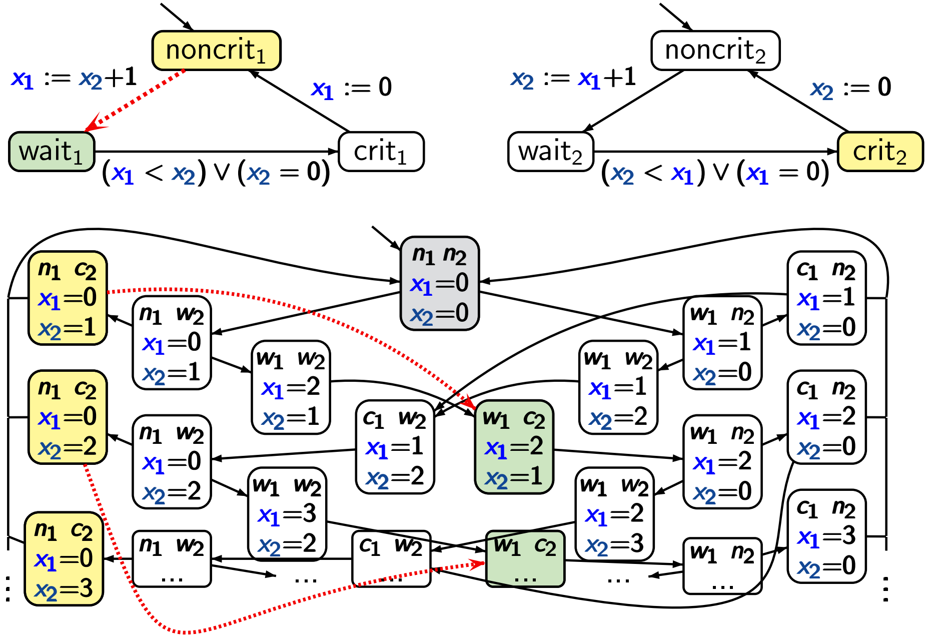
Transition system for the Bakery algorithm



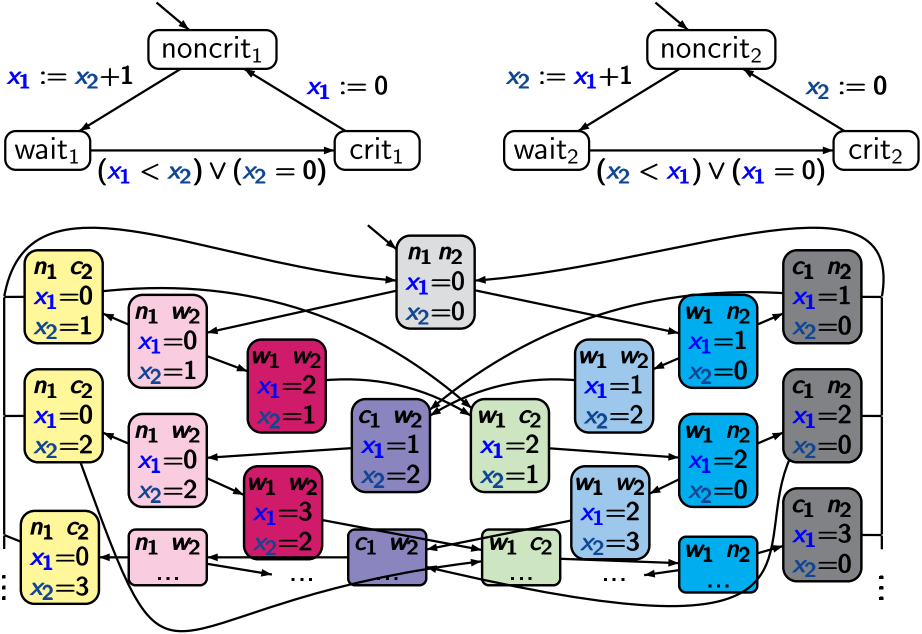
Transition system for the Bakery algorithm



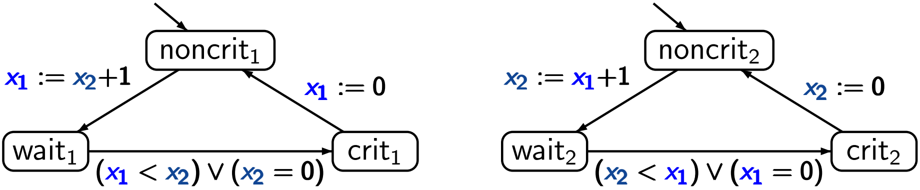
Transition system for the Bakery algorithm



Transition system for the Bakery algorithm

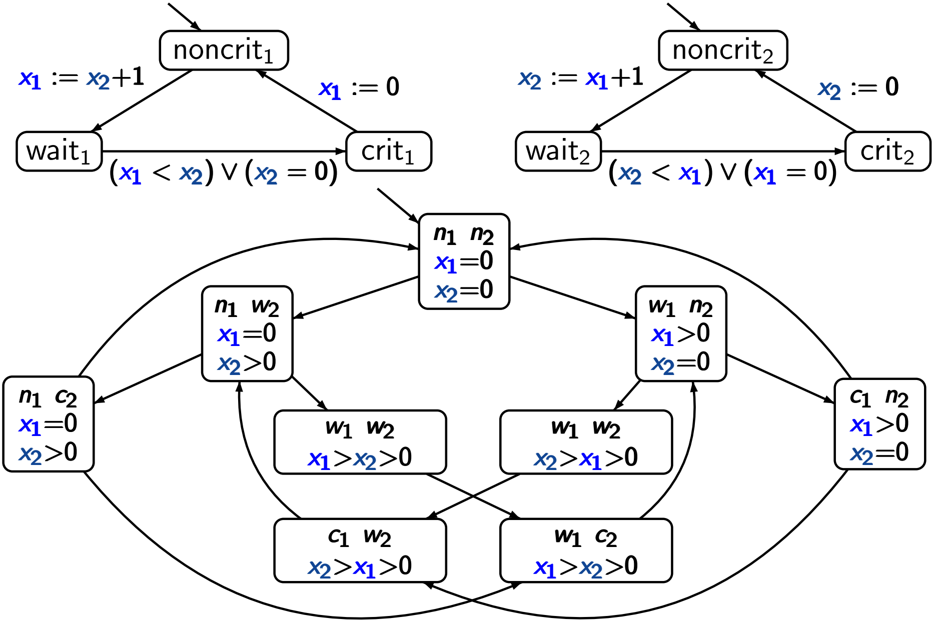


Bakery algorithm: bisimulation quotient

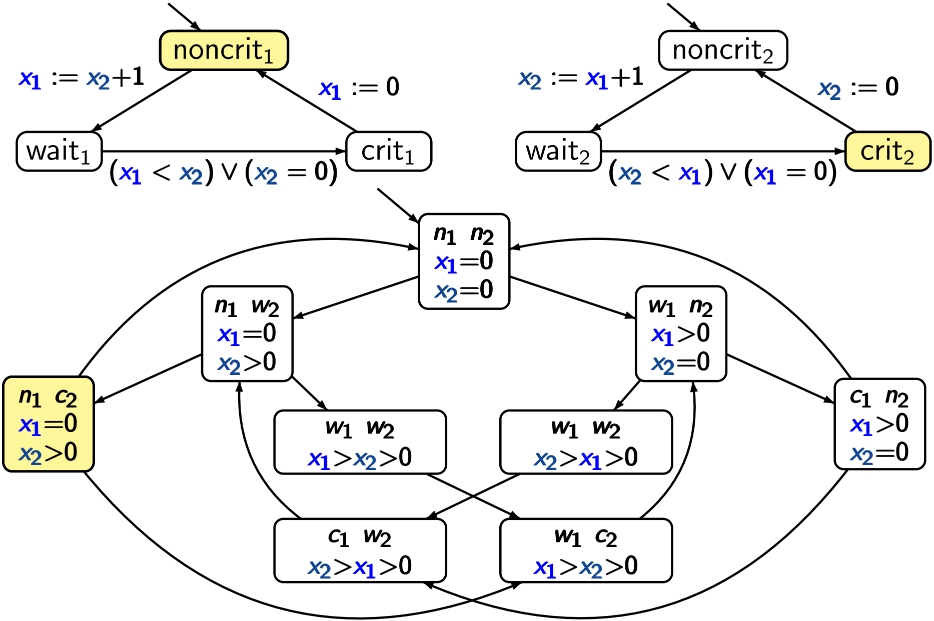


infinite transition system with a
finite bisimulation quotient

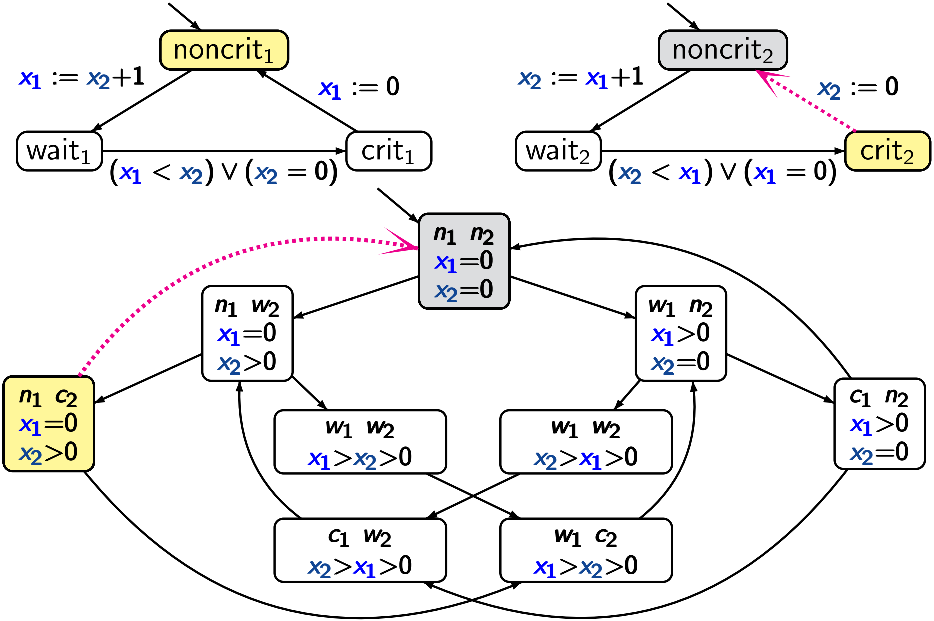
Bakery algorithm: bisimulation quotient



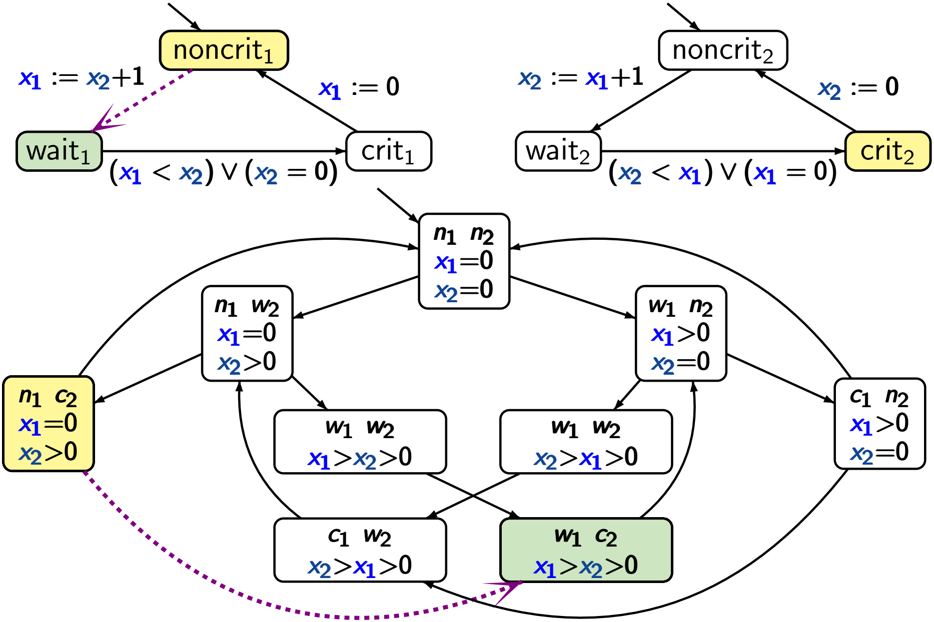
Bakery algorithm: bisimulation quotient



Bakery algorithm: bisimulation quotient



Bakery algorithm: bisimulation quotient



Overview

Recall: CTL*

CTL* state formulas

$$\Phi \ ::= \ \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

CTL* path formulas

$$\varphi \ ::= \ \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \, \mathsf{U} \, \varphi_2$$

derived operators:

- $\Diamond$, $\Box$, . . . as in **LTL**
- universal quantification: $\forall\varphi \stackrel{\text{def}}{=} \neg\exists\neg\varphi$

Recall: CTL* and CTL

CTL* state formulas

$$\Phi ::= \mathit{true} \mid a \mid \Phi_1 \wedge \Phi_2 \mid \neg\Phi \mid \exists\varphi$$

CTL* path formulas

$$\varphi ::= \Phi \mid \varphi_1 \wedge \varphi_2 \mid \neg\varphi \mid \bigcirc\varphi \mid \varphi_1 \,\mathsf{U}\, \varphi_2$$

CTL: sublogic of **CTL***

- with path quantifiers $\exists$ and $\forall$
- restricted syntax of path formulas:
  - *no* boolean combinations of path formulas
  - arguments of temporal operators $\bigcirc$ and U are state formulas

CTL equivalence

CTL equivalence

Let s_1, s_2 be states of a TS $\mathcal{T}$ without terminal states

s_1, s_2 are **CTL** equivalent if for all **CTL** formulas Φ:

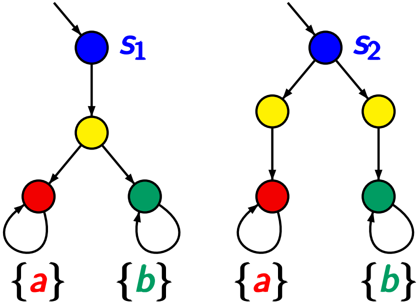
$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

CTL equivalence

Let s_1, s_2 be states of a TS $\mathcal{T}$ without terminal states

s_1, s_2 are **CTL** equivalent if for all **CTL** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

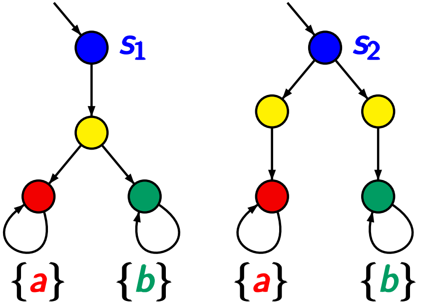


CTL equivalence

Let s_1, s_2 be states of a TS $\mathcal{T}$ without terminal states

s_1, s_2 are **CTL** equivalent if for all **CTL** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$



s_1, s_2 are not **CTL** equivalent

$s_1 \models \exists\bigcirc(\exists\bigcirc a \wedge \exists\bigcirc b)$

$s_2 \not\models \exists\bigcirc(\exists\bigcirc a \wedge \exists\bigcirc b)$

CTL, CTL* and LTL equivalence

Let s_1, s_2 be states of a TS $\mathcal{T}$ without terminal states

s_1, s_2 are **CTL** equivalent if for all **CTL** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

analogous definition for **CTL*** and **LTL**

CTL, CTL* and LTL equivalence

Let s_1, s_2 be states of a TS $\mathcal{T}$ without terminal states

s_1, s_2 are **CTL** equivalent if for all **CTL** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

s_1, s_2 are **CTL*** equivalent if for all **CTL*** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

s_1, s_2 are **LTL** equivalent if for all **LTL** formulas φ:

$$s_1 \models \varphi \quad \text{iff} \quad s_2 \models \varphi$$

CTL/CTL* and bisimulation

CTL/CTL* and bisimulation

bisimulation equivalence

= **CTL** equivalence

= **CTL*** equivalence

CTL/CTL* and bisimulation

	bisimulation equivalence
=	**CTL** equivalence
=	**CTL*** equivalence

$\longleftarrow$ for finite TS

	bisimulation equivalence	
=	**CTL** equivalence	⟵ for finite TS
=	**CTL*** equivalence	

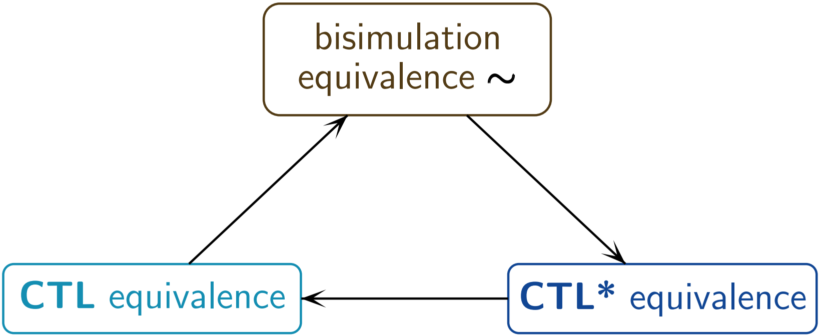
Let $\mathcal{T}$ be a finite TS without terminal states, and s_1, s_2 states in $\mathcal{T}$. Then:

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

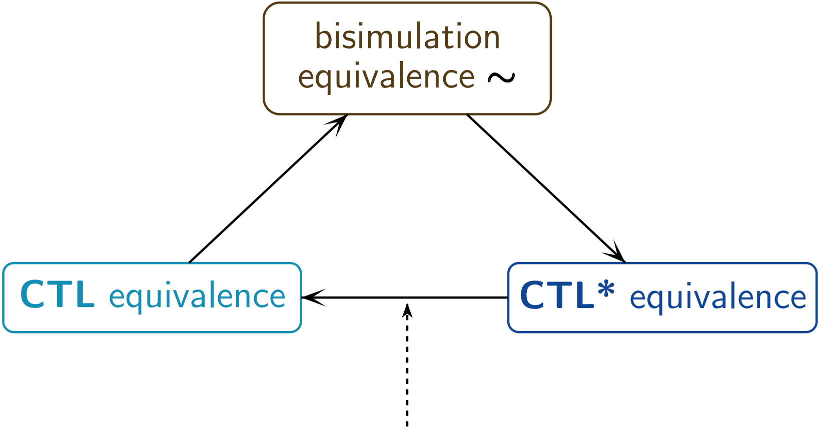
iff s_1 and s_2 are **CTL** equivalent

iff s_1 and s_2 are **CTL*** equivalent

CTL/CTL* and bisimulation

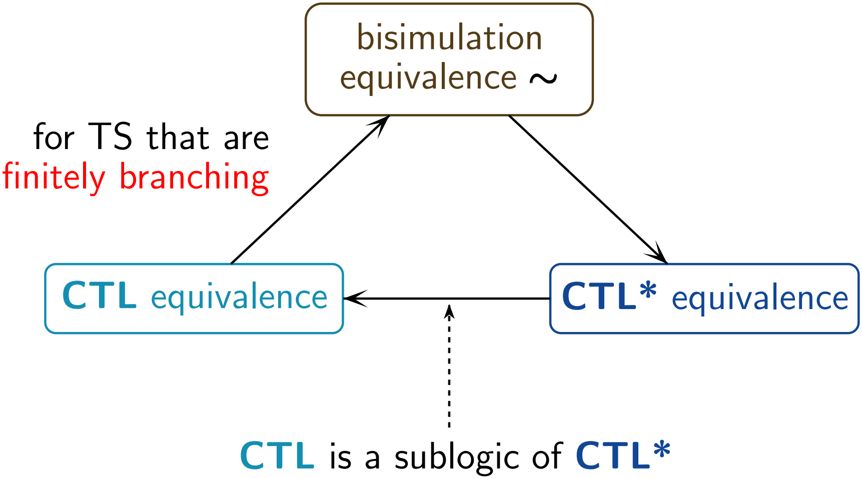


CTL/CTL* and bisimulation

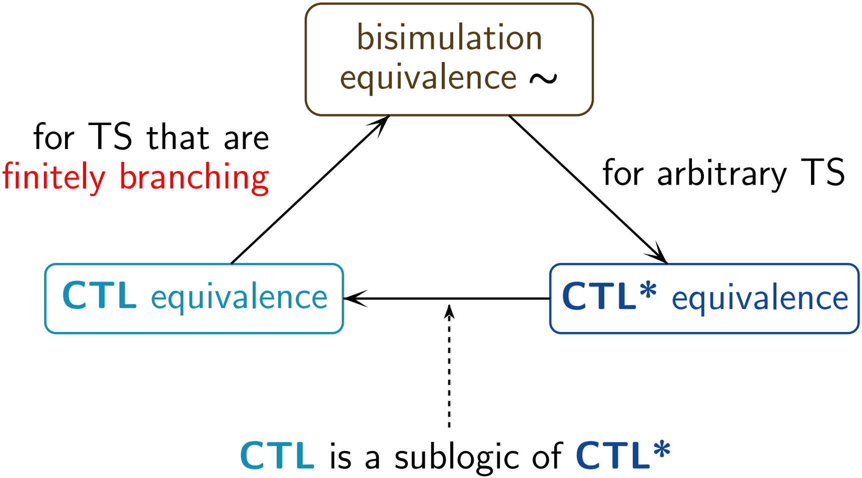


CTL is a sublogic of CTL*

CTL/CTL* and bisimulation



CTL/CTL* and bisimulation



Bisimulation equivalence $\Rightarrow$ CTL* equivalence

For arbitrary (possibly infinite) transition systems without terminal states:

If s_1, s_2 are states with $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL*** formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

Bisimulation equivalence $\Rightarrow$ CTL* equivalence

show by structural induction on **CTL*** formulas:

(a) if s_1, s_2 are states with $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL*** state formulas Φ:

$$s_1 \models \Phi \quad \text{iff} \quad s_2 \models \Phi$$

(b) if π_1, π_2 are paths with $\pi_1 \sim_{\mathcal{T}} \pi_2$ then for all **CTL*** path formulas φ:

$$\pi_1 \models \varphi \quad \text{iff} \quad \pi_2 \models \varphi$$

Bisimulation equivalence $\Rightarrow$ CTL* equivalence

show by structural induction on **CTL*** formulas:

(a) if s_1, s_2 are states with $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL*** state formulas Φ:

$$s_1 \models \Phi \text{ iff } s_2 \models \Phi$$

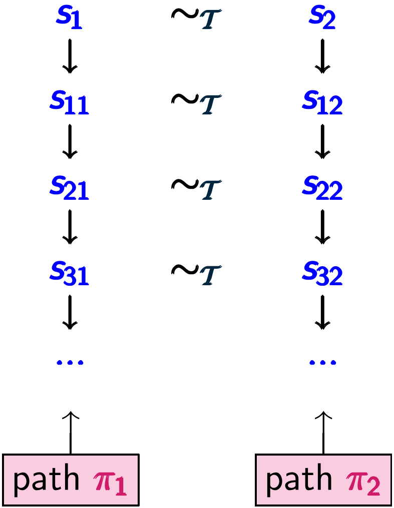
(b) if π_1, π_2 are paths with $\pi_1 \sim_{\mathcal{T}} \pi_2$ then for all **CTL*** path formulas φ:

$$\pi_1 \models \varphi \text{ iff } \pi_2 \models \varphi$$

$\pi_1 \sim_{\mathcal{T}} \pi_2 \quad \overset{\text{def}}{\Longleftrightarrow} \quad \pi_1$ and π_2 are statewise bisimulation equivalent

Bisimulation equivalence ⇒ CTL* equivalence

statewise bisimulation equivalent paths:



Bisimulation equivalence ⇒ CTL* equivalence

For all CTL* state formulas Φ and path formulas φ:

(a) if $s_1 \sim_{\mathcal{T}} s_2$ then: $s_1 \models \Phi$ iff $s_2 \models \Phi$

(b) if $\pi_1 \sim_{\mathcal{T}} \pi_2$ then: $\pi_1 \models \varphi$ iff $\pi_2 \models \varphi$

Bisimulation equivalence $\Rightarrow$ CTL* equivalence

For all CTL* state formulas Φ and path formulas φ:

(a) if $s_1 \sim_{\mathcal{T}} s_2$ then: $s_1 \models \Phi$ iff $s_2 \models \Phi$

(b) if $\pi_1 \sim_{\mathcal{T}} \pi_2$ then: $\pi_1 \models \varphi$ iff $\pi_2 \models \varphi$

Proof by structural induction

Bisimulation equivalence $\Rightarrow$ CTL* equivalence

For all CTL* state formulas Φ and path formulas φ:

(a) if $s_1 \sim_{\mathcal{T}} s_2$ then: $s_1 \models \Phi$ iff $s_2 \models \Phi$

(b) if $\pi_1 \sim_{\mathcal{T}} \pi_2$ then: $\pi_1 \models \varphi$ iff $\pi_2 \models \varphi$

Proof by structural induction

base of induction:

(a) $\Phi = \mathit{true}$ or $\Phi = a \in AP$

(b) $\varphi = \Phi$ for some state formula Φ
s.t. statement (a) holds for Φ

Bisimulation equivalence $\Rightarrow$ CTL* equivalence

For all CTL* state formulas Φ and path formulas φ:

(a) if $s_1 \sim_{\mathcal{T}} s_2$ then: $s_1 \models \Phi$ iff $s_2 \models \Phi$

(b) if $\pi_1 \sim_{\mathcal{T}} \pi_2$ then: $\pi_1 \models \varphi$ iff $\pi_2 \models \varphi$

Proof by structural induction

step of induction:

(a) consider $\Phi = \Phi_1 \wedge \Phi_2$, $\neg\Psi$ or $\exists\varphi$ s.t.

- (a) holds for Φ_1, Φ_2, Ψ
- (b) holds for φ

(b) consider $\varphi = \varphi_1 \wedge \varphi_2$, $\neg\varphi'$, $\bigcirc\varphi'$, $\varphi_1 \mathsf{U} \varphi_2$ s.t.

- (a) holds for $\varphi_1, \varphi_2, \varphi'$

Path lifting for $\sim_{\mathcal{T}}$

$$
\begin{array}{ccc}
s_1 & \sim_{\mathcal{T}} & s_2 \\
\downarrow & & \\
s_{11} & & \\
\downarrow & & \\
s_{21} & & \\
\downarrow & & \\
s_{31} & & \\
\downarrow & & \\
\vdots & &
\end{array}
$$

can be
completed to

$$
\begin{array}{ccc}
s_1 & \sim_{\mathcal{T}} & s_2 \\
\downarrow & & \downarrow \\
s_{11} & \sim_{\mathcal{T}} & s_{12} \\
\downarrow & & \downarrow \\
s_{21} & \sim_{\mathcal{T}} & s_{22} \\
\downarrow & & \downarrow \\
s_{31} & \sim_{\mathcal{T}} & s_{32} \\
\downarrow & & \downarrow \\
\vdots & & \vdots
\end{array}
$$

Path lifting for $\sim_{\mathcal{T}}$

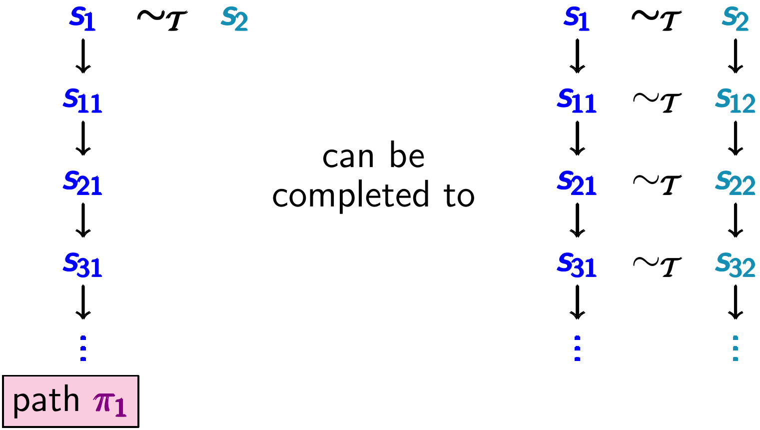
$$\begin{array}{ccc} s_1 & \sim_{\mathcal{T}} & s_2 \\ \downarrow & & \\ s_{11} & & \\ \downarrow & & \\ s_{21} & & \\ \downarrow & & \\ s_{31} & & \\ \downarrow & & \\ \vdots & & \end{array}$$

can be completed to

$$\begin{array}{ccc} s_1 & \sim_{\mathcal{T}} & s_2 \\ \downarrow & & \downarrow \\ s_{11} & \sim_{\mathcal{T}} & s_{12} \\ \downarrow & & \downarrow \\ s_{21} & \sim_{\mathcal{T}} & s_{22} \\ \downarrow & & \downarrow \\ s_{31} & \sim_{\mathcal{T}} & s_{32} \\ \downarrow & & \downarrow \\ \vdots & & \vdots \end{array}$$

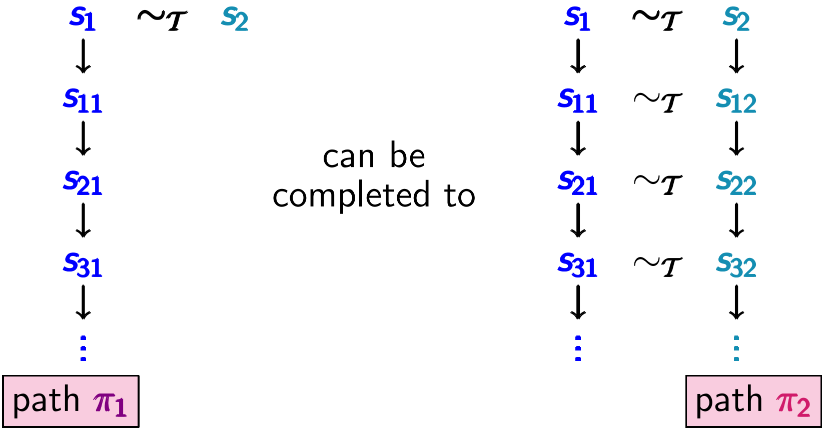
If $s_1 \sim_{\mathcal{T}} s_2$ then for all $\pi_1 \in Paths(s_1)$ there exists $\pi_2 \in Paths(s_2)$ with $\pi_1 \sim_{\mathcal{T}} \pi_2$

Path lifting for $\sim_{\mathcal{T}}$



If $s_1 \sim_{\mathcal{T}} s_2$ then for all $\pi_1 \in \mathit{Paths}(s_1)$
there exists $\pi_2 \in \mathit{Paths}(s_2)$ with $\pi_1 \sim_{\mathcal{T}} \pi_2$

Path lifting for $\sim_{\mathcal{T}}$



If $s_1 \sim_{\mathcal{T}} s_2$ then for all $\pi_1 \in Paths(s_1)$ there exists $\pi_2 \in Paths(s_2)$ with $\pi_1 \sim_{\mathcal{T}} \pi_2$

Correct or wrong?

If s_1, s_2 are not **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

Correct or wrong?

If s_1, s_2 are not **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

correct.

Correct or wrong?

If s_1, s_2 are not **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

correct.

If s_1, s_2 are not **LTL** equivalent then there exists a **LTL** formula φ with $s_1 \models \varphi$ and $s_2 \not\models \varphi$

Correct or wrong?

If s_1, s_2 are not **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

correct.

If s_1, s_2 are not **LTL** equivalent then there exists a **LTL** formula φ with $s_1 \models \varphi$ and $s_2 \not\models \varphi$

wrong.

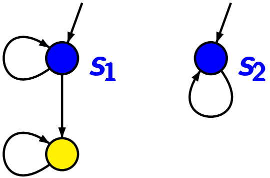
Correct or wrong?

If s_1, s_2 are not **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

correct.

If s_1, s_2 are not **LTL** equivalent then there exists a **LTL** formula φ with $s_1 \models \varphi$ and $s_2 \not\models \varphi$

wrong.



Correct or wrong?

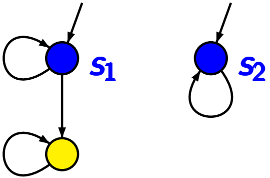
> If s_1, s_2 are <u>not</u> **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

correct.

> If s_1, s_2 are <u>not</u> **LTL** equivalent then there exists a **LTL** formula φ with $s_1 \models \varphi$ and $s_2 \not\models \varphi$

wrong.

$Traces(s_2) \subset Traces(s_1)$



If s_1, s_2 are <u>not</u> **CTL** equivalent then there exists a **CTL** formula Φ with $s_1 \models \Phi$ and $s_2 \not\models \Phi$

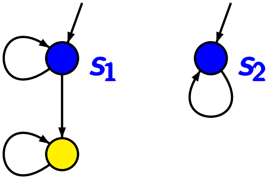
correct.

If s_1, s_2 are <u>not</u> **LTL** equivalent then there exists a **LTL** formula φ with $s_1 \models \varphi$ and $s_2 \not\models \varphi$

wrong.

$Traces(s_2) \subset Traces(s_1)$

hence: $s_1 \models \varphi$ implies $s_2 \models \varphi$



CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

CTL equivalence $\Longrightarrow$ bisimulation equivalence

> If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:
>
> if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

Proof: show that

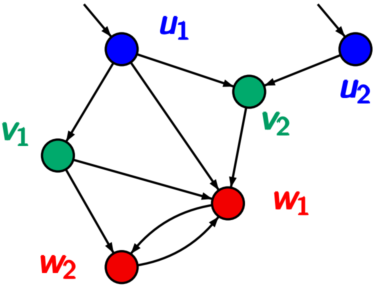
$$\mathcal{R} \stackrel{\text{def}}{=} \{ (s_1, s_2) : s_1, s_2 \text{ satisfy the same } \textbf{CTL} \text{ formulas} \}$$

is a bisimulation, i.e., for all $(s_1, s_2) \in \mathcal{R}$:

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

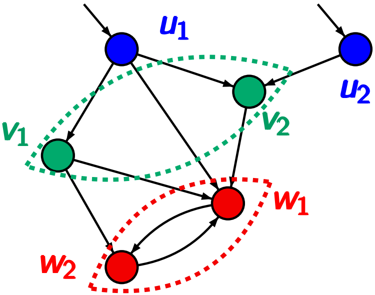
(2) if $s_1 \longrightarrow t_1$ then there exists a transition $s_2 \longrightarrow t_2$ s.t. $(t_1, t_2) \in \mathcal{R}$

Example: CTL master formulas



- (blue) $\widehat{=} \{a\}$
- (red) $\widehat{=} \{b\}$
- (green) $\widehat{=} \varnothing$

Example: CTL master formulas

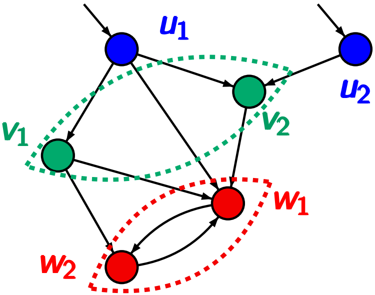


bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

 $\widehat{=} \{a\}$

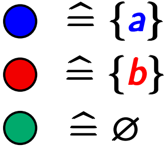
 $\widehat{=} \{b\}$

 $\widehat{=} \varnothing$

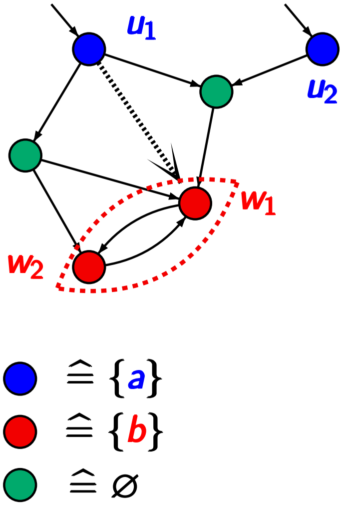


bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

but $u_1 \not\sim_{\mathcal{T}} u_2$



Example: CTL master formulas



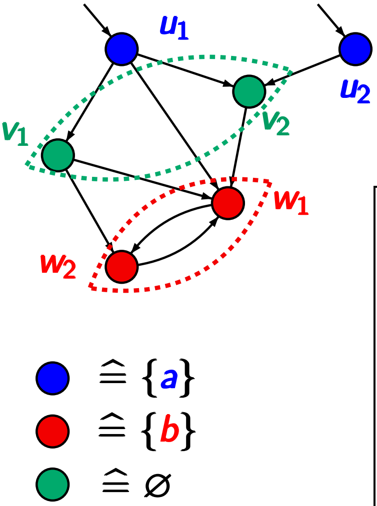
bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

but $u_1 \not\sim_{\mathcal{T}} u_2$

as $u_1 \rightarrow \{w_1, w_2\}$

$u_2 \not\rightarrow \{w_1, w_2\}$

Example: CTL master formulas



bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

CTL master formulas:

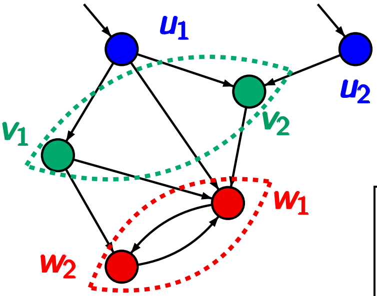
$w_1, w_2 \models$?

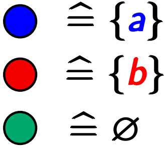
$v_1, v_2 \models$?

$u_1 \models$?

$u_2 \models$?

Example: CTL master formulas





bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

CTL master formulas:

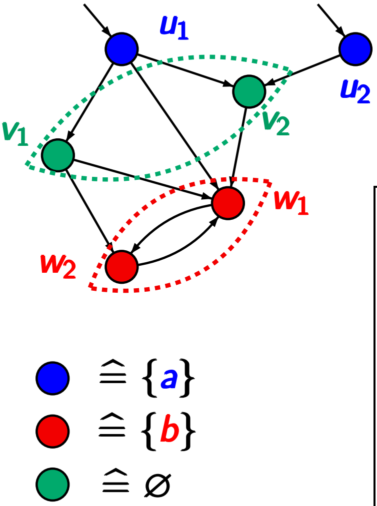
$w_1, w_2 \models b$

$v_1, v_2 \models$?

$u_1 \models$?

$u_2 \models$?

Example: CTL master formulas



bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$

CTL master formulas:

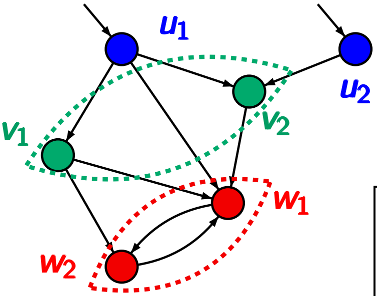
$w_1, w_2 \models b$

$v_1, v_2 \models \neg a \wedge \neg b$

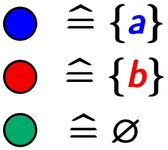
$u_1 \models$?

$u_2 \models$?

Example: CTL master formulas



bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \ldots\}$



CTL master formulas:

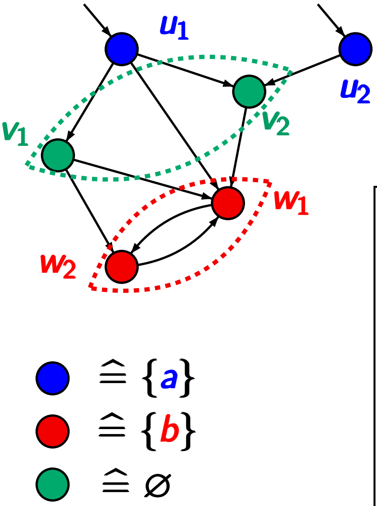
$w_1, w_2 \models b$

$v_1, v_2 \models \neg a \wedge \neg b$

$u_1 \models (\exists \bigcirc b) \wedge a$

$u_2 \models ?$

Example: CTL master formulas



bisimulation equivalence $\sim_{\mathcal{T}}$
$= \{(v_1, v_2), (w_1, w_2), \dots\}$

CTL master formulas:

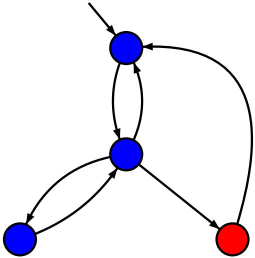
$w_1, w_2 \models b$

$v_1, v_2 \models \neg a \wedge \neg b$

$u_1 \models (\exists \bigcirc b) \wedge a$

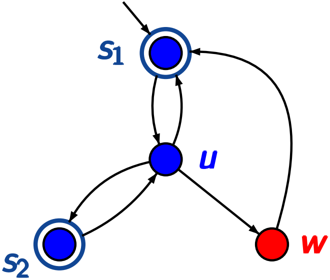
$u_2 \models (\neg \exists \bigcirc b) \wedge a$

...master formulas for $\sim_{\mathcal{T}}$-classes?



$AP = \{blue, red\}$

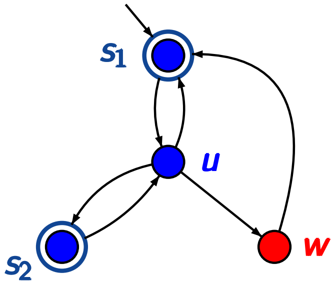
...master formulas for $\sim_{\mathcal{T}}$-classes?



$AP = \{blue, red\}$

$s_1 \sim_{\mathcal{T}} s_2 \not\sim_{\mathcal{T}} u$

...master formulas for $\sim_{\mathcal{T}}$-classes?



$AP = \{blue, red\}$

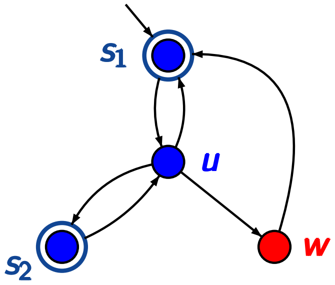
$s_1 \sim_{\mathcal{T}} s_2 \not\sim_{\mathcal{T}} u$

$\Phi_w = ?$

$\Phi_C = ?$ where $C = \{s_1, s_2\}$

$\Phi_u = ?$

...master formulas for $\sim_{\mathcal{T}}$-classes?



$AP = \{blue, red\}$

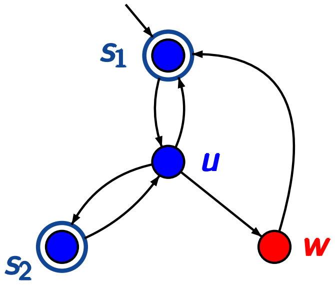
$s_1 \sim_{\mathcal{T}} s_2 \not\sim_{\mathcal{T}} u$

$$\Phi_w = red$$

$$\Phi_C = ?$$ where $C = \{s_1, s_2\}$

$$\Phi_u = ?$$

...master formulas for $\sim_{\mathcal{T}}$-classes?



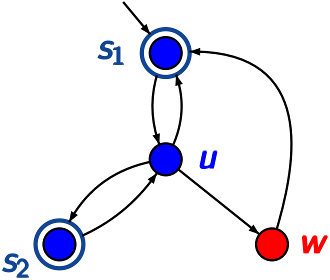
$AP = \{blue, red\}$

$s_1 \sim_{\mathcal{T}} s_2 \not\sim_{\mathcal{T}} u$

$\Phi_w = red$

$\Phi_C = blue \wedge \forall \bigcirc blue$ where $C = \{s_1, s_2\}$

$\Phi_u = ?$



$AP = \{blue, red\}$

$s_1 \sim_{\mathcal{T}} s_2 \not\sim_{\mathcal{T}} u$

$$\Phi_w = red$$

$$\Phi_C = blue \wedge \forall \bigcirc blue \quad \text{where } C = \{s_1, s_2\}$$

$$\Phi_u = \exists \bigcirc red$$

CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

- wrong for infinite TS

CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

- wrong for infinite TS
- but also holds for finitely branching TS

CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finite TS then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

- wrong for infinite TS
- but also holds for finitely branching TS

↑

possibly infinite-state TS such that

* the number of initial states is finite
* for each state the number of successors is finite

CTL equivalence $\Longrightarrow$ bisimulation equivalence

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

CTL equivalence $\Longrightarrow$ bisimulation equivalence

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

- S_0 is finite
- $Post(s)$ is finite for all $s \in S$

CTL equivalence $\Longrightarrow$ bisimulation equivalence

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

- * S_0 is finite
- * $Post(s)$ is finite for all $s \in S$

Then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

CTL equivalence $\Longrightarrow$ bisimulation equivalence

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

- S_0 is finite
- $Post(s)$ is finite for all $s \in S$

Then, for all states s_1, s_2 in $\mathcal{T}$:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

Proof: as for finite TS.

CTL equivalence $\Longrightarrow$ bisimulation equivalence

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

- S_0 is finite
- $Post(s)$ is finite for all $s \in S$

Then, for all states s_1, s_2 in $\mathcal{T}$:

> if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

Proof: as for finite TS. Amounts showing that

$$\mathcal{R} \stackrel{\text{def}}{=} \{ (s_1, s_2) : s_1, s_2 \text{ satisfy the same } \textbf{CTL} \text{ formulas} \}$$

is a bisimulation.

CTL equivalence $\Longrightarrow$ bisimulation equivalence

If $\mathcal{T}$ is a finitely branching TS then for all states s_1, s_2:

if s_1, s_2 are **CTL** equivalent then $s_1 \sim_{\mathcal{T}} s_2$

Proof: show that

$$\mathcal{R} \stackrel{\text{def}}{=} \{ (s_1, s_2) : s_1, s_2 \text{ satisfy the same } \textbf{CTL} \text{ formulas} \}$$

is a bisimulation, i.e., for $(s_1, s_2) \in \mathcal{R}$:

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

(2) if $s_1 \rightarrow t_1$ then there exists a transition $s_2 \rightarrow t_2$ s.t. $(t_1, t_2) \in \mathcal{R}$

Summary: CTL/CTL* and bisimulation

Summary: CTL/CTL* and bisimulation

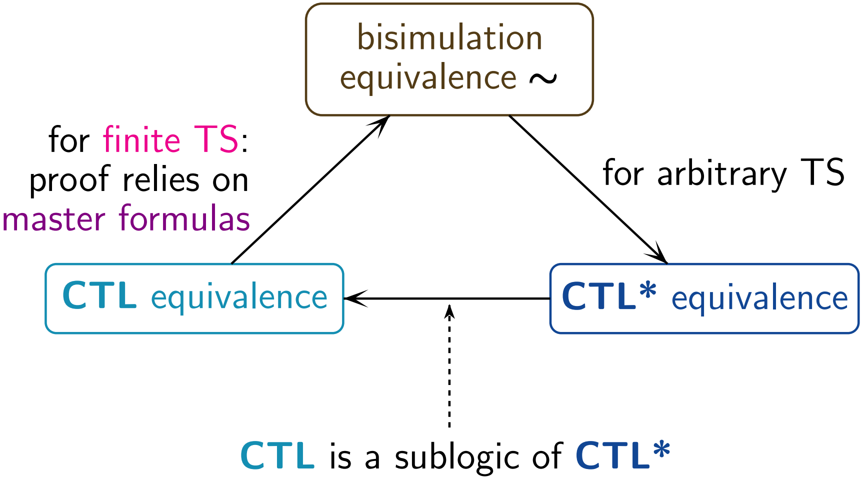
Let $\mathcal{T}$ be a finitely branching TS without terminal states, and s_1, s_2 states in $\mathcal{T}$. Then:

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

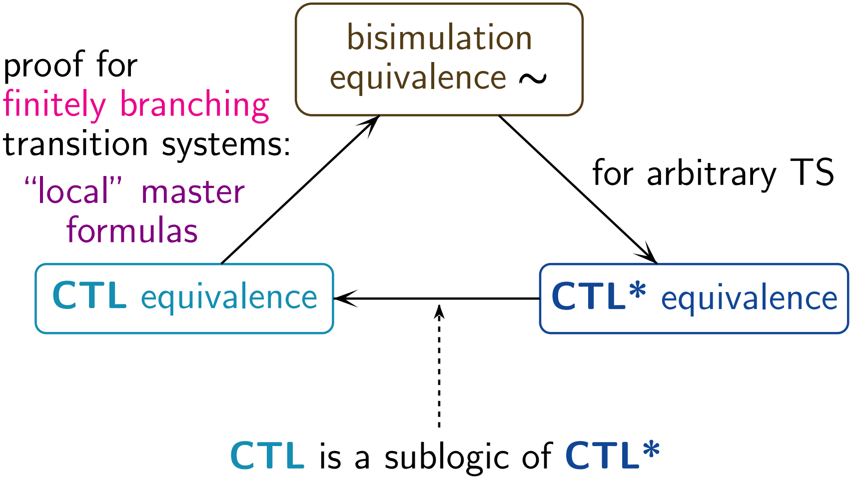
iff s_1 and s_2 are **CTL** equivalent

iff s_1 and s_2 are **CTL*** equivalent

Summary: CTL/CTL* and bisimulation



Summary: CTL/CTL* and bisimulation



CTL/CTL* and bisimulation for TS

CTL/CTL* and bisimulation for TS

so far: we considered

- **CTL**/**CTL*** equivalence
- bisimulation equivalence $\sim_{\mathcal{T}}$

for the states of a single transition system $\mathcal{T}$

CTL/CTL* and bisimulation for TS

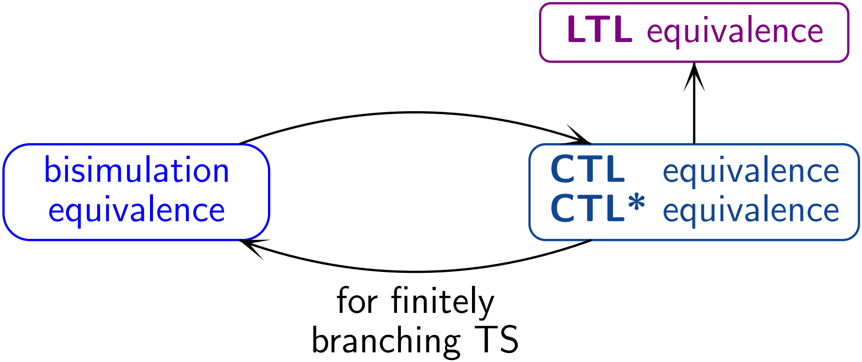
If $\mathcal{T}_1$, $\mathcal{T}_2$ are finitely branching TS over AP without terminal states then:

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

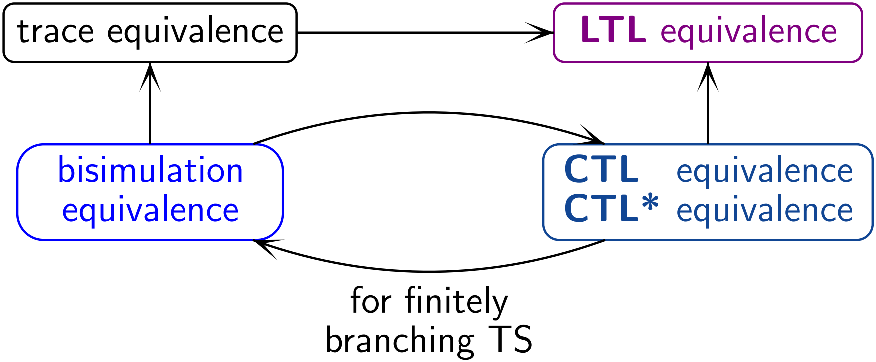
iff $\mathcal{T}_1$ and $\mathcal{T}_2$ satisfy the same **CTL** formulas

iff $\mathcal{T}_1$ and $\mathcal{T}_2$ satisfy the same **CTL*** formulas

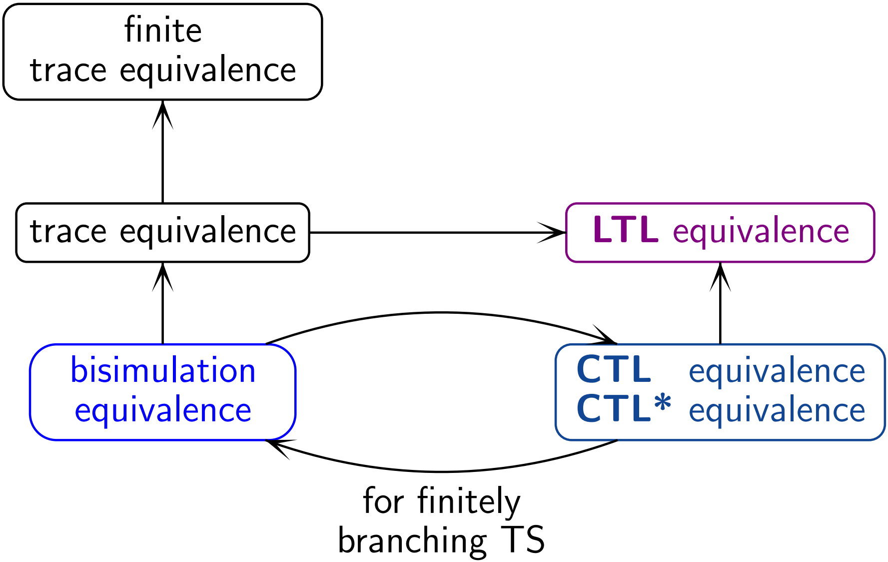
Summary: equivalences



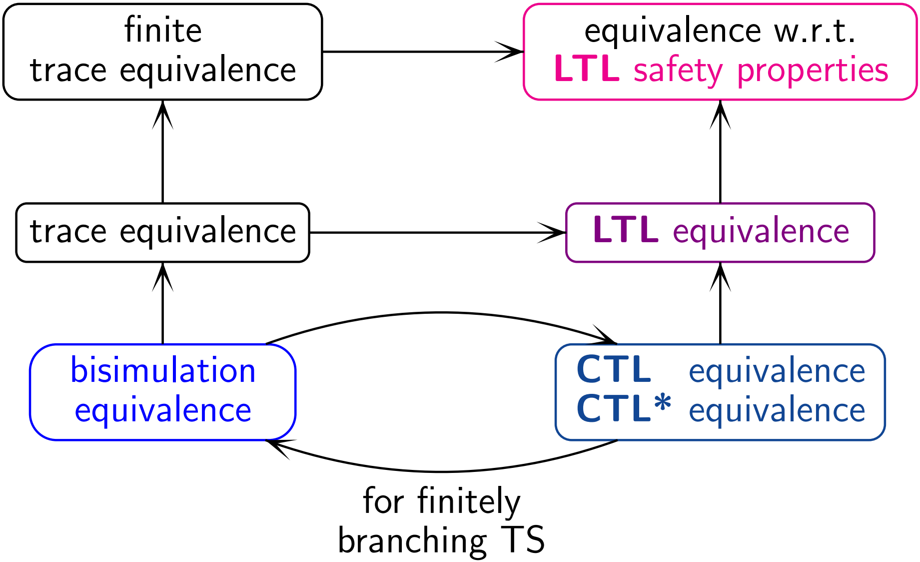
Summary: equivalences



Summary: equivalences



Summary: equivalences



Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

If s_1, s_2 satisfy the same $\mathrm{CTL}_{\setminus \mathrm{U}}$ formulas then $s_1 \sim_{\mathcal{T}} s_2$.

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

If s_1, s_2 satisfy the same $\mathrm{CTL}_{\setminus \mathbf{U}}$ formulas then

$$s_1 \sim_{\mathcal{T}} s_2.$$

where $\mathrm{CTL}_{\setminus \mathbf{U}} \mathrel{\widehat{=}} \mathrm{CTL}$ without until operator $\mathbf{U}$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $\text{CTL}_{\setminus \text{U}}$ formulas then
> $s_1 \sim_{\mathcal{T}} s_2$.

where $\text{CTL}_{\setminus \text{U}} \mathrel{\widehat{=}} \text{CTL}$ without until operator $\mathbf{U}$

correct.

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

If s_1, s_2 satisfy the same $\text{CTL}_{\setminus \text{U}}$ formulas then $s_1 \sim_{\mathcal{T}} s_2$.

where $\text{CTL}_{\setminus \text{U}} \mathrel{\widehat{=}} \text{CTL}$ without until operator **U**

correct. see the proof

"**CTL** equivalence $\Longrightarrow$ bisimulation equivalence"

$\text{CTL}_{\setminus U}$-equivalence $\Rightarrow$ bisimulation equivalence

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $\text{CTL}_{\setminus U}$ formulas then
> $s_1 \sim_{\mathcal{T}} s_2$.

Proof. Show that $\text{CTL}_{\setminus U}$ equivalence is a bisimulation

$\mathrm{CTL}_{\setminus \mathrm{U}}$-equivalence $\Rightarrow$ bisimulation equivalence

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $\mathrm{CTL}_{\setminus \mathrm{U}}$ formulas then
> $s_1 \sim_{\mathcal{T}} s_2$.

Proof. Show that $\mathrm{CTL}_{\setminus \mathrm{U}}$ equivalence is a bisimulation

- labeling condition only uses atomic propositions

$\mathrm{CTL}_{\setminus U}$-equivalence $\Rightarrow$ bisimulation equivalence

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $\mathrm{CTL}_{\setminus U}$ formulas then
> $$s_1 \sim_{\mathcal{T}} s_2.$$

Proof. Show that $\mathrm{CTL}_{\setminus U}$ equivalence is a bisimulation

- labeling condition only uses atomic propositions
- simulation condition can be established by $\mathrm{CTL}_{\setminus U}$ master formulas of the form:

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $\mathrm{CTL}_{\setminus U}$ formulas then
> $s_1 \sim_{\mathcal{T}} s_2$.

Proof. Show that $\mathrm{CTL}_{\setminus U}$ equivalence is a bisimulation

- labeling condition only uses atomic propositions
- simulation condition can be established by $\mathrm{CTL}_{\setminus U}$ master formulas of the form:

$$\exists \bigcirc \Phi_C \quad \text{where} \quad \Phi_C = \bigwedge_D \Phi_{C,D}$$

Let $\mathcal{T}$ be a finite TS without terminal states and s_1, s_2 states of $\mathcal{T}$.

> If s_1, s_2 satisfy the same $CTL_{\setminus U}$ formulas then
> $s_1 \sim_{\mathcal{T}} s_2$.

Proof. Show that $CTL_{\setminus U}$ equivalence is a bisimulation

- labeling condition only uses atomic propositions
- simulation condition can be established by $CTL_{\setminus U}$ master formulas of the form:

$$\exists \bigcirc \Phi_C \quad \text{where} \quad \Phi_C = \bigwedge_D \Phi_{C,D}$$

$$\text{and} \quad Sat(\Phi_{C,D}) \subseteq C \setminus D$$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states.

> $\mathcal{T}$ and its bisimulation quotient $\mathcal{T}/\sim$ satisfy the same **CTL*** formulas.

Let $\mathcal{T}$ be a finite TS without terminal states.

$\mathcal{T}$ and its bisimulation quotient $\mathcal{T}/\sim$ satisfy the same **CTL*** formulas.

correct.

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states.

$\mathcal{T}$ and its bisimulation quotient $\mathcal{T}/\!\sim$ satisfy the same **CTL*** formulas.

correct. Recall that $\mathcal{T} \sim \mathcal{T}/\!\sim$

Let $\mathcal{T}$ be a finite TS without terminal states.

> $\mathcal{T}$ and its bisimulation quotient $\mathcal{T}/\!\sim$ satisfy the same **CTL*** formulas.

correct. Recall that $\mathcal{T} \sim \mathcal{T}/\!\sim$ as

$$\mathcal{R} = \{(s, [s]) : s \in S\}$$

is a bisimulation for $(\mathcal{T}, \mathcal{T}/\!\sim)$

here: $[s] = \sim_{\mathcal{T}}$-equivalence class of state s

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

correct

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

correct, as $\models_{fair}$ is "**CTL***-definable"

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

correct, as $\models_{fair}$ is "**CTL***-definable"

For each **CTL*** state formula Φ there exists a **CTL*** formula Ψ s.t. $s \models \Psi$ iff $s \models_{fair} \Phi$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

correct, as $\models_{fair}$ is "**CTL***-definable"

For each **CTL*** state formula Φ there exists a **CTL*** formula Ψ s.t. $s \models \Psi$ iff $s \models_{fair} \Phi$

Example: for $\Phi = \exists\Box(a \wedge \forall\Diamond b)$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS without terminal states and let *fair* be a **CTL** fairness assumption.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **CTL** formulas Φ:

$$s_1 \models_{fair} \Phi \quad \text{iff} \quad s_2 \models_{fair} \Phi$$

correct, as $\models_{fair}$ is "**CTL***-definable"

↑

For each **CTL*** state formula Φ there exists a **CTL*** formula Ψ s.t. $s \models \Psi$ iff $s \models_{fair} \Phi$

Example: for $\Phi = \exists\Box(a \wedge \forall\Diamond b)$

$$\Psi = \exists\big(fair \wedge \Box(a \wedge \forall(fair \rightarrow \Diamond b))\big)$$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS over AP without terminal states.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models E \quad \text{iff} \quad s_2 \models E$$

Correct or wrong?

Let $\mathcal{T}$ be a finite TS over AP without terminal states.

> If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:
>
> $$s_1 \models E \quad \text{iff} \quad s_2 \models E$$

correct.

Correct or wrong?

Let $\mathcal{T}$ be a finite TS over AP without terminal states.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^{\omega}$:

$$s_1 \models E \quad \text{iff} \quad s_2 \models E$$

correct.

Note that:

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

Correct or wrong?

Let $\mathcal{T}$ be a finite TS over AP without terminal states.

> If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:
>
> $$s_1 \models E \quad \text{iff} \quad s_2 \models E$$

correct.

Note that:

$$(1) \quad s_1 \sim_{\mathcal{T}} s_2 \quad \Longrightarrow \quad Traces(s_1) = Traces(s_2)$$

$$(2) \quad s \models E \quad \Longleftrightarrow \quad Traces(s) \subseteq E$$

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

e.g., strong fairness for a single action α

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

e.g., strong fairness for a single action α

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

Correct or wrong?

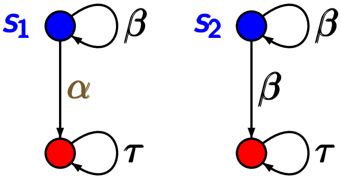
Let $\mathcal{F}$ be an action-based strong fairness assumption

e.g., strong fairness for a single action α

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.



$\mathcal{F} \mathrel{\widehat{=}}$ strong fairness assumption for action α

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

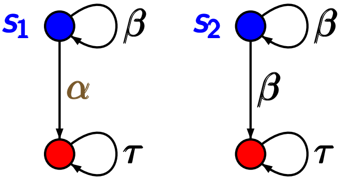
e.g., strong fairness for a single action α

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

$E \mathrel{\widehat{=}} \Diamond red$



$\mathcal{F} \mathrel{\widehat{=}}$ strong fairness assumption for action α

Let $\mathcal{F}$ be an action-based strong fairness assumption

e.g., strong fairness for a single action α

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

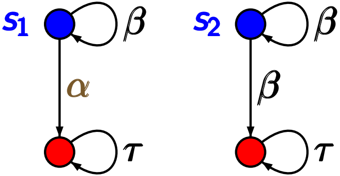
$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

$E \mathrel{\widehat{=}} \Diamond \mathit{red}$

$s_1 \models_{\mathcal{F}} E$

$s_2 \not\models_{\mathcal{F}} E$



$\mathcal{F} \mathrel{\widehat{=}}$ strong fairness assumption for action α

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^{\omega}$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all safety properties $E \subseteq (2^{AP})^{\omega}$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all safety properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

correct.

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

$$\text{If } s_1 \sim_{\mathcal{T}} s_2 \text{ then for all } \mathbf{LT} \text{ properties } E \subseteq (2^{AP})^\omega: \quad s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

$$\text{If } s_1 \sim_{\mathcal{T}} s_2 \text{ then for all safety properties } E \subseteq (2^{AP})^\omega: \quad s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

correct.

- realizable fairness irrelevant for safety properties

Correct or wrong?

Let $\mathcal{F}$ be an action-based strong fairness assumption

If $s_1 \sim_{\mathcal{T}} s_2$ then for all **LT** properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

wrong.

If $s_1 \sim_{\mathcal{T}} s_2$ then for all safety properties $E \subseteq (2^{AP})^\omega$:

$$s_1 \models_{\mathcal{F}} E \quad \text{iff} \quad s_2 \models_{\mathcal{F}} E$$

correct.

- realizable fairness irrelevant for safety properties
- strong action-based fairness assumptions are realizable