Overview

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

Equivalences and Abstraction
Model checking

- System: $\mathcal{T}$
- Requirements: $\Phi$

Temporal formula $\Phi$

```
TS $\vdash \Phi$ ?
```
Model checking

heterogeneous approach

system

requirements

TS $\mathcal{T}$

temporal formula $\Phi$

model checking

"$\mathcal{T} \models \Phi$?"

yes

no
Model checking

**heterogeneous approach**

- system
- requirements
- temporal formula $\Phi$

model checking

"$\mathcal{T} \models \Phi$ ?"

yes  no

**homogeneous approach**

- system
- requirements
- $\mathcal{T}_1$
- $\mathcal{T}_2$

model checking

"$\mathcal{T}_1$ impl $\mathcal{T}_2$ ?"

yes  no
Implementation relations

requirements

specification formula $\varphi$

abstract model $\mathcal{T}_2$

refined model $\mathcal{T}_1$
Implementation relations

- **requirements**
- **specification formula** $\varphi$
- **abstract model** $TS \mathcal{T}_2$
- **refined model** $TS \mathcal{T}_1$

**implementation relation** $\text{impl}$

$\mathcal{T}_1 \text{impl} \mathcal{T}_2$ iff “$\mathcal{T}_1$ is a correct implementation of $\mathcal{T}_2$”
Implementation relations

trace inclusion:

\[ \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \]

implementation relation \( \text{impl} \)

\[ \mathcal{T}_1 \text{ impl } \mathcal{T}_2 \text{ iff } "\mathcal{T}_1 \text{ is a correct implementation of } \mathcal{T}_2" \]
Implementation relations

trace inclusion:

\( T_1 \impl T_2 \) iff

\[ \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \]

compatible with LTL,

implementation relation \( \impl \)

\( T_1 \impl T_2 \) iff “\( T_1 \) is a correct implementation of \( T_2 \)”
Implementation relations

trace inclusion:
\[ \text{Traces}(\mathcal{I}_1) \subseteq \text{Traces}(\mathcal{I}_2) \]

compatible with \( \text{LTL} \),
i.e., if \( \mathcal{I}_2 \models \varphi \) then \( \mathcal{I}_1 \models \varphi \)

implementation relation \( \text{impl} \)

\[ \mathcal{I}_1 \text{ impl } \mathcal{I}_2 \text{ iff } \text{“} \mathcal{I}_1 \text{ is a correct implementation of } \mathcal{I}_2 \text{“} \]
Trace equivalence

$\mathcal{T}_1$: $\Rightarrow \emptyset$

$\mathcal{T}_2$: $\Rightarrow \{a\}, \Rightarrow \{b\}$
Trace equivalence

\[ \mathcal{T}_1 : \]

\[ \mathcal{T}_2 : \]

\[ \begin{align*}
\text{Traces}(\mathcal{T}_1) &= \{ \emptyset \emptyset a^\omega, \emptyset \emptyset b^\omega \} = \text{Traces}(\mathcal{T}_1) \\
\end{align*} \]
Trace equivalence

\[ \mathcal{T}_1 : \]

\[ \mathcal{T}_2 : \]

\[
\begin{align*}
\text{Traces}(\mathcal{T}_1) &= \{ \emptyset \emptyset a^\omega, \emptyset \emptyset b^\omega \} = \text{Traces}(\mathcal{T}_1) \\
\text{CTL-formula } \Phi &= \exists \Box (\exists \Box a \land \exists \Box b)
\end{align*}
\]
Trace equivalence

\[ \mathcal{T}_1: \]

\[ \mathcal{T}_2: \]

\[ \text{Traces}(\mathcal{T}_1) = \{ \emptyset \emptyset a^\omega, \emptyset \emptyset b^\omega \} = \text{Traces}(\mathcal{T}_1) \]

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

\( \mathcal{T}_1 \not\models \Phi \) and \( \mathcal{T}_2 \models \Phi \)
Trace equivalence is not compatible with CTL

\[
\mathcal{T}_1:
\]

\[
\mathcal{T}_2:
\]

\[
Traces(\mathcal{T}_1) = \{ \emptyset \emptyset a^\omega, \emptyset \emptyset b^\omega \} = Traces(\mathcal{T}_1)
\]

CTL-formula \( \Phi = \exists\Box(\exists\Box a \land \exists\Box b) \)

\( \mathcal{T}_1 \not\models \Phi \) and \( \mathcal{T}_2 \models \Phi \)
Implementation relations

- for the design of complex systems
  \( \sim \) 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
  ~\rightleftharpoons~ \text{comparison of 2 transition systems}

- for the **analysis** of complex systems
  ~\rightleftharpoons~ \text{homogeneous model checking approach}
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
Implementation relations

- for the **design** of complex systems
  \[\sim\rightarrow\] comparison of 2 transition systems

- for the **analysis** of complex systems
  \[\sim\rightarrow\] homogeneous model checking approach
  \[\sim\rightarrow\] graph minimization

\[\text{use equivalence relation } \sim\text{ for the states of a single transition system } T \text{ and analyze the quotient } T/\sim\]
Implementation relations

- for the **design** of complex systems
  ⇝ comparison of 2 transition systems

- for the **analysis** of complex systems
  ⇝ homogeneous model checking approach
  ⇝ graph minimization

use equivalence relation \( \sim \) for the states of a single transition system \( T \) and analyze the quotient \( T / \sim \)

**goal:** define the equivalence \( \sim \) in such a way that

\[
T \models \Phi \text{ iff } T / \sim \models \Phi
\]

for all “relevant” properties \( \Phi \)
Linear-time implementation relations
finite trace inclusion and equivalence:

\[ \text{Traces}_f(T_1) \subseteq \text{Traces}_f(T_2) \]

trace inclusion and trace equivalence:

\[ \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \]
Linear-time implementation relations

finite trace inclusion and equivalence:
  e.g., $\text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2)$
  preserves all linear-time safety properties

trace inclusion and trace equivalence:
  e.g., $\text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2)$
Linear-time implementation relations

finite trace inclusion and equivalence:
  e.g., $\text{Traces}_{\text{fin}}(T_1) \subseteq \text{Traces}_{\text{fin}}(T_2)$
  preserves all linear-time safety properties

trace inclusion and trace equivalence:
  e.g., $\text{Traces}(T_1) \subseteq \text{Traces}(T_2)$
  preserves all LTL properties
Linear-time implementation relations

finite trace inclusion and equivalence:
  e.g., $\text{Traces}_{\text{fin}}(T_1) \subseteq \text{Traces}_{\text{fin}}(T_2)$
  preserves all linear-time safety properties

trace inclusion and trace equivalence:
  e.g., $\text{Traces}(T_1) \subseteq \text{Traces}(T_2)$
  preserves all LTL properties

* none of the LT relations is compatible with CTL
Linear-time implementation relations

finite trace inclusion and equivalence:
  e.g., $\text{Tracesfin}(\mathcal{T}_1) \subseteq \text{Tracesfin}(\mathcal{T}_2)$
  preserves all linear-time safety properties

trace inclusion and trace equivalence:
  e.g., $\text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2)$
  preserves all LTL properties

∗ none of the LT relations is compatible with CTL
∗ checking LT relations is computationally hard
Linear-time implementation relations

finite trace inclusion and equivalence:

\[ \text{Traces}_{\text{fin}}(\mathcal{T}_1) \subseteq \text{Traces}_{\text{fin}}(\mathcal{T}_2) \]

preserves all linear-time \textcolor{red}{safety} properties

trace inclusion and trace equivalence:

\[ \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \]

preserves all \textcolor{red}{LTL} properties

---

* none of the LT relations is compatible with \textcolor{red}{CTL}
* checking LT relations is \textcolor{red}{computationally hard}
* \textcolor{red}{minimization} ???
Minimization w.r.t. trace equivalence?

$\mathcal{T}_1$: 

$\mathcal{T}_2$: 

BSEQOR5.1-MIN-LT
Minimization w.r.t. trace equivalence?

\( \mathcal{T}_1 \):

\( \mathcal{T}_2 \):

- \( \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \)
Minimization w.r.t. trace equivalence?

\[ \mathcal{T}_1: \]

\[ \mathcal{T}_2: \]

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

but \( \mathcal{T}_1 \) and \( \mathcal{T}_2 \) are not isomorphic
Minimization w.r.t. trace equivalence?

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

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

\[ \mathcal{T}_1: \quad \mathcal{T}_2: \]

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

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

let $T_1 = (S_1, Act_1, \rightarrow_1, S_{0,1}, AP, L_1)$,

$T_2 = (S_2, Act_2, \rightarrow_2, S_{0,2}, AP, L_2)$

be two transition systems
Bisimulation for two transition systems

let \( \mathcal{T}_1 = (S_1, \text{Act}_1, \rightarrow_1, S_{0,1}, AP, L_1) \),
\( \mathcal{T}_2 = (S_2, \text{Act}_2, \rightarrow_2, S_{0,2}, AP, L_2) \)

be two transition systems

- with the same set \( AP \)
Bisimulation for two transition systems

let \( T_1 = (S_1, Act_1, \rightarrow_1, S_{0,1}, AP, L_1) \),
\( T_2 = (S_2, Act_2, \rightarrow_2, S_{0,2}, AP, L_2) \)

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 \( T_1 = (S_1, \text{Act}_1, \rightarrow_1, S_{0,1}, AP, L_1) \), 
\( T_2 = (S_2, \text{Act}_2, \rightarrow_2, S_{0,2}, AP, L_2) \)

be two transition systems

- with the same set \( AP \) \( \leftarrow \) observables
- possibly containing terminal states

Bisimulation equivalence of \( T_1 \) and \( T_2 \) requires that \( T_1 \) and \( T_2 \) 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 \((T_1, T_2)\)

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

\[(1) \quad L_1(s_1) = L_2(s_2)\]
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. \(\forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2)\) 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. \((s'_1, s'_2) \in \mathcal{R}\)

\[\begin{array}{c|c|c}
    s_1 & \mathcal{R} & s_2 \\
    \downarrow & \downarrow & \downarrow \\
    s'_1 & & s'_2
  \end{array}\]

can be completed to

\[\begin{array}{c|c|c}
    s_1 & \mathcal{R} & s_2 \\
    \downarrow & \downarrow & \downarrow \\
    s'_1 & \mathcal{R} & s'_2
  \end{array}\]
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. $\forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2)$ s.t. $(s'_1, s'_2) \in \mathcal{R}$

(3) $\forall s'_2 \in \text{Post}(s_2) \exists s'_1 \in \text{Post}(s_1)$ s.t. $(s'_1, s'_2) \in \mathcal{R}$

\[
\begin{array}{ccc}
\text{s}_1 & \mathcal{R} & \text{s}_2 \\
\downarrow & & \downarrow \\
\text{s}'_1 & & \text{s}'_2
\end{array}
\]

can be completed to

\[
\begin{array}{ccc}
\text{s}_1 & \mathcal{R} & \text{s}_2 \\
\downarrow & & \downarrow \\
\text{s}'_1 & \mathcal{R} & \text{s}'_2
\end{array}
\]
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. \((s'_1, s'_2) \in \mathcal{R}\)

\[
\begin{array}{ccc}
 s_1 & \mathcal{R} & s_2 \\
\downarrow & & \downarrow \\
 s'_1 & \mathcal{R} & s'_2 \\
\end{array}
\]

can be completed to

\[
\begin{array}{ccc}
 s_1 & \mathcal{R} & s_2 \\
\downarrow & & \downarrow \\
 s'_1 & \mathcal{R} & s'_2 \\
\end{array}
\]

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:

4. \(\forall s_{0,1} \in S_{0,1} \exists s_{0,2} \in S_{0,2}\) 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. \(\forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2)\) s.t. \((s'_1, s'_2) \in \mathcal{R}\)

3. \(\forall s'_2 \in \text{Post}(s_2) \exists s'_1 \in \text{Post}(s_1)\) 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 ~
Bisimulation equivalence \( \sim \)

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

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

\[\begin{align*}
(1) & \quad \text{labeling condition} \\
(2) & \quad \text{mutual stepwise simulation} \\
(3) & \quad \text{initial condition}
\end{align*}\]

and initial condition \((I)\)
Bisimulation equivalence ~

bisimulation for \((T_1, T_2)\): relation \(R \subseteq S_1 \times S_2\) s.t.

for all \((s_1, s_2) \in R\):

1. labeling condition
2. \(\) mutual stepwise simulation
3. \(\)

and initial condition \(I\)

bisimulation equivalence ~ for TS:
Bisimulation equivalence \( \sim \)

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

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

\begin{align*}
(1) & \quad \text{labeling condition} \\
(2) & \quad \text{mutual stepwise simulation} \\
(3) & \quad \text{initial condition} \quad (I)
\end{align*}

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. \(s_1 \sim s_2\) labeling condition
2. mutual stepwise simulation
3. \((s_1, s_2) \in \mathcal{R}\)

and initial condition \((I)\)

Bisimulation equivalence \(\sim\) for TS:

\[ \mathcal{T}_1 \sim \mathcal{T}_2 \] if 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 \(\mathcal{T}_2\):

\[ s_1 \sim s_2 \] if 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

\[ T_1 \]
- **pay**
- **select**
- **coke**
- **soda**

\[ T_2 \]
- **pay**
- **select**
- **coke_1**
- **coke_2**
- **soda**

**AP** = \{ **pay**, **coke**, **soda** \}
Two beverage machines

\[ \mathcal{T}_1 \]

\[ \mathcal{T}_2 \]

\[ AP = \{ \text{pay, coke, soda} \} \]
Two beverage machines

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

$AP = \{ \text{pay}, \text{coke}, \text{soda} \}$
Two beverage machines

\[ T_1 \sim T_2 \] as there is a bisimulation for \((T_1, T_2)\):

\[ \mathcal{AP} = \{\text{pay, coke, soda}\} \]
Two beverage machines

\[ T_1 \sim T_2 \]  as there is a bisimulation for \((T_1, T_2)\):

\[ \{ (\text{pay}, \text{pay}), (\text{select}, \text{select}), (\text{soda}, \text{soda}), (\text{coke}, \text{coke}_1), (\text{coke}, \text{coke}_2) \} \]

\[ AP = \{ \text{pay}, \text{coke}, \text{soda} \} \]
Two beverage machines

$\mathcal{T}_1$

$\mathcal{T}_2$

$AP = \{pay, coke, soda\}$
Two beverage machines

\[ \mathcal{T}_1 \]

\[ \begin{align*}
    &\text{pay} \\
    &\text{paid}_1 \rightarrow \text{coke} \\
    &\text{paid}_2 \rightarrow \text{soda} \\
\end{align*} \]

\[ \mathcal{T}_2 \]

\[ \begin{align*}
    &\text{pay} \\
    &\text{select} \rightarrow \text{coke} \\
    &\text{coke} \rightarrow \text{soda} \\
\end{align*} \]

\[ AP = \{\text{pay, coke, soda}\} \]

\[ \mathcal{T}_1 \not\succ \mathcal{T}_2 \]
Two beverage machines

\[ \mathcal{T}_1 \]

\[ \begin{align*} &\text{pay} \\
&\text{paid}_1 \\
&\text{paid}_2 \\
&\text{coke} \\
&\text{soda} \end{align*} \]

\[ \mathcal{T}_2 \]

\[ \begin{align*} &\text{pay} \\
&\text{select} \\
&\text{coke} \\
&\text{soda} \end{align*} \]

\[ \mathcal{AP} = \{\text{pay}, \text{coke}, \text{soda}\} \]

\[ \mathcal{T}_1 \not\sim \mathcal{T}_2 \]

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

can be completed to
Path lifting for bisimulation $\mathcal{R}$
Path lifting for bisimulation $\mathcal{R}$

can be completed to
Path lifting for bisimulation $\mathcal{R}$

- $s_1$ $\not\sim \mathcal{R} \sim$ $s_2$
  - $s_{1,1}$
  - $s_{1,2}$
  - $s_{1,3}$
  - $s_{1,4}$
  - $\vdots$

- $s_1$ $\not\sim \mathcal{R} \sim$ $s_2$
  - $s_{1,1}$
  - $s_{1,2}$
  - $s_{1,3}$
  - $s_{1,4}$
  - $\vdots$

- $s_{1,1}$ $\not\sim \mathcal{R} \sim$ $s_{2,1}$
  - $s_{1,2}$
  - $s_{1,3}$
  - $s_{1,4}$
  - $\vdots$

can be completed to
Path lifting for bisimulation $\mathcal{R}$

$S_1 \xrightarrow{\mathcal{R}} S_2$

\[
\begin{array}{c}
S_1 \\
\downarrow \\
S_{1,1} \\
\downarrow \\
S_{1,2} \\
\downarrow \\
S_{1,3} \\
\downarrow \\
S_{1,4} \\
\downarrow \\
\vdots \\
\end{array}
\]

\[
\begin{array}{c}
S_2 \\
\downarrow \\
S_{2,1} \\
\downarrow \\
S_{2,2} \\
\downarrow \\
\vdots \\
\end{array}
\]

can be completed to
Path lifting for bisimulation $\mathcal{R}$

\[
\begin{array}{ccc}
s_1 & \not\sim & s_2 \\
\downarrow & & \downarrow \\
s_1,1 & \not\sim & s_2 \\
\downarrow & & \downarrow \\
s_1,2 & \not\sim & s_2 \\
\downarrow & & \downarrow \\
s_1,3 & \not\sim & s_2 \\
\downarrow & & \downarrow \\
s_1,4 & \not\sim & s_2 \\
\vdots & & \vdots \\
\end{array}
\]

can be completed to

\[
\begin{array}{ccc}
s_1 & \not\sim & s_2 \\
\downarrow & & \downarrow \\
s_{1,1} & \not\sim & s_{2,1} \\
\downarrow & & \downarrow \\
s_{1,2} & \not\sim & s_{2,2} \\
\downarrow & & \downarrow \\
s_{1,3} & \not\sim & s_{2,3} \\
\downarrow & & \downarrow \\
s_{1,4} & \not\sim & s_{2,4} \\
\vdots & & \vdots \\
\end{array}
\]
Path lifting for bisimulation $\mathcal{R}$

\[ S_1 \xleftarrow{\mathcal{R}} S_2 \]

\[ \downarrow \]

\[ S_{1,1} \]

\[ \downarrow \]

\[ S_{1,2} \]

\[ \downarrow \]

\[ S_{1,3} \]

\[ \downarrow \]

\[ S_{1,4} \]

\[ \downarrow \]

\[ \vdots \]

---

can be completed to

\[ S_1 \xleftarrow{\mathcal{R}} S_2 \]

\[ \downarrow \]

\[ S_{1,1} \xleftarrow{\mathcal{R}} S_{2,1} \]

\[ \downarrow \]

\[ S_{1,2} \xleftarrow{\mathcal{R}} S_{2,2} \]

\[ \downarrow \]

\[ S_{1,3} \xleftarrow{\mathcal{R}} S_{2,3} \]

\[ \downarrow \]

\[ S_{1,4} \xleftarrow{\mathcal{R}} S_{2,4} \]

\[ \downarrow \]

\[ \vdots \]
Properties of bisimulation equivalence
Properties of bisimulation equivalence

\[ \sim \text{ is an equivalence} \]
Properties of bisimulation equivalence

\sim \text{ is an equivalence, i.e.,}

\bullet \text{ reflexivity: } \mathcal{T} \sim \mathcal{T} \text{ for all transition systems } \mathcal{T}
Properties of bisimulation equivalence

\[ \sim \] is an equivalence, i.e.,

- reflexivity: \( T \sim T \) for all transition systems \( T \)

\begin{align*}
\mathcal{R} &= \{ (s, s) : s \in S \} \\
\text{is a bisimulation for } (T, T)
\end{align*}
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 \( \mathcal{T}_2 \sim \mathcal{T}_1 \)
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 \( \mathcal{T}_2 \sim \mathcal{T}_1 \)

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

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 \(\mathcal{T}_1 \sim \mathcal{T}_3\)
Properties of bisimulation equivalence

~ is an equivalence, i.e.,

- reflexivity: \( T \sim T \) for all transition systems \( T \)
- symmetry: \( T_1 \sim T_2 \) implies \( T_2 \sim T_1 \)
- transitivity: if \( T_1 \sim T_2 \) and \( T_2 \sim T_3 \) then \( T_1 \sim T_3 \)

Let \( R_{1,2} \) be a bisimulation for \((T_1, T_2)\), \( R_{2,3} \) be a bisimulation for \((T_2, T_3)\).
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 \((\mathcal{T}_2, \mathcal{T}_3)\).

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

is a bisimulation for \((\mathcal{T}_1, \mathcal{T}_3)\)
Correct or wrong?

BSEQOR5.1-19
Correct or wrong?

---

wrong

BSEQOR5.1-19
Correct or wrong?

\[ s_1 \rightarrow u, \text{ but } s_2 \nrightarrow blue \]  
(\text{thus } s_1 \not\sim s_2)
Correct or wrong?

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

---

---
Correct or wrong?

$s_1 \leadsto u$, but $s_2 \not\leadsto \text{blue}$ (thus $s_1 \not\sim s_2$)

---

---

$s_1 \sim u$, but $s_2 \not\sim \text{blue}$ (thus $s_1 \not\sim s_2$)
Correct or wrong?

\[ s_1 \rightarrow u, \text{ but } s_2 \not\rightarrow \text{blue} \quad \text{(thus } s_1 \not\sim s_2) \]

bisimulation:
\[ \{(w_1, w_2), (w_1', w_2), (s_1, s_2), (s_1, s_2'), (u, x), (u, y)\} \]
Correct or wrong?
Correct or wrong?

BSEQOR5.1-20

correct
Correct or wrong?

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

bisimulation

Correct
Correct or wrong?

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?

\[ \{ (s_1, s_2), (s'_1, s'_2), (s'_1, s''_2), (t_1, t_2), (t'_1, t_2), (t''_1, t_2) \} \]

bisimulation

Correct
Correct or wrong?

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

Correct: \{ (s_1, s_2), (s_1', s_2'), (s_1', s_2''), (t_1, t_2), (t_1', t_2), (t_1'', t_2) \}
Bisimulation vs. trace equivalence
Bisimulation vs. trace equivalence

\[ \mathcal{I}_1 \sim \mathcal{I}_2 \implies \text{Traces}(\mathcal{I}_1) = \text{Traces}(\mathcal{I}_2) \]
Bisimulation vs. trace equivalence

\[ T_1 \sim T_2 \implies \text{Traces}(T_1) = \text{Traces}(T_2) \]

proof: ... path fragment lifting ...
Bisimulation vs. trace equivalence

\[ \mathcal{I}_1 \sim \mathcal{I}_2 \implies \text{Traces}(\mathcal{I}_1) = \text{Traces}(\mathcal{I}_2) \]

Proof: ... path fragment lifting ...

\[ \text{Traces}(\mathcal{I}_1) = \text{Traces}(\mathcal{I}_2) \iff \mathcal{I}_1 \sim \mathcal{I}_2 \]
Bisimulation vs. trace equivalence

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

**proof:** ... path fragment lifting ...

$$\text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \not\iff \mathcal{T}_1 \sim \mathcal{T}_2$$

trace equivalent, but not bisimulation equivalent
Bisimulation vs. trace equivalence

\[ \mathcal{T}_1 \sim \mathcal{T}_2 \implies \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \]

proof: ... path fragment lifting ...

\[ \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \nrightarrow \mathcal{T}_1 \sim \mathcal{T}_2 \]

Trace equivalence is strictly coarser than bisimulation equivalence.
Bisimulation vs. trace equivalence

\[ \mathcal{T}_1 \sim \mathcal{T}_2 \implies \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \]

proof: ... path fragment lifting ...

\[ \text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2) \nRightarrow \mathcal{T}_1 \sim \mathcal{T}_2 \]

Trace equivalence is strictly coarser than bisimulation equivalence.

Bisimulation equivalent transition systems satisfy the same \textbf{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

\[ \mathcal{T}_1 \quad \mathcal{T}_2 \]

- 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

\[ s_1 \sim s_2 \quad \text{iff} \quad T_{s_1} \sim T_{s_2} \]
Bisimulation equivalence ...

- as a relation that compares 2 transition systems

\[ T_1 \sim T_2 \]

- as a relation on the states of 1 transition system

\[ s_1 \sim s_2 \text{ iff } T_{s_1} \sim T_{s_2} \]
Bisimulation equivalence...

- as a relation that compares 2 transition systems

\[ T_1 \sim T_2 \]

- as a relation on the states of 1 transition system

\[ T \]

\[ s_1 \sim s_2 \iff T_{s_1} \sim T_{s_2} \iff \text{there exists a bisimulation } R \text{ for } T \text{ s.t. } (s_1, s_2) \in R \]
Bisimulations on a single TS

Let $T$ be a TS with proposition set $AP$. 
Bisimulations on a single TS

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

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. \( \forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R} \)
3. \( \forall s'_2 \in \text{Post}(s_2) \exists s'_1 \in \text{Post}(s_1) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R} \)
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. $\forall s'_1 \in Post(s_1) \exists s'_2 \in Post(s_2) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R}$
3. $\forall s'_2 \in Post(s_2) \exists s'_1 \in Post(s_1) \text{ s.t. } (s'_1, s'_2) \in \mathcal{R}$

Bisimulation equivalence $\sim_T$: 

$s_1 \sim_T s_2$ iff there exists a bisimulation $\mathcal{R}$ for $\mathcal{T}$ s.t. $(s_1, s_2) \in \mathcal{R}$
Bisimulation equivalence \( \sim_T \) on a single TS

Let \( T \) be a TS with proposition set \( AP \).

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

1. \( L(s_1) = L(s_2) \)
2. \( \forall s'_1 \in \text{Post}(s_1) \exists s'_2 \in \text{Post}(s_2) \text{ s.t. } (s'_1, s'_2) \in R \)
3. \( \forall s'_2 \in \text{Post}(s_2) \exists s'_1 \in \text{Post}(s_1) \text{ s.t. } (s'_1, s'_2) \in R \)

coinductive definition of \( \sim_T \):

\( s_1 \sim_T s_2 \) iff there exists a bisimulation \( R \) for \( T \) s.t. \( (s_1, s_2) \in R \)
Bisimulation equivalence

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

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

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

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

- the coarsest bisimulation on $\mathcal{T}$
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 $s_1 \sim_\mathcal{T} s_2$: 
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 $s_2$: 
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 $s_2$:

\[
\begin{array}{ccc}
  s_1 & \sim_{\mathcal{T}} & s_2 \\
  \downarrow & & \downarrow \\
  s'_1 & \sim_{\mathcal{T}} & s'_2
\end{array}
\]

which can be completed to

\[
\begin{array}{ccc}
  s_1 & \sim_{\mathcal{T}} & s_2 \\
  \downarrow & & \downarrow \\
  s'_1 & \sim_{\mathcal{T}} & s'_2
\end{array}
\]
Two variants of bisimulation equivalence

\( \sim \) relation that compares 2 transition systems

\( \sim_T \) equivalence on the state space of a single TS \( T \)
Two variants of bisimulation equivalence

∼ relation that compares 2 transition systems
∼_T equivalence on the state space of a single TS T

1. ∼_T can be derived from ∼
Two variants of bisimulation equivalence

~ relation that compares 2 transition systems

\(\sim_T\) equivalence on the state space of a single TS \(\mathcal{T}\)

1. \(\sim_T\) can be derived from ~

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

\[s_1 \sim_T s_2\] \(\iff\) \(\mathcal{T}_{s_1} \sim \mathcal{T}_{s_1}\)
Two variants of bisimulation equivalence

\(\sim\) relation that compares 2 transition systems

\(\sim_T\) equivalence on the state space of a single TS \(T\)

1. \(\sim_T\) can be derived from \(\sim\)

| for all states \(s_1\) and \(s_2\) of \(T\):
| \(s_1 \sim_T s_2\) iff \(T_{s_1} \sim T_{s_1}\) |

where \(T_s\) agrees with \(T\), except that state \(s\) is declared to be the unique initial state
Two variants of bisimulation equivalence

relation that compares 2 transition systems

\( \sim \) equivalence on the state space of a single TS \( \mathcal{T} \)

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

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

\[
    s_1 \sim_\mathcal{T} s_2 \iff \mathcal{T}_{s_1} \sim \mathcal{T}_{s_1}
\]

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 \( \sim_T \)

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 $\sim_T$

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 $\sim_T$

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 $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$

(state space $S_1 \cup S_2$)

$\mathcal{T}_1 \sim \mathcal{T}_2$ iff $\forall$ initial states $s_1$ of $\mathcal{T}_1$

$\exists$ initial state $s_2$ of $\mathcal{T}_2$ s.t. $s_1 \sim_T s_2$. 
Derivation of $\sim$ from $\sim_T$

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. $s_1 \sim_T s_2$,
and vice versa
Bisimulation quotient
Let $T = (S, Act, \rightarrow, S_0, AP, L)$ be a TS.
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:

$\mathcal{T} / \sim = (S', Act', \rightarrow', S'_0, AP, L')$
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS.

bisimulation quotient:

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

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

set of bisimulation equivalence classes
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS.

bisimulation quotient:

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

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

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

bisimulation quotient:

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

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

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

bisimulation quotient:

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

- state space: \( S' = S/\sim_T \)
- set of initial states: \( S'_0 = \{ [s]_\sim_T : s \in S_0 \} \)
- labeling function: \( L'([s]_\sim_T) = L(s) \)

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: $S' = S/\sim_T$
- set of initial states: $S_0' = \{[s]_{\sim_T} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_T}) = L(s)$
- transition relation:

$$\frac{s \rightarrow s'}{[s]_{\sim_T} \rightarrow [s']_{\sim_T}}$$
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS.

bisimulation quotient:

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

- state space: $S' = S/\sim_T$
- set of initial states: $S'_0 = \{[s]_{\sim_T} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_T}) = L(s)$
- transition relation:

$[s]_{\sim_T} \rightarrow [s']_{\sim_T}$

action labels irrelevant
Bisimulation quotient

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

bisimulation quotient:

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

- state space: $S' = S/\sim_T$
- set of initial states: $S'_0 = \{[s]_{\sim_T} : s \in S_0\}$
- labeling function: $L'([s]_{\sim_T}) = L(s)$
- transition relation:

\[
\begin{align*}
  s \xrightarrow{\alpha} s' \\
  [s]_{\sim_T} \xrightarrow{\tau} [s']_{\sim_T}
\end{align*}
\]

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

**Bisimulation quotient:**

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

- **state space:** $S' = S / \sim_T$
- **set of initial states:** $S'_0 = \{[s]_{\sim_T} : s \in S_0\}$
- **labeling function:** $L'([s]_{\sim_T}) = L(s)$
- **transition relation:**
  $$\frac{s \xrightarrow{\alpha} s'}{[s]_{\sim_T} \xrightarrow{\tau} [s']_{\sim_T}}$$

$$T \sim T / \sim$$
Example: interleaving of $n$ printers

parallel system $\mathcal{T} = \underbrace{\text{Printer} \ || \ \text{Printer} \ || \ \ldots \ || \ \text{Printer}}_{n \ \text{printer}}$
Example: interleaving of $n$ printers

parallel system $\mathcal{T} = Printer \ || \ Printer \ || \ldots \ || \ Printer$

$n$ printer

transition system for each printer

ready_to_print

is_printing
Example: interleaving of \( n \) printers

parallel system \( \mathcal{T} = \text{Printer} \parallel \text{Printer} \parallel \ldots \parallel \text{Printer} \)

\( n \) printer

\( AP = \{0, 1, \ldots, n\} \) “number of available printers”

transition system for each printer

ready\_to\_print

is\_printing
Example: \( n = 3 \) printers

Parallel system \( T = \text{Printer} \parallel \text{Printer} \parallel \ldots \parallel \text{Printer} \)

\[ AP = \{0, 1, 2, 3\} \]

\( p \): is printing

\( r \): ready to print
Example: \( n = 3 \) printers

parallel system \( \mathcal{T} = \text{Printer} \parallel \cdots \parallel \text{Printer} \)

\( AP = \{0, 1, 2, 3\} \)

\( p \): is printing

\( r \): ready to print
Example: \( n=3 \) printers

parallel system \( \mathcal{I} = \text{Printer} || \text{Printer} || \ldots || \text{Printer} \)

\( AP = \{0, 1, 2, 3\} \)

\( p: \) is printing

\( r: \) ready to print

bisimulation quotient
Example: \( n=3 \) printers

parallel system \( \mathcal{T} = Printer \parallel Printer \parallel \ldots \parallel Printer \)

\( AP = \{0, 1, 2, 3\} \)

\( 2^n \) 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 \( P_1 \):

\[
\text{LOOP FOREVER} \\
\text{noncritical actions} \\
x_1 := x_2 + 1 \\
\text{AWAIT} (x_1 < x_2) \lor (x_2 = 0); \\
critical section; \\
x_1 := 0 \\
\text{END LOOP}
\]

symmetric protocol for \( P_2 \)
protocol for $P_1$:

```
LOOP FOREVER
    noncritical actions
    $x_1 := x_2 + 1$
    AWAIT ($x_1 < x_2$) OR ($x_2 = 0$);
    critical section;
    $x_1 := 0$
END LOOP
```

initially:

- $x_1 = x_2 = 0$

symmetric protocol for $P_2$
Program graphs for the Bakery algorithm

\begin{itemize}
\item \textbf{noncrit}_1
  \begin{align*}
  x_1 &:= x_2 + 1 \\
  \text{wait}_1 &\rightarrow \text{crit}_1 \\
  (x_1 < x_2) \lor (x_2 = 0) \\
  x_1 &:= 0
  \end{align*}
\end{itemize}

\begin{itemize}
\item \textbf{noncrit}_2
  \begin{align*}
  x_2 &:= x_1 + 1 \\
  \text{wait}_2 &\rightarrow \text{crit}_2 \\
  (x_2 < x_1) \lor (x_1 = 0) \\
  x_2 &:= 0
  \end{align*}
\end{itemize}
Transition system for the Bakery algorithm

\[
\begin{align*}
\text{wait}_1: & \quad (x_1 < x_2) \lor (x_2 = 0) \\
\text{crit}_1: & \quad x_1 := x_2 + 1 \\
\text{noncrit}_1: & \quad x_1 := 0 \\
\text{wait}_2: & \quad (x_2 < x_1) \lor (x_1 = 0) \\
\text{crit}_2: & \quad x_2 := x_1 + 1 \\
\text{noncrit}_2: & \quad x_2 := 0
\end{align*}
\]
Transition system for the Bakery algorithm

1. \( x_1 := x_2 + 1 \) from \( \text{wait}_1 \) to \( \text{noncrit}_1 \)
   \( (x_1 < x_2) \lor (x_2 = 0) \) from \( \text{noncrit}_1 \) to \( \text{crit}_1 \)

2. \( x_1 := 0 \) from \( \text{noncrit}_1 \) to \( \text{crit}_1 \)

3. \( x_2 := x_1 + 1 \) from \( \text{wait}_2 \) to \( \text{noncrit}_2 \)
   \( (x_2 < x_1) \lor (x_1 = 0) \) from \( \text{noncrit}_2 \) to \( \text{crit}_2 \)

4. \( x_2 := 0 \) from \( \text{noncrit}_2 \) to \( \text{crit}_2 \)
Transition system for the Bakery algorithm

\[ x_1 := x_2 + 1 \quad \text{or} \quad x_1 := 0 \]

(\( x_1 < x_2 \) \( \lor \) \( x_2 = 0 \))

\[ x_2 := x_1 + 1 \quad \text{or} \quad x_2 := 0 \]

(\( x_2 < x_1 \) \( \lor \) \( x_1 = 0 \))
Transition system for the Bakery algorithm

\[ x_1 := x_2 + 1 \]
\[ (x_1 < x_2) \lor (x_2 = 0) \]

\[ x_1 := 0 \]

\[ x_2 := x_1 + 1 \]
\[ (x_2 < x_1) \lor (x_1 = 0) \]

\[ x_2 := 0 \]
Transition system for the Bakery algorithm

\[ x_1 := x_2 + 1 \]  
\[ x_1 := 0 \]  
\[ (x_1 < x_2) \lor (x_2 = 0) \]  
\[ x_2 := x_1 + 1 \]  
\[ x_2 := 0 \]  
\[ (x_2 < x_1) \lor (x_1 = 0) \]
Transition system for the Bakery algorithm

\[ x_1 := x_2 + 1 \]
\[ x_1 := 0 \]

\[ (x_1 < x_2) \lor (x_2 = 0) \]

\[ x_2 := x_1 + 1 \]
\[ x_2 := 0 \]

\[ (x_2 < x_1) \lor (x_1 = 0) \]
Infinite transition system with a finite bisimulation quotient
Bakery algorithm: bisimulation quotient

\[ x_1 := x_2 + 1 \]

\[ (x_1 < x_2) \lor (x_2 = 0) \]

\[ x_2 := x_1 + 1 \]

\[ (x_2 < x_1) \lor (x_1 = 0) \]
Bakery algorithm: bisimulation quotient

\( x_1 := x_2 + 1 \)

\( \text{noncrit}_1 \)

\( x_1 := 0 \)

\( \text{crit}_1 \)

\( (x_1 < x_2) \lor (x_2 = 0) \)

\( \text{wait}_1 \)

\( x_2 := x_1 + 1 \)

\( \text{noncrit}_2 \)

\( x_2 := 0 \)

\( \text{crit}_2 \)

\( (x_2 < x_1) \lor (x_1 = 0) \)

\( \text{wait}_2 \)

\( x_1 = 0 \)

\( x_2 = 0 \)

\( n_1 \)

\( w_1 \)

\( x_1 = 0 \)

\( x_2 = 0 \)

\( n_2 \)

\( w_2 \)

\( x_1 > 0 \)

\( x_2 = 0 \)

\( c_1 \)

\( w_1 \)

\( x_1 > 0 \)

\( x_2 > 0 \)

\( c_2 \)

\( w_2 \)

\( x_1 > 0 \)

\( x_2 > 0 \)

\( w_1 \)

\( x_2 > 0 \)

\( x_1 > 0 \)
Bakery algorithm: bisimulation quotient

\[ x_1 := x_2 + 1 \]
\[ x_1 := 0 \]
\[ x_2 := x_1 + 1 \]
\[ x_2 := 0 \]

\[ (x_1 < x_2) \lor (x_2 = 0) \]
\[ (x_2 < x_1) \lor (x_1 = 0) \]
Bakery algorithm: bisimulation quotient

\[ x_1 := x_2 + 1 \]

\[ x_1 := 0 \]

\[ (x_1 < x_2) \lor (x_2 = 0) \]

\[ x_2 := x_1 + 1 \]

\[ x_2 := 0 \]

\[ (x_2 < x_1) \lor (x_1 = 0) \]