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
Classification of implementation relations
Classification of implementation relations

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

- **(nonsymmetric) preorders vs. equivalences:**
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
Classification of implementation relations

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

- **(nonsymmetric) preorders vs. equivalences:**
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
Classification of implementation relations

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

- **(nonsymmetric) preorders vs. equivalences:**
  - preorders: trace inclusion, simulation
  - equivalences: trace equivalence, bisimulation

- **strong vs. weak relations**
  - strong: reasoning about all transitions
  - weak: abstraction from stutter steps
Design by stepwise refinement

specification

abstract model
TS $T_1$

refinement
TS $T_2$
Design by stepwise refinement

specification

abstract model TS $\mathcal{T}_1$

refinement TS $\mathcal{T}_2$

transition $s_1 \xrightarrow{\alpha} t_1$
Design by stepwise refinement

- Specification
- Abstract model TS $\mathcal{I}_1$
- Refinement TS $\mathcal{I}_2$
- Transition $s_1 \xrightarrow{\alpha} t_1$
- Execution fragment $s_2 \xrightarrow{} u_1 \xrightarrow{} \ldots \xrightarrow{} u_n \xrightarrow{\alpha} t_2$
Design by stepwise refinement

specification

abstract model
TS $\mathcal{T}_1$

refinement
TS $\mathcal{T}_2$

transition $s_1 \xrightarrow{\alpha} t_1$

execution fragment
$s_2 \rightarrow u_1 \rightarrow \ldots \rightarrow u_n \xrightarrow{\alpha} t_2$

internal computation prior to the execution of action $\alpha$

- access on auxiliary variables of $\mathcal{T}_2$
- no access on variables of $\mathcal{T}_1$
Design by stepwise refinement

\[ AP \subseteq AP_1 \subseteq AP_2 \]

specification

abstract model

\[ TS \mathcal{T}_1 \]

transition \( s_1 \xrightarrow{\alpha} t_1 \)

refinement

\[ TS \mathcal{T}_2 \]

execution fragment

\[ s_2 \rightarrow u_1 \rightarrow \ldots \rightarrow u_n \xrightarrow{\alpha} t_2 \]

internal computation prior to the execution of action \( \alpha \)

- access on auxiliary variables of \( \mathcal{T}_2 \)
- no access on variables of \( \mathcal{T}_1 \)
Design by stepwise refinement

\( \text{AP} \)

\( \subseteq \)

\( \text{AP}_1 \)

\( \subseteq \)

\( \text{AP}_2 \)

specification

abstract model

TS \( \mathcal{T}_1 \)

transition \( s_1 \xrightarrow{\alpha} t_1 \)

eduction fragment

\( s_2 \rightarrow u_1 \rightarrow \ldots \rightarrow u_n \xrightarrow{\alpha} t_2 \)

internal computation prior to the execution of action \( \alpha \)

- access on auxiliary variables of \( \mathcal{T}_2 \)
- no access on variables of \( \mathcal{T}_1 \)

\( s_2 \rightarrow u_1 \rightarrow \ldots \rightarrow u_n \): stutter steps w.r.t. \( \text{AP}_1 \) (or \( \text{AP} \))
Mutual exclusion (with arbiter)

Abstract representation for process $P_i$
Mutual exclusion (with arbiter)

abstract representation for process $P_i$

refined representation for process $P_i$

$noncrit_i$  

$crit_i$

$n_0$

$n_1$

$n_2$

$n_3$

$n_4$

$crit_{i,1}$

$crit_{i,2}$

$crit_{i,3}$

request

release

release

request

request
Example: abstraction from stutter steps

process $P$

```
LOOP FOREVER
  $x := y \bmod 3$
  $y := (x + y) \bmod 3$
  $z := (2y - x) \div 3$
END LOOP
```
Example: abstraction from stutter steps

process $P \rightsquigarrow$ transition system $T_P$

\begin{align*}
\ell_0 & \quad \text{LOOP FOREVER} \\
\ell_1 & \quad x := \text{y MOD 3} \\
\ell_2 & \quad y := (x + y) \text{ MOD 3} \\
\ell_3 & \quad z := (2y - x) \text{ DIV 3} \\
\ell_4 & \quad \text{END LOOP}
\end{align*}
Example: abstraction from stutter steps

process $P \rightsquigarrow$ transition system $\mathcal{T}_P$

\begin{align*}
\ell_0 & \quad \text{LOOP FOREVER} \\
\ell_1 & \quad x := y \mod 3 \\
\ell_2 & \quad y := (x + y) \mod 3 \\
\ell_3 & \quad z := (2y - x) \div 3 \\
\ell_4 & \quad \text{END LOOP}
\end{align*}

$\text{CTL}^*$ property: does $\mathcal{T}_P \models \forall \square \Diamond (z = 1)$ hold?
Example: abstraction from stutter steps

process $P \leadsto$ transition system $\mathcal{T}_P$ over $AP = \text{Eval}(z)$

\begin{align*}
\ell_0 & \quad \text{LOOP FOREVER} \\
\ell_1 & \quad x := y \mod 3 \\
\ell_2 & \quad y := (x + y) \mod 3 \\
\ell_3 & \quad z := (2y - x) \div 3 \\
\ell_4 & \quad \text{END LOOP}
\end{align*}

$CTL^*$ property: does $\mathcal{T}_P \models \forall \Box \Diamond (z = 1)$ hold?
Example: abstraction from stutter steps

process $P \rightsquigarrow$ transition system $\mathcal{T}_P$ over $AP = \text{Eval}(z)$

\begin{align*}
\ell_0 & \quad \text{LOOP FOREVER} \\
\ell_1 & \quad x := y \mod 3 \\
\ell_2 & \quad y := (x + y) \mod 3 \\
\ell_3 & \quad z := (2y - x) \div 3 \\
\ell_4 & \quad \text{END LOOP}
\end{align*}

\textbf{CTL* property:} does $\mathcal{T}_P \models \forall \Box \Diamond (z = 1)$ hold?
Transition system for process $P$

\[
\begin{align*}
&\ell_1 \ x=2 \ y=4 \ z=3 \\
&\ell_2 \ x=1 \ y=4 \ z=3 \\
&\ell_3 \ x=1 \ y=2 \ z=3 \\
&\ell_1 \ x=1 \ y=2 \ z=1 \\
&\ell_2 \ x=2 \ y=2 \ z=1 \\
&\ell_3 \ x=2 \ y=1 \ z=1 \\
&\ell_1 \ x=2 \ y=1 \ z=0 \\
\cdots
\end{align*}
\]
Analysis by abstraction from stutter steps

\[ \ell_1 \ x=2 \ y=4 \ z=3 \]

\[ \ell_2 \ x=1 \ y=4 \ z=3 \]

\[ \ell_3 \ x=1 \ y=2 \ z=3 \]

\[ \ell_1 \ x=1 \ y=2 \ z=1 \]

\[ \ell_2 \ x=2 \ y=2 \ z=1 \]

\[ \ell_3 \ x=2 \ y=1 \ z=1 \]

\[ \ell_1 \ x=2 \ y=1 \ z=0 \]

\[ \ldots \]
Analysis by abstraction from stutter steps

\[ \ell_1 \ x=2 \ y=4 \ z=3 \]

\[ \ell_2 \ x=1 \ y=4 \ z=3 \]

\[ \ell_3 \ x=1 \ y=2 \ z=3 \]

\[ \ell_1 \ x=1 \ y=2 \ z=1 \]

\[ \ell_2 \ x=2 \ y=2 \ z=1 \]

\[ \ell_3 \ x=2 \ y=1 \ z=1 \]

\[ \ell_1 \ x=2 \ y=1 \ z=0 \]

simplified TS representation

\[ z=3 \]

\[ z=1 \]

\[ z=0 \]

\[ \ldots \]
Overview

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

**Equivalences and Abstraction**

- bisimulation, CTL/CTL*-equivalence
- computing the bisimulation quotient
- abstraction stutter steps
  - stutter LT relations
  - stutter bisimulation
- simulation relations
Remind: trace relations
Remind: trace relations

trace equivalence for paths

\( \pi_1, \pi_2 \) are trace equivalent iff  \( \text{trace}(\pi_1) = \text{trace}(\pi_2) \)
Remind: trace relations

trace equivalence for paths

\[ \pi_1, \pi_2 \text{ are trace equivalent iff } \text{trace}(\pi_1) = \text{trace}(\pi_2) \]

trace inclusion for TS:

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

\[ \forall \pi_1 \in \text{Traces}(T_1) \exists \pi_2 \in \text{Traces}(T_2) \text{ s.t. } \pi_1, \pi_2 \text{ are trace equivalent} \]
Remind: trace relations

trace equivalence for paths
\[ \pi_1, \pi_2 \text{ are trace equivalent } \iff \text{trace}(\pi_1) = \text{trace}(\pi_2) \]

trace inclusion for TS:
\[ \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \]
\[ \forall \pi_1 \in \text{Traces}(T_1) \exists \pi_2 \in \text{Traces}(T_2) \]
\[ \text{s.t. } \pi_1, \pi_2 \text{ are trace equivalent} \]

trace equivalence for TS:
\[ \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \land \text{Traces}(T_2) \subseteq \text{Traces}(T_1) \]
Remind: trace relations

trace equivalence for paths

\[ \pi_1, \pi_2 \text{ are trace equivalent iff } \text{trace}(\pi_1) = \text{trace}(\pi_2) \]

trace inclusion for TS: \( \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \)

\[ \forall \pi_1 \in \text{Traces}(\mathcal{T}_1) \exists \pi_2 \in \text{Traces}(\mathcal{T}_2) \]

s.t. \( \pi_1, \pi_2 \) are trace equivalent

trace equivalence for TS:

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

\[ \text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2) \text{ iff for each LT property } E : \]

\[ \mathcal{T}_2 \models E \text{ implies } \mathcal{T}_1 \models E \]
Remind: trace relations

trace equivalence for paths

\( \pi_1, \pi_2 \) are trace equivalent \iff \( \text{trace}(\pi_1) = \text{trace}(\pi_2) \)

trace inclusion for TS: \( \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \)

\( \forall \pi_1 \in \text{Traces}(T_1) \exists \pi_2 \in \text{Traces}(T_2) \)

s.t. \( \pi_1, \pi_2 \) are trace equivalent

trace equivalence for TS:

\( \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \quad \land \quad \text{Traces}(T_2) \subseteq \text{Traces}(T_1) \)

\( \text{Traces}(T_1) \subseteq \text{Traces}(T_2) \) iff for each LTL property \( E \):

\( T_2 \models E \) implies \( T_1 \models E \)

trace equivalent TS satisfy the same LTL formulas
Stutter equivalence for paths

stutter equivalence for infinite path fragments:
Stutter equivalence for paths

Stutter equivalence for infinite path fragments:

\[ \pi_1 \triangleq \pi_2 \quad \text{iff} \quad \text{there exists an infinite word} \]

\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \quad \text{s.t. the} \]

traces of \( \pi_1 \) and \( \pi_2 \) are of the form

\[ A_0 \ldots A_0 A_1 \ldots A_1 A_2 \ldots A_2 \ldots \]
Stutter equivalence for paths

stutter equivalence for infinite path fragments:

\[ \pi_1 \triangleq \pi_2 \iff \text{there exists an infinite word} \]

\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \text{ s.t. the} \]

traces of \( \pi_1 \) and \( \pi_2 \) are of the form

\[ A_0^{n_0} A_1^{n_1} A_2^{n_2} \ldots \]

where \( n_0, n_1, n_2, \ldots \) are natural numbers \( \geq 1 \)
Stutter equivalence for paths

stutter equivalence for infinite path fragments:

\[ \pi_1 \triangleq \pi_2 \quad \text{iff} \quad \text{there exists an infinite word} \]

\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \quad \text{s.t. the} \]

traces of \( \pi_1 \) and \( \pi_2 \) are of the form

\[ A_0^+ A_1^+ A_2^+ \ldots \]
Stutter equivalence for paths

stutter equivalence for infinite path fragments:

\[ \pi_1 \cong \pi_2 \text{ iff there exists an infinite word } \]

\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \text{ s.t. the traces of } \pi_1 \text{ and } \pi_2 \text{ are of the form } \]

\[ A_0^+ A_1^+ A_2^+ \ldots \]

stutter equivalence for finite path fragments:

\[ \hat{\pi}_1 \cong \hat{\pi}_2 \text{ iff there exists a finite word } \]

\[ A_0 A_1 A_2 \ldots A_n \in (2^{AP})^+ \text{ s.t. the traces of } \hat{\pi}_1 \text{ and } \hat{\pi}_2 \text{ are in } \]

\[ A_0^+ A_1^+ A_2^+ \ldots A_n^+ \]
Stutter trace relations for TS

stutter equivalence for infinite path fragments:

\[ \pi_1 \Delta \pi_2 \iff \text{there exists an infinite word} \]
\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \text{ s.t. the} \]
\[ \text{traces of } \pi_1 \text{ and } \pi_2 \text{ are of the form} \]
\[ A_0^+ A_1^+ A_2^+ \ldots \]
stutter trace relations for TS

stutter equivalence for infinite path fragments:

\[ \pi_1 \triangleq \pi_2 \iff \text{there exists an infinite word} \]

\[ A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \text{ s.t. the} \]

traces of \( \pi_1 \) and \( \pi_2 \) are of the form

\[ A_0^+ A_1^+ A_2^+ \ldots \]

stutter trace inclusion for transition systems:

\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \iff \text{for all paths } \pi_1 \text{ of } \mathcal{T}_1 \]

there exists a path \( \pi_2 \) of \( \mathcal{T}_2 \)

s.t. \( \pi_1 \triangleq \pi_2 \)
Example: stutter trace inclusion ⊆

\[ \mathcal{I}_1 \subseteq \mathcal{I}_2 \ \iff \ \forall \pi_1 \in \text{Paths}(\mathcal{I}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{I}_2) \ \text{s.t.} \ \pi_1 \Delta = \pi_2 \]

\[
\begin{align*}
\text{gray} & = \emptyset \\
\text{red} & = \{a\} \\
\text{green} & = \{b\}
\end{align*}
\]
Example: stutter trace inclusion $\trianglerighteq$

\[ I_1 \trianglerighteq I_2 \text{ iff } \forall \pi_1 \in \text{Paths}(I_1) \exists \pi_2 \in \text{Paths}(I_2) \text{ s.t. } \pi_1 \Delta = \pi_2 \]
Example: stutter trace inclusion $\preceq$

$$\mathcal{I}_1 \preceq \mathcal{I}_2 \text{ iff } \forall \pi_1 \in \text{Paths}(\mathcal{I}_1) \exists \pi_2 \in \text{Paths}(\mathcal{I}_2)$$

$$\text{s.t. } \pi_1 \triangleright = \pi_2$$

all traces have the form $$(\emptyset^+\{b\}^++\{a\}^+)^\omega$$

or $$(\emptyset^+\{b\}^++\{a\}^+)^*\emptyset^\omega$$
Stutter trace inclusion and LTL

\[ \mathcal{I}_1 \preceq \mathcal{I}_2 \text{ iff } \forall \pi_1 \in \text{Paths}(\mathcal{I}_1) \exists \pi_2 \in \text{Paths}(\mathcal{I}_2) \]

s.t. \( \pi_1 \triangleright= \pi_2 \)

Does stutter trace inclusion preserve LTL properties?
Stutter trace inclusion and LTL

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \text{ iff } \forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \]
\[ \text{s.t. } \pi_1 \Delta = \pi_2 \]

Does stutter trace inclusion preserve LTL properties?

i.e., for all LTL formulas \( \varphi \):

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \land \mathcal{T}_2 \models \varphi \text{ implies } \mathcal{T}_1 \models \varphi \]
Stutter trace inclusion and LTL

\[ \mathcal{I}_1 \preceq \mathcal{I}_2 \iff \forall \pi_1 \in \text{Paths}(\mathcal{I}_1) \exists \pi_2 \in \text{Paths}(\mathcal{I}_2) \]

s.t. \( \pi_1 \overset{\Delta}{=} \pi_2 \)

Does stutter trace inclusion preserve LTL properties?

i.e., for all LTL formulas \( \varphi \):

\[ \mathcal{I}_1 \preceq \mathcal{I}_2 \land \mathcal{I}_2 \models \varphi \] implies \( \mathcal{I}_1 \models \varphi \)

answer: no
Stutter trace inclusion and LTL

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \text{ iff } \forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \]

s.t. \( \pi_1 \uparrow \Delta \pi_2 \)

Does stutter trace inclusion preserve LTL properties?

i.e., for all LTL formulas \( \varphi \):

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \land \mathcal{T}_2 \models \varphi \text{ implies } \mathcal{T}_1 \models \varphi \]

answer: no

Example: LTL formulas of the form \( \bigcirc a \)
Stutter trace inclusion and LTL

\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \text{ iff } \forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \]

s.t. \( \pi_1 \triangleq \pi_2 \)

Let \( \mathcal{T}_1 \) and \( \mathcal{T}_2 \) are TS without terminal states and \( \varphi \) an LTL formula. Then:

\[ \mathcal{T}_1 \preceq \mathcal{T}_2 \land \mathcal{T}_2 \models \varphi \text{ implies } \mathcal{T}_1 \models \varphi \]
Stutter trace inclusion and LTL\$\bigcirc\$

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \iff \forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \]

s.t. \( \pi_1 \triangleleft \pi_2 \)

Let \( \mathcal{T}_1 \) and \( \mathcal{T}_2 \) are TS without terminal states and \( \varphi \) an LTL\$\bigcirc\$ formula. Then:

\[ \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \land \mathcal{T}_2 \models \varphi \text{ implies } \mathcal{T}_1 \models \varphi \]

where LTL\$\bigcirc\$ = LTL without the next operator \( \bigcirc \)
Stutter trace equivalence $\Delta$ for TS
Stutter trace equivalence \( \triangleq \) for TS

stutter trace inclusion \( \mathcal{T}_1 \triangleleft \mathcal{T}_2 \)

\[ \forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \ \text{s.t.} \ \pi_1 \triangleq \pi_2 \]
Stutter trace equivalence $\trianglerighteq$ for TS

stutter trace inclusion $\mathcal{T}_1 \triangleleft \mathcal{T}_2$

$$\forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \text{ s.t. } \pi_1 \trianglerighteq \pi_2$$

stutter trace equivalence

$$\mathcal{T}_1 \trianglerighteq \mathcal{T}_2 \ \text{iff} \ \mathcal{T}_1 \triangleleft \mathcal{T}_2 \ \text{and} \ \mathcal{T}_2 \triangleleft \mathcal{T}_1$$
Stutter trace equivalence $\Delta$ for TS

stutter trace inclusion $\mathcal{T}_1 \trianglelefteq \mathcal{T}_2$

$$\forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \text{ s.t. } \pi_1 \Delta \pi_2$$

stutter trace equivalence

$$\mathcal{T}_1 \Delta \mathcal{T}_2 \text{ iff } \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \trianglelefteq \mathcal{T}_1$$

kernel of $\trianglelefteq$, i.e.,

coarsest equivalence that refines $\trianglelefteq$
Stutter trace equivalence $\triangleq$ for TS

stutter trace inclusion $\mathcal{T}_1 \triangleleft \mathcal{T}_2$

$\forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \text{ s.t. } \pi_1 \triangleq \pi_2$

For all LTL $\bigcirc$ formulas $\varphi$:

$\mathcal{T}_1 \triangleleft \mathcal{T}_2 \wedge \mathcal{T}_2 \models \varphi$ implies $\mathcal{T}_1 \models \varphi$

stutter trace equivalence

$\mathcal{T}_1 \triangleq \mathcal{T}_2$ iff $\mathcal{T}_1 \triangleleft \mathcal{T}_2$ and $\mathcal{T}_2 \triangleleft \mathcal{T}_1$

kernel of $\triangleleft$, i.e.,

coarsest equivalence that refines $\triangleleft$
Stutter trace equivalence $\Delta$ for TS

stutter trace inclusion $\mathcal{T}_1 \trianglelefteq \mathcal{T}_2$

$$\forall \pi_1 \in \text{Paths}(\mathcal{T}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{T}_2) \text{ s.t. } \pi_1 \overset{\Delta}{=} \pi_2$$

For all LTL\(\bigcirc\) formulas $\varphi$:

$$\mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \land \mathcal{T}_2 \models \varphi \implies \mathcal{T}_1 \models \varphi$$

stutter trace equivalence

$$\mathcal{T}_1 \overset{\Delta}{=} \mathcal{T}_2 \text{ iff } \mathcal{T}_1 \trianglelefteq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \trianglelefteq \mathcal{T}_1$$

If $\mathcal{T}_1 \overset{\Delta}{=} \mathcal{T}_2$ then $\mathcal{T}_1$ and $\mathcal{T}_2$ are LTL\(\bigcirc\) equivalent.
Correct or wrong?

\[
\Delta = \Delta = \Delta = \frac{52}{444}
\]
Correct or wrong?

\[ \Delta \]

\begin{align*}
\text{correct} & \\
\end{align*}
Correct or wrong?

The traces of $\mathcal{T}_1$ and $\mathcal{T}_2$ have the form $\bullet^+ \bullet^+$ or $\bullet^\omega$
Correct or wrong?

The traces of $\mathcal{T}_1$ and $\mathcal{T}_2$ have the form $\bullet + \bullet \ +$ or $\bullet \omega$
Correct or wrong?

The traces of $T_1$ and $T_2$ have the form \( \bullet^{++} \) or \( \bullet\omega \)

\[ \Delta \]

\[ \Delta \]

Correct

\[ \Delta \]

Wrong

\[ \Delta \]

Wrong
Correct or wrong?

The traces of $\mathcal{T}_1$ and $\mathcal{T}_2$ have the form $\bullet + \bullet +$ or $\bullet \omega$

$\mathcal{T}_1$ has a finite trace $\bullet + \bullet$, while $\mathcal{T}_2$ has not
Correct or wrong?

If $\mathcal{T}_1$ and $\mathcal{T}_2$ are TS over $AP$ then:

$\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\mathcal{T}_1 \equiv \mathcal{T}_2$
Correct or wrong?

If $\mathcal{T}_1$ and $\mathcal{T}_2$ are TS over $AP$ then:

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

bisimulation equivalence

stutter trace equivalence
Correct or wrong?

If $\mathcal{T}_1$ and $\mathcal{T}_2$ are TS over $AP$ then:

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

bisimulation equivalence

stutter trace equivalence

correct
If $\mathcal{I}_1$ and $\mathcal{I}_2$ are TS over $AP$ then:

$$\mathcal{I}_1 \sim \mathcal{I}_2 \quad \text{implies} \quad \mathcal{I}_1 \models \mathcal{I}_2$$

bisimulation equivalence

stutter trace equivalence

correct, as

- $\mathcal{I}_1 \sim \mathcal{I}_2$ implies $\text{Traces}(\mathcal{I}_1) = \text{Traces}(\mathcal{I}_2)$
- trace equivalent paths are stutter trace equivalent
Correct or wrong?

If $\mathcal{T}_1$ and $\mathcal{T}_2$ are TS over $AP$ then:

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

bisimulation equivalence

stutter trace equivalence

Correct, as

- $\mathcal{T}_1 \sim \mathcal{T}_2$ implies $\text{Traces}(\mathcal{T}_1) = \text{Traces}(\mathcal{T}_2)$
- trace equivalent paths are stutter trace equivalent

obviously: $\text{Traces}(\mathcal{T}_1) \subseteq \text{Traces}(\mathcal{T}_2)$ implies $\mathcal{T}_1 \preccurlyeq \mathcal{T}_2$
Stutter-insensitive LT properties
stutter equivalence for infinite words
stutter equivalence for infinite words $\sigma_1, \sigma_2 \in (2^{AP})^\omega$:
Stutter-insensitive LT properties

stutter equivalence for infinite words $\sigma_1, \sigma_2 \in (2^{AP})^\omega$:

$\sigma_1 \triangleq \sigma_2 \iff$ there exists an infinite word

$A_0 A_1 A_2 \ldots \in (2^{AP})^\omega$ s.t. $\sigma_1$ and $\sigma_2$

are in $A_0^+ A_1^+ A_2^+ \ldots$
Stutter-insensitive LT properties

Let $E \subseteq (2^{\mathcal{AP}})^\omega$ be an LT property. $E$ is called stutter-insensitive iff for all $\sigma_1, \sigma_2 \in (2^{\mathcal{AP}})^\omega$:

if $\sigma_1 \in E$ and $\sigma_1 \triangleq \sigma_2$ then $\sigma_2 \in E$
Stutter-insensitive LT properties

Stutter equivalence for infinite words $\sigma_1, \sigma_2 \in (2^{AP})^\omega$:

$$\sigma_1 \triangleq \sigma_2 \text{ iff there exists an infinite word } A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \text{ s.t. } \sigma_1 \text{ and } \sigma_2 \text{ are in } A_0^+ A_1^+ A_2^+ \ldots$$

Let $E \subseteq (2^{AP})^\omega$ be an LT property. $E$ is called stutter-insensitive iff for all $\sigma_1, \sigma_2 \in (2^{AP})^\omega$:

if $\sigma_1 \in E$ and $\sigma_1 \triangleq \sigma_2$ then $\sigma_2 \in E$

Example: if $\varphi$ is an $\text{LTL} \setminus \bigcirc$ formula then

$$E = \text{Words}(\varphi)$$ is stutter-insensitive
Let $T_1$, $T_2$ be two TS and $E$ a stutter-insensitive LT-property. Then:

$$T_1 \preceq T_2 \text{ and } T_2 \models E \text{ implies } T_1 \models E$$
Stutter-insensitive LT properties

Let $\mathcal{T}_1$, $\mathcal{T}_2$ be two TS and $E$ a stutter-insensitive LT-property. Then:

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \models E \implies \mathcal{T}_1 \models E$$

Let $\mathcal{T}_1$, $\mathcal{T}_2$ be two TS and $\varphi$ an $\text{LTL} \setminus \Box$ formula.

$$\mathcal{T}_1 \preceq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \models \varphi \implies \mathcal{T}_1 \models \varphi$$
Stutter-insensitive LT properties

Let $\mathcal{T}_1$, $\mathcal{T}_2$ be two TS and $E$ a stutter-insensitive LT-property. Then:

$$\mathcal{T}_1 \triangleq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \models E \implies \mathcal{T}_1 \models E$$

Let $\mathcal{T}_1$, $\mathcal{T}_2$ be two TS and $\varphi$ an $\mathbf{LTL}\setminus\circ$ formula.

$$\mathcal{T}_1 \triangleq \mathcal{T}_2 \text{ and } \mathcal{T}_2 \models \varphi \implies \mathcal{T}_1 \models \varphi$$

remind: if $\varphi$ is an $\mathbf{LTL}\setminus\circ$ formula then

$$E = \text{Words}(\varphi) \text{ is stutter-insensitive}$$
Overview

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

Equivalences and Abstraction

bisimulation, CTL/CTL*-equivalence
computing the bisimulation quotient
abstraction stutter steps
stutter LT relations
stutter bisimulation
simulation relations
Stutter bisimulation

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a TS, possibly with terminal states.
Stutter bisimulation

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

A stutter bisimulation for $\mathcal{T}$ is ....
Stutter bisimulation

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

A *stutter bisimulation* for $\mathcal{T}$ is a binary relation $R$ on $S$ s.t.
Let $T = (S, Act, \rightarrow, S_0, AP, L)$ be a TS, possibly with terminal states.

A **stutter bisimulation** for $T$ is a binary relation $R$ on $S$ s.t. for all $(s_1, s_2) \in R$:

1. labeling condition
2. simulation condition up to stuttering: "$s_2$ can mimick all transitions of of $s_1$"
3. simulation condition up to stuttering: "$s_1$ can mimick all transitions of of $s_2$"
Stutter bisimulation

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

A *stutter bisimulation* for $\mathcal{T}$ is a binary relation $R$ on $S$ s.t. for all $(s_1, s_2) \in R$:

1. **labeling condition:** $L(s_1) = L(s_2)$

2. **simulation condition up to stuttering**
   "$s_2$ can mimic all transitions of $s_1$"

3. **simulation condition up to stuttering**
   "$s_1$ can mimic all transitions of $s_2$"
Stutter bisimulation

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

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

1. labeling condition: $L(s_1) = L(s_2)$

2. simulation condition up to stuttering
   “$s_2$ can mimick all transitions of $s_1$”

3. simulation condition up to stuttering
   “$s_1$ can mimick all transitions of $s_2$”
A stutter bisimulation for \( \mathcal{T} \) is a binary relation \( \mathcal{R} \) on \( S \) s.t. for all \( (s_1, s_2) \in \mathcal{R} \):

\[
\vdash \vdash
\]

(2) simulation condition up to stuttering

\[
s_1 \mathcal{R} s_2
\]
A stutter bisimulation for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $\mathcal{S}$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

\[
\vdash \vdash \vdash
\]

(2) simulation condition up to stuttering

\[s_1 \mathcal{R} s_2\]

with $(s_1', s_2) \not\in \mathcal{R}$
A stutter bisimulation for $\mathcal{I}$ is a binary relation $\mathcal{R}$ on $S$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

\[
\vdots \quad \vdots
\]

(2) simulation condition up to stuttering

\[
s_1 \xrightarrow{\mathcal{R}} s_2
\]

\[
s_1' \xrightarrow{u_1} s_2
\]

\[
s_1' \xrightarrow{\mathcal{R}} s_2
\]

\[
s_1' \xrightarrow{u_n} s_2'
\]

with $(s_1', s_2) \notin \mathcal{R}$
A stutter bisimulation for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $\mathcal{S}$ s.t. for all $(s_1, s_2) \in \mathcal{R}$:

(2) simulation condition up to stuttering

with $(s'_1, s_2) \notin \mathcal{R}$
Stutter bisimulation for a TS

Let $\mathcal{T}$ be a transition system with state space $S$. A **stutter bisimulation** for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $S$ such that for all $(s_1, s_2) \in \mathcal{R}$:

1. $L(s_1) = L(s_2)$
2. for each transition $s_1 \rightarrow s'_1$ with $(s'_1, s_2) \notin \mathcal{R}$ there exists a path fragment $s_2 u_1 u_2 \ldots u_n s'_2$ s.t. . . .
3. . . .
Stutter bisimulation for a TS

Let $T$ be a transition system with state space $S$.

A *stutter bisimulation* for $T$ is a binary relation $R$ on $S$ such that for all $(s_1, s_2) \in R$:

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

2. For each transition $s_1 \rightarrow s_1'$ with $(s_1', s_2) \notin R$ there exists a path fragment $s_2 u_1 u_2 \ldots u_n s_2'$ s.t. $n \geq 0$ and $(s_1, u_i) \in R$ for $1 \leq i \leq n$

3. ...
Let $\mathcal{T}$ be a transition system with state space $S$.

A *stutter bisimulation* for $\mathcal{T}$ is a binary relation $R$ on $S$ such that for all $(s_1, s_2) \in R$:

1. $L(s_1) = L(s_2)$
2. for each transition $s_1 \rightarrow s_1'$ with $(s_1', s_2) \notin R$ there exists a path fragment $s_2 u_1 u_2 \ldots u_n s_2'$ s.t. $n \geq 0$ and $(s_1, u_i) \in R$ for $1 \leq i \leq n$
3. symmetric condition
Stutter bisimulation for a TS

Let $T$ be a transition system with state space $S$.

A \textit{stutter bisimulation} for $T$ is a binary relation $R$ on $S$ such that for all $(s_1, s_2) \in R$:

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

(2) for each transition $s_1 \rightarrow s_1'$ with $(s_1', s_2) \notin R$ there exists a path fragment $s_2 u_1 u_2 \ldots u_n s_2'$ s.t. $n \geq 0$ and $(s_1, u_i) \in R$ for $1 \leq i \leq n$

(3) for each transition $s_2 \rightarrow s_2'$ with $(s_1, s_2') \notin R$ there exists a path fragment $s_1 v_1 v_2 \ldots v_n s_1'$ s.t. $n \geq 0$ and $(v_i, s_2) \in R$ for $1 \leq i \leq n$
Stutter bisimulation equivalence $\approx_T$
Let $\mathcal{T}$ be a transition system with state space $S$.

A *stutter bisimulation* for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $S$ such that for all $(s_1, s_2) \in \mathcal{R}$:

1. labeling condition
2. and (3) mutual simulation condition
Let $\mathcal{T}$ be a transition system with state space $S$.

A \textit{stutter bisimulation} for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $S$ such that for all $(s_1, s_2) \in \mathcal{R}$:

1. labeling condition
2. and (3) mutual simulation condition

\textit{Stutter bisimulation equivalence} $\cong_T$:
Let $\mathcal{T}$ be a transition system with state space $S$.

A stutter bisimulation for $\mathcal{T}$ is a binary relation $\mathcal{R}$ on $S$ such that for all $(s_1, s_2) \in \mathcal{R}$:

(1) labeling condition

(2) and (3) mutual simulation condition

stutter bisimulation equivalence $\approx_{\mathcal{T}}$:

$s_1 \approx_{\mathcal{T}} s_2$ iff there exists a stutter bisimulation $\mathcal{R}$ for $\mathcal{T}$ such that $(s_1, s_2) \in \mathcal{R}$
\(\approx_T\) is an equivalence
≈_T is an equivalence

symmetry: \( s_1 \approx_T s_2 \) then \( s_2 \approx_T s_1 \)
\[ \approx_T \] is an equivalence

symmetry: \( s_1 \approx_T s_2 \) then \( s_2 \approx_T s_1 \)

proof:

if \( \mathcal{R} \) is a stutter bisimulation with \((s_1, s_2) \in \mathcal{R}\) then

\[ \mathcal{R}^{-1} = \{(t_2, t_1) : (t_1, t_2) \in \mathcal{R}\} \]

is a stutter bisimulation that contains \((s_2, s_1)\).
\( \approx_T \) is an equivalence

**Symmetry:** if \( s_1 \approx_T s_2 \) then \( s_2 \approx_T s_1 \)

**Reflexivity:** \( s \approx_T s \) for all states \( s \)
\( \approx_T \) is an equivalence

symmetry: if \( s_1 \approx_T s_2 \) then \( s_2 \approx_T s_1 \)

reflexivity: \( s \approx_T s \) for all states \( s \)

proof:

\[
\mathcal{R} = \{(s, s) : s \in S\} \text{ is a stutter bisimulation}
\]
\( \approx_T \) is an equivalence

**Symmetry:** if \( s_1 \approx_T s_2 \) then \( s_2 \approx_T s_1 \)

**Reflexivity:** \( s \approx_T s \) for all states \( s \)

**Transitivity:** \( s_1 \approx_T s_2 \) and \( s_2 \approx_T s_3 \) implies \( s_1 \approx_T s_3 \)
≈_T is an equivalence

symmetry: if s_1 ≈_T s_2 then s_2 ≈_T s_1

reflexivity: s ≈_T s for all states s

transitivity: s_1 ≈_T s_2 and s_2 ≈_T s_3 implies s_1 ≈_T s_3

Proof: Let R_{1,2} and R_{2,3} be stutter bisimulations s.t.

\((s_1, s_2) \in R_{1,2}, \ (s_2, s_3) \in R_{2,3}\)

Show that R = R_{1,2} \circ R_{2,3} is a stutter bisimulation.
$s_1 \xrightarrow{\mathcal{R}_{1,2}} s_2 \xrightarrow{\mathcal{R}_{2,3}} s_3$

$s'_1$
\[ s_1 \xrightarrow{\mathcal{R}_{1,2}} s_2 \xrightarrow{\mathcal{R}_{2,3}} s_3 \]

\[ s'_1 \xrightarrow{\mathcal{R}_{1,2}} s'_2 \]

\[ u_{j-1} \]
\[ u_j \]
\[ u_{k-1} \]
\[ u_k \]
\[ u_m \]
Stutter bisimulation equivalence

\( \approx_T \) is an equivalence on state space \( S \) of \( T \) such that for all states \( s_1, s_2 \) with \( s_1 \approx_T s_2 \):

1. \( L(s_1) = L(s_2) \)
2. simulation condition up to stuttering

\[ s_1 \approx_T s_2 \]

\[ s'_1 \approx_T s_2 \]

with \( s'_1 \not\approx_T s_2 \)
Stutter bisimulation equivalence

≈_T is the coarsest equivalence on state space S of T such that for all states s_1, s_2 with s_1 ≈_T s_2:

1. \( L(s_1) = L(s_2) \)
2. simulation condition up to stuttering

\[ s_1 \approx_T s_2 \]
\[ s_1' \approx_T s_2 \]

with \( s_1' \not\approx_T s_2 \)
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]

stutter bisimulation with three equivalence classes
Peterson algorithm

protocol for $P_1$

```
LOOP FOREVER
  noncritical section
  $b_1 := \text{true}$; $x := 2$
  AWAIT ($x = 1$) $\lor \neg b_2$
  critical section
  $b_1 := \text{false}$
END LOOP
```
Peterson algorithm

protocol for $P_1$

\[
\text{LOOP FOREVER}
\]
\[
\text{noncritical section}
\]
\[
\begin{align*}
&b_1 := \text{true}; \ x := 2 \\
\text{AWAIT (}x=1\text{)} \lor \neg b_2
\end{align*}
\]
\[
\text{critical section}
\]
\[
\begin{align*}
&b_1 := \text{false} \\
\text{END LOOP}
\end{align*}
\]
Peterson algorithm

protocol for \( P_1 \)

\[
\text{LOOP FOREVER} \\
\text{noncritical section} \\
\quad b_1 := true; \ x := 2 \\
\quad \text{AWAIT } (x=1) \lor \neg b_2 \\
\text{critical section} \\
\quad b_1 := false \\
\text{END LOOP}
\]

protocol for \( P_2 \)

\[
\text{LOOP FOREVER} \\
\text{noncritical section} \\
\quad b_2 := true; \ x := 1 \\
\quad \text{AWAIT } (x=2) \lor \neg b_1 \\
\text{critical section} \\
\quad b_2 := false \\
\text{END LOOP}
\]
TS for the Peterson algorithm

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
TS for the Peterson algorithm

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
TS for the Peterson algorithm

\[ \begin{align*}
\text{AP} &= \{ \text{crit}_1, \text{crit}_2 \} \\
\end{align*} \]
TS for the Peterson algorithm

\[ \text{AP} = \{ \text{crit}_1, \text{crit}_2 \} \]
TS for the Peterson algorithm

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
TS for the Peterson algorithm

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
TS for the Peterson algorithm

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]

9 stutter bisimulation equivalence classes
Stutter bisimulation equivalence for two TS
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

state space $S_1$

transition system $\mathcal{T}_2$

state space $S_2$
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

state space $S_1$

transition system $\mathcal{T}_2$

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $\mathcal{R}$ for $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ such that
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

transition system $\mathcal{T}_2$

state space $S_1$

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $R$

for $\mathcal{T} = \mathcal{T}_1 \uplus \mathcal{T}_2$ such that

$\forall$ initial states $s_1$ of $\mathcal{T}_1 \exists$ initial state $s_2$ of $\mathcal{T}_2$

s.t. $s_1 \approx_{\mathcal{T}} s_2$

$\forall$ initial states $s_2$ of $\mathcal{T}_2 \exists$ initial state $s_1$ of $\mathcal{T}_1$

s.t. $s_1 \approx_{\mathcal{T}} s_2$
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$  

state space $S_1$

transition system $\mathcal{T}_2$  

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $R$ for $(\mathcal{T}_1, \mathcal{T}_2)$
Stutter bisimulation equivalence for two TS

Transition system $\mathcal{T}_1$  

Transition system $\mathcal{T}_2$  

State space $S_1$  

State space $S_2$  

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $R$ for $(\mathcal{T}_1, \mathcal{T}_2)$, i.e., $R \subseteq S_1 \times S_2$ s.t.
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

transition system $\mathcal{T}_2$

state space $S_1$

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$, i.e., $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

(1) if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

state space $S_1$

transition system $\mathcal{T}_2$

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$, i.e., $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

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

(2) and (3) ...
Stutter bisimulation equivalence for two TS

transition system $\mathcal{T}_1$

state space $S_1$

transition system $\mathcal{T}_2$

state space $S_2$

$\mathcal{T}_1 \approx \mathcal{T}_2$ iff there exists a stutter bisimulation $\mathcal{R}$ for $(\mathcal{T}_1, \mathcal{T}_2)$, i.e., $\mathcal{R} \subseteq S_1 \times S_2$ s.t.

1. if $(s_1, s_2) \in \mathcal{R}$ then $L_1(s_1) = L_2(s_2)$
2. and (3) ...
3. $\forall$ initial state $s_1$ of $\mathcal{T}_1$ $\exists$ initial state $s_2$ of $\mathcal{T}_2$ with $(s_1, s_2) \in \mathcal{R}$, and vice versa
Example: door opener

abstract model $\mathcal{T}_1$

$AP = \{\text{closed, open, alarm}\}$
Example: door opener with code no. 181

abstract model $T_1$

refinement $T_2$

$AP = \{\text{closed, open, alarm}\}$
Example: door opener with code no. 181

abstract model $\mathcal{T}_1$

refinement $\text{TS } \mathcal{T}_2$

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

$AP = \{ \text{closed, open, alarm} \}$
Example: door opener with code no. 181

abstract model $\mathcal{T}_1$

wrong code

refinement $\text{TS } \mathcal{T}_2$

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

abstraction from stutter steps:

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

$AP = \{\text{closed, open, alarm}\}$
Correct or wrong?

$\mathcal{T}_1 \approx \mathcal{T}_2$
Correct or wrong?

\[ T_1 \approx T_2 \]

Wrong
Correct or wrong?

\[ T_1 \approx T_2 \]

\[ T_2 \text{ does not contain an equivalent state to } s \text{ and } s' \]
Correct or wrong?

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

Wrong

\[ \mathcal{T}_1 \approx \mathcal{T}_2 \]
Correct or wrong?

\[ T_1 \approx T_2 \]

- [Diagram showing two graphs, one correct, one incorrect]

- [Diagram showing two graphs, one correct, one incorrect]
Correct or wrong?

\[ T_1 \approx T_2 \]

\[ \text{wrong} \]

\[ \text{correct} \]

stutter bisimulation for \((T_1, T_2)\):
\[ \{(s_1, s_2), (t_1, s_2), (u_1, s_2), (w_1, s_2), (v_1, v_2)\} \]
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \approx_T s_2$

remind: $\sim_T$ bisimulation equivalence for $T$

$\approx_T$ stutter bisimulation equivalence for $T$
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \approx_T s_2$

correct

remind: $\sim_T$ bisimulation equivalence for $T$

$\approx_T$ stutter bisimulation equivalence for $T$
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \cong_T s_2$

correct

as $\sim_T$ is a stutter bisimulation for $\mathcal{T}$

remind: $\sim_T$ bisimulation equivalence for $\mathcal{T}$

$\cong_T$ stutter bisimulation equivalence for $\mathcal{T}$
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \approx_T s_2$

correct

as $\sim_T$ is a stutter bisimulation for $\mathcal{T}$

If $s_1 \approx_T s_2$ then $s_1 \sim_T s_2$
Correct or wrong?

If $s_1 \sim T s_2$ then $s_1 \approx T s_2$

correct

as $\sim T$ is a stutter bisimulation for $T$

If $s_1 \approx T s_2$ then $s_1 \sim T s_2$

wrong
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \simeq_T s_2$

correct

as $\sim_T$ is a stutter bisimulation for $\mathcal{T}$

If $s_1 \simeq_T s_2$ then $s_1 \sim_T s_2$

wrong, e.g.:
Correct or wrong?

If $s_1 \sim_T s_2$ then $s_1 \approx_T s_2$

correct

as $\sim_T$ is a stutter bisimulation for $T$

If $s_1 \approx_T s_2$ then $s_1 \sim_T s_2$

wrong, e.g.:

$s_1 \approx_T s_2$

$s_1 \not\sim_T s_2$
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_T s_2$ implies $s_1 \sim_T s_2$. 
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_T s_2$ implies $s_1 \sim_T s_2$

correct
Correct or wrong?

Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_T s_2$ implies $s_1 \sim_T s_2$.

correct, as $\approx_T$ is a bisimulation for $\mathcal{T}$.
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_\mathcal{T} s_2$ implies $s_1 \sim_\mathcal{T} s_2$

**correct**, as $\approx_\mathcal{T}$ is a bisimulation for $\mathcal{T}$

(1) labeling condition: √
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_{\mathcal{T}} s_2$ implies $s_1 \sim_{\mathcal{T}} s_2$

**correct**, as $\approx_{\mathcal{T}}$ is a bisimulation for $\mathcal{T}$

1. labeling condition: $\checkmark$

2. Suppose $s_1 \rightarrow s'_1$. 
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_\mathcal{T} s_2$ implies $s_1 \sim_\mathcal{T} s_2$

**Correct**, as $\approx_\mathcal{T}$ is a **bisimulation** for $\mathcal{T}$

(1) labeling condition: $\checkmark$

(2) Suppose $s_1 \rightarrow s'_1$. Then: $L(s_1) \neq L(s'_1)$
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx^\mathcal{T} s_2$ implies $s_1 \sim^\mathcal{T} s_2$

correct, as $\approx^\mathcal{T}$ is a bisimulation for $\mathcal{T}$

(1) labeling condition: $\checkmark$

(2) Suppose $s_1 \rightarrow s'_1$. Then: $L(s_1) \neq L(s'_1)$

$$
\Rightarrow s_1 \not\approx^\mathcal{T} s'_1
$$
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_T s_2$ implies $s_1 \sim_T s_2$

correct, as $\approx_T$ is a bisimulation for $\mathcal{T}$

(1) labeling condition: $\checkmark$

(2) Suppose $s_1 \rightarrow s'_1$. Then: $L(s_1) \neq L(s'_1)$

$\implies s_1 \not\approx_T s'_1$

$\implies$ there is a path fragment $s_2 u_1 \ldots u_m s'_2$

with $m \geq 0$ and $s_1 \approx_T u_i \land s'_1 \approx_T s'_2$
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_{\mathcal{T}} s_2$ implies $s_1 \sim_{\mathcal{T}} s_2$.

**Correct**, as $\approx_{\mathcal{T}}$ is a bisimulation for $\mathcal{T}$

1. **labeling condition:** √

2. **Suppose** $s_1 \to s'_1$. Then: $L(s_1) \neq L(s'_1)$

   $$\implies s_1 \not\approx_{\mathcal{T}} s'_1$$

   $$\implies$$ there is a path fragment $s_2 u_1 \ldots u_m s'_2$

   with $m \geq 0$ and $s_1 \approx_{\mathcal{T}} u_i$ and $s'_1 \approx_{\mathcal{T}} s'_2$

   $$\implies m = 0.$$
Let $\mathcal{T}$ be a transition system without stutter steps. Then $s_1 \approx_{\mathcal{T}} s_2$ implies $s_1 \sim_{\mathcal{T}} s_2$

correct, as $\approx_{\mathcal{T}}$ is a bisimulation for $\mathcal{T}$

(1) labeling condition: $\checkmark$

(2) Suppose $s_1 \rightarrow s'_1$. Then: $L(s_1) \neq L(s'_1)$

$\Rightarrow$ $s_1 \not\approx_{\mathcal{T}} s'_1$

$\Rightarrow$ there is a path fragment $s_2u_1\ldots u_ms'_2$

with $m \geq 0$ and $s_1 \approx_{\mathcal{T}} u_i \land s'_1 \approx_{\mathcal{T}} s'_2$

$\Rightarrow$ $m=0$. Hence: $s_2 \rightarrow s'_2$ and $s'_1 \approx_{\mathcal{T}} s'_2$
Stutter bisimulation quotient
Stutter bisimulation quotient

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

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

stutter bisimulation quotient of $\mathcal{T}$:

$\mathcal{T}/\approx = (S/\approx_T, \text{Act}', \rightarrow\approx, S'_0, AP, L')$
Stutter bisimulation quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

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

• state space: $S/\approx_T$ ← set of stutter bisimulation equivalence classes
Stutter bisimulation quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

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

- state space: $S/\approx$
- initial states: $S'_0 = \{[s] : s \in S_0\}$

$$[s] = [s]_{\approx_T} = \{s' \in S : s \approx_T s'\}$$
equivalence class of state $s$
Stutter bisimulation quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

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

- state space: $S/\approx_T$
- initial states: $S'_0 = \{[s] : s \in S_0\}$
- labeling: $L'([s]) = L(s)$

$[s] = [s]_{\approx_T} = \{s' \in S : s \approx_T s'\}$
equivalence class of state $s$
Stutter bisimulation quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

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

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

$$s \rightarrow s' \land s \not\approx_{\mathcal{T}} s' \quad \Rightarrow \quad [s] \rightarrow_{\approx} [s']$$
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)$ be a TS.

stutter bisimulation quotient of $\mathcal{T}$:

$\mathcal{T} / \approx = (S / \approx_T, \text{Act}', \rightarrow \approx, S_0', \text{AP}, L')$

- state space: $S / \approx_T$
- initial states: $S_0' = \{ [s] : s \in S_0 \}$
- labeling: $L'([s]) = L(s)$
- transition relation: $s \rightarrow s' \land s \not\approx_T s' \Rightarrow [s] \rightarrow \approx [s']$  
  \hspace{0.4cm} actions irrelevant
Let $\mathcal{T} = (S, \text{Act}, \rightarrow, S_0, AP, L)$ be a TS.

stutter bisimulation quotient of $\mathcal{T}$:

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

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

transition relation:

\[
\begin{array}{c}
s \rightarrow s' \land s \not\approx_{\mathcal{T}} s' \\
\hline
[s] \rightarrow_{\approx} [s']
\end{array}
\]
Equivalence of $\mathcal{T}$ and its quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

$\mathcal{T}/\approx = (S/\approx_T, Act', \rightarrow_\approx, S'_0, AP, L')$

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

transition relation:

$s \rightarrow s' \land s \not\approx_T s' \\
\frac{[s] \rightarrow_\approx [s']}{\mathcal{T} \approx \mathcal{T}/\approx}$
Equivalence of $\mathcal{T}$ and its quotient

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

stutter bisimulation quotient of $\mathcal{T}$:

$$\mathcal{T}/\approx = (S/\approx_T, Act', \rightarrow_\approx, S'_0, AP, L')$$

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

transition relation:

$$s \rightarrow s' \land s \not\approx_T s' \implies [s] \rightarrow_\approx [s']$$

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

is a stutter bisimulation for $(\mathcal{T}, \mathcal{T}/\approx)$
Example: mutual exclusion with semaphore

\[ AP = \{\text{crit}_1, \text{crit}_2\} \]
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]

stutter bisimulation with three equivalence classes
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
Example: mutual exclusion with semaphore

\[ AP = \{ \text{crit}_1, \text{crit}_2 \} \]
Alternating bit protocol

Sender

acknowledgement (bit)

message + bit

Receiver

Timer

Sender

Receiver

Timer
Alternating bit protocol

- formalization by a closed channel system

\[ \text{[Sender \mid Timer \mid Receiver]} \]
Alternating bit protocol

- formalization by a closed channel system

$$[\text{Sender} \mid \text{Timer} \mid \text{Receiver}]$$

- TS with about $$2^{30}$$ states for channels of capacity 10
Alternating bit protocol

Sender

Receiver

Timer

acknowledgement (bit)

message + bit

program graph for sender

generate message(0)

generate message(1)

send(0)

send(1)

d?x

c!0

c!1

lost

lost

timeout!

timeout!

...
Alternating bit protocol

$SMode = 0$  $SMode = 1$  $RMode = 0$  $RMode = 1$
Alternating bit protocol

\[ \Phi \Rightarrow \forall \square \diamond SMode = 0 \land \forall \square \diamond SMode = 1 \]

\[ \text{AP} = \{ SMode = 0, SMode = 1, RMode = 0, RMode = 1 \} \]

SMode = 0    SMode = 1

RMode = 0    RMode = 1
Alternating bit protocol

\[ AP = \{ SMode=0, SMode=1, RMode=0, RMode=1 \} \]

\[ \phi = \forall \Box \Diamond SMode=0 \land \forall \Box \Diamond SMode=1 \]

\[ \text{ABP} \not\models \phi \]
Alternating bit protocol

\[ AP = \{ SMode=0, SMode=1, RMode=0, RMode=1 \} \]

\[ \Phi = \forall \Box \Diamond SMode=0 \land \forall \Box \Diamond SMode=1 \]

\[ ABP \not\models \Phi, \text{ but } ABP/\approx \models \Phi \]

stutter bisimulation quotient
Alternating bit protocol

stutter bisimulation quotient:
Correct or wrong?

If $\mathcal{I}_1 \approx \mathcal{I}_2$ then $\mathcal{I}_1$ and $\mathcal{I}_2$ are $\text{LTL} \setminus \mathcal{O}$-equivalent.
Correct or wrong?

If $\mathcal{T}_1 \approx \mathcal{T}_2$ then $\mathcal{T}_1$ and $\mathcal{T}_2$ are $\text{LTL}\backslash\text{O}$-equivalent.

Wrong.
Correct or wrong?

If $\mathcal{T}_1 \approx \mathcal{T}_2$ then $\mathcal{T}_1$ and $\mathcal{T}_2$ are LTL$\setminus$O-equivalent.

Wrong.

$AP = \{a\}$
If $T_1 \approx T_2$ then $T_1$ and $T_2$ are $\text{LTL}_\emptyset$-equivalent.

wrong.

$AP = \{a\}$
If $\mathcal{T}_1 \approx \mathcal{T}_2$ then $\mathcal{T}_1$ and $\mathcal{T}_2$ are $\text{LTL} \setminus \mathcal{O}$-equivalent.

Wrong.

$$AP = \{a\}$$

$\emptyset^\omega \in \text{Traces}(\mathcal{T}_1)$

$\emptyset^\omega \notin \text{Traces}(\mathcal{T}_2)$
stutter trace equivalence: $\mathcal{I}_1 \preceq \mathcal{I}_2$ iff

\[
\forall \pi_1 \in \text{Paths}(\mathcal{I}_1) \ \exists \pi_2 \in \text{Paths}(\mathcal{I}_2) \ \text{s.t.} \ \pi_1 \preceq \pi_2
\]

\[
\forall \pi_2 \in \text{Paths}(\mathcal{I}_2) \ \exists \pi_1 \in \text{Paths}(\mathcal{I}_1) \ \text{s.t.} \ \pi_1 \preceq \pi_2
\]

stutter bisimulation equivalence $\approx$
\[ \Delta \equiv \text{stutter trace equivalence} \]

\[ \approx \text{stutter bisimulation equivalence} \]
Stutter bisimulation/stutter trace equivalence

\[ \Delta \equiv \text{stutter trace equivalence} \]

\[ \approx \text{stutter bisimulation equivalence} \]
Stutter bisimulation/stutter trace equivalence

\[ \triangleq \text{ stutter trace equivalence} \]

\[ \approx \text{ stutter bisimulation equivalence} \]
Stutter bisimulation/stutter trace equivalence

\[\Delta \approx \Delta \approx \Delta \approx \not\approx \approx \approx \]

\[\Delta \approx \text{stutter trace equivalence} \]

\[\approx \text{stutter bisimulation equivalence} \]
Stutter bisimulation/stutter trace equivalence

\[ \triangleq \]

\[ \cong \]

\[ \not\cong \]

\[ \triangleq \textit{ stutter trace equivalence} \]

\[ \cong \textit{ stutter bisimulation equivalence} \]
Stutter bisimulation/stutter trace equivalence

$\Delta \equiv$ stutter trace equivalence

$\simeq$ stutter bisimulation equivalence
Stutter bisimulation/stutter trace equivalence

\[ \triangleq \quad \text{stutter trace equivalence} \]

\[ \approx \quad \text{stutter bisimulation equivalence} \]

\[ \approx \text{ and } \triangleq \text{ are incomparable} \]