Stutter Trace and Bisimulation Equivalence

Lecture #6 of Advanced Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

E-mail: katoen@cs.rwth-aachen.de

November 9, 2006

Motivation

- Bisimulation, simulation and trace equivalence are *strong*
  - each transition $s \longrightarrow s'$ must be matched by a transition of a related state
  - for comparing models at different abstraction levels, this is too fine
  - consider e.g., modeling an abstract action by a sequence of concrete actions
- Idea: allow for sequences of “invisible” actions
  - each transition $s \longrightarrow s'$ must be matched by a path fragment of a related state
  - matching means: ending in a state related to s', and all previous states invisible
- Abstraction of such internal computations yields coarser quotients
  - but: what kind of properties are preserved?
  - but: can such quotients still be obtained efficiently?
  - but: how to treat infinite internal computations?

Motivating example

Stuttering equivalence

- $s \to s'$ in transition system *TS* is a *stutter step* if $L(s) = L(s')$
  - stutter steps do not affect the state labels of successor states
- Paths π_1 and π_2 are *stuttering equivalent*, denoted $\pi_1 \cong \pi_2$:
  - if there exists an infinite sequence $A_0A_1A_2\ldots$ with $A_i \subseteq AP$ and
  - natural numbers $n_0, n_1, n_2, \ldots, m_0, m_1, m_2, \ldots \geqslant 1$ such that:

$$\begin{aligned} \mathit{trace}(\pi_1) &= \underbrace{A_0 \ldots A_0}_{n_0\text{-times}} \underbrace{A_1 \ldots A_1}_{n_1\text{-times}} \underbrace{A_2 \ldots A_2}_{n_2\text{-times}} \ldots \\ \mathit{trace}(\pi_2) &= \underbrace{A_0, \ldots, A_0}_{m_0\text{-times}} \underbrace{A_1 \ldots A_1}_{m_1\text{-times}} \underbrace{A_2 \ldots A_2}_{m_2\text{-times}} \ldots \end{aligned}$$

$\pi_1 \cong \pi_2$ if their traces only differ in their stutter steps

i.e., if both their traces are of the form $A_0^+A_1^+A_2^+\ldots$ for $A_i \subseteq AP$

Semaphore-based mutual exclusion

Stutter trace equivalence

Transition systems TS_i over AP, $i{=}1,2$, are *stutter-trace equivalent*:

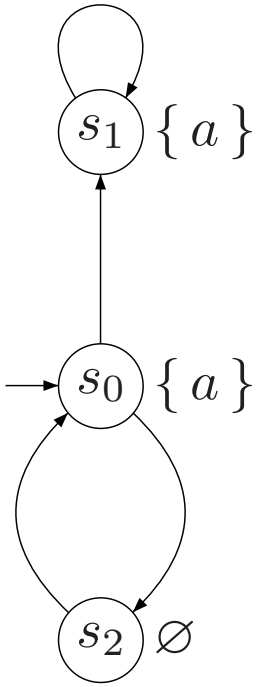
$$TS_1 \cong TS_2 \quad \text{if and only if} \quad TS_1 \sqsubseteq TS_2 \text{ and } TS_2 \sqsubseteq TS_1$$

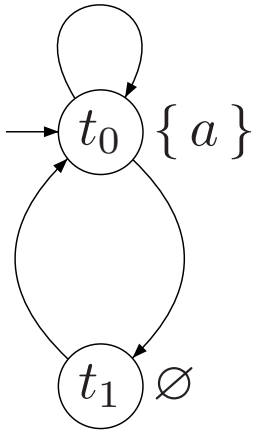
where $\sqsubseteq$ is defined by:

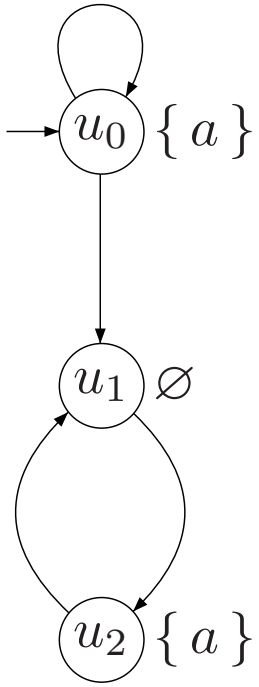
$$TS_1 \sqsubseteq TS_2 \quad \text{iff} \quad \forall \sigma_1 \in \mathit{Traces}(TS_1)\ (\exists \sigma_2 \in \mathit{Traces}(TS_2).\ \sigma_1 \cong \sigma_2\ )$$

clearly: $\mathit{Traces}(TS_1) = \mathit{Traces}(TS_2)$ implies $TS_1 \cong TS_2$, but not always the reverse

Example


s_1 $\{ a \}$
s_0 $\{ a \}$
s_2 $\varnothing$


t_0 $\{ a \}$
t_1 $\varnothing$


u_0 $\{ a \}$
u_1 $\varnothing$
u_2 $\{ a \}$

The $\bigcirc$ operator

Stuttering equivalence does not preserve the validity of next-formulas:

$\sigma_1 = A\,B\,B\,B\ldots$ and $\sigma_2 = A\,A\,A\,B\,B\,B\,B\ldots$ for $A, B \subseteq AP$ and $A \neq B$

Then for $b \in B \setminus A$:

$$\sigma_1 \cong \sigma_2 \quad \text{but} \quad \sigma_1 \models \bigcirc b \quad \text{and} \quad \sigma_2 \not\models \bigcirc b.$$

$\Rightarrow$ a logical characterization of $\cong$ can only be obtained by omitting $\bigcirc$

in fact, it turns out that this is the only modal operator that is not preserved by $\cong$!

Stutter trace and LTL$_{\setminus\bigcirc}$ equivalence

For traces σ_1 and σ_2 over 2^{AP} it holds:

$\sigma_1 \cong \sigma_2 \;\Rightarrow\; (\sigma_1 \models \varphi$ if and only if $\sigma_2 \models \varphi)$

for any LTL$_{\setminus\bigcirc}$ formula φ over *AP*

LTL$_{\setminus\bigcirc}$ denotes the class of LTL formulas without the next step operator $\bigcirc$

Proof

Stutter trace and LTL$_{\setminus\bigcirc}$ equivalence

For transition systems TS_1, TS_2 (over AP) without terminal states:

(a) $TS_1 \cong TS_2$ implies $TS_1 \equiv_{LTL\setminus\bigcirc} TS_2$

(b) if $TS_1 \sqsubseteq TS_2$ then for any LTL$_{\setminus\bigcirc}$ formula φ: $TS_2 \models \varphi$ implies $TS_1 \models \varphi$

A more general result can be established by considering stutter-insensitive LT properties . . . . . .

Stutter insensitivity

- LT property P is *stutter-insensitive* if $[\sigma]_{\cong} \subseteq P$, for any $\sigma \in P$
  - P is stutter insensitive if it is closed under stutter equivalence
- For any stutter-insensitive LT property P:

$$TS_1 \cong TS_2 \quad \text{implies} \quad TS_1 \models P \text{ iff } TS_2 \models P$$

- Moreover: $TS_1 \sqsubseteq TS_2$ and $TS_2 \models P$ implies $TS_1 \models P$
- For any $\text{LTL}_{\setminus\bigcirc}$ formula φ, LT property $\mathit{Words}(\varphi)$ is stutter insensitive
  - but: some stutter insensitive LT properties cannot be expressed in $\text{LTL}_{\setminus\bigcirc}$
  - for LTL formula φ with $\mathit{Words}(\varphi)$ stutter insensitive:

$$\text{there exists } \psi \in \text{LTL}_{\setminus\bigcirc} \text{ such that } \psi \equiv_{LTL} \varphi$$

Stutter bisimulation

Let $TS = (S, Act, \rightarrow, I, AP, L)$ be a transition system and $\mathcal{R} \subseteq S \times S$

$\mathcal{R}$ is a *stutter-bisimulation* for TS if for all $(s_1, s_2) \in \mathcal{R}$:

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

2. if $s_1' \in Post(s_1)$ with $(s_1, s_1') \notin \mathcal{R}$, then there exists a finite path fragment $s_2\, u_1 \ldots u_n\, s_2'$ with $n \geqslant 0$ and $(s_2, u_i) \in \mathcal{R}$ and $(s_1', s_2') \in \mathcal{R}$

3. if $s_2' \in Post(s_2)$ with $(s_2, s_2') \notin \mathcal{R}$, then there exists a finite path fragment $s_1\, v_1 \ldots v_n\, s_1'$ with $n \geqslant 0$ and $(s_1, v_i) \in \mathcal{R}$ and $(s_1', s_2') \in \mathcal{R}$

s_1, s_2 are *stutter-bisimulation equivalent*, denoted $s_1 \approx_{TS} s_2$, if there exists a stutter bisimulation $\mathcal{R}$ for TS with $(s_1, s_2) \in \mathcal{R}$

Stutter bisimulation

$$\begin{array}{ccc} s_1 & \approx & s_2 \\ \downarrow & & \\ s_1' & & \end{array}$$

(with $s_1 \not\approx s_1'$)

can be completed to

$$\begin{array}{ccc} s_1 & \approx & s_2 \\ & & \downarrow \\ s_1 & \approx & u_1 \\ & & \downarrow \\ s_1 & \approx & u_2 \\ & & \downarrow \\ & & \vdots \\ & & \downarrow \\ s_1 & \approx & u_n \\ \downarrow & & \downarrow \\ s_1' & \approx & s_2' \end{array}$$

Semaphore-based mutual exclusion

Stutter-bisimilar transition systems

Let $TS_i = (S_i, Act_i, \rightarrow_i, I_i, AP, L_i)$, $i = 1, 2$, be transition systems over AP

A *stutter bisimulation* for (TS_1, TS_2) is a binary relation $\mathcal{R} \subseteq S_1 \times S_2$ such that:

1. $\mathcal{R}$ and $\mathcal{R}^{-1}$ are stutter-bisimulations for $TS_1 \oplus TS_2$, and

2. $\forall s_1 \in I_1. (\exists s_2 \in I_2. (s_1, s_2) \in \mathcal{R})$ and $\forall s_2 \in I_2. (\exists s_1 \in I_1. (s_1, s_2) \in \mathcal{R})$.

TS_1 and TS_2 are stutter-bisimulation equivalent (stutter-bisimilar, for short), denoted $TS_1 \approx TS_2$, if there exists a stutter bisimulation for (TS_1, TS_2)

Stutter bisimulation quotient

For $TS = (S, Act, \rightarrow, I, AP, L)$ and stutter bisimulation $\approx \subseteq S \times S$ let

$$TS/\approx = (S', \{\tau\}, \rightarrow', I', AP, L'), \quad \text{the } \textit{quotient} \text{ of } TS \text{ under } \approx$$

where

- $S' = S/\approx = \{ [s]_\approx \mid s \in S \}$
- $\rightarrow'$ is defined by: $$\frac{s \xrightarrow{\alpha} s' \text{ and } s \not\approx s'}{[s]_\approx \xrightarrow{\tau}{}' [s']_\approx}$$
- $I' = \{ [s]_\approx \mid s \in I \}$
- $L'([s]_\approx) = L(s)$

note that (a) no self-loops occur in $TS/\approx$ and (b) $TS \approx TS/\approx$ Why?

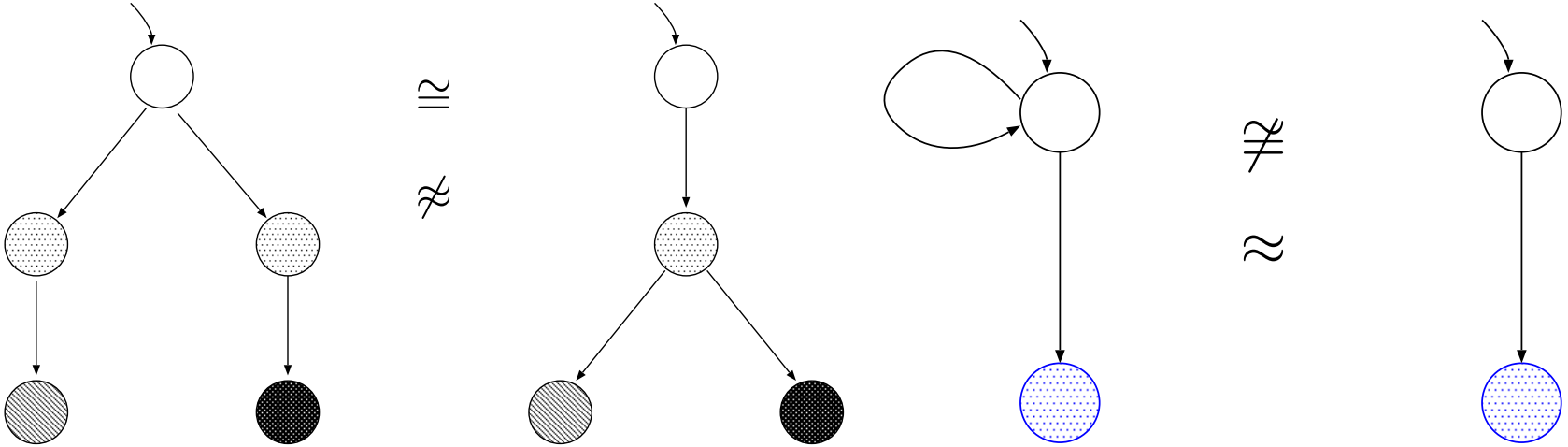
Semaphore-based mutual exclusion

Stutter trace and stutter bisimulation

For transition systems TS_1 and TS_2 over AP:

- Known fact: $TS_1 \sim TS_2$ implies $Traces(TS_1) = Traces(TS_2)$
- But *not*: $TS_1 \approx TS_2$ implies $TS_1 \cong TS_2$!
- So:
  - bisimilar transition systems are trace equivalent
  - but stutter-bisimilar transition systems are not always stutter trace-equivalent!
- Why? Stutter paths!
  - stutter bisimulation does not impose any constraint on such paths
  - but $\cong$ requires the existence of a stuttering equivalent trace

Stutter trace and stutter bisimulation are incomparable



Stutter bisimulation does not preserve LTL$_{\setminus\bigcirc}$

$TS_{left} \approx TS_{right}$ but $TS_{left} \not\models \Diamond a$ and $TS_{right} \models \Diamond a$

Summary

stutter-trace inclusion:

$TS_1 \sqsubseteq TS_2$ iff $\forall \sigma_1 \in \mathit{Traces}(TS_1)\ \exists \sigma_2 \in \mathit{Traces}(TS_2).\ \pi_1 \cong \pi_2$

stutter-trace equivalence:

$TS_1 \cong TS_2$ iff $TS_1 \sqsubseteq TS_2$ and $TS_2 \sqsubseteq TS_1$

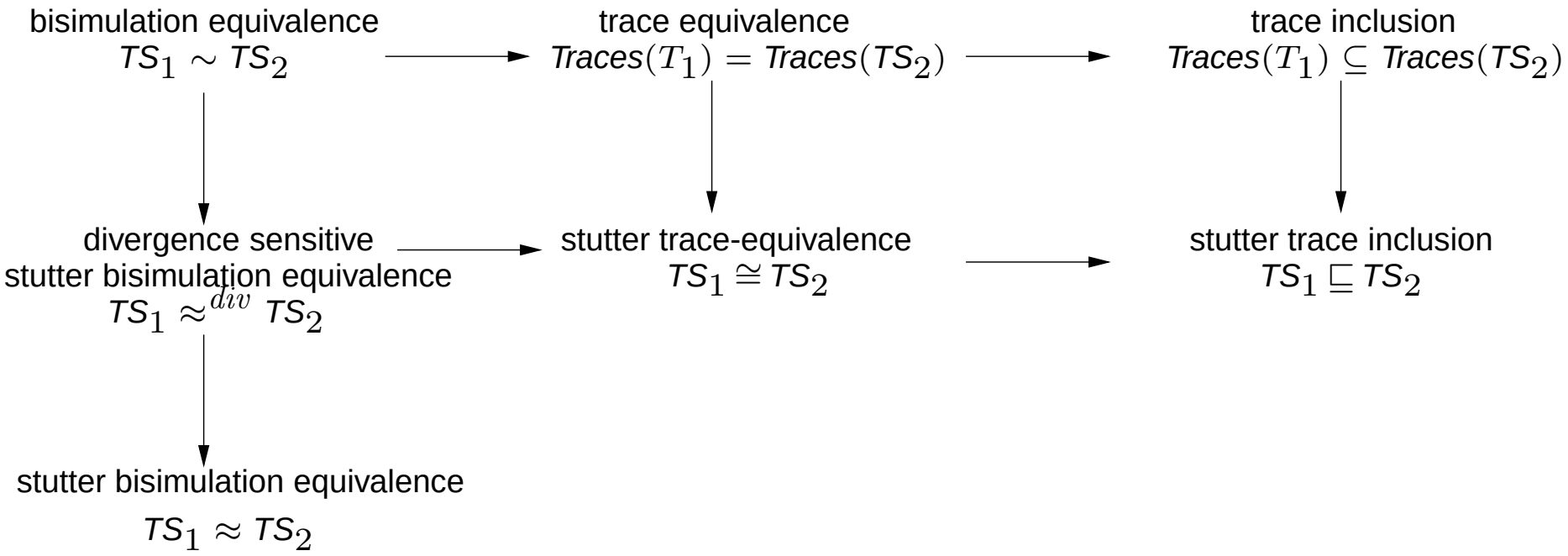
stutter-bisimulation equivalence:

$TS_1 \approx TS_2$ iff there exists a stutter-bisimulation for (TS_1, TS_2)

stutter-bisimulation equivalence with divergence:

$TS_1 \approx^{div} TS_2$ iff there exists a divergence-sensitive stutter bisimulation for (TS_1, TS_2)

Comparison



$\approx^{div}$ will be the topic of the next lecture