Stutter Bisimulation Quotienting

Lecture #6 of Advanced Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

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

May 6, 2009

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?

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

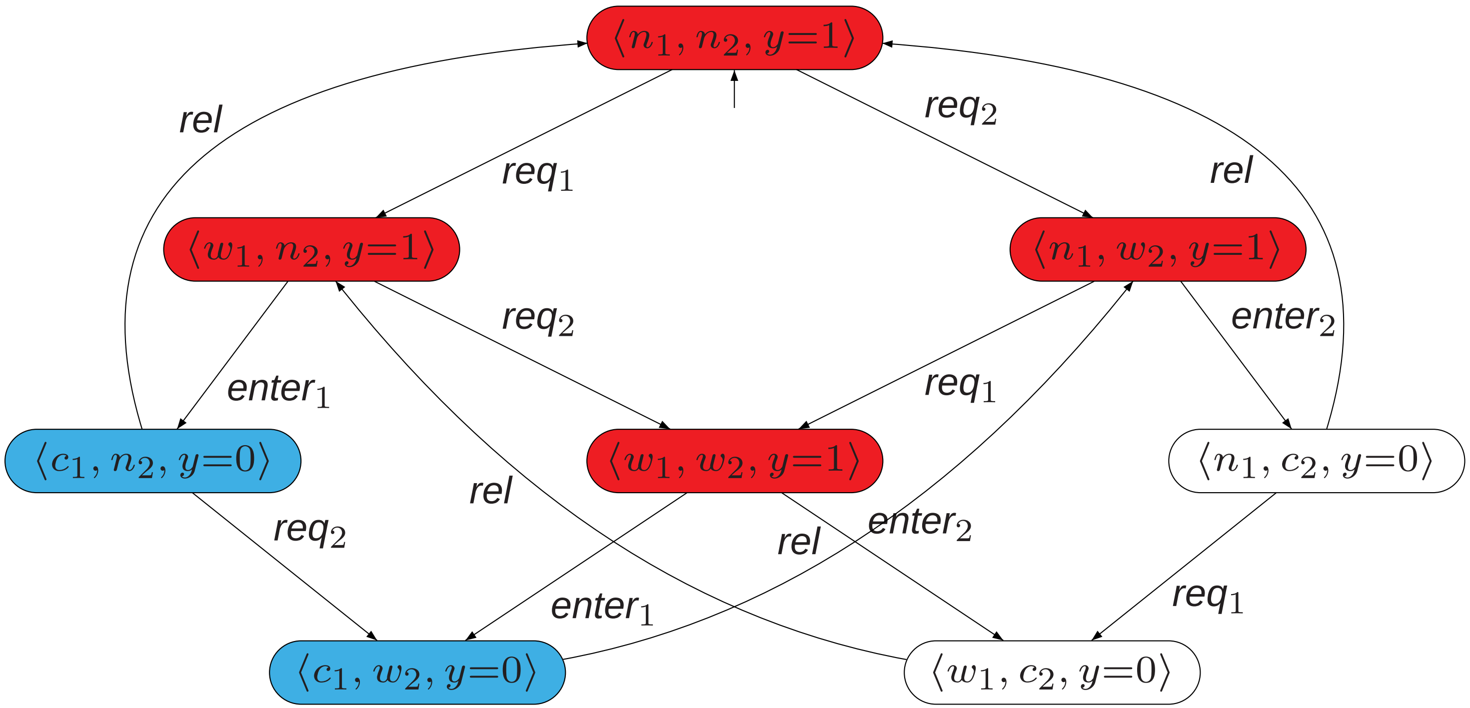
$$\begin{array}{ccc} s_1 & \approx & s_2 \\ \downarrow & & \\ s_1' & & \end{array}$$

(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 states for $AP = \{ crit_1, crit_2 \}$

Stutter-bisimilar transition systems

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

TS_1 and TS_2 are stutter bisimilar, denoted $TS_1 \approx TS_2$, if there exists a stutter bisimulation $\mathcal{R}$ on $TS_1 \oplus TS_2$ such that:

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

Divergence sensitivity

- *Stutter paths* are paths that only consist of stutter steps
  - no restrictions are imposed on such paths by a stutter bisimulation

  $\Rightarrow$ stutter trace-equivalence ($\triangleq$) and stutter bisimulation ($\approx$) are incomparable

  $\Rightarrow$ $\approx$ and $\text{LTL}_{\setminus \bigcirc}$ equivalence are incomparable

- Stutter paths *diverge*: they never leave an equivalence class

- Remedy: only relate *divergent* states or *non-divergent* states
  - divergent state = a state that has a stutter path

  $\Rightarrow$ relate states only if they either both have stutter paths or none of them

- This yields *divergence-sensitive stutter bisimulation* ($\approx^{div}$)

  $\Rightarrow$ $\approx^{div}$ is strictly finer than $\triangleq$ (and $\approx$)

Outlook

formal relation	trace equivalence	bisimulation	simulation
complexity	PSPACE-complete	PTIME	PTIME
logical fragment	LTL	CTL^*	$\forall CTL^*$
preservation	strong	strong match	weak match

formal relation	stutter trace equivalence	divergence-sensitive stutter bisimulation
complexity	PSPACE-complete	PTIME
logical fragment	$LTL_{\setminus \bigcirc}$	$CTL^*_{\setminus \bigcirc}$
preservation	strong	strong match

Divergence sensitivity

Let *TS* be a transition system and $\mathcal{R}$ an equivalence relation on S

- s is *$\mathcal{R}$-divergent* if there exists an infinite path fragment $s\, s_1\, s_2 \ldots \in$ *Paths*(s) such that $(s, s_j) \in \mathcal{R}$ for all $j > 0$
  - s is $\mathcal{R}$-divergent if there is an infinite path starting in s that only visits $[s]_{\mathcal{R}}$
- $\mathcal{R}$ is *divergence sensitive* if for any $(s_1, s_2) \in \mathcal{R}$:

  s_1 is $\mathcal{R}$-divergent implies s_2 is $\mathcal{R}$-divergent

  - $\mathcal{R}$ is divergence-sensitive if in any $[s]_{\mathcal{R}}$ either all or none states are $\mathcal{R}$-divergent

Divergent-sensitive stutter bisimulation

s_1, s_2 are *divergent-sensitive stutter-bisimilar*, denoted $s_1 \approx^{\mathit{div}}_{\mathit{TS}} s_2$, if:

$\exists$ divergent-sensitive stutter bisimulation $\mathcal{R}$ on *TS* such that $(s_1, s_2) \in \mathcal{R}$

$\approx^{div}_{\mathit{TS}}$ is an equivalence, the coarsest divergence-sensitive stutter bisimulation for *TS*

and the union of all divergence-sensitive stutter bisimulations for *TS*

Divergence-sensitive stutter bisimilar paths

For infinite path fragments $\pi_i = s_{0,i}\, s_{1,i}\, s_{2,i} \ldots$, $i = 1, 2$, in *TS*, let:

$$\pi_1 \approx^{\mathit{div}}_{\mathit{TS}} \pi_2$$

if and only if there exists an infinite sequence of indexes

$$0 = j_0 < j_1 < j_2 < \ldots \quad \text{and} \quad 0 = k_0 < k_1 < k_2 < \ldots$$

with:

$$s_{j,1} \approx^{\mathit{div}}_{\mathit{TS}} s_{k,2} \text{ for all } j_{r-1} \leqslant j < j_r \text{ and } k_{r-1} \leqslant k < k_r \text{ with } r = 1, 2, \ldots.$$

State vs path equivalence

Let $\mathit{TS} = (S, \mathit{Act}, \longrightarrow, I, \mathit{AP}, L)$ be a transition system, $s_1, s_2 \in S$. Then:

$$s_1 \approx^{\mathit{div}}_{\mathit{TS}} s_2 \quad \text{implies} \quad \forall \pi_1 \in \mathit{Paths}(s_1).\ \left(\exists \pi_2 \in \mathit{Paths}(s_2).\ \pi_1 \approx^{\mathit{div}}_{\mathit{TS}} \pi_2 \right)$$

Proof

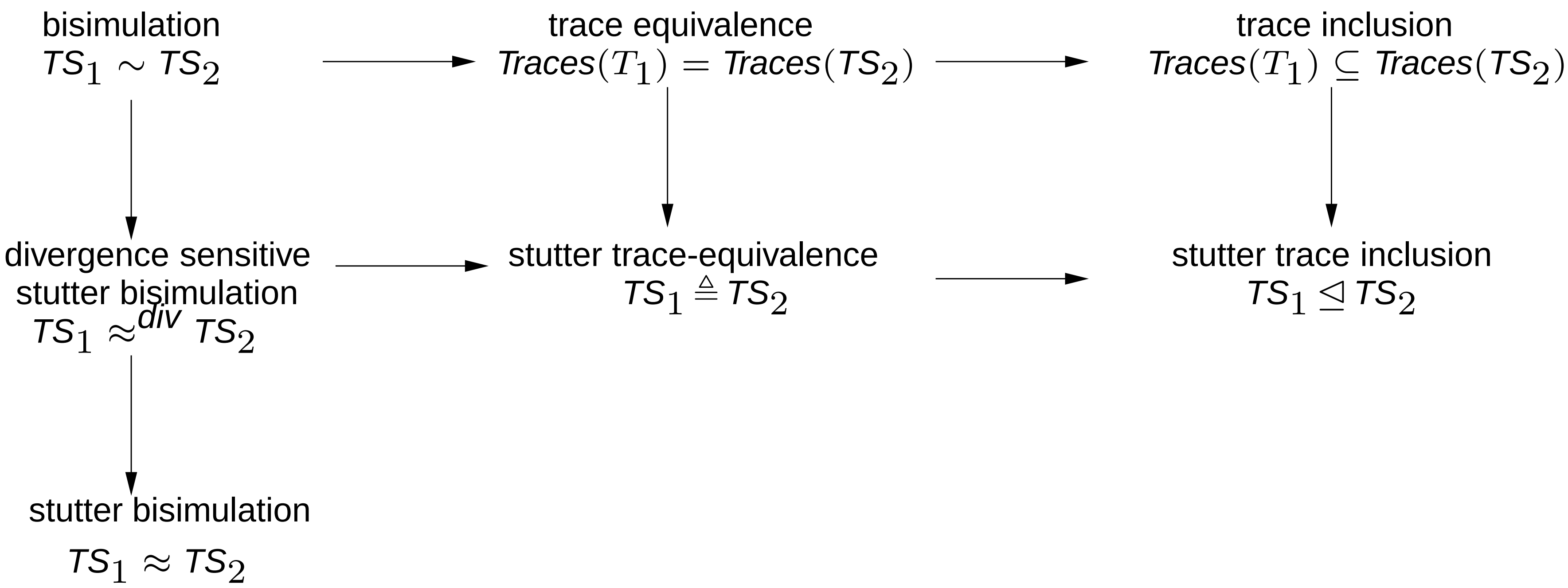
Stutter trace vs stutter bisimulation

Let TS_1 and TS_2 be transition systems over AP. Then:

$$\underbrace{TS_1 \approx^{div} TS_2}_{\text{stutter-bisimulation equivalence with divergence}} \quad \text{implies} \quad \underbrace{TS_1 \triangleq TS_2}_{\text{stutter-trace equivalence}}$$

whereas the reverse implication does not hold in general

Relationship between equivalences



CTL$^*_{\setminus\bigcirc}$ and CTL$_{\setminus\bigcirc}$ equivalence vs $\approx^{div}$

For finite transition system *TS* without terminal states, and s_1, s_2 in *TS*:

$$s_1 \approx^{div}_{TS} s_2 \quad \text{iff} \quad s_1 \equiv_{CTL^*_{\setminus\bigcirc}} s_2 \quad \text{iff} \quad s_1 \equiv_{CTL_{\setminus\bigcirc}} s_2$$

divergent-sensitive stutter bisimulation coincides with CTL$_{\setminus\bigcirc}$ and CTL$^*_{\setminus\bigcirc}$ equivalence

Proof of $\equiv_{CTL_{\setminus\bigcirc}} \subseteq \approx_{TS}^{div}$

A producer-consumer example

Producer

```
in := 0;
while true {
    produce d_1, ..., d_n;
    for i = 1 to n {
    wait until (buffer[in] = ⊥) {
        buffer[in] := d_i;
        in := (in + 1) mod m; }
    }
}
```

Consumer

```
out := 0;
while true {
    for j = 1 to n {
    wait until (buffer[out] ≠ ⊥) {
        e_j := buffer[out];
        buffer[out] := ⊥;
        out := (out + 1) mod m; }
    }
    consume e_1, ..., e_n
}
```

An abstraction

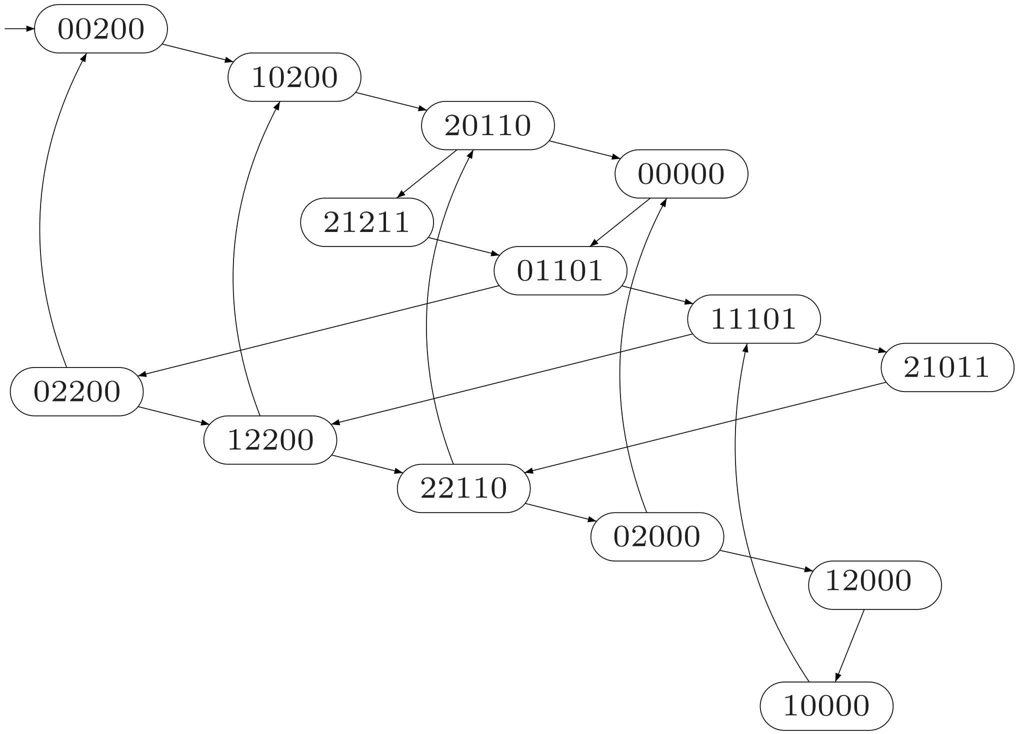
Producer

```
while true {
    produce;
    for i = 1 to n {
    wait until (free > 0) {
        free := free − 1;
    }
}
```

Consumer

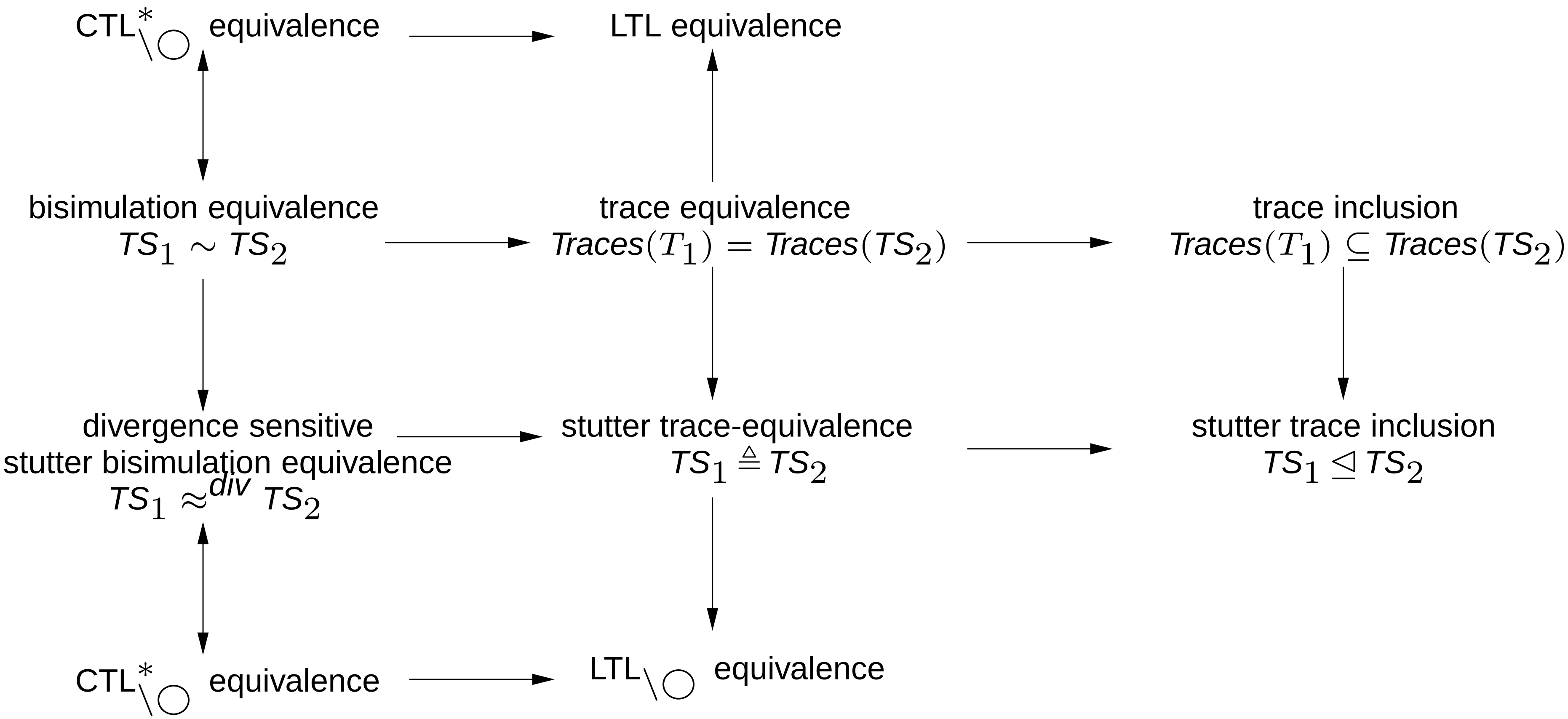
```
while true {
    for j = 1 to n {
    wait until (free < m) {
        free := free + 1;
    }
    consume
}
```

Abstract transition system



$$\ell_0 \quad : \quad produce$$
$$\ell_1 \quad : \quad \langle \textbf{if } (free > 0) \textbf{ then } i := 1; free{-}{-} \textbf{ fi}\rangle$$
$$\ell_2 \quad : \quad \langle \textbf{if } (free > 0) \textbf{ then } i := 0; free{-}{-} \textbf{ fi}\rangle \; ; \; \textbf{goto } \ell_0$$

Equivalences and logical equivalence



Quotienting: Motivation

- Quotienting wrt. $\approx^{div}$ allows to *abstract from stutter steps*
  - in particular $TS \approx^{div} TS/\approx^{div}$
  - typically we have $|TS| \gg |TS/\approx^{div}|$
- $TS_1 \approx^{div} TS_2$ if and only if ($TS_1 \models \Phi$ iff $TS_2 \models \Phi$)
  - for any $\text{CTL}^*_{\setminus \bigcirc}$ (or $\text{CTL}_{\setminus \bigcirc}$) formula Φ

$\Rightarrow$ To check $TS \models \Phi$, if suffices to check whether $TS/\approx^{div} \models \Phi$

  - quotienting with respect to $\approx^{div}$ is a useful preprocessing step of model checking
  - quotienting can be used to determine whether $TS_1 \approx^{div} TS_2$

Quotienting: A two-phase approach

[Groote and Vaandrager, 1990]

1. A quotienting algorithm to determine $TS/\approx$:
   - remove *stutter cycles* from TS
   - a refine operator to *efficiently split* (blocks of) partitions
   - exploit partition-refinement (as for bisimulation $\sim$)

2. A quotienting algorithm to determine $TS/\approx^{div}$:
   - *transform* TS into a (divergence-sensitive) transition system $\overline{TS}$
   - $\overline{TS}$ is divergent-sensitive, i.e., $\approx_{\overline{TS}}$ and $\approx^{div}_{\overline{TS}}$ coincide
   - determine $\overline{TS}/\approx$ using the quotienting algorithm for $\approx$
   - "distill" $TS/\approx^{div}$ from $\overline{TS}/\approx$

Partition-refinement

from now on, we assume that *TS* is finite

- Iteratively compute a partition of S
- Initially: Π_0 equals $\Pi_{AP} = \{ (s,t) \in S \times S \mid L(s) = L(t) \}$ as before
- Repeat until no change: $\boxed{\Pi_{i+1} := \mathit{Refine}_{\approx}(\Pi_i)}$
  - loop invariant: Π_i is coarser than $S/\approx$ and finer than $\{ S \}$
- Return Π_i
  - termination: $\mathcal{R}_{\Pi_0} \supsetneq \mathcal{R}_{\Pi_1} \supsetneq \mathcal{R}_{\Pi_2} \supsetneq \ldots \supsetneq \mathcal{R}_{\Pi_i} = \approx_{TS}$
  - time complexity: maximally $|\, S \,|$ iterations needed

Theorem

> $S/\approx$ is the *coarsest* partition Π of S such that:
>
> (i) Π is finer than the initial partition Π_{AP}, and
>
> (ii) $B \cap \mathit{Pre}(C) = \varnothing$ or $B \subseteq \mathit{Pre}^*_\Pi(C)$ for all $B, C \in \Pi$

for partition Π of S and blocks B, C in Π we have:

$$s \in \mathit{Pre}^*_\Pi(C) \text{ whenever } \underbrace{s = s_1\, s_2 \ldots s_{n-1}}_{\in B}\, \underbrace{s_n}_{\in C} \in \mathit{Paths}(s)$$

state s can reach C via a path that is completely in B (= $[s]_\Pi$)

The refinement operator

- Let: $\mathit{Refine}_{\approx}(\Pi, C) = \bigcup_{B \in \Pi} \mathit{Refine}_{\approx}(B, C)$ for C a block in Π
  - where $\mathit{Refine}_{\approx}(B, C) = \left\{ B \cap \mathit{Pre}(C),\ B \setminus \mathit{Pre}^{*}_{\Pi}(C) \right\} \setminus \{ \varnothing \}$
- Basic properties:
  - for Π finer than Π_{AP} and coarser than $S/\approx$:

    $\mathit{Refine}_{\approx}(\Pi, C)$ is finer than Π and $\mathit{Refine}_{\approx}(\Pi, C)$ is coarser than $S/\approx$
  - Π is strictly coarser than $S/\approx$ if and only if there exists a *splitter* for Π

what is an appropriate splitter for $\approx$?

Splitter for $\approx$

Let Π be a partition of S and let C, $B \in \Pi$.

1. C is a Π-*splitter* for B if and only if:

$$B \neq C \quad \text{and} \quad B \cap \mathit{Pre}(C) \neq \varnothing \quad \text{and} \quad B \setminus \mathit{Pre}^*_\Pi(C) \neq \varnothing$$

2. Π is C-*stable* if there is no $B \in \Pi$ such that C is a Π-splitter for B

3. Π is *stable* if Π is C-stable for all blocks $C \in \Pi$

Partition-refinement

Input: finite transition system *TS* with state space S
Output: stutter-bisimulation quotient space $S/\approx$

$\Pi := \Pi_{AP}$; (* as before *)
while ($\exists B, C \in \Pi.\ C$ is a Π-splitter for B) **do**
  choose such $B, C \in \Pi$;
  $\Pi := (\Pi \setminus \{ B \}) \cup \{ \underbrace{B \cap Pre^*_\Pi(C)}_{B_1}, \underbrace{B \setminus Pre^*_\Pi(C)}_{B_2} \} \setminus \{ \varnothing \}$; (* refine Π *)
od
return Π

Stutter cycles

- $s_0\, s_1\, \ldots \underbrace{s_n}_{=\, s_0}$ is a *stutter cycle* if $s_i\, s_{i+1}$ is a stutter step
- For stutter cycle $s_0\, s_1\, s_2\, \ldots s_n$ in transition system *TS*:

$$s_0 \approx^{\mathit{div}}_{\mathit{TS}} s_1 \approx^{\mathit{div}}_{\mathit{TS}} \ldots \approx^{\mathit{div}}_{\mathit{TS}} s_n$$

- Corollary: for finite *TS* and state s in *TS*:

> s is $\approx^{\mathit{div}}$ –divergent if and only if
>
> a stutter cycle is reachable from s via a path in $[s]_{\mathit{div}}$

⇒ simplify refinement by removing stutter cycles

Removal of stutter cycles: How?

1. Determine the SCCs in $G(\mathit{TS})$ that only contain stutter steps
   - use depth-first search to find these strongly connected components (SCCs)
2. Collapse any stutter SCC into a single state
   - $C \rightarrow' C'$ with $C \neq C'$ whenever $s \rightarrow s'$ in *TS* with $s \in C$ and $s' \in C'$

$\Rightarrow$ Resulting TS' has no stutter cycles

- $s_1 \approx_{\mathit{TS}} s_2$ if and only if $\underbrace{C_1}_{s_1 \in C_1} \approx_{\mathit{TS}'} \underbrace{C_2}_{s_2 \in C_2}$

from now on, assume transition systems have no stutter cycles

A "local" splitter characterization

- C is a Π-*splitter* for B if and only if:

$$B \neq C \quad \text{and} \quad B \cap \mathit{Pre}(C) \neq \varnothing \quad \text{and} \quad B \setminus \mathit{Pre}^*_\Pi(C) \neq \varnothing$$

- How to avoid the computation of $\mathit{Pre}^*_\Pi(C)$ for $C \in \Pi$?

- No stutter cycles $\Rightarrow$ block $B \in \Pi$ has at least one *exit state*
  - exit state = a state with only direct successors outside B:

$$\mathit{Bottom}(B) = \left\{ s \in B \mid \mathit{Post}(s) \cap B = \varnothing \right\}$$

- For finite *TS* without stutter cycles, C is a Π-*splitter* for B iff:

$$B \neq C \quad \text{and} \quad B \cap \mathit{Pre}(C) \neq \varnothing \quad \text{and} \quad \mathit{Bottom}(B) \setminus \mathit{Pre}(C) \neq \varnothing$$

Proof

Time complexity

For *TS* $= (S, \mathit{Act}, \rightarrow, I, \mathit{AP}, L)$ with $M \geqslant |S|$, the $\#$ edges in *TS*:

The partition-refinement algorithm to compute $\mathit{TS}/\approx$

has a worst-case time complexity in $\mathcal{O}\Big(|S| \cdot (|\mathit{AP}| + M) \Big)$

Approach

1. A quotienting algorithm to determine $TS/\approx$:
   - remove *stutter cycles* from TS
   - a refine operator to *efficiently split* (blocks of) partitions
   - exploit partition-refinement (as for bisimulation $\sim$)

$\Rightarrow$ A quotienting algorithm to determine $TS/\approx^{div}$:

- *transform* TS into a (divergence-sensitive) transition system $\overline{TS}$
- $\overline{TS}$ is divergent-sensitive, i.e., $\approx_{\overline{TS}}$ and $\approx^{div}_{\overline{TS}}$ coincide
- determine $\overline{TS}/\approx$ using the quotienting algorithm for $\approx$
- "distill" $TS/\approx^{div}$ from $\overline{TS}/\approx$

Divergence-sensitive stutter bisimulation

Let *TS* be a transition system and $\mathcal{R}$ an equivalence relation on S

- $\mathcal{R}$ is *divergence sensitive* if for any $(s_1, s_2) \in \mathcal{R}$:

  s_1 is $\mathcal{R}$-divergent implies s_2 is $\mathcal{R}$-divergent

  - $\mathcal{R}$ is divergence-sensitive if in any $[s]_{\mathcal{R}}$ either all or none states are $\mathcal{R}$-divergent

- s_1, s_2 in *TS* are *divergent stutter-bisimilar*, denoted $s_1 \approx^{div}_{TS} s_2$, if:
  - $\exists$ divergent-sensitive stutter bisimulation $\mathcal{R}$ on *TS* such that $(s_1, s_2) \in \mathcal{R}$

- *TS* is *divergence sensitive* if $\approx_{TS}$ is so

Quotient transition system under $\approx^{div}$

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

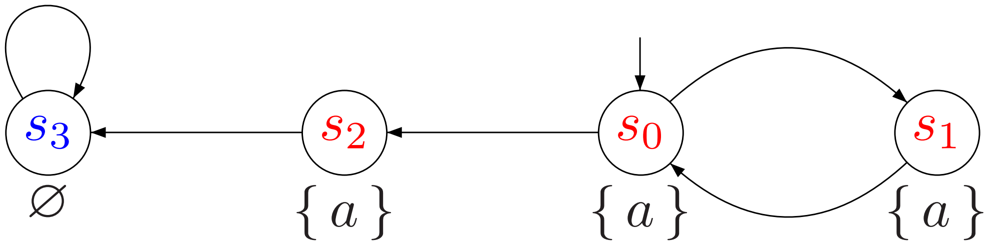
where

- S', I' and L' are defined as usual (for eq. classes $[s]_{div}$ under $\approx^{div}$)
- $\rightarrow'$ is defined by:

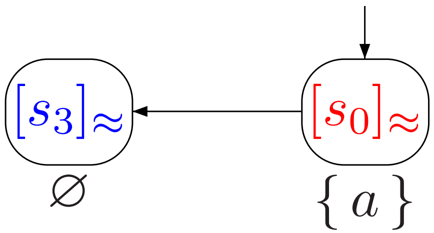
$$\frac{s \xrightarrow{\alpha} s' \;\wedge\; s \not\approx^{div} s'}{[s]_{div} \xrightarrow{\tau}'_{div} [s']_{div}} \quad \text{and} \quad \frac{s \text{ is } \approx^{div}\text{-divergent}}{[s]_{div} \xrightarrow{\tau}'_{div} [s]_{div}}$$

note that $TS \approx^{div} TS/\approx^{div}$

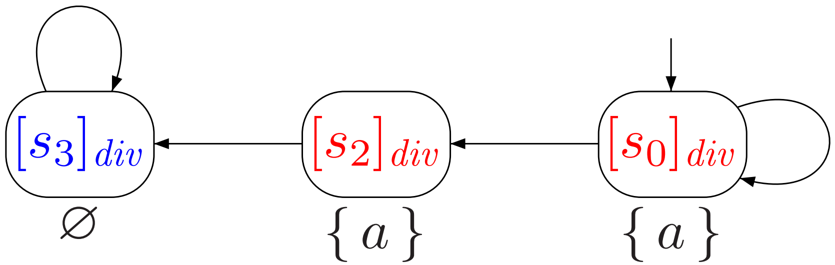
Example



transition system *TS*



transition system $TS/\approx$



transition system $TS/\approx^{div}$

Divergence expansion

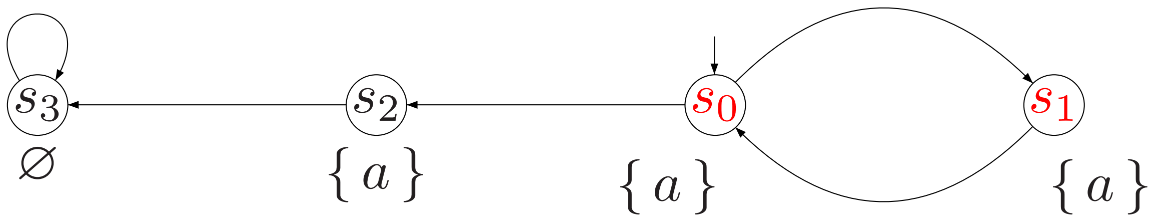
Divergence-sensitive expansion of finite $TS = (S, Act, \rightarrow, I, AP, L)$ is:

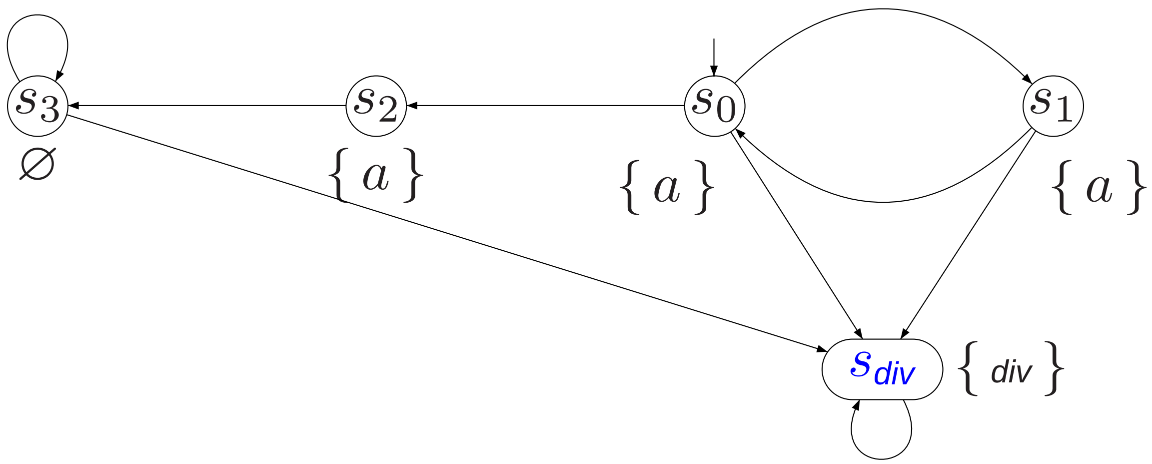
$$\overline{TS} \;=\; \left(S \cup \{ s_{div} \}, Act \cup \{ \tau \}, \rightarrow, I, AP \cup \{ div \}, \overline{L}\right) \quad \text{where}$$

- $s_{div} \notin S$
- $\rightarrow$ extends the transition relation of *TS* by:
  - $s_{div} \xrightarrow{\tau} s_{div}$ and
  - $s \xrightarrow{\tau} s_{div}$ for every state $s \in S$ on a stutter cycle in *TS*
- $\overline{L}(s) = L(s)$ if $s \in S$ and $\overline{L}(s_{div}) = \{ div \}$

$s_{div} \not\approx s$ for any $s \in S$ and s_{div} can only be reached from a $\approx^{div}$-divergent state

Example





Correctness

For finite transition system *TS*:

1. $\overline{TS}$ is divergence-sensitive, and

2. for all $s_1, s_2 \in S$: $s_1 \approx_{TS}^{div} s_2$ if and only if $s_1 \approx_{\overline{TS}} s_2$

Proof

Recipe for computing $TS/\approx^{div}$

1. Construct the divergence-sensitive expansion $\overline{TS}$
   - determine the SCCs in $G_{stutter}(TS)$, and insert transitions $s_{div} \longrightarrow s_{div}$ and
   - $s \longrightarrow s_{div}$ for any state s in a non-trivial SCC of $G_{stutter}$
2. Apply partition-refinement to $\overline{TS}$ to obtain $S/\approx^{div}_{TS} = S/\approx_{\overline{TS}}$
3. Generate $\overline{TS}/\approx$
   - any $C \in S/\approx^{div}$ that contains an initial state of TS is an initial state
   - the labeling of $C \in S/\approx^{div}$ equals the labeling of any $s \in C$
   - any transition $s \longrightarrow s'$ with $s \not\approx^{div}_{TS} s'$ yields a transition between C_s and $C_{s'}$
4. "Distill" $TS\approx^{div}$ from $\overline{TS}/\approx$:
   - replace transition $s \longrightarrow s_{div}$ in $\overline{TS}$ by the self-loop $[s]_{div} \longrightarrow [s]_{div}$
   - delete state s_{div}

Example

Time complexity

For $\mathit{TS} = (S, \mathit{Act}, \rightarrow, I, \mathit{AP}, L)$ with $M \geqslant |S|$, the $\#$ edges in TS:

> The quotient transition system $\mathit{TS}/\approx^{\mathit{div}}$ can be determined
>
> with a worst-case time complexity in $\mathcal{O}\Big(|S|+M \ + \ |S| \cdot (|\mathit{AP}|+M)\Big)$

Summary

formal relation	trace equivalence	bisimulation	simulation
complexity	PSPACE-complete	$\mathcal{O}(M \cdot \log \lvert S \rvert)$	$\mathcal{O}(M \cdot \lvert S \rvert)$
logical fragment	LTL	CTL*	$\forall$CTL*
preservation	strong	strong match	weak match

formal relation	stutter trace equivalence	divergence-sensitive stutter bisimulation
complexity	PSPACE-complete	$\mathcal{O}(M \cdot \lvert S \rvert)$
logical fragment	LTL$_{\setminus \bigcirc}$	CTL$^*_{\setminus \bigcirc}$
preservation	strong	strong match