Divergence-Sensitive Stutter Bisimulation

Lecture #7 of Advanced Model Checking

Joost-Pieter Katoen

Lehrstuhl 2: Software Modeling & Verification

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

November 16, 2006

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

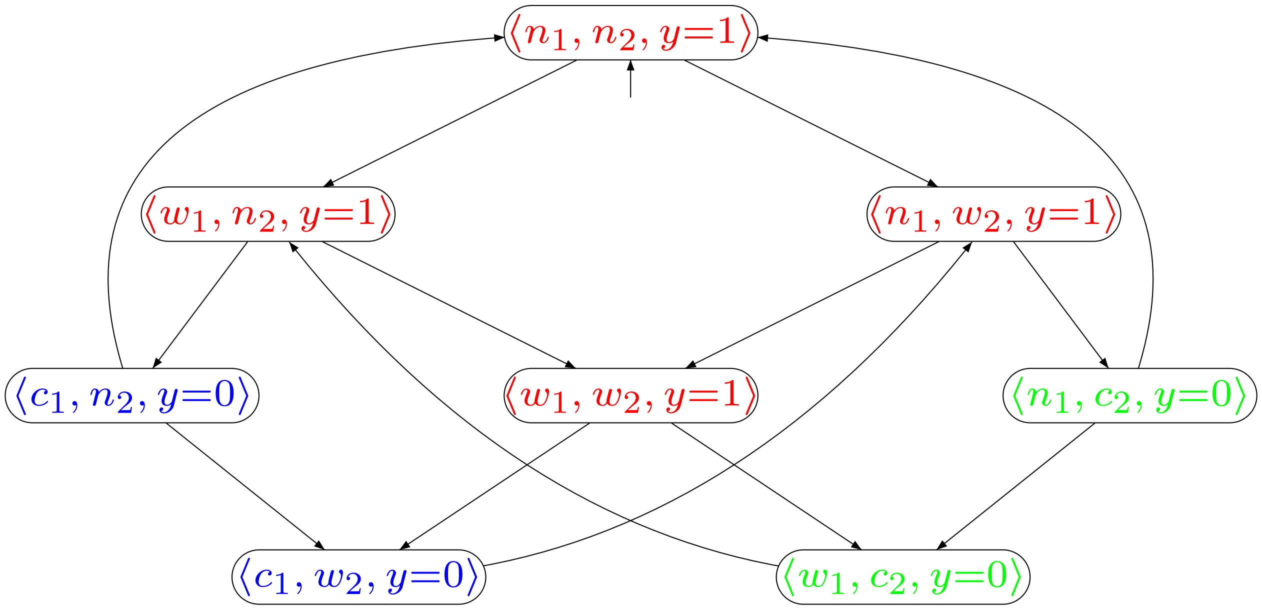
$$\begin{array}{ccc} s_1 & \approx_{TS} & s_2 \\ \downarrow & & \\ s'_1 & & \end{array}$$

(with $s_1 \not\approx_{TS} s'_1$)

can be completed to

$$\begin{array}{ccc} s_1 & \approx_{TS} & s_2 \\ & & \downarrow \\ s_1 & \approx_{TS} & u_1 \\ & & \downarrow \\ s_1 & \approx_{TS} & u_2 \\ & & \downarrow \\ & & \vdots \\ & & \downarrow \\ s_1 & \approx_{TS} & u_n \\ \downarrow & & \downarrow \\ s'_1 & \approx_{TS} & s'_2 \end{array}$$

Example

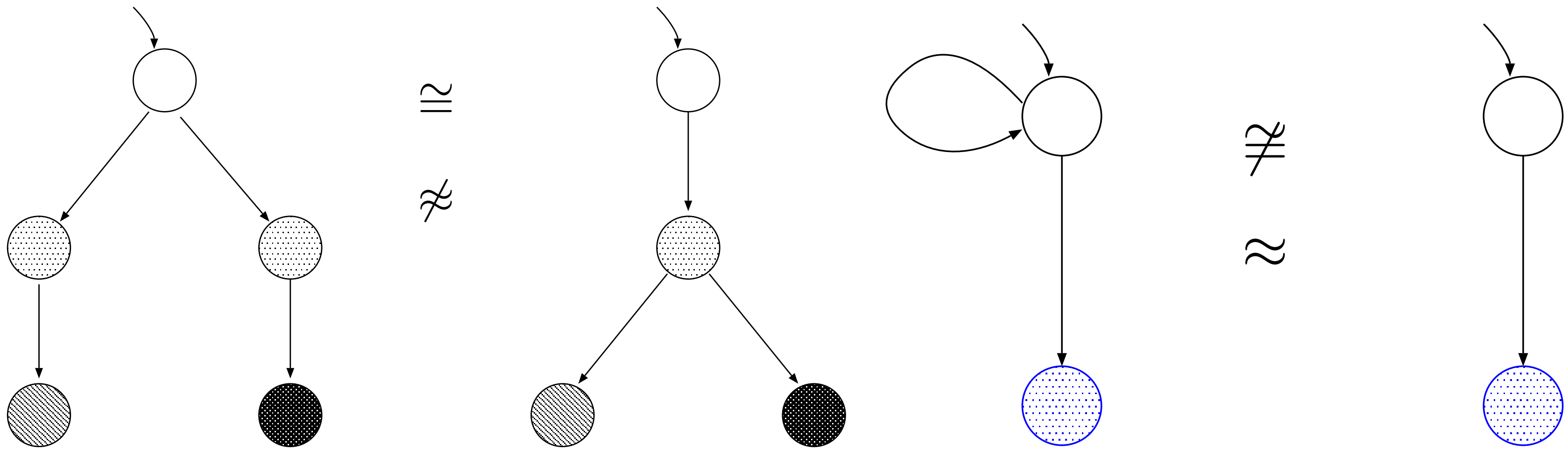


$\mathcal{R}$ inducing the following partitioning of the state space is a stutter bisimulation:

$$\{\{\langle n_1, n_2\rangle, \langle n_1, w_2\rangle, \langle w_1, n_2\rangle, \langle w_1, w_2\rangle\}, \{\langle c_1, n_2\rangle, \langle c_1, w_2\rangle\}, \{\langle c_2, n_1\rangle, \langle w_1, c_2\rangle\}\}$$

In fact, this is the coarsest stutter bisimulation, i.e., $\mathcal{R}$ equals $\approx_{TS}$

Stutter trace and stutter bisimulation are incomparable



main reason: ≈ does not impose constraints on stutter paths

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

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

main reason: presence of stutter paths

Divergence sensitivity

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

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

  $\Rightarrow$ $\approx$ and 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 $\cong$ (and $\approx$)

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

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

Example

Divergent-sensitive stutter bisimulation

s_1, s_2 in *TS* are *divergent stutter-bisimilar*, denoted $s_1 \approx_{TS}^{div} s_2$, if:

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

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

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

Example

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

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

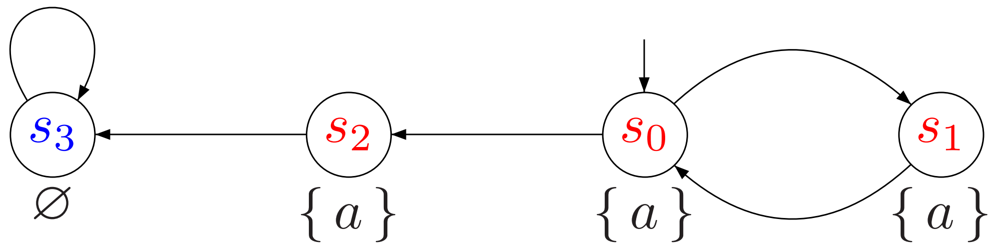
where

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

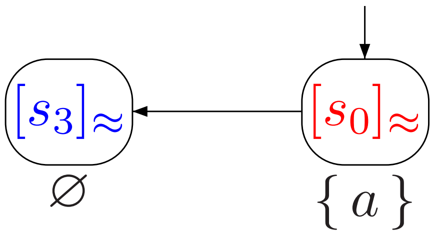
$$\frac{s \xrightarrow{\alpha} s' \;\wedge\; s \not\approx^{div} s'}{[s]_{div} \xrightarrow{\tau}'_{div} [s']_{div}} \quad \text{and} \quad \frac{s \text{ is } \approx^{div}\text{-divergent}}{[s]_{div} \xrightarrow{\tau}'_{div} [s]_{div}}$$

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

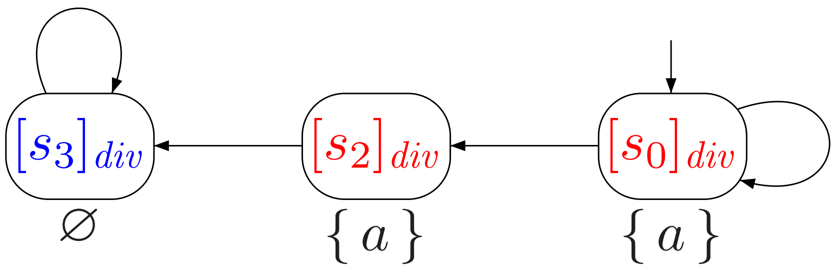
Example



transition system *TS*



transition system $TS/\approx$



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

A remark on purely divergent states

- s_{pd} is *purely divergent* if all paths of s are infinite and divergent
- s_{term} is a terminal state if it has no outgoing transitions
- if $L(s_{pd}) = L(s_{term})$ then $s_{term} \approx_{\mathsf{TS}} s_{pd}$ and $s_{term} \not\approx_{\mathsf{TS}}^{div} s_{pd}$
- $s_{term} \approx_{\mathsf{TS}}^{div} s$ implies
  - $L(s) = L(s_{term})$ and each path of s is finite and divergent

$\approx^{div}$ on paths

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

$$\pi_1 \;\approx_{TS}^{div}\; \pi_2$$

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:

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

$\approx^{div}$ on finite paths can be defined similarly

Example

Comparing paths by $\approx^{div}$

Let *TS* $= (S, Act, \rightarrow, I, AP, L)$ be a transition system, $s_1, s_2 \in S$. Then:

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

Proof

Stutter equivalence versus $\approx^{div}$

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

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

whereas the reverse implication does not hold in general

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

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 $\text{CTL}_{\setminus\bigcirc}$ and $\text{CTL}^*_{\setminus\bigcirc}$ equivalence

Proof

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 {

$produce$;

$\mathbf{for}\ i = 1\ \textbf{to}\ n$ {

$\mathbf{wait\ until}\ (\mathit{free} > 0)$ {

$\mathit{free} := \mathit{free} - 1;$

}

}

Consumer

while true {

$\mathbf{for}\ j = 1\ \textbf{to}\ n$ {

$\mathbf{wait\ until}\ (\mathit{free} < m)$ {

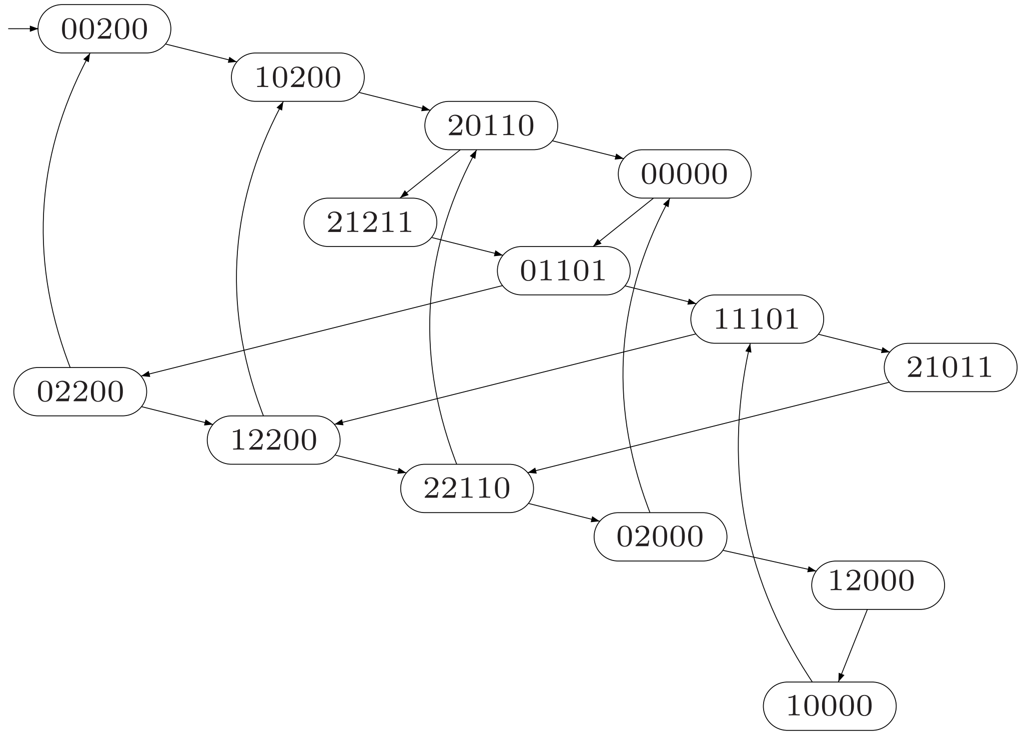
$\mathit{free} := \mathit{free} + 1;$

}

$consume$

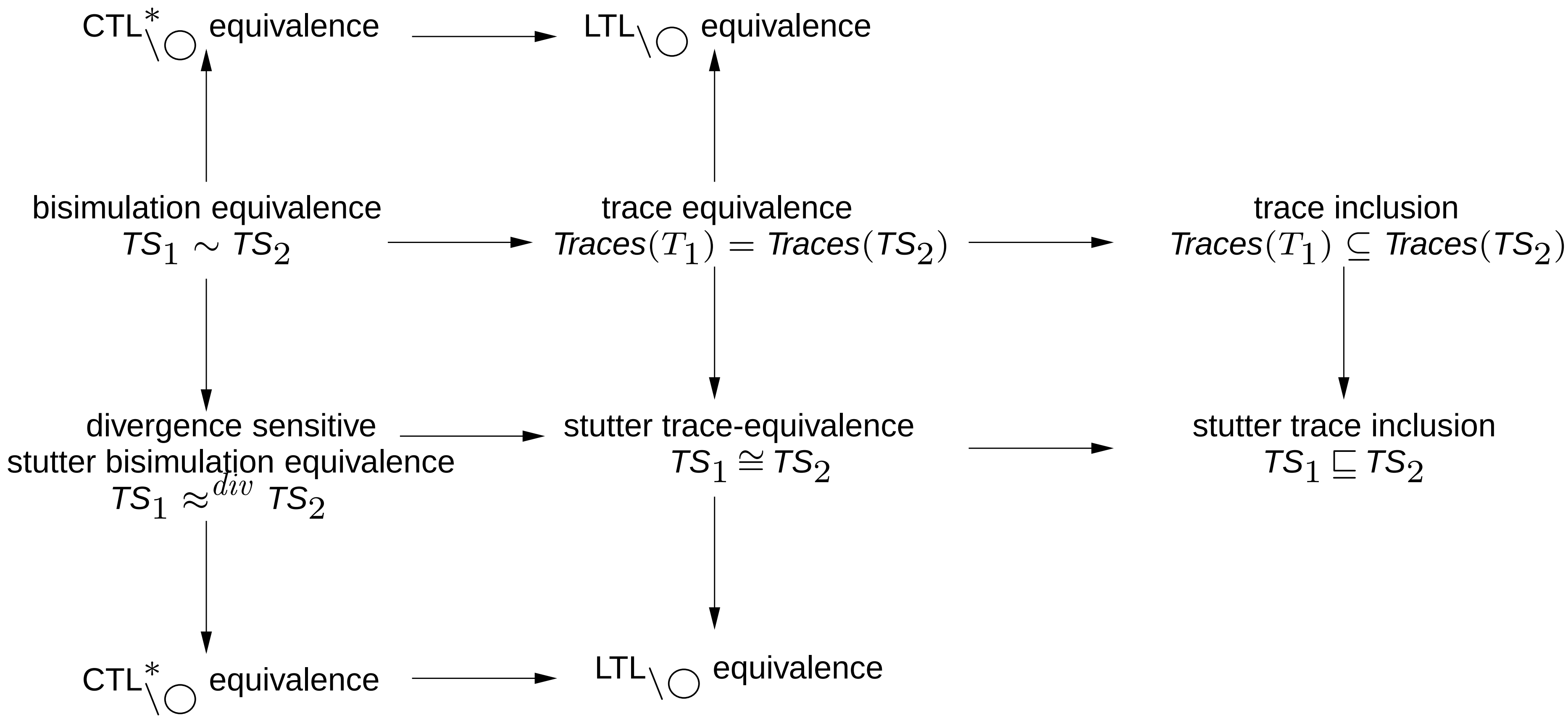
}

Abstract transition system



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

Comparative semantics



AP determinism