Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

Analyzing Divergence in Bisimulation Semantics

于婷婷

中科院软件所
中国科学院大学

(合作者 柳欣欣 张文辉)

SAVE 2016, 长沙

Motivation

- Divergence: existence of infinite internal computation sequences, an important semantic issue
  - termination
  - progress property: eventually one of the pending procedure calls will be returned
- Bisimulation: a corner stone in concurrency theory
  - it has been successfully used to define semantic equivalences of various abstraction levels
  - it provides verification methods (bisimulation techniques) for these equivalences. Some equivalences are useful in program verifications:
    - $\approx$ with weak bisimulation method
    - $\approx_b$ with branching bisimulation method

Motivation

- However, divergence is not preserved by some popular bisimulation equivalences including weak bisimilarity $\approx$ and branching bisimilarity $\approx_b$

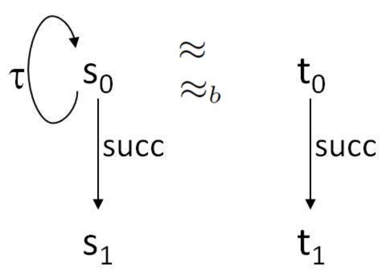


Figure: Divergence is not preserved by $\approx$ and $\approx_b$

Motivation

- Usual solution: treat divergence as a basic observation and strengthen the definitions to obtain
  - divergence preserving weak bisimilarity $\approx^{\Uparrow}$ (dates back to Hennessy and Plotkin 1980):
    if $s \approx^{\Uparrow} t$ then
    ... (the action matching requirment)
    and moreover: $s \Uparrow$ if and only if $t \Uparrow$
  - branching bisimularity with explicit divergence $\approx_b^{\Delta}$ (van Glabbeek and Weijland 1996):
    if $s \approx_b^{\Delta} t$ then
    ... (the action matching requirment)
    and moreover: $s \Uparrow_{\approx_b^{\Delta}}$ if and only if $t \Uparrow_{\approx_b^{\Delta}}$

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

Motivation

- A recent work by Xiaoxiao Yang *et al.* proposed an original idea of using $\approx_b^\Delta$ to prove correctness and progress of concurrent objects, and developed a set of methods supported by many significant case studies to illustrate the idea. Their work shows that divergence preserving bisimulation equivalences can be used in verifying correctness and progress of concurrent objects.
- Question: What more needs to be done?

Motivation

- Problem: because divergence is not a primitive observation (the outcome of a primitive observation should at least be decidable), it is difficult to device a verification method for an equivalence containing non-primitive observations.

  for all $(s,t) \in R$ the following hold

  - if $s \xrightarrow{\alpha} s'$ then $t \stackrel{\widehat{\alpha}}{\Longrightarrow} t'$ and $(s',t') \in R$ with t';
  - if $s \Uparrow$ then $t \Uparrow$;
  - ...

  Thus with the current description of divergence preservation, the divergence preserving bisimulation equivalences are not supported by verification methods.

Motivation

- Our idea for solving the problem: introduce induction into the notion of bisimulation to identify pairs of states which have the same divergence behaviour.

  The hope is that with this solution we may only consider (internal) transitions as basic observations (they are clearly primitive), then it is easy to device a kind of bisimulation technique which is good for verification.

Summary of results

We develop this idea and obtain the following results:

- introduce a new divergence sensitive weak bisimulation equivalence, weak bisimilarity with explicit divergence $\approx^{\Delta}$, characterized by inductive weak bisimulation.
- provide inductive characterizations (thus verification methods) for two known divergence-sensitive equalities: branching bisimilarity with explicit divergence $\approx_b^{\Delta}$, and divergence preserving weak bisimilarity $\approx^{\Uparrow}$.
- introduce complete weak (branching) bisimulation, a very useful theoretical notion which builds connections for different notions.
- verify the correctness of HSY collision stack using the proposed bisimulation technique, which demonstrates that the technique is not over restrictive.

Outline

1. Motivation
2. Weak bisimulation
3. Weak bisimulation with explicit divergence
4. Complete weak bisimulation
5. Inductive weak bisimulation
6. Characterization theorem
7. Inductive branching bisimulation
8. Generalized inductive weak bisimulation
9. A verification example
10. Conclusion and related works

Weak bisimulation

A binary relation $R \subseteq S \times S$ on states of an LTS is a weak bisimulation if for all $(s,t) \in R$ the following hold:

1. whenever $s \xrightarrow{\alpha} s'$ then $\exists (s',t') \in R.\ t \overset{\widehat{\alpha}}{\Longrightarrow} t'$;
2. whenever $t \xrightarrow{\alpha} t'$ then $\exists (s',t') \in R.\ s \overset{\widehat{\alpha}}{\Longrightarrow} s'$.

weak bisimilarity, written $\approx$, is defined by

$$\approx\ = \bigcup \{R \mid R \text{ is a weak bisimulation}\}.$$

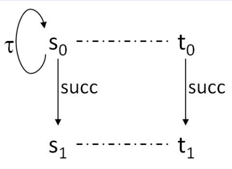
Theorem

1. *$\approx$ is an equivalence relation, and*
2. *it is the largest weak bisimulation.*

The proof of this theorem is a routine application of Knaster-Tarski fixed-point theorem.

Weak bisimulation

- $\approx$ is not divergence preserving.



For an equivalence $\equiv$, write $s \Uparrow_{\equiv}$ if there is an infinite sequence $s_1 s_2 \ldots$ such that $s \xrightarrow{\tau} s_1$, $s_i \xrightarrow{\tau} s_{i+1}$ and $s_i \equiv s$ for $i \geq 1$.
A simple fact: for an equivalence relation $\equiv$ which is a weak bisimulation, if $\equiv$ preserves $\Uparrow_{\equiv}$ then it preserves divergence ($\Uparrow$).

Weak bisimulation with explicit divergence

An equivalence relation $\equiv$ is called a weak bisimulation with explicit divergence if it is a weak bisimulation, and moreover whenever $s \equiv t$ then $s \Uparrow_{\equiv}$ if and only if $t \Uparrow_{\equiv}$.
Now we may define weak bisimilarity with explicit divergence, written $\approx^{\Delta}$, as

$$\approx^{\Delta} = \bigcup \{\equiv \mid \equiv \text{ is a weak bisim. with explicit div.}\}.$$

Theorem

1. $\approx^{\Delta}$ *an equivalence relation, and moreover*
2. *it is the largest weak bisim. with explicit divergence.*

Proof needed! (Here, Knaster-Tarski fixed point theorem is no longer applicable.)

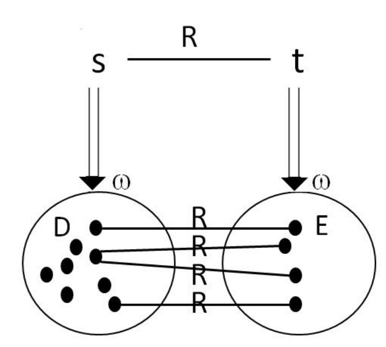
Complete weak bisimulation

A binary relation $R \subseteq S \times S$ on states of an LTS is a complete weak bisimulation if R is a weak bisimulation and moreover for all $(s, t) \in R$ the following hold:

3 whenever $s \Longrightarrow_\omega D$ then $\exists E.\ t \Longrightarrow_\omega E$ & $D \sqsupseteq_R E$;

4 whenever $t \Longrightarrow_\omega E$ then $\exists D.\ s \Longrightarrow_\omega D$ & $D \sqsubseteq_R E$;

Complete weak bisimulation



Notation

- $s \Longrightarrow_\omega D$ if there is an infinite τ-run which passes D infinitely often
- $D \sqsupseteq_R E$ for all $t \in E$ there is $s \in D$ such that $(s, t) \in R$

Complete weak bisimulation

Complete weak bisimilarity, written $\approx_c$, is defined by

$$\approx_c = \bigcup\{R \mid R \text{ is a complete weak bisimulation}\}.$$

Theorem

1. *$\approx_c$ is an equivalence relation, and*
2. *it is the largest complete weak bisimulation.*

Lemma

Closed under composition: If R_1, R_2 are complete weak bisimulations, then $R_1 \cdot R_2$ is a complete weak bisimulation.

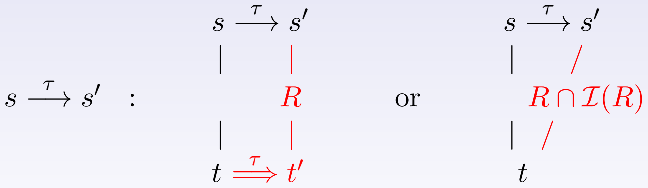
Lemma

Closed under union: If $\{R_i\}_I$ is a set of comp. weak bisimulations, then $\bigcup\{R_i \mid i \in I\}$ is a comp. weak bisim.

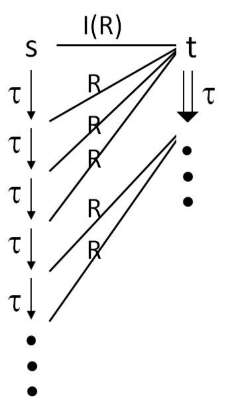
Inductive weak bisimulation

For a binary relation $R \subseteq S \times S$ on states of a LTS, let $\mathcal{I}(R)$ be the smallest binary relation closed in the sense that for $s, t \in S$, if the following hold then $(s,t) \in \mathcal{I}(R)$:

1. whenever $s \xrightarrow{\tau} s'$ then either there exists $t' \in S$ such that $t \overset{\tau}{\Longrightarrow} t'$ and $(s', t') \in R$, or $(s', t) \in R \cap \mathcal{I}(R)$;
2. whenever $t \xrightarrow{\tau} t'$ then ....



Inductive weak bisimulation



$\mathcal{I}(R)$ is inductively defined, intuitively it captures those pairs which have the same divergence behaviour with respect to R.

Inductive weak bisimulation

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

A relation R is an inductive weak bisimulation if it is a weak bisimulation and moreover $R \subseteq \mathcal{I}(R)$.
Inductive weak bisimilarity, written $\approx_i$, is defined by

$$\approx_i = \bigcup\{R \mid R \text{ is an inductive weak bisimulation}\}.$$

Theorem

1. *$\approx_i$ is an equivalence relation, and*
2. *it is the largest inductive weak bisimulation.*

1. needs to be proved. 2. follows from the following lemma.

Lemma

Closed under union: If $\{R_i\}_I$ is a set of ind. weak bisimulations, then $\bigcup\{R_i \mid i \in I\}$ is an ind. weak bisim.

A brief summary of the notions

We have defined three notions of weak bisimulation, corresponding to three relations

1. $\approx^{\Delta} = \bigcup\{\equiv \mid \equiv$ is a weak bisim. with explicit div.$\}$
   $\approx^{\Delta}$ is an equivalence(?)
   $\approx^{\Delta}$ is a weak bisimulation with explicit divergence(?)
2. $\approx_c = \bigcup\{R \mid R$ is a complete weak bisimulation$\}$
   $\approx_c$ is an equivalence
   $\approx_c$ is a complete weak bisimulation.
3. $\approx_i = \bigcup\{R \mid R$ is an ind. weak bisimulatoin$\}$
   $\approx_i$ is an equivalence(?)
   $\approx_i$ is an inductive weak bisimulation.

Characterization theorem

To answer the questions we study properties of and the relationships among the notions

Lemma

$\mathcal{W}(\approx_c)$ is a complete weak bisimulation, where

$$\mathcal{W}(\approx_c) = \{(s,t) | \textit{whenever } s \xrightarrow{\alpha} s' \textit{ then } t \overset{\widehat{\alpha}}{\Longrightarrow} t' \& s' \approx_c t' \quad \textit{whenever } t \xrightarrow{\alpha} t' \textit{ then } s \overset{\widehat{\alpha}}{\Longrightarrow} s' \& s' \approx_c t'\}.$$

Thus $\mathcal{W}(\approx_c) = \approx_c$.

$\approx_c$ satisfies the following so-called stuttering property or computation lemma.

Lemma

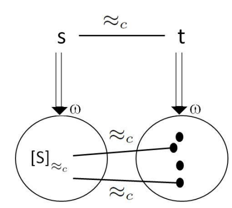
If $s \Longrightarrow t \Longrightarrow s'$, $s \approx_c s'$, then $(s,t) \in \mathcal{W}(\approx_c)$, thus $s \approx_c t$.

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

Characterization theorem

Lemma

If $s \approx_c t$ and $s \Uparrow_{\approx_c}$, then $t \Uparrow_{\approx_c}$. Thus $\approx_c$ is a weak bisimulaiton with explicit divergence.



Lemma

1. *Every weak bisim. w. expl. div. is an ind. weak bisim.*
2. *Every ind. weak bisimulation is a complete weak bisim.*

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

Characterization theorem

weak bisim. with explicit div.		
$\downarrow$		
ind. weak bisim.		$\uparrow$
$\downarrow$		
comp. weak bisim.	$\subseteq$	$\approx_c$

Theorem

$\approx_c$ and $\approx^\Delta$ and $\approx_i$ coincide.

Proof. $\approx_c$ is a weak bisim. w. expl. div. $\Rightarrow$ $\approx_c \subseteq \approx^\Delta$. Every weak bisim. w. expl. div. is an ind. weak bisim. $\Rightarrow$ $\approx^\Delta \subseteq \approx_i$. Every ind. weak bisim. is a comp. weak bisim. $\Rightarrow$ $\approx_i \subseteq \approx_c$. □

Characterization theorem

This theorem can answer the three early questions:

1. $\approx^{\Delta}$ is an equivalence.
2. $\approx^{\Delta}$ is a weak bisimulation with explicit divergence.
3. $\approx_i$ is an equivalence.

The three notions provide three different perspectives for the understanding of the equivalence relation:

1. weak bisim. with explicit div. is simple and direct;
2. comp. weak bisim. is theoretically powerful;
3. inductive weak bisimulation is verification friendly.

Inductive branching bisimulation

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

The same approach can be applied to branching bisimulation to obtain inductive characterization of branching bisimilarity with explicit divergence

1. $\approx_b^\Delta = \bigcup\{\equiv \mid \equiv$ is a bran. bisim. with explicit div.$\}$
   $\approx_b^\Delta$ is an equivalence
   $\approx_b^\Delta$ is a branching bisimulation with explicit divergence
2. $\approx_{cb} = \bigcup\{R \mid R$ is a complete bran. bisim.$\}$
   $\approx_{cb}$ is an equivalence
   $\approx_{cb}$ is a complete branching bisimulation.
3. $\approx_{ib} = \bigcup\{R \mid R$ is an ind. bran. bisim.$\}$
   $\approx_{ib}$ is an equivalence(?)
   $\approx_{ib}$ is an inductive branching bisimulation.

Inductive branching bisimulation

branching bisim. with explicit div.

$\downarrow$

ind. branching bisim. $\qquad\qquad \uparrow$

$\downarrow$

comp. branching bisim. $\quad \subseteq \quad \approx_{cb}$

Theorem

$\approx_b^\Delta$ and $\approx_{cb}$ and $\approx_{ib}$ coincide.

Divergence preserving weak bisimulation

A binary relation R is called a divergence preserving weak bisimulation if

1. R is a weak bisimulation
2. for all $(s, t) \in R$: $s \Uparrow$ if and only if $t \Uparrow$.

Divergence preserving weak bisimilarity, written $\approx^{\Uparrow}$, is defined by

$$\approx^{\Uparrow} = \bigcup \{R | R \text{ is a divergence preserving weak bisimulation}\}.$$

Theorem

1. *$\approx^{\Uparrow}$ is an equivalence relation, and*
2. *it is the largest divergence preserving weak bisimulation.*

Theorem

$\approx_b^{\Delta} \subseteq \approx^{\Delta} \subseteq \approx^{\Uparrow}$.

Generalized inductive weak bisimulation

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

A set of states $D \subseteq S$ is called a divergence set if for all $s \in D$ there is $s' \in D$ such that $s \stackrel{\tau}{\Longrightarrow} s'$.

A relation R is a generalized inductive weak bisimulation if R is a weak bisimulation and moreover there is a divergence set D such that $R \subseteq (\mathcal{I}(R) \cup D \times D)$.

Obviously, generalized inductive weak bisimulation is a generalization of inductive weak bisimulation: every inductive weak bisimulation is also a generalized one.

Theorem

For $s, t \in S$, $s \approx^{\Uparrow} t$ if and only if there is a generalized inductive weak bisimulation R such that $(s, t) \in R$.

A verification example


```
          type Node = { val:  Val,
                        next:  ptr_to Node}
          Top:  ptr_to Node

          push(v:  Val) =
      E0  n:  ptr_to Node := newNode()
      E1  atomic {
      E2    n-> val := v
      E3    n->next := Top
      E4    Top := n
      E5  }
      E6  return
```

Figure: Pseudo-code for lock-free stack specification using atomic blocks

A verification example

```
      pop(): Val =
F0    atomic {
F1      n:ptr_to Node:= Top
F2      if n <> null then
F3        {Top:=n->next
F4        v: Val :=n-> val
F5         }
F6    }
F7    if n = null then
F8      return empty
F9    else
F10     return v
F11   fi
```

Figure: Pseudo-code for lock-free stack specification using atomic blocks

Transitions of thread with atomic blocks

$$\langle H, M, (t, \texttt{EFidle}, m)\rangle \xrightarrow{(t,\text{call } \texttt{push}(d))} \langle H, M, (t, \texttt{E0}, m[\texttt{v} \mapsto d])\rangle$$

$$\langle H, M, (t, \texttt{E0}, m)\rangle \xrightarrow{\tau} \langle H \uplus [q \mapsto \{\texttt{Val} : \bot, \texttt{next} : \texttt{null}\}], M, (t, \texttt{E1}, m[\texttt{n} \mapsto q])\rangle$$

$$\langle H[q \mapsto \{\texttt{Val} : d, \texttt{next} : r\}], M, (t, \texttt{E1}, m[\texttt{n} \mapsto q])\rangle \xrightarrow{\tau} \langle H[q \mapsto \{\texttt{Val} : m(\texttt{v}), \texttt{next} : M(\texttt{Top})\})], M[\texttt{Top} \mapsto q], (t, \texttt{E6}, m[\texttt{n} \mapsto q])\rangle$$

$$\langle H, M, (t, \texttt{E6}, m)\rangle \xrightarrow{(t,\text{ret}()\texttt{push})} \langle H, M, (t, \texttt{EFidle}, m)\rangle$$

$$\langle H, M, (t, \texttt{EFidle}, m)\rangle \xrightarrow{(t,\text{call } \texttt{pop}())} \langle H, M, (t, \texttt{F0}, m)\rangle$$

$$\langle H, M, (t, \texttt{F0}, m)\rangle \xrightarrow{\tau} \langle H, M, (t, \texttt{F7}, m[\texttt{n} \mapsto null])\rangle \quad (M(\texttt{Top}) = null)$$

$$\langle H, M, (t, \texttt{F7}, m)\rangle \xrightarrow{\tau} \langle H, M, (t, \texttt{F8}, m)\rangle \quad (m(\texttt{n}) = null)$$

$$\langle H, M, (t, \texttt{F8}, m)\rangle \xrightarrow{(t,\text{ret}(\texttt{empty})\texttt{pop})} \langle H, M, (t, \texttt{EFidle}, m)\rangle$$

$$\langle H[p \mapsto \{\texttt{Val} : d, \texttt{next} : q\}], M[\texttt{Top} \mapsto p], (t, \texttt{F0}, m)\rangle \xrightarrow{\tau} \langle H, M[\texttt{Top} \mapsto q], (t, \texttt{F7}, m[\texttt{v} \mapsto d, \texttt{n} \mapsto p])\rangle$$

$$\langle H, M, (t, \texttt{F7}, m)\rangle \xrightarrow{\tau} \langle H, M, (t, \texttt{F10}, m)\rangle \quad (m(\texttt{n}) \neq null)$$

$$\langle H, M, (t, \texttt{F10}, m)\rangle \xrightarrow{(t,\text{ret}(m(\texttt{v}))\texttt{pop})} \langle H, M, (t, \texttt{EFidle}, m)\rangle$$

```
    type Node = { val: Val,
                  next: ptr_to Node}
    Top: ptr_to Node
    type Op=enum{NONE,PUSH,POP}
    type opInfo={op:OP,
                  node:ptr_to Node}
    opInfos: array[numprocs] of opInfo
    collision: array[size] of ProcessId

    push(v: Val) =
A0  n: ptr_to Node := newNode()
A1    n-> val := v
A2    info: opInfo :=(PUSH,n)
A3  loop
A4  if tryPush(n) then exit
A5  if tryElimination(& info) then exit
A6  endloop
A7  return
```

```
    tryPush(n:ptr_to Node):boolean=
C1  ss:ptr_to Node := Top
C2  n->next:= ss
C3  return CAS(&Top, ss, n)
```

```
     pop(): Val =
B0   info: opInfo:=(POP,null)
B1   loop
B2     if tryPop(info.node) then exit
B3     if tryEliminate(&info) then exit
B4   endloop
B5   if info.node=null then
B6       return empty
B7   else
B8      v: Val:=info.node-> val
B10    return v
B11  fi
```

```
     tryPop(n:ptr_to Node):  boolean=
D1   ss:  ptr_to Node:= Top
D2   if ss=null then
D3      n:=null
D4      return true
D5   else
D6       n:= ss
D7       ssn:  ptr_to Node:= ss-> next
D8       return CAS(&Top, ss, ssn)
D9   fi
```

A verification example

$$\frac{\langle H, M, (t_i, l_i, m_i)\rangle \xrightarrow{\alpha} \langle H', M', (t_i, l'_i, m'_i)\rangle}{\langle H, M, \ldots (t_i, l_i, m_i) \ldots\rangle \xrightarrow{\alpha} \langle H', M', \ldots (t_i, l'_i, m'_i) \ldots\rangle}$$

We need to establish:
$\langle \epsilon, M, \ldots (t_i, \texttt{ABidle}, m_i) \ldots\rangle \approx^{\Delta} \langle \epsilon, M, \ldots (t_i, \texttt{EFidle}, m_i) \ldots\rangle$

For that we construct an inductive weak bisimulation R which contains
$(\langle \epsilon, M, \ldots (t_i, \texttt{ABidle}, m_i) \ldots\rangle, \langle \epsilon, M, \ldots (t_i, \texttt{EFidle}, m_i) \ldots\rangle)$.

A verification example

R is defined such that

$$(\langle H, M, (t_1, l_1, m_1) \ldots (t_n, l_n, m_n) \rangle, \\ \langle H', M', (t_1, l_1', m_1') \ldots (t_n, l_n', m_n') \rangle) \in R$$

if and only if the following hold:

1. $\langle H, M, (t_1, l_1, m_1) \ldots (t_n, l_n, m_n) \rangle$ is a type AB configuration which is reachable from $\langle \epsilon, M, \ldots (t_i, \texttt{ABidle}, m_i) \ldots \rangle$, and $\langle H', M', (t_1, l_1', m_1') \ldots (t_n, l_n', m_n') \rangle$ is a type EF configuration which is reachable from $\langle \epsilon, M, \ldots (t_i, \texttt{EFidle}, m_i) \ldots \rangle$.
2. $H = H', M(\texttt{Top}) = M'(\texttt{Top})$.

A verification example

3. for each i, (t_i, l_i, m_i) and (t_i, l'_i, m'_i) satisfy one of the following conditions

   idle: *both in idle states;*

   push: *both in pre-linearization push states or both in post-linearization push states,* $m_i(\mathtt{n}) = m'_i(\mathtt{n}), m_i(\mathtt{v}) = m'_i(\mathtt{v})$*;*

   pre-pop: *both in pre-linearization pop states;*

   post-pop: *both in post-linearization pop states,* $m_i(\mathtt{ss}) = m'_i(\mathtt{n})$, $m_i(\mathtt{v}) = m'_i(\mathtt{v})$.

For this R, we can prove that

1. R is a weak bisimulation,
2. and $R \subseteq \mathcal{I}(R)$.

Conclusion and Related Works

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

1. Introduced weak bisimilarity with explicit divergence $\approx^{\Delta}$, characterized by inductive weak bisimulation which supports verification.
2. As an application example, used inductive weak bisimulation to verify the correctness of HSY collision stack, which shows that the proposed method is not over restrictive.
3. The method can be adapted for branching bisimilarity with explicit divergence $\approx_b^{\Delta}$ and divergence preserving weak bisimilarity $\approx^{\Uparrow}$.

Conclusion and Related Works

1. van Glabbeek and Weijland's work on branching bisimilarity with explicit divergence.
2. Namjoshi's work on well-founded stutter equivalence.
3. Gotsman and Yang, Liang *et al.*'s work on linearizability plus progress conditions.
4. Xiaoxiao Yang *et al.*'s work on using branching bisimilarity with explicit divergence to verify correctness and progress of concurrent data structures.

References

- Rob J. van Glabbeek, Peter Weijland: Branching time and abstraction in bisimulation semantics. J. ACM 43(3):555-600.1996
- Rob J. van Glabbeek, Bas Luttik, Nikola Trcka: Branching Bisimilarity with Explicit Divergence. Fundam. Inform. 93(4): 371-392 (2009)
- K. S. Namjoshi. A simple characterization of stuttering bisimulation. In 17th Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS), volume 1346 of Lecture Notes in Computer Science, pages 284 - 296.
- X. Yang, J. Katoen, H. Lin, and Hao, Wu. Proving Linearizability via Branching Bisimulation. CoRR abs/1690.07546(2016)

- Hongjin Liang, Jan Hoffmann, Xinyu Feng, Zhong Shao: Characterizing Progress Properties of Concurrent Objects via Contextual Refinements. CONCUR 2013: 227-241.
- Hongjin Liang, Xinyu Feng, Zhong Shao: Compositional verification of termination-preserving refinement of concurrent programs. CSL-LICS 2014: 65:1-10.
- Alexey Gotsman, Hongseok Yang: Liveness-Preserving Atomicity Abstraction. ICALP (2) 2011: 453-465.
- M. Hennessy and G. Plotkin. A term model for CCS, Lecture notes in computer science, Vol.88, Springer-Verlag, 1980.
- D.J. Walker. Bisimulation and divergence, Information and Computation, vol. 85, pp. 212-241, 1990.

- M.C. Browne and E. M. Clarke and O. Grümberg, *Characterizing finite kripke structures in propositional temporal logic*, Theoretical computer science 59(1988) pp.115-131
- M. C. Browne, E. M. Clarke, O. Grumberg. Reasoning about Networks with Many Identical Finite State Processes, Information and Computation, vol. 81, no. 1, pp. 13-31, April 1989.
- Rob J. van Glabbeek: The Linear Time - Branching Time Spectrum II. CONCUR 1993: 66-81.
- R. de Nicola, F. Vaandrager. Three logics for branching bisimulation, Journal of the ACM, 42(2):458-487, 1995.
- David Park: Concurrency and automata on infinite sequences. Lecture Notes in Computer Science 104,1981. Proceedings of 5th GI Conference.

Motivation
Weak bisim.
Weak bisim. w. exp. div.
Comp. weak bisimulation
Ind. weak bisimulation
Characteriza. theorem
Ind. bran. bisimulation
Gen. ind. weak bisim.
A case stud.
Conclusion and related works

- Robin Milner: Communication and concurrency. Prentice-Hall,1989.
- Christel Baier and Joost-Pieter Katoen. Principles of Model Checking. The MIT Press. 2008.
- E. Allen Emerson, Joseph Y. Halpern: “Sometimes” and “Not Never” revisited: on branching versus linear time temporal logic. J. ACM 33(1): 151-178. 1986.
- J.F. Groote, F.W. Vaandrager: An efficiant algorithm for branching bisimulation and stuttering equivalence: In proceedings of ICALP’, M.W.Paterson ed. Lecture notes in computer science, vol. 433. Springer-Verlag, New York, pp. 626-638, 1990.