Bisimilarity of Pushdown Automata is Nonelementary

Michael Benedikt (Oxford)
Stefan Göller (Bremen)
Stefan Kiefer (Oxford)
Andrzej Murawski (Warwick)

Highlights 2013, Paris
September 19th
Pushdown automata

A pushdown automaton $P$ is given by

- a finite set of control states $p, q, \ldots$
- a finite set of stack symbols $\bot, A, B, C, \ldots$
- labeled rules of the kind
  - $pA \xrightarrow{a} q$ (pop).
  - $pA \xrightarrow{b} qB$ (internal), where $A = \bot$ iff $B = \bot$.
  - $pA \xrightarrow{c} qBC$ (push), where $A = \bot$ iff $\bot \in \{B, C\}$.
Pushdown automata

A pushdown automaton $P$ is given by

- a finite set of control states $p, q, \ldots$
- a finite set of stack symbols $\bot, A, B, C, \ldots$
- labeled rules of the kind
  - $pA \xrightarrow{a} q$ (pop).
  - $pA \xrightarrow{b} qB$ (internal), where $A = \bot$ iff $B = \bot$.
  - $pA \xrightarrow{c} qBC$ (push), where $A = \bot$ iff $\bot \in \{B, C\}$.

An example for a pushdown automaton $P$

Rules: $qA \xrightarrow{a} qAA$  $qA \xrightarrow{b} qBA$  $qB \xrightarrow{a} qAB$  $qB \xrightarrow{b} qBB$
Pushdown automata

A pushdown automaton $P$ is given by

- a finite set of control states $p, q, \ldots$
- a finite set of stack symbols $\bot, A, B, C, \ldots$
- labeled rules of the kind
  - $pA \xrightarrow{a} q$ (pop).
  - $pA \xrightarrow{b} qB$ (internal), where $A = \bot$ iff $B = \bot$.
  - $pA \xrightarrow{c} qBC$ (push), where $A = \bot$ iff $\bot \in \{B, C\}$.

An example for a pushdown automaton $P$

Rules: $qA \xrightarrow{a} qAA$ $qA \xrightarrow{b} qBA$ $qB \xrightarrow{a} qAB$ $qB \xrightarrow{b} qBB$

Transition system $T(P)$:
Bisimilarity checking of pushdown processes: Results

Theorem (Sénizergues 1998)

*Bisimilarity of pushdown processes is *decidable.*

(two semi-decision procedures, no prim. rec. upper bound known)

Theorem (Kučera, Mayr 2002)

*Bisimilarity of pushdown processes is EXP-hard.*
Two useful gadgets from Jančar/Srba and Chen/v. Breugel/Worrell

Proposition

We have
- in the left picture: $s_1 \sim s_2$ iff $(t_1 \sim t_2 \text{ OR } t'_1 \sim t'_2)$.
- in the right picture: $s_1 \sim s_2$ iff $(t_1 \sim t_2 \text{ AND } t'_1 \sim t'_2)$. 
Proving EXPSPACE-hardness

Fix an input of length \( n \).

A **0-counter** is a sequence \( c = b_0 \cdots b_{n-1} \in \{0, 1\}^n \).

Its value is \( \text{val}(c) \overset{\text{def}}{=} \sum_{i=0}^{n-1} 2^i \cdot b_i \in \{0, \ldots, 2^n - 1\} \).

**Convention:** For each 0-counter write \( c_i \) we assume \( \text{val}(c_i) = i \).
Proving EXPSPACE-hardness

Fix an input of length \( n \).

A 0-counter is a sequence \( c = b_0 \cdots b_{n-1} \in \{0, 1\}^n \).
Its value is \( \text{val}(c) \overset{\text{def}}{=} \sum_{i=0}^{n-1} 2^i \cdot b_i \in \{0, \ldots, 2^n - 1\} \).

Convention: For each 0-counter write \( c_i \) we assume \( \text{val}(c_i) = i \).

**Proposition**

\( \exists \) gadget with control states \( q, q' \) such that the following holds:

Assume the stack content

\[ \sigma = c_j b c_i w \perp, \]

where \( c_j \) and \( c_i \) are the 0-counters of value \( j \) and \( i \), resp. We have

\[ q(\sigma) \sim q'(\sigma) \iff j = i - 1 \]
Proving EXPSPACE-hardness

\[ q(\boxed{c_j b c_i \ w \perp}) \sim q'(\boxed{c_j b c_i \ w \perp}) \quad \text{iff} \quad j = i - 1 \]
Proving EXPSPACE-hardness

\[ q(\texttt{cjbc}_{i \infty} \cdot w \perp) \sim q'(\texttt{cjbc}_{i \infty} \cdot w \perp) \iff j = i - 1 \]

**Idea:**

1. Lead the game to

\[ p(\texttt{c}_{0 \infty} \cdot c_{jbc} \cdot w \perp) \sim p'(\texttt{c}_{0 \infty} \cdot c_{jbc} \cdot w \perp) \]

by pushing bit 0 followed by the 0-counter \( c_0 \).
Proving EXPSPACE-hardness

\[ q(cjbc_i w \bot) \sim q'(cjbc_i w \bot) \text{ iff } j = i - 1 \]

Idea:

1. Lead the game to \( p(c_00jc_0bc_i w \bot) \sim p'(c_00jc_0bc_i w \bot) \) by pushing bit 0 followed by the 0-counter \( c_0 \).
2. De-synchronize the two stacks by
   - popping the two top-most 0-counters from the left stack and
   - and popping the top-most 0-counter from the right stack.
Proving EXPSPACE-hardness

$q(c_j b c_i w \perp) \sim q'(c_j b c_i w \perp)$ iff $j = i - 1$

Idea:

1. Lead the game to $p(c_0 0 c_j b c_i w \perp) \sim p'(c_0 0 c_j b c_i w \perp)$ by pushing bit 0 followed by the 0-counter $c_0$.

2. De-synchronize the two stacks by
   - popping the two top-most 0-counters from the left stack and
   - and popping the top-most 0-counter from the right stack.

leading us to $p(c_i w \perp) \sim p'(c_j b c_i w \perp)$
Proving EXPSPACE-hardness

\[ q(\overline{c_jbc_iw\bot}) \sim q'(\overline{c_jbc_iw\bot}) \text{ iff } j = i - 1 \]

Idea:

1. Lead the game to \( p(\overline{c_00c_jbc_iw\bot}) \sim p'(\overline{c_00c_jbc_iw\bot}) \) by pushing bit 0 followed by the 0-counter \( c_0 \).

2. De-synchronize the two stacks by
   - popping the two top-most 0-counters from the left stack and
   - and popping the top-most 0-counter from the right stack.

   leading us to \( p(\overline{c_iw\bot}) \sim p'(\overline{c_jbc_iw\bot}) \)

3. Check if \( j = i - 1 \) deterministically as follows:
   3.1 **Left automaton** outputs \( c_i \)
   3.2 **Right automaton** outputs \( T(c_j) \), where \( T \) is a deterministic letter-to-letter transducer that outputs the successor function.
Proving EXPSPACE-hardness

How to push

\[ c_0 1 \cdots c_1 1 c_{2n-1} 1 \]

onto both stacks?

**Idea:**

1. First, we explicitly push \( c_{2n-1} 1 \) onto the stacks.
Proving EXPSPACE-hardness

How to push
\[ c_01 \cdots c_11c_{2^n-1}1 \]
onto both stacks?

**Idea:**

1. First, we explicitly push \( c_{2^n-1}1 \) onto the stacks.
2. We are thus in \( q(c_{2^n-1}1\perp) \overset{?}{\sim} q'(c_{2^n-1}1\perp) \).
Proving EXPSPACE-hardness

How to push

\[ c_0 1 \cdots c_1 1 c_{2^{n-1}} 1 \]

onto both stacks?

**Idea:**

1. First, we explicitly push \( c_{2^{n-1}} 1 \) onto the stacks.
2. We are thus in \( q(c_{2^{n-1}} 1 \perp) \sim q'(c_{2^{n-1}} 1 \perp) \).
3. How push the remaining sequence \( c_0 1 c_1 1 \cdots c_{2^{n-2}} 1 \)?
Proving EXPSPACE-hardness

How to push
\[ c_0 1 \cdots c_1 1 c_{2^n - 1} 1 \]
ono onto both stacks?

**Idea:**

1. First, we explicitly push \( c_{2^n - 1} 1 \) onto the stacks.
2. We are thus in \( q(c_{2^n - 1} 1 \perp) \sim q'(c_{2^n - 1} 1 \perp) \).
3. How push the remaining sequence \( c_0 1 c_1 1 \cdots c_{2^n - 2} 1 \)?
4. Being in situation

\[ q(c_i 1 c_{i+1} 1 \cdots c_{2^n - 1} 1 \perp) \sim q(c_i 1 c_{i+1} 1 \cdots c_{2^n - 1} 1 \perp) \]

Duplicator’s job is to push \( c_j 1 \) with \( j = i - 1 \) onto the stacks.
Proving \textsc{EXPSPACE}-hardness

How to push
\[ c_01 \cdots c_11c_{2^n-1}1 \]
on onto both stacks?

\textbf{Idea:}

1. First, we explicitly push $c_{2^n-1}1$ onto the stacks.
2. We are thus in $q(c_{2^n-1}1\perp) \overset{?}{\sim} q'(c_{2^n-1}1\perp)$.
3. How push the remaining sequence $c_01c_11\cdots c_{2^n-2}1$?
4. Being in situation
\[ q(c_i1c_{i+1}1\cdots c_{2^n-1}1\perp) \overset{?}{\sim} q(c_i1c_{i+1}1\cdots c_{2^n-1}1\perp) \]

Duplicator’s job is to push $c_j1$ with $j = i - 1$ onto the stacks.
5. She pushes some $c_j$ onto the stacks (via $n$ OR-gadgets)
   - If Spoiler believes $j = i - 1$, then goto 4.
   - If Spoiler does not believe $j = i - 1$, then play subgame!
We now know how Duplicator can push $\#d_{2^n-1}$ onto the stacks.

$\Rightarrow$ We now know how Duplicator can push any $\#d_i$ onto the stacks.

$\Rightarrow$ Duplicator can push

$$d_0 \#d_1 \cdots \#d_{2^n-1}$$

onto both stacks by using the same ideas!
Open questions

There are (too) many of them!

- Primitive recursive lower/upper bounds for PDA bisimilarity
- Complexity of DPDA language equivalence
- Decidability of equivalence of Deterministic Higher-Order PDA
- Decidability of bisimilarity of PA-processes
- Decidability of bisimilarity of Ground Tree Rewrite Systems
- Decidability of weak bisimilarity of Single-State PDA
- Decidability of weak bisimilarity of Basic Parallel Processes
- ...

Thank you for your attention!