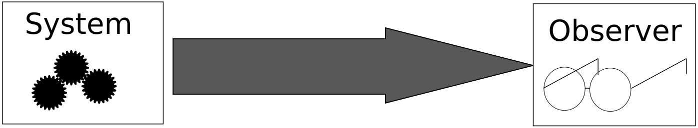
Downward closures and complexity

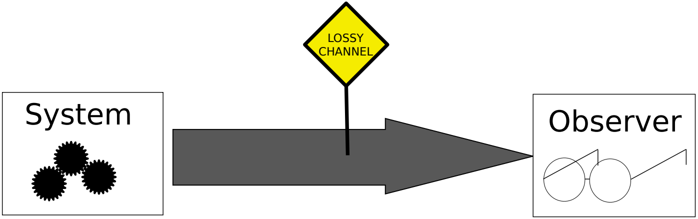
Georg Zetzsche[1]

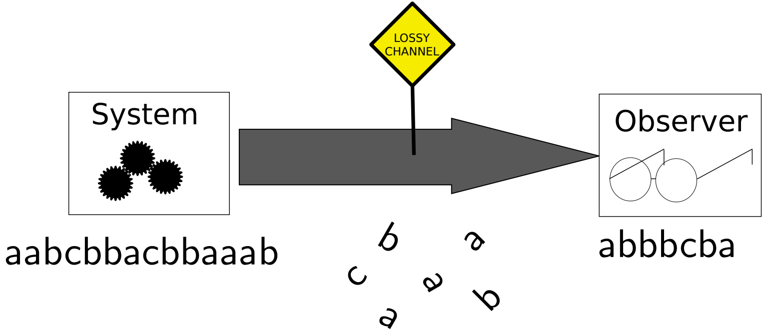
LSV, CNRS & ENS Cachan
Université Paris-Saclay

Higher-Order Model Checking
Shonan Meeting, March 14-17, 2016

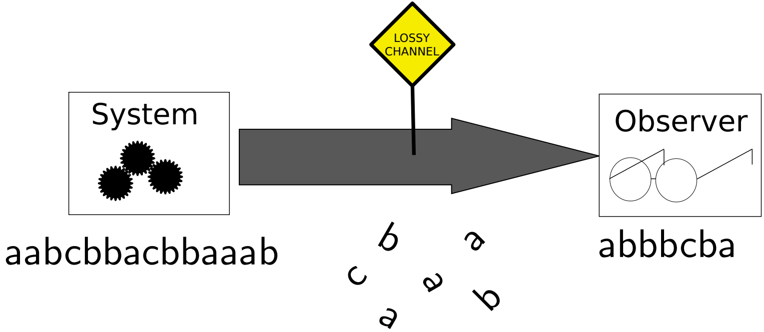
[1] Supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD).


System
Observer


LOSSY
CHANNEL
System
Observer



$\underline{a}ab\underline{c}bb\underline{a}cb\underline{b}a\underline{aab}$



aabcbbacbbaaab

Downward Closures

- $u \preccurlyeq v$: u is a subsequence of v
- $L{\downarrow} = \{u \in X^* \mid \exists v \in L\colon u \preccurlyeq v\}$
- Observer sees precisely $L{\downarrow}$

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L{\downarrow}$ is regular.

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L\downarrow$ is regular.

Applications

Given an automaton for $L\downarrow$, many things are decidable:

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L{\downarrow}$ is regular.

Applications

Given an automaton for $L{\downarrow}$, many things are decidable:

- Inclusion of behavior under lossy observation ($K{\downarrow} \subseteq L{\downarrow}$)
  Ordinary inclusion almost always undecidable!

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L\downarrow$ is regular.

Applications

Given an automaton for $L\downarrow$, many things are decidable:

- Inclusion of behavior under lossy observation ($K\downarrow \subseteq L\downarrow$)
  Ordinary inclusion almost always undecidable!
- Which actions occur arbitrarily often? ($a^* \subseteq L\downarrow$)

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L{\downarrow}$ is regular.

Applications

Given an automaton for $L{\downarrow}$, many things are decidable:

- Inclusion of behavior under lossy observation ($K{\downarrow} \subseteq L{\downarrow}$)
  Ordinary inclusion almost always undecidable!
- Which actions occur arbitrarily often? ($a^* \subseteq L{\downarrow}$)
- Can the system run arbitrarily long? ($L{\downarrow}$ infinite)

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L{\downarrow}$ is regular.

Applications

Given an automaton for $L{\downarrow}$, many things are decidable:

- Inclusion of behavior under lossy observation ($K{\downarrow} \subseteq L{\downarrow}$)
  Ordinary inclusion almost always undecidable!
- Which actions occur arbitrarily often? ($a^* \subseteq L{\downarrow}$)
- Can the system run arbitrarily long? ($L{\downarrow}$ infinite)
- Safety verification of parametrized asynchronous shared-memory systems (La Torre, Muscholl, Walukiewicz, FSTTCS 2015)

Downward Closures

Theorem (Higman/Haines)

For every language $L \subseteq X^*$, $L{\downarrow}$ is regular.

Applications

Given an automaton for $L{\downarrow}$, many things are decidable:

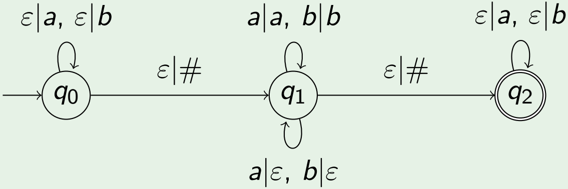
- Inclusion of behavior under lossy observation ($K{\downarrow} \subseteq L{\downarrow}$)
  Ordinary inclusion almost always undecidable!
- Which actions occur arbitrarily often? ($a^* \subseteq L{\downarrow}$)
- Can the system run arbitrarily long? ($L{\downarrow}$ infinite)
- Safety verification of parametrized asynchronous shared-memory systems (La Torre, Muscholl, Walukiewicz, FSTTCS 2015)

Problem

- Finite automaton for $L{\downarrow}$ exists for every L.
- How can we compute it?

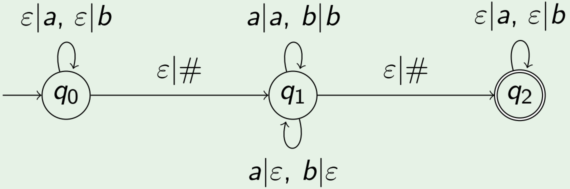
A general approach

Example (Transducer)



A general approach

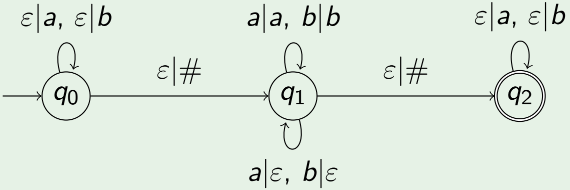
Example (Transducer)



$$T(A) = \{(x, u\#v\#w) \mid u, v, w, x \in \{a,b\}^*,\ v \leqslant x\}$$

A general approach

Example (Transducer)



$$T(A) = \{(x, u\#v\#w) \mid u, v, w, x \in \{a, b\}^*,\ \ v \leqslant x\}$$

Definition

- *Rational transduction*: set of pairs given by a finite state transducer.
- For rational transduction $T \subseteq X^* \times Y^*$ and language $L \subseteq Y^*$, let

$$TL = \{y \in Y^* \mid \exists x \in L : (x, y) \in T\}$$

Definition

$\mathcal{C}$ is a *full trio* if $LR \in \mathcal{C}$ for each $L \in \mathcal{C}$ and rational transduction R.

Definition

$\mathcal{C}$ is a *full trio* if $LR \in \mathcal{C}$ for each $L \in \mathcal{C}$ and rational transduction R.

Theorem (Z., ICALP 2015)

If $\mathcal{C}$ is a full trio, then downward closures are computable for $\mathcal{C}$ if and only if the simultaneous unboundedness problem is decidable:

Given A language $L \subseteq a_1^* \cdots a_n^*$ in $\mathcal{C}$

Question Is $a_1^* \cdots a_n^*$ included in $L\downarrow$?

Definition

$\mathcal{C}$ is a *full trio* if $LR \in \mathcal{C}$ for each $L \in \mathcal{C}$ and rational transduction R.

Theorem (Z., ICALP 2015)

If $\mathcal{C}$ is a full trio, then downward closures are computable for $\mathcal{C}$ if and only if the simultaneous unboundedness problem is decidable:

Given A language $L \subseteq a_1^* \cdots a_n^*$ in $\mathcal{C}$

Question Is $a_1^* \cdots a_n^*$ included in $L\!\downarrow$?

Equivalently, we check whether it is true that:

$$\text{for each } k \geqslant 0, \text{ there are } x_1, \ldots, x_n \geqslant k \text{ with } a_1^{x_1} \cdots a_n^{x_n} \in L$$

Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L\!\downarrow$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\ldots,x_n$ are letters and $Y_0,\ldots,Y_n$ are alphabets.

Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L\!\downarrow$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\ldots,x_n$ are letters and $Y_0,\ldots,Y_n$ are alphabets.

Ideal decompositions: currently also studied by Lazić, Leroux, Schmitz

Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L{\downarrow}$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\ldots,x_n$ are letters and $Y_0,\ldots,Y_n$ are alphabets.

Ideal decompositions: currently also studied by Lazić, Leroux, Schmitz

Algorithm

Suppose $L \subseteq X^*$ is given.
Enumerate simple regular languages R.
Decide whether $L{\downarrow} = R$:

Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L\downarrow$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\ldots,x_n$ are letters and $Y_0,\ldots,Y_n$ are alphabets.

Ideal decompositions: currently also studied by Lazić, Leroux, Schmitz

Algorithm

Suppose $L \subseteq X^*$ is given.
Enumerate simple regular languages R.
Decide whether $L\downarrow = R$:

- $L\downarrow \subseteq R$ iff $L\downarrow \cap (X^*\setminus R) = \emptyset \rightsquigarrow$ emptiness.

Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L\downarrow$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\ldots,x_n$ are letters and $Y_0,\ldots,Y_n$ are alphabets.

Ideal decompositions: currently also studied by Lazić, Leroux, Schmitz

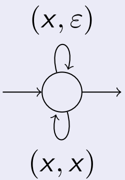
Algorithm

Suppose $L \subseteq X^*$ is given.
Enumerate simple regular languages R.
Decide whether $L\downarrow = R$:

- $L\downarrow \subseteq R$ iff $L\downarrow \cap (X^* \setminus R) = \emptyset \rightsquigarrow$ emptiness.

Observation

$L\downarrow$ is in $\mathcal{C}$:



Theorem (Jullien 1969, Abdulla et. al. 2004)

Every language $L\!\downarrow$ can be written as a finite union of ideals:

$$Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^*,$$

where $x_1,\dots,x_n$ are letters and $Y_0,\dots,Y_n$ are alphabets.

Ideal decompositions: currently also studied by Lazić, Leroux, Schmitz

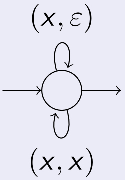
Algorithm

Suppose $L \subseteq X^*$ is given.
Enumerate simple regular languages R.
Decide whether $L\!\downarrow = R$:

- $L\!\downarrow \subseteq R$ iff $L\!\downarrow \cap (X^* \setminus R) = \emptyset \rightsquigarrow$ emptiness.
- $R \subseteq L\!\downarrow \rightsquigarrow Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L\!\downarrow$

Observation

$L\!\downarrow$ is in $\mathcal{C}$:



Observation

- It suffices to check whether $Y_0^*\{x_1, \varepsilon\}Y_1^* \cdots \{x_n, \varepsilon\}Y_n^* \subseteq L{\downarrow}$.

Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$.
- $L{\downarrow}$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L\downarrow$.
- $L\downarrow$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

$$abc\ abc\ abc\ abc\ abc$$

Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$.
- $L{\downarrow}$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

$$abc\ abc\ abc\ abc\ abc$$

$$bacca$$

Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$.
- $L{\downarrow}$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

$$abc\ abc\ abc\ abc\ abc$$

$$bacca$$

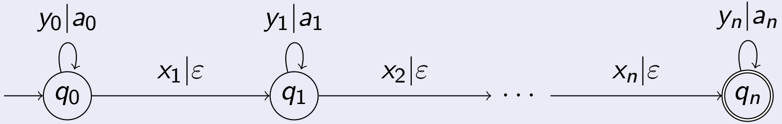
Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$.
- $L{\downarrow}$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

$$abc\ abc\ abc\ abc\ abc$$

$$bacca$$

Transduction T



y_i: word containing each letter of Y_i once.

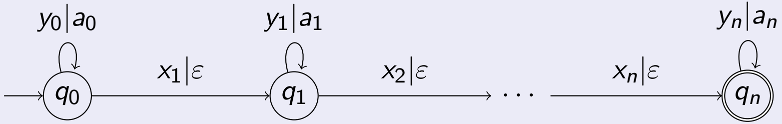
Observation

- It suffices to check whether $Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$.
- $L{\downarrow}$ includes $\{a,b,c\}^*$ if and only if it contains $(abc)^*$.

$$abc\ abc\ abc\ abc\ abc$$

$$bacca$$

Transduction T



y_i: word containing each letter of Y_i once. Then:

$$T(L{\downarrow}){\downarrow} = a_0^*\cdots a_n^* \quad \text{iff} \quad Y_0^*\{x_1,\varepsilon\}Y_1^*\cdots\{x_n,\varepsilon\}Y_n^* \subseteq L{\downarrow}$$

- New algorithm for every known computable case

- New algorithm for every known computable case
- Additional language classes:

- New algorithm for every known computable case
- Additional language classes:

Corollary (Z., ICALP 2015)

Downward closures are computable for order-2 pushdown automata.

- New algorithm for every known computable case
- Additional language classes:

Corollary (Z., ICALP 2015)

Downward closures are computable for order-2 pushdown automata.

Theorem (Hague, Kochems, Ong, POPL 2016)

Downward closures are computable for higher-order pushdown automata.

- New algorithm for every known computable case
- Additional language classes:

Corollary (Z., ICALP 2015)

Downward closures are computable for order-2 pushdown automata.

Theorem (Hague, Kochems, Ong, POPL 2016)

Downward closures are computable for higher-order pushdown automata.

- Igor's talk: higher-order recursion schemes

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\mathrm{pop}_k([s_1 \cdots s_m]_k) \quad = \quad [s_2 \cdots s_m]_k$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\begin{array}{lcll} \mathrm{pop}_k([s_1 \cdots s_m]_k) & = & [s_2 \cdots s_m]_k & \\ \mathrm{pop}_k([s_1 \cdots s_m]_n) & = & [\mathrm{pop}_k(s_1) s_2 \cdots s_m]_k & n > k \end{array}$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\begin{array}{rcll}
\mathrm{pop}_k([s_1 \cdots s_m]_k) & = & [s_2 \cdots s_m]_k & \\
\mathrm{pop}_k([s_1 \cdots s_m]_n) & = & [\mathrm{pop}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{push}_k([s_1 \cdots s_m]_k) & = & [s_1 s_1 \cdots s_m]_k &
\end{array}$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\begin{array}{rcll}
\mathrm{pop}_k([s_1 \cdots s_m]_k) & = & [s_2 \cdots s_m]_k & \\
\mathrm{pop}_k([s_1 \cdots s_m]_n) & = & [\mathrm{pop}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{push}_k([s_1 \cdots s_m]_k) & = & [s_1 s_1 \cdots s_m]_k & \\
\mathrm{push}_k([s_1 \cdots s_m]_n) & = & [\mathrm{push}_k(s_1) s_2 \cdots s_m]_k & n > k
\end{array}$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\begin{array}{rcll}
\mathrm{pop}_k([s_1 \cdots s_m]_k) & = & [s_2 \cdots s_m]_k & \\
\mathrm{pop}_k([s_1 \cdots s_m]_n) & = & [\mathrm{pop}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{push}_k([s_1 \cdots s_m]_k) & = & [s_1 s_1 \cdots s_m]_k & \\
\mathrm{push}_k([s_1 \cdots s_m]_n) & = & [\mathrm{push}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{rew}_\gamma([\gamma_1 \cdots \gamma_m]_1) & = & [\gamma\gamma_2 \cdots \gamma_m]_1 &
\end{array}$$

Higher-Order Pushdown Automata

Let Γ be a stack alphabet.
S_n^Γ is the set of order-n stacks:

$$S_0^\Gamma = \Gamma \qquad\qquad S_{k+1}^\Gamma = \{[s_1 \cdots s_m]_{k+1} \mid s_1, \ldots, s_m \in S_k^\Gamma\}.$$

Operations on order-n stacks

$$\begin{array}{rcll}
\mathrm{pop}_k([s_1 \cdots s_m]_k) & = & [s_2 \cdots s_m]_k & \\
\mathrm{pop}_k([s_1 \cdots s_m]_n) & = & [\mathrm{pop}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{push}_k([s_1 \cdots s_m]_k) & = & [s_1 s_1 \cdots s_m]_k & \\
\mathrm{push}_k([s_1 \cdots s_m]_n) & = & [\mathrm{push}_k(s_1) s_2 \cdots s_m]_k & n > k \\
\mathrm{rew}_\gamma([\gamma_1 \cdots \gamma_m]_1) & = & [\gamma \gamma_2 \cdots \gamma_m]_1 & \\
\mathrm{rew}_\gamma([s_1 \cdots s_m]_n) & = & [\mathrm{rew}_\gamma(s_1) s_2 \cdots s_m]_n & n > 1
\end{array}$$

Higher-Order Pushdown Automata

Let $\Gamma = \{\bot, A_0, \ldots, A_k\}$. $\bot$: Initial stack symbol.

Higher-Order Pushdown Automata

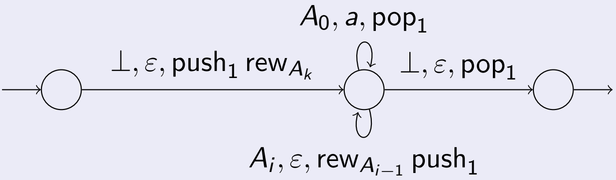
Let $\Gamma = \{\bot, A_0, \ldots, A_k\}$. $\bot$: Initial stack symbol.

Order-1 Pushdown Automaton

Higher-Order Pushdown Automata

Let $\Gamma = \{\bot, A_0, \ldots, A_k\}$. $\bot$: Initial stack symbol.

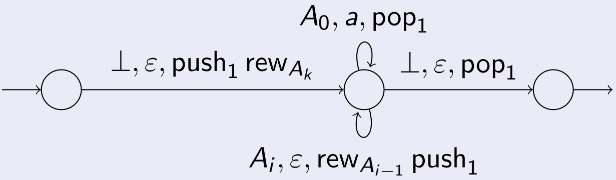
Order-1 Pushdown Automaton



Higher-Order Pushdown Automata

Let $\Gamma = \{\perp, A_0, \ldots, A_k\}$. $\perp$: Initial stack symbol.

Order-1 Pushdown Automaton

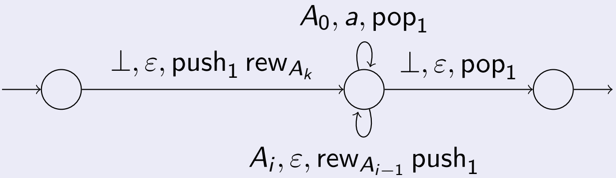


Accepts $\{a^{2^k}\}$.

Higher-Order Pushdown Automata

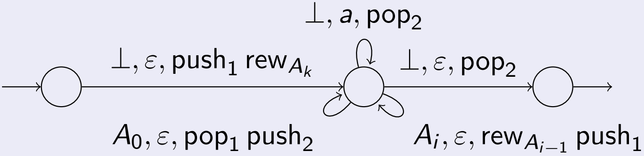
Let $\Gamma = \{\bot, A_0, \ldots, A_k\}$. $\bot$: Initial stack symbol.

Order-1 Pushdown Automaton



Accepts $\{a^{2^k}\}$.

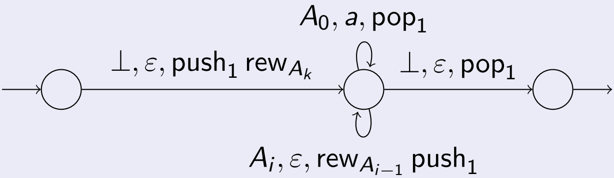
Order-2 Pushdown Automaton



Higher-Order Pushdown Automata

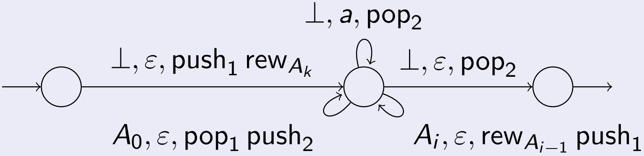
Let $\Gamma = \{\bot, A_0, \ldots, A_k\}$. $\bot$: Initial stack symbol.

Order-1 Pushdown Automaton



Accepts $\{a^{2^k}\}$.

Order-2 Pushdown Automaton



Accepts $\{a^{2^{2^k}}\}$.

Complexity

Descriptional complexity

- What is the size of an automaton for $L\downarrow$?
- How expensive the construction?

Complexity

Descriptional complexity

- What is the size of an automaton for $L{\downarrow}$?
- How expensive the construction?

Inclusion problem

For models $\mathcal{M}$ and $\mathcal{N}$ the *downward closure inclusion problem* $\mathcal{M} \subseteq_{\downarrow} \mathcal{N}$ is the following:

Given: Language K, L described in $\mathcal{M}$ and $\mathcal{N}$, respectively.

Question: Does $K{\downarrow} \subseteq L{\downarrow}$?

Complexity

Descriptional complexity

- What is the size of an automaton for $L\downarrow$?
- How expensive the construction?

Inclusion problem

For models $\mathcal{M}$ and $\mathcal{N}$ the *downward closure inclusion problem* $\mathcal{M} \subseteq_\downarrow \mathcal{N}$ is the following:

Given: Language K, L described in $\mathcal{M}$ and $\mathcal{N}$, respectively.

Question: Does $K\downarrow \subseteq L\downarrow$?

Equality problem

For models $\mathcal{M}$ and $\mathcal{N}$ the *downward closure inclusion problem* $\mathcal{M} =_\downarrow \mathcal{N}$ is the following:

Given: Language K, L described in $\mathcal{M}$ and $\mathcal{N}$, respectively.

Question: Does $K\downarrow = L\downarrow$?

Witnesses

Suppose we have an NFA for the downward closure.

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K{\downarrow} \nsubseteq L(\mathcal{A}){\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L(\mathcal{A}){\downarrow}$ with $|w| \leqslant |\mathcal{A}| + 1$.

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K\downarrow \nsubseteq L(\mathcal{A})\downarrow$, then there exists a $w \in K\downarrow \setminus L(\mathcal{A})\downarrow$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K{\downarrow} \nsubseteq L(\mathcal{A}){\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L(\mathcal{A}){\downarrow}$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges
- For an input word $w = x_1 \cdots x_n$, consider the sets Q_i of words reachable by $x_1 \cdots x_i$. Then $Q_0 \supseteq Q_1 \supseteq \cdots$

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K{\downarrow} \nsubseteq L(\mathcal{A}){\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L(\mathcal{A}){\downarrow}$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges
- For an input word $w = x_1 \cdots x_n$, consider the sets Q_i of words reachable by $x_1 \cdots x_i$. Then $Q_0 \supseteq Q_1 \supseteq \cdots$
- Hence, if $n > |\mathcal{A}| + 1$, then we can remove some symbol from w

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K{\downarrow} \nsubseteq L(\mathcal{A}){\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L(\mathcal{A}){\downarrow}$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges
- For an input word $w = x_1 \cdots x_n$, consider the sets Q_i of words reachable by $x_1 \cdots x_i$. Then $Q_0 \supseteq Q_1 \supseteq \cdots$
- Hence, if $n > |\mathcal{A}| + 1$, then we can remove some symbol from w

Suppose we have no good upper bound on an NFA.

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K\!\downarrow \not\subseteq L(\mathcal{A})\!\downarrow$, then there exists a $w \in K\!\downarrow \setminus L(\mathcal{A})\!\downarrow$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges
- For an input word $w = x_1 \cdots x_n$, consider the sets Q_i of words reachable by $x_1 \cdots x_i$. Then $Q_0 \supseteq Q_1 \supseteq \cdots$
- Hence, if $n > |\mathcal{A}| + 1$, then we can remove some symbol from w

Suppose we have no good upper bound on an NFA.

Ideal length

For $I = Y_0^* \{x_1, \varepsilon\} Y_1^* \cdots \{x_n, \varepsilon\} Y_n^*$, the *length* $|I|$ of I is the smallest n such that it can be written in this form.

Witnesses

Suppose we have an NFA for the downward closure. "Short witness":

Proposition (Bachmeier, Luttenberger, Schlund 2015)

If $\mathcal{A}$ is an NFA and $K{\downarrow} \nsubseteq L(\mathcal{A}){\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L(\mathcal{A}){\downarrow}$ with $|w| \leqslant |\mathcal{A}| + 1$.

- Suppose $\mathcal{A}$ has parallel ε-edges
- For an input word $w = x_1 \cdots x_n$, consider the sets Q_i of words reachable by $x_1 \cdots x_i$. Then $Q_0 \supseteq Q_1 \supseteq \cdots$
- Hence, if $n > |\mathcal{A}| + 1$, then we can remove some symbol from w

Suppose we have no good upper bound on an NFA.

Ideal length

For $I = Y_0^*\{x_1, \varepsilon\}Y_1^* \cdots \{x_n, \varepsilon\}Y_n^*$, the *length* $|I|$ of I is the smallest n such that it can be written in this form. Measure for languages:

$$|L| = \min\left\{\max_j |I_j| \quad : \quad L{\downarrow} = I_1 \cup \cdots \cup I_k \text{ for ideals } I_1, \ldots, I_k\right\}$$

Pumping

Putting a bound on $|L|$ amounts to proving a pumping lemma:
$|L| \leqslant m$ if and only if for every $w \in L$, there is an ideal I such that $|I| \leqslant m$ and $w \in I \subseteq L{\downarrow}$.

Pumping

Putting a bound on $|L|$ amounts to proving a pumping lemma:
$|L| \leqslant m$ if and only if for every $w \in L$, there is an ideal I such that $|I| \leqslant m$ and $w \in I \subseteq L\downarrow$.

Proposition (Ideal witness)

Let $I = Y_0^*\{x_1, \varepsilon\}Y_1^* \cdots \{x_n, \varepsilon\}Y_n^*$. Then the following are equivalent:

1. $I \subseteq L\downarrow$.
2. $w_{Y_0}^m x_1 w_{Y_1}^m \cdots x_n w_{Y_n}^m \in L\downarrow$ *for every* $m \geqslant |L| + 1$.
3. $w_{Y_0}^m x_1 w_{Y_1}^m \cdots x_n w_{Y_n}^m \in L\downarrow$ *for some* $m \geqslant |L| + 1$.

Pumping

Putting a bound on $|L|$ amounts to proving a pumping lemma: $|L| \leqslant m$ if and only if for every $w \in L$, there is an ideal I such that $|I| \leqslant m$ and $w \in I \subseteq L{\downarrow}$.

Proposition (Ideal witness)

Let $I = Y_0^*\{x_1, \varepsilon\}Y_1^* \cdots \{x_n, \varepsilon\}Y_n^*$. Then the following are equivalent:

1. $I \subseteq L{\downarrow}$.
2. $w_{Y_0}^m x_1 w_{Y_1}^m \cdots x_n w_{Y_n}^m \in L{\downarrow}$ *for every* $m \geqslant |L| + 1$.
3. $w_{Y_0}^m x_1 w_{Y_1}^m \cdots x_n w_{Y_n}^m \in L{\downarrow}$ *for some* $m \geqslant |L| + 1$.

Strategy for $K{\downarrow} \subseteq L{\downarrow}$

- Suppose $|K|$ is polynomial and $|L|$ exponential.
- Guess an ideal I of length $\leqslant |K|$ and verify $I \subseteq K$, $I \nsubseteq L$.
- Represent witness above succinctly.

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K{\downarrow} \nsubseteq L{\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L{\downarrow}$ with $|w| \leqslant |X| \cdot (|L| + 1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K{\downarrow} \nsubseteq L{\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L{\downarrow}$ with $|w| \leqslant |X| \cdot (|L|+1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.
- Turn every ideal into an *ordered DFA*: no cycles except self-loops

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K{\downarrow} \nsubseteq L{\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L{\downarrow}$ with $|w| \leqslant |X| \cdot (|L| + 1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.
- Turn every ideal into an *ordered DFA*: no cycles except self-loops
- Then prove the following:

Lemma

If $w \in X^*$ with $|w| > |X| \cdot (n-1)^{|X|}$, then w has a position at which every ordered n-state DFA cycles.

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K{\downarrow} \not\subseteq L{\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L{\downarrow}$ with $|w| \leqslant |X| \cdot (|L| + 1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.
- Turn every ideal into an *ordered DFA*: no cycles except self-loops
- Then prove the following:

Lemma

If $w \in X^*$ with $|w| > |X| \cdot (n-1)^{|X|}$, then w has a position at which every ordered n-state DFA cycles.

- Induction yields length bound for subwords with less than $|X|$ symbols

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K\!\downarrow \not\subseteq L\!\downarrow$, then there exists a $w \in K\!\downarrow \setminus L\!\downarrow$ with $|w| \leqslant |X| \cdot (|L| + 1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.
- Turn every ideal into an *ordered DFA*: no cycles except self-loops
- Then prove the following:

Lemma

If $w \in X^*$ with $|w| > |X| \cdot (n-1)^{|X|}$, then w has a position at which every ordered n-state DFA cycles.

- Induction yields length bound for subwords with less than $|X|$ symbols
- Decompose w into at least n factors each of which sees all of X

Sometimes, we have a small bound on $|L|$ but only a large bound on $|K|$.

Proposition (Small alphabet witness)

Let $K, L \subseteq X^*$. If $K{\downarrow} \not\subseteq L{\downarrow}$, then there exists a $w \in K{\downarrow} \setminus L{\downarrow}$ with $|w| \leqslant |X| \cdot (|L| + 1)^{|X|}$.

- This yields existence of small witnesses for fixed alphabets.
- Turn every ideal into an *ordered DFA*: no cycles except self-loops
- Then prove the following:

Lemma

If $w \in X^*$ with $|w| > |X| \cdot (n-1)^{|X|}$, then w has a position at which every ordered n-state DFA cycles.

- Induction yields length bound for subwords with less than $|X|$ symbols
- Decompose w into at least n factors each of which sees all of X
- Each DFA has to repeat a state, hence cycle on the last letter of w

Lower bounds

Theorem

There are order-n pushdown automata whose downward closure NFAs are at least n-fold exponential.

Lower bounds

Theorem

There are order-n pushdown automata whose downward closure NFAs are at least n-fold exponential.

- An NFA for $\{w\}{\downarrow}$ requires $|w| + 1$ states.

Lower bounds

Theorem

There are order-n pushdown automata whose downward closure NFAs are at least n-fold exponential.

- An NFA for $\{w\}\downarrow$ requires $|w| + 1$ states.
- Examples for $\{a^{2^k}\}$ and $\{a^{2^{2^k}}\}$ extend easily to order n.

Lower bounds

Theorem

There are order-n pushdown automata whose downward closure NFAs are at least n-fold exponential.

- An NFA for $\{w\}{\downarrow}$ requires $|w| + 1$ states.
- Examples for $\{a^{2^k}\}$ and $\{a^{2^{2^k}}\}$ extend easily to order n.

Theorem

Under mild conditions on the models $\mathcal{M}$ and $\mathcal{N}$: Suppose for each n we have a description of $\{a^{t(n)}\}$ in $\mathcal{M}$ and $\mathcal{N}$ of polynomial size. Then $\mathcal{M} \subseteq_{\downarrow} \mathcal{N}$ is hard for $\mathsf{coNTIME}(t)$.

Lower bounds

Theorem

There are order-n pushdown automata whose downward closure NFAs are at least n-fold exponential.

- An NFA for $\{w\}{\downarrow}$ requires $|w| + 1$ states.
- Examples for $\{a^{2^k}\}$ and $\{a^{2^{2^k}}\}$ extend easily to order n.

Theorem

Under mild conditions on the models $\mathcal{M}$ and $\mathcal{N}$: Suppose for each n we have a description of $\{a^{t(n)}\}$ in $\mathcal{M}$ and $\mathcal{N}$ of polynomial size. Then $\mathcal{M} \subseteq_{\downarrow} \mathcal{N}$ is hard for coNTIME(t).

Corollary

The problem $\mathrm{HOPA}_n \subseteq_{\downarrow} \mathrm{HOPA}_n$ *is* co-n-NEXP*-hard.*

	Ideal	NFA	OCA	$RBC_{k,r}$	CFG	RBC
Ideal	$\in$ L	NL	NL	NL	P	NP
NFA	NL	coNP	coNP	coNP	coNP	Π_2^P
OCA	NL	coNP	coNP	coNP	coNP	Π_2^P
$RBC_{k,r}$	NL	coNP	coNP	coNP	coNP	Π_2^P
CFG	P	coNP	coNP	$coNP^{\dagger}$	coNEXP	coNEXP
RBC	coNP	coNP	coNP	$coNP^{\dagger}$	coNEXP	coNEXP

	Ideal	NFA	OCA	$RBC_{k,r}$	CFG	RBC
Ideal	$\in$ L	NL	NL	NL	P	NP
NFA	NL	coNP	coNP	coNP	coNP	Π_2^P
OCA	NL	coNP	coNP	coNP	coNP	Π_2^P
$RBC_{k,r}$	NL	coNP	coNP	coNP	coNP	Π_2^P
CFG	P	coNP	coNP	coNP†	coNEXP	coNEXP
RBC	coNP	coNP	coNP	coNP†	coNEXP	coNEXP

- Often better complexity than constructing an NFA:
  - NFA $\subseteq_\downarrow$ CFG: NFA for the CFG would be exponential, but problem is coNP-complete.

	Ideal	NFA	OCA	$RBC_{k,r}$	CFG	RBC
Ideal	$\in$ L	NL	NL	NL	P	NP
NFA	NL	coNP	coNP	coNP	coNP	Π_2^P
OCA	NL	coNP	coNP	coNP	coNP	Π_2^P
$RBC_{k,r}$	NL	coNP	coNP	coNP	coNP	Π_2^P
CFG	P	coNP	coNP	$coNP^\dagger$	coNEXP	coNEXP
RBC	coNP	coNP	coNP	$coNP^\dagger$	coNEXP	coNEXP

- Often better complexity than constructing an NFA:
  - NFA $\subseteq_\downarrow$ CFG: NFA for the CFG would be exponential, but problem is coNP-complete.
  - $HOPA_2 \subseteq_\downarrow$ NFA is EXPTIME-complete: Using "short witness". Hardness inherited from emptiness.

	Ideal	NFA	OCA	$RBC_{k,r}$	CFG	RBC
Ideal	$\in L$	NL	NL	NL	P	NP
NFA	NL	coNP	coNP	coNP	coNP	Π_2^P
OCA	NL	coNP	coNP	coNP	coNP	Π_2^P
$RBC_{k,r}$	NL	coNP	coNP	coNP	coNP	Π_2^P
CFG	P	coNP	coNP	$coNP^{\dagger}$	coNEXP	coNEXP
RBC	coNP	coNP	coNP	$coNP^{\dagger}$	coNEXP	coNEXP

- Often better complexity than constructing an NFA:
  - NFA $\subseteq_\downarrow$ CFG: NFA for the CFG would be exponential, but problem is coNP-complete.
  - $HOPA_2 \subseteq_\downarrow$ NFA is EXPTIME-complete: Using "short witness". Hardness inherited from emptiness.
- Pumping lemmas may give us bounds on ideal lengths

	Ideal	NFA	OCA	$\mathrm{RBC}_{k,r}$	CFG	RBC
Ideal	$\in L$	NL	NL	NL	P	NP
NFA	NL	coNP	coNP	coNP	coNP	Π_2^P
OCA	NL	coNP	coNP	coNP	coNP	Π_2^P
$\mathrm{RBC}_{k,r}$	NL	coNP	coNP	coNP	coNP	Π_2^P
CFG	P	coNP	coNP	$\mathrm{coNP}^\dagger$	coNEXP	coNEXP
RBC	coNP	coNP	coNP	$\mathrm{coNP}^\dagger$	coNEXP	coNEXP

- Often better complexity than constructing an NFA:
  - NFA $\subseteq_\downarrow$ CFG: NFA for the CFG would be exponential, but problem is coNP-complete.
  - $\mathrm{HOPA}_2 \subseteq_\downarrow$ NFA is EXPTIME-complete: Using "short witness". Hardness inherited from emptiness.
- Pumping lemmas may give us bounds on ideal lengths
- The problem $\mathrm{HOPA}_n \subseteq_\downarrow \mathrm{HOPA}_n$ is co-n-NEXP-hard.