Martin Zimmermann
Aalborg University

History-deterministic Pushdown Automata

Based on joint work with Shibashis Guha, Ismaël Jecker and Karoliina Lehtinen

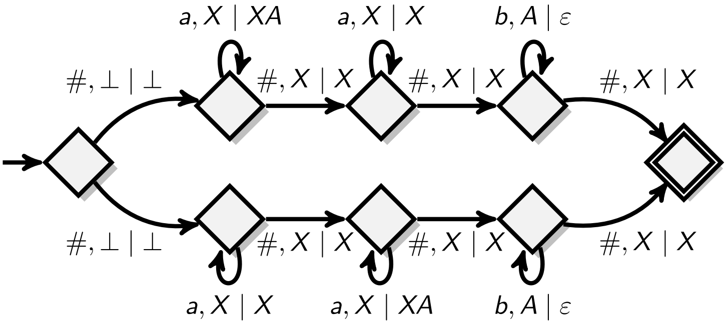
IRIF, Paris, August 2024

Let's Get Started

$$L_2 = \{\# a^i \# a^j \# b^k \# \mid k \leq i \text{ or } k \leq j\}$$

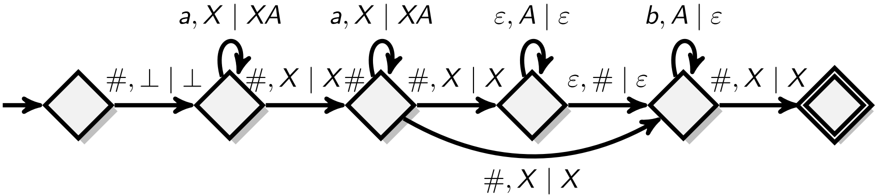
Let's Get Started

$$L_2 = \{\#\, a^i \,\#\, a^j \,\#\, b^k \,\# \mid k \leq i \text{ or } k \leq j\}$$



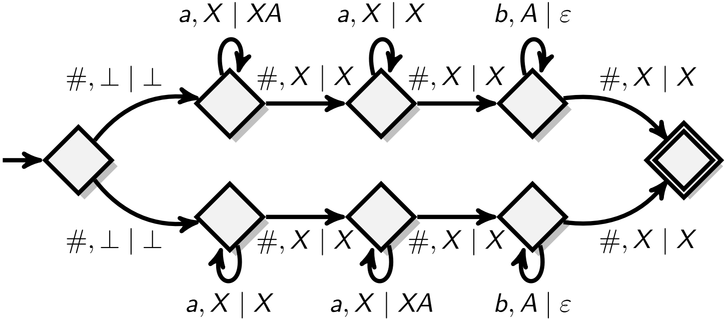
Let's Get Started

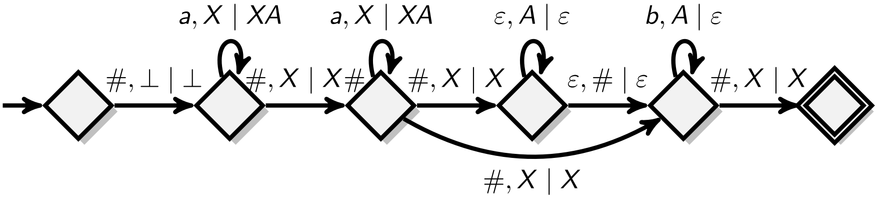
$$L_2 = \{\#\, a^i \,\#\, a^j \,\#\, b^k \,\# \mid k \leq i \text{ or } k \leq j\}$$



Let's Get Started

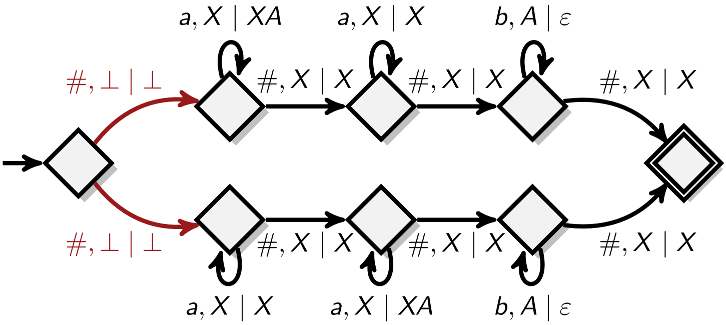
$$L_2 = \{\#\, a^i \,\#\, a^j \,\#\, b^k \,\# \mid k \leq i \text{ or } k \leq j\}$$





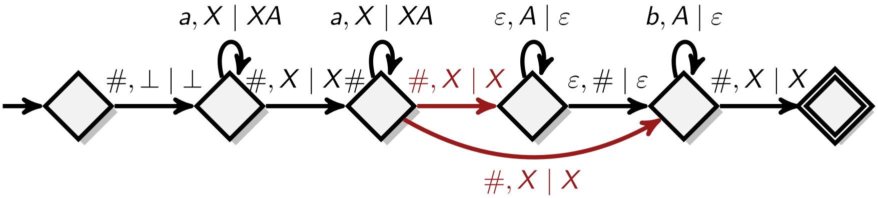
Not All Nondeterminism is Equal

- In the first automaton, one needs to know the whole word to make the nondeterministic choice.



Not All Nondeterminism is Equal

- In the first automaton, one needs to know the whole word to make the nondeterministic choice.
- In the second automaton, one only needs to know the prefix processed so far to make the nondeterministic choice.



Another Language

- Let $I = \{0, +, -\}$ and define the energy level $EL \colon I^* \to \mathbb{Z}$ of a finite word over I as

$$EL(w) = |w|_+ - |w|_-,$$

where $|w|_\circ$ is the number of $\circ$ in w, for $\circ \in I$.

Another Language

- Let $I = \{0, +, -\}$ and define the energy level $EL \colon I^* \to \mathbb{Z}$ of a finite word over I as

$$EL(w) = |w|_+ - |w|_-,$$

  where $|w|_\circ$ is the number of $\circ$ in w, for $\circ \in I$.
- An infinite word $w \in I^\omega$ is safe if $EL(w(0) \cdots w(n)) \geq 0$ for every $n \geq 0$.
- A word $w \in I^\omega$ is eventually safe if it has a safe suffix.

Another Language

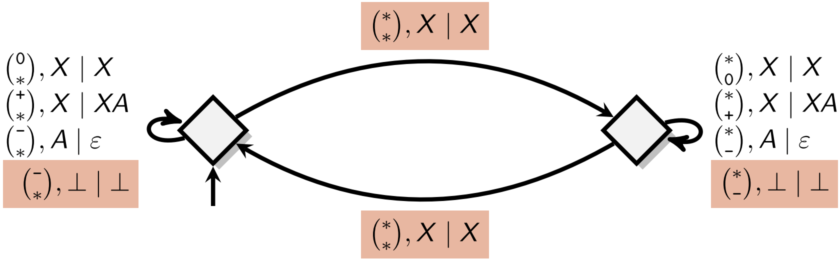
- Let $I = \{0, +, -\}$ and define the energy level $EL \colon I^* \to \mathbb{Z}$ of a finite word over I as

$$EL(w) = |w|_+ - |w|_-,$$

  where $|w|_\circ$ is the number of $\circ$ in w, for $\circ \in I$.
- An infinite word $w \in I^\omega$ is safe if $EL(w(0) \cdots w(n)) \geq 0$ for every $n \geq 0$.
- A word $w \in I^\omega$ is eventually safe if it has a safe suffix.
- Let $\Sigma = I \times I$ and

$$L_{es} = \left\{ \binom{w_0}{w_1} \in \Sigma^\omega \;\middle|\; \text{some } w_i \text{ is eventually safe} \right\}.$$

An Automaton for L_{es}



Acceptance condition: Red transitions may only be taken finitely often.

Why Should You Care?

Why Should You Care?

Because you want to solve infinite games with contextfree winning conditions!

Why Should You Care?

Because you want to solve infinite games with contextfree winning conditions!

- Two players move a token through a finite graph without dead ends whose vertices are labeled by Σ, thereby constructing an infinite word ρ over Σ.
- Protagonist wins if ρ is in some fixed winning condition, an ω-language over Σ.
- Many applications in automata theory, logic, reactive synthesis, etc.

Why Should You Care?

Because you want to solve infinite games with contextfree winning conditions!

- Two players move a token through a finite graph without dead ends whose vertices are labeled by Σ, thereby constructing an infinite word ρ over Σ.
- Protagonist wins if ρ is in some fixed winning condition, an ω-language over Σ.
- Many applications in automata theory, logic, reactive synthesis, etc.

Theorem (Finkel 2001, Walukiewicz 2001)

- *The following problem is undecidable: Does protagonist win a game with ω-contextfree winning condition?*

Why Should You Care?

Because you want to solve infinite games with contextfree winning conditions!

- Two players move a token through a finite graph without dead ends whose vertices are labeled by Σ, thereby constructing an infinite word ρ over Σ.
- Protagonist wins if ρ is in some fixed winning condition, an ω-language over Σ.
- Many applications in automata theory, logic, reactive synthesis, etc.

Theorem (Finkel 2001, Walukiewicz 2001)

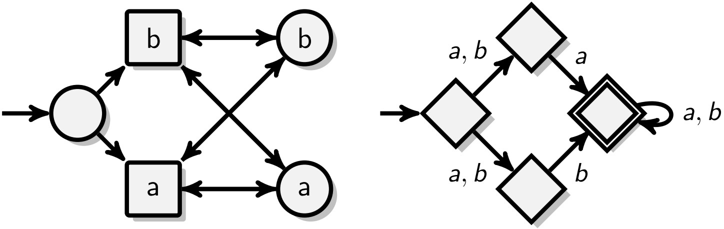
- *The following problem is undecidable: Does protagonist win a game with ω-contextfree winning condition?*
- *The following problem is* ExpTime*-complete: Does protagonist win a game with deterministic ω-contextfree winning condition?*

But..

.. why can't we just let protagonist "resolve" nondeterministic choices on-the-fly during a play?

But..

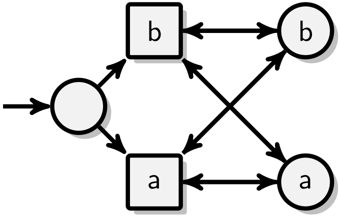
.. why can't we just let protagonist "resolve" nondeterministic choices on-the-fly during a play?

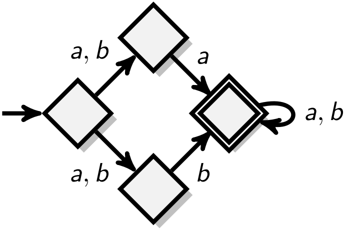


- Every play is winning for protagonist (round vertices), but..

But..

.. why can't we just let protagonist "resolve" nondeterministic choices on-the-fly during a play?





- Every play is winning for protagonist (round vertices), but..
- protagonist loses if she has to resolve nondeterminism on-the-fly.

History-Determinism

Let $\mathcal{P} = (Q, \Sigma, \Gamma, q_I, \Delta, \Omega)$ be a nondeterministic PDA.

- A resolver for $\mathcal{P}$ is a function $r\colon \Delta^* \times \Sigma \to \Delta$ such that for every $w \in L(\mathcal{P})$ the sequence $\tau_0\tau_1\tau_2 \cdots \in \Delta^\omega$ defined as

  $$\tau_n = r(\tau_0 \cdots \tau_{n-1}, w(|(\tau_0 \cdots \tau_{n-1})|_\Sigma))$$

  induces an accepting run of $\mathcal{P}$ on w.

  Here, $|(\tau_0 \cdots \tau_{n-1})|_\Sigma$ denotes the number of letters processed by the transitions $\tau_0 \cdots \tau_{n-1}$ ($\mathcal{P}$ may have ε-transitions), i.e., $w(|(\tau_0 \cdots \tau_{n-1})|_\Sigma)$ is the first letter of w not processed by $\tau_0 \cdots \tau_{n-1}$.

- $\mathcal{P}$ is history-deterministic if it has a resolver.

History-Determinism

Let $\mathcal{P} = (Q, \Sigma, \Gamma, q_I, \Delta, \Omega)$ be a nondeterministic PDA.

- A resolver for $\mathcal{P}$ is a function $r\colon \Delta^* \times \Sigma \to \Delta$ such that for every $w \in L(\mathcal{P})$ the sequence $\tau_0\tau_1\tau_2\cdots \in \Delta^\omega$ defined as

  $$\tau_n = r(\tau_0 \cdots \tau_{n-1}, w(|(\tau_0 \cdots \tau_{n-1})|_\Sigma))$$

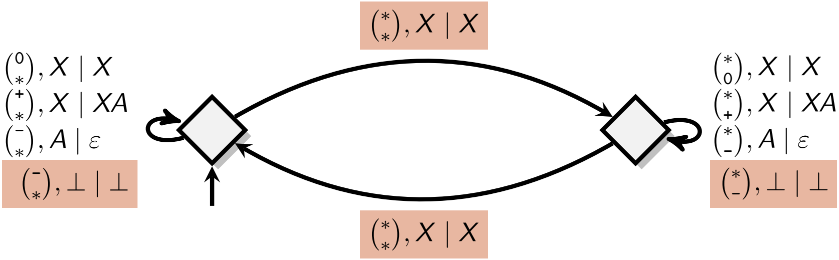
  induces an accepting run of $\mathcal{P}$ on w.

  Here, $|(\tau_0 \cdots \tau_{n-1})|_\Sigma$ denotes the number of letters processed by the transitions $\tau_0 \cdots \tau_{n-1}$ ($\mathcal{P}$ may have ε-transitions), i.e., $w(|(\tau_0 \cdots \tau_{n-1})|_\Sigma)$ is the first letter of w not processed by $\tau_0 \cdots \tau_{n-1}$.

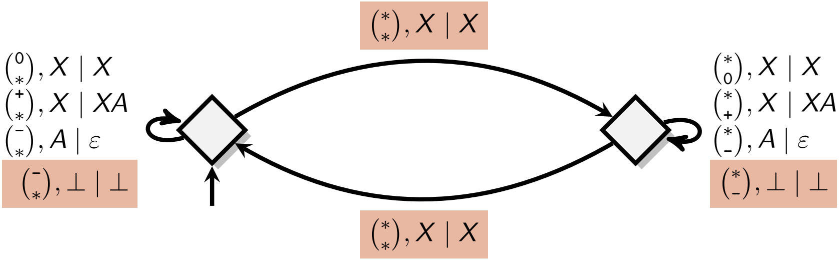
- $\mathcal{P}$ is history-deterministic if it has a resolver.

- Definition slightly more tedious for finite words.

Back to the Example

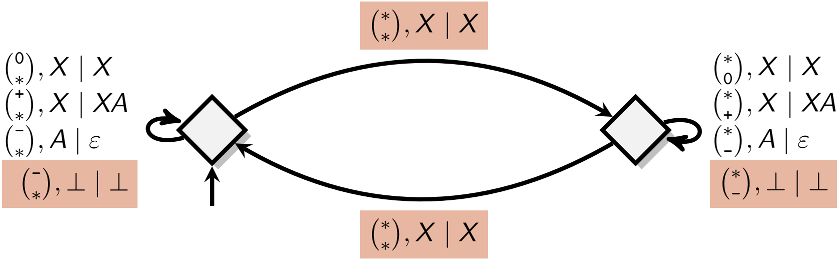


Back to the Example



- Let us define a resolver for the ω-PDA above.
- Given $w = \binom{w_0^0}{w_0^1} \cdots \binom{w_n^0}{w_n^1}$ let m^i for $i \in \{0, 1\}$ be the minimal m such that $w_m^i \cdots w_n^i$ is safe.

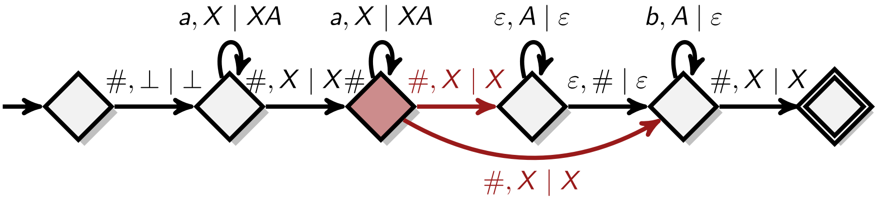
Back to the Example



- Let us define a resolver for the ω-PDA above.
- Given $w = \binom{w_0^0}{w_0^1} \cdots \binom{w_n^0}{w_n^1}$ let m^i for $i \in \{0, 1\}$ be the minimal m such that $w_m^i \cdots w_n^i$ is safe.
- Then, we define the resolver to guide the run
  - to the left state, if $m^0 \leq m^1$, and
  - to the right state otherwise.

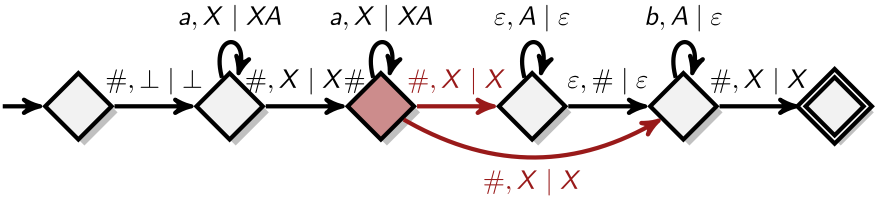
And the Other One

$$L_2 = \{\# \, a^i \, \# \, a^j \, \# \, b^k \, \# \mid k \le i \text{ or } k \le j\}$$



And the Other One

$$L_2 = \{\# \, a^i \, \# \, a^j \, \# \, b^k \, \# \mid k \leq i \text{ or } k \leq j\}$$



- Let us again define a resolver for the PDA.
- In the red state, after having processed $\# \, a^i \, \# \, a^j$ and the next letter being $\#$, select
  - the upper transition if $j < i$, and
  - the lower transition otherwise.

But Have we Gained Anything?

Theorem (Lehtinen, Z. 2020)

- *L_{es} is history-deterministic ω-contextfree, but not deterministic ω-contextfree.*

But Have we Gained Anything?

Theorem (Lehtinen, Z. 2020)

- *L_{es} is history-deterministic ω-contextfree, but not deterministic ω-contextfree.*
- *There are ω-contextfree languages that are not history-deterministic ω-contextfree.*

But Have we Gained Anything?

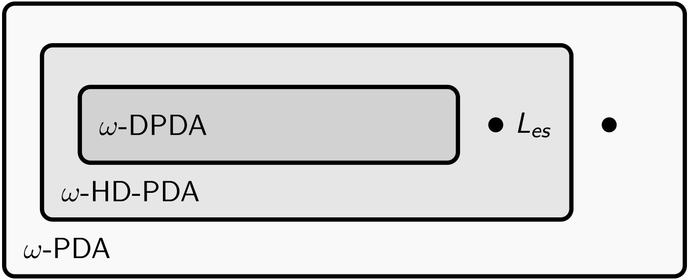
Theorem (Lehtinen, Z. 2020)

- *L_{es} is history-deterministic ω-contextfree, but not deterministic ω-contextfree.*
- *There are ω-contextfree languages that are not history-deterministic ω-contextfree.*

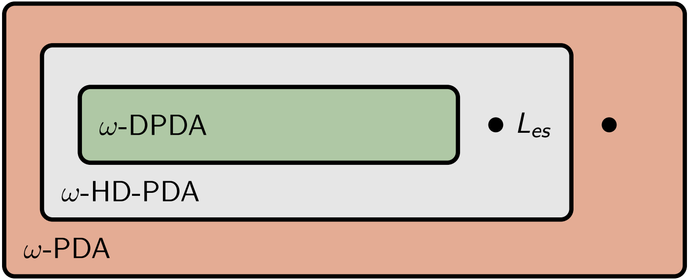
Theorem (Guha, Jecker, Lehtinen, Z. 2021)

- *L_2 is history-deterministic contextfree, but not deterministic contextfree.*
- *There are contextfree languages that are not history-deterministic contextfree.*

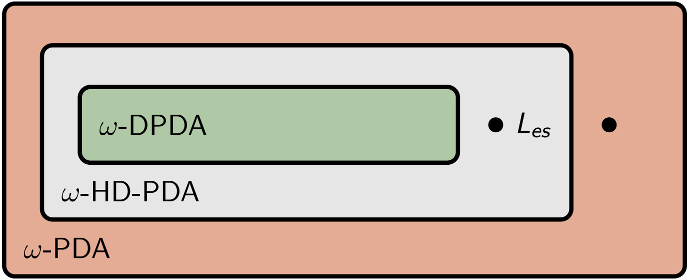
The Big Picture



The Big Picture



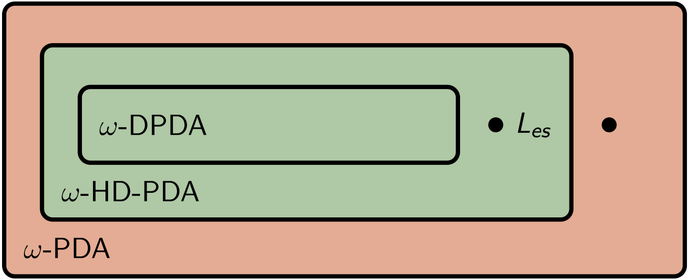
The Big Picture



Theorem (Lehtinen, Z. 2020)

The following problem is EXPTIME*-complete: Does protagonist win a game with history-deterministic* ω*-contextfree winning condition?*

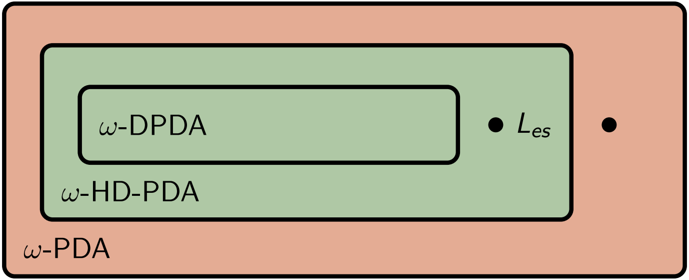
The Big Picture



Theorem (Lehtinen, Z. 2020)

The following problem is ExpTime*-complete: Does protagonist win a game with history-deterministic ω-contextfree winning condition?*

The Big Picture



Theorem (Lehtinen, Z. 2020)

The following problem is EXPTIME*-complete: Does protagonist win a game with history-deterministic ω-contextfree winning condition?*

Corollary

Universality for history-deterministic PDA is in EXPTIME.

Closure Properties

Theorem (Lehtinen, Z. 2020)

The history-deterministic contextfree languages have poor closure properties.

Theorem (Lehtinen, Z. 2020)

The history-deterministic contextfree languages have poor closure properties.

	$\cap$	$\cup$	$\overline{}$	$\setminus$	h
ω-DPDA	✗	✗	✓	✗	✗
ω-HD-PDA	✗	✗	✗	✗	✗
ω-VPA	✓	✓	✓	✓	✗
ω-PDA	✗	✓	✗	✗	✓

Decidability

History-determinism is a semantic definition, so it is not trivial to determine whether a PDA is history-deterministic.

Decidability

History-determinism is a semantic definition, so it is not trivial to determine whether a PDA is history-deterministic.

Theorem (Lehtinen, Z. 2020)

The following problems are undecidable:

1. *Given an ω-PDA $\mathcal{P}$, is $\mathcal{P}$ history-deterministic?*
2. *Is a given ω-PDA $\mathcal{P}$ equivalent to some history-deterministic PDA?*

Similar results hold for PDA on finite words.

Also in the Papers

- A comparison to unambiguous contextfree languages: incomparable (over finite words)

Also in the Papers

- A comparison to unambiguous contextfree languages: incomparable (over finite words)
- A comparison to visibly pushdown languages: incomparable (over infinite words)

Also in the Papers

- A comparison to unambiguous contextfree languages: incomparable (over finite words)
- A comparison to visibly pushdown languages: incomparable (over infinite words)
- History-deterministic visibly pushdown languages: better properties*

Also in the Papers

- A comparison to unambiguous contextfree languages: incomparable (over finite words)
- A comparison to visibly pushdown languages: incomparable (over infinite words)
- History-deterministic visibly pushdown languages: better properties*
- The parity-index hierarchy of history-deterministic context-free language: infinite

Also in the Papers

- A comparison to unambiguous contextfree languages: incomparable (over finite words)
- A comparison to visibly pushdown languages: incomparable (over infinite words)
- History-deterministic visibly pushdown languages: better properties*
- The parity-index hierarchy of history-deterministic context-free language: infinite
- A comparison to good-for-games pushdown automata: not equivalent!!!

How Hard is it to Resolve Nondeterminism?

- Recall: A resolver is a function $r\colon \Delta^* \times \Sigma \to \Delta$ that induces an accepting run for every accepted word.

How Hard is it to Resolve Nondeterminism?

- Recall: A resolver is a function $r\colon \Delta^* \times \Sigma \to \Delta$ that induces an accepting run for every accepted word.
- Such functions can be implemented by automata with output.

How Hard is it to Resolve Nondeterminism?

- Recall: A resolver is a function $r\colon \Delta^* \times \Sigma \to \Delta$ that induces an accepting run for every accepted word.
- Such functions can be implemented by automata with output.
- But how "complex" do resolvers for history-deterministic PDA need to be?

How Hard is it to Resolve Nondeterminism?

- Recall: A resolver is a function $r\colon \Delta^* \times \Sigma \rightarrow \Delta$ that induces an accepting run for every accepted word.
- Such functions can be implemented by automata with output.
- But how "complex" do resolvers for history-deterministic PDA need to be?

Theorem (Lehtinen, Z. 2020)

- *If a history-deterministic PDA has a finite-state resolver, then it can be determinized by taking the product of the PDA and the (finite-state implementation of the) resolver.*

How Hard is it to Resolve Nondeterminism?

- Recall: A resolver is a function $r\colon \Delta^* \times \Sigma \to \Delta$ that induces an accepting run for every accepted word.
- Such functions can be implemented by automata with output.
- But how "complex" do resolvers for history-deterministic PDA need to be?

Theorem (Lehtinen, Z. 2020)

- *If a history-deterministic PDA has a finite-state resolver, then it can be determinized by taking the product of the PDA and the (finite-state implementation of the) resolver.*
- *Not every history-deterministic PDA has a resolver that is implementable by a PDA with output.*

Counterexample

- Recall

$$L_2 = \{\# a^i \# a^j \# b^k \# \mid k \leq i \text{ or } k \leq j\}.$$

It has resolver that is implementable by a PDA with output (after processing $\# a^i \# a^j$, we need to check $j < i$).

Counterexample

- Recall

$$L_2 = \{\# a^i \# a^j \# b^k \# \mid k \leq i \text{ or } k \leq j\}.$$

It has resolver that is implementable by a PDA with output (after processing $\# a^i \# a^j$, we need to check $j < i$).

- So, consider

$$L_3 = \{\# a^h \# a^i \# a^j \# b^k \# \mid k \leq h \text{ or } k \leq i \text{ or } k \leq j\},$$

which is also history-deterministic.

Counterexample

- Recall

$$L_2 = \{\# a^i \# a^j \# b^k \# \mid k \leq i \text{ or } k \leq j\}.$$

It has resolver that is implementable by a PDA with output (after processing $\# a^i \# a^j$, we need to check $j < i$).

- So, consider

$$L_3 = \{\# a^h \# a^i \# a^j \# b^k \# \mid k \leq h \text{ or } k \leq i \text{ or } k \leq j\},$$

which is also history-deterministic.

- To resolve the nondeterminism here, we need to check, e.g., $i < h \wedge j < h$. This cannot be achieved by a PDA.

Counterexample

- Recall

$$L_2 = \{\#\, a^i \,\#\, a^j \,\#\, b^k \,\#\mid k \leq i \text{ or } k \leq j\}.$$

It has resolver that is implementable by a PDA with output (after processing $\#\, a^i \,\#\, a^j$, we need to check $j < i$).

- So, consider

$$L_3 = \{\#\, a^h \,\#\, a^i \,\#\, a^j \,\#\, b^k \,\#\mid k \leq h \text{ or } k \leq i \text{ or } k \leq j\},$$

which is also history-deterministic.

- To resolve the nondeterminism here, we need to check, e.g., $i < h \wedge j < h$. This cannot be achieved by a PDA.

Open Problem

Does every history-deterministic PDA have a computable resolver?

Succinctness

Theorem (Valiant 1976)

There is no computable function f with the following property: For every n-state PDA accepting a deterministic contextfree language, there is an equivalent deterministic PDA with $f(n)$ states.

Succinctness

Theorem (Valiant 1976)

There is no computable function f with the following property: For every n-state PDA accepting a deterministic contextfree language, there is an equivalent deterministic PDA with $f(n)$ states.

- Due to ω-DCFL $\subsetneq$ ω-HD-CFL $\subsetneq$ ω-CFL, we know that at least one of the succinctness gaps involving history-deterministic PDA must be noncomputable.

Open Problem

Which of the two succinctness gaps is noncomputable?

Succinctness

Theorem (Valiant 1976)

There is no computable function f with the following property: For every n-state PDA accepting a deterministic contextfree language, there is an equivalent deterministic PDA with $f(n)$ states.

- Due to ω-DCFL $\subsetneq$ ω-HD-CFL $\subsetneq$ ω-CFL, we know that at least one of the succinctness gaps involving history-deterministic PDA must be noncomputable.

Open Problem

Which of the two succinctness gaps is noncomputable?

Theorem (Guha, Jecker, Lehtinen, Z. 2021)

There are exponential and doubly-exponential lower bounds.

More Open Problems

- Is universality for history-deterministic PDA EXPTIME-hard?

More Open Problems

- Is universality for history-deterministic PDA EXPTIME-hard?
- Is it decidable whether a history-deterministic PDA has an equivalent deterministic PDA?

More Open Problems

- Is universality for history-deterministic PDA ExpTime-hard?
- Is it decidable whether a history-deterministic PDA has an equivalent deterministic PDA?
- Is equivalence of history-deterministic PDA (over finite words) decidable?

More Open Problems

- Is universality for history-deterministic PDA ExpTime-hard?
- Is it decidable whether a history-deterministic PDA has an equivalent deterministic PDA?
- Is equivalence of history-deterministic PDA (over finite words) decidable?
- Is there a class of grammars that is equivalent to history-deterministic PDA?

More Open Problems

- Is universality for history-deterministic PDA EXPTIME-hard?
- Is it decidable whether a history-deterministic PDA has an equivalent deterministic PDA?
- Is equivalence of history-deterministic PDA (over finite words) decidable?
- Is there a class of grammars that is equivalent to history-deterministic PDA?
- Is there an extension of MSO that is equivalent to history-deterministic PDA?

More Open Problems

- Is universality for history-deterministic PDA ExpTime-hard?
- Is it decidable whether a history-deterministic PDA has an equivalent deterministic PDA?
- Is equivalence of history-deterministic PDA (over finite words) decidable?
- Is there a class of grammars that is equivalent to history-deterministic PDA?
- Is there an extension of MSO that is equivalent to history-deterministic PDA?

Thank you!
Questions and more open problems?