On the Universality Problem for Unambiguous Register Automata

joint work with Wojtek Czerwinski & Antoine Mottet

Karin Quaas
Universität Leipzig

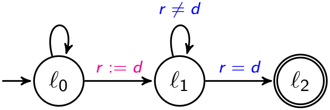
- Extension of finite automata to infinite alphabets ($\mathbb{N}$)

- An accepting run on the input word 2 5 2 5 3 5:

$$(\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_1, 5) \xrightarrow{3} (\ell_1, 5) \xrightarrow{5} (\ell_2, 5)$$

- $L(\mathcal{A}) = \{d_1 \dots d_k \mid \exists 1 \leq i < k. d_i = d_k\}$
- Emptiness problem is decidable (PSPACE-complete)

- Extension of finite automata to infinite alphabets ($\mathbb{N}$)

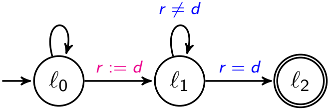


- An accepting run on the input word 2 5 2 5 3 5:

$$(\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_1, 5) \xrightarrow{3} (\ell_1, 5) \xrightarrow{5} (\ell_2, 5)$$

- $L(\mathcal{A}) = \{d_1 \ldots d_k \mid \exists 1 \leq i < k. d_i = d_k\}$
- Emptiness problem is decidable (PSPACE-complete)

- Extension of finite automata to infinite alphabets ($\mathbb{N}$)

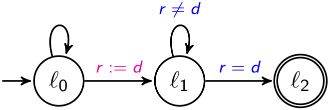


- An accepting run on the input word 2 5 2 5 3 5:

$$(\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_0, \bot) \xrightarrow{2} (\ell_0, \bot) \xrightarrow{5} (\ell_1, 5) \xrightarrow{3} (\ell_1, 5) \xrightarrow{5} (\ell_2, 5)$$

- $L(\mathcal{A}) = \{d_1 \dots d_k \mid \exists 1 \leq i < k . d_i = d_k\}$
- Emptiness problem is decidable (PSPACE-complete)

- Extension of finite automata to infinite alphabets ($\mathbb{N}$)

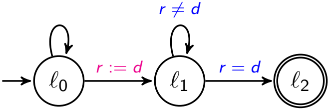


- An accepting run on the input word 2 5 2 5 3 5:

$$(\ell_0, \perp) \xrightarrow{2} (\ell_0, \perp) \xrightarrow{5} (\ell_0, \perp) \xrightarrow{2} (\ell_0, \perp) \xrightarrow{5} (\ell_1, 5) \xrightarrow{3} (\ell_1, 5) \xrightarrow{5} (\ell_2, 5)$$

- $L(\mathcal{A}) = \{d_1 \dots d_k \mid \exists 1 \leq i < k . d_i = d_k\}$
- Emptiness problem is decidable (PSPACE-complete)

- Extension of finite automata to infinite alphabets ($\mathbb{N}$)



- An accepting run on the input word 2 5 2 5 3 5:

$$(\ell_0,\bot) \xrightarrow{2} (\ell_0,\bot) \xrightarrow{5} (\ell_0,\bot) \xrightarrow{2} (\ell_0,\bot) \xrightarrow{5} (\ell_1,5) \xrightarrow{3} (\ell_1,5) \xrightarrow{5} (\ell_2,5)$$

- $L(\mathcal{A}) = \{d_1 \dots d_k \mid \exists 1 \le i < k. d_i = d_k\}$
- Emptiness problem is decidable (PSPACE-complete)

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

Definition

An automaton is unambiguous if every input word has at most one accepting run

- DETerministic $\subseteq$ UNAmbiguous $\subseteq$ NONDETerministic
- Collapses or non-collapses depending on model of computation
- UNA automata are often more succinct than DET automata
- UNA automata often better complexity than NONDET automata
- UNA interesting from theoretical point of view

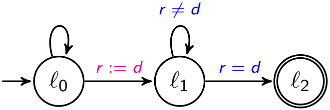
- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \dots d_k \mid \exists 1 \leq i < k . d_i = d_k\}$

- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k . d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \dots d_k \mid \exists 1 \leq i < k . d_i = d_k\}$

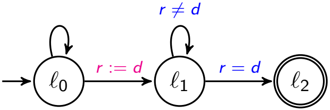
- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k . d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \dots d_k \mid \exists 1 \leq i < k. d_i = d_k\}$



- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

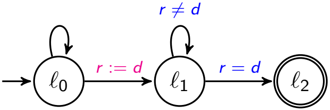
- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \ldots d_k \mid \exists 1 \leq i < k. d_i = d_k\}$



- $\overline{L} = \{d_1 \ldots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

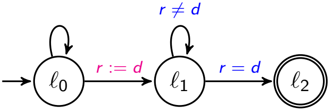
DET $\subsetneq$ UNA for Register Automata

- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \dots d_k \mid \exists 1 \leq i < k. d_i = d_k\}$



- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

- For register automata we have DET $\subsetneq$ UNA
- This follows from the non-closure of UNA under complement:
- $L = \{d_1 \ldots d_k \mid \exists 1 \leq i < k . d_i = d_k\}$



- $\overline{L} = \{d_1 \ldots d_k \mid \forall 1 \leq i < k . d_i \neq d_k\}$
- $\overline{L}$ not recognizable by any register automaton
- In particular L not recognizable by DET register automaton

- Generalize register automata: registers can store any "guessed" value
- $\overline{L} = \{d_1 \ldots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$ is recognizable by register automata with guessing

- An accepting run on the input word 1 2 1 4:

$$(\ell_0, \bot) \xrightarrow{1} (\ell_1, 4) \xrightarrow{2} (\ell_1, 4) \xrightarrow{1} (\ell_1, 4) \xrightarrow{4} (\ell_2, 4)$$

- Note that this is an unambiguous automaton

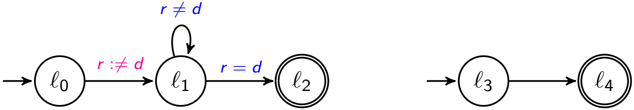
- Generalize register automata: registers can store any "guessed" value
- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$ is recognizable by register automata with guessing

- An accepting run on the input word 1 2 1 4:

$$(\ell_0, \bot) \xrightarrow{1} (\ell_1, 4) \xrightarrow{2} (\ell_1, 4) \xrightarrow{1} (\ell_1, 4) \xrightarrow{4} (\ell_2, 4)$$

- Note that this is an unambiguous automaton

- Generalize register automata: registers can store any "guessed" value
- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k . d_i \neq d_k\}$ is recognizable by register automata with guessing

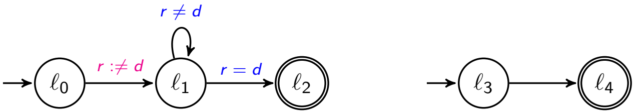


- An accepting run on the input word 1 2 1 4:

$$(\ell_0, \bot) \xrightarrow{1} (\ell_1, 4) \xrightarrow{2} (\ell_1, 4) \xrightarrow{1} (\ell_1, 4) \xrightarrow{4} (\ell_2, 4)$$

- Note that this is an unambiguous automaton

- Generalize register automata: registers can store any "guessed" value
- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$ is recognizable by register automata with guessing

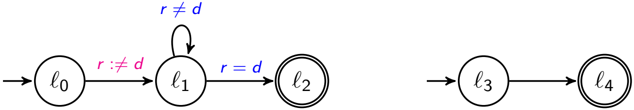


- An accepting run on the input word 1 2 1 4:

$$(\ell_0, \perp) \xrightarrow{1} (\ell_1, 4) \xrightarrow{2} (\ell_1, 4) \xrightarrow{1} (\ell_1, 4) \xrightarrow{4} (\ell_2, 4)$$

- Note that this is an unambiguous automaton

- Generalize register automata: registers can store any "guessed" value
- $\overline{L} = \{d_1 \dots d_k \mid \forall 1 \leq i < k. d_i \neq d_k\}$ is recognizable by register automata with guessing



- An accepting run on the input word 1 2 1 4:

$$(\ell_0, \bot) \xrightarrow{1} (\ell_1, 4) \xrightarrow{2} (\ell_1, 4) \xrightarrow{1} (\ell_1, 4) \xrightarrow{4} (\ell_2, 4)$$

- Note that this is an unambiguous automaton

- Universality Problem:
  given automaton $\mathcal{A}$, decide if $L(\mathcal{A}) = \mathbb{N}^*$
- State of the art in 2018:
- For register automata

# registers	DET	UNA	NONDET
1	PSPACE-complete	?	Ackermann-complete
> 1	PSPACE-complete	?	Undecidable

- For register automata with guessing

# registers	UNA	NONDET
1	?	Undecidable
> 1	?	Undecidable

- Universality Problem:
  given automaton $\mathcal{A}$, decide if $L(\mathcal{A}) = \mathbb{N}^*$
- State of the art in 2018:
- For register automata

# registers	DET	UNA	NONDET
1	PSPACE-complete	?	Ackermann-complete
> 1	PSPACE-complete	?	Undecidable

- For register automata with guessing

# registers	UNA	NONDET
1	?	Undecidable
> 1	?	Undecidable

- Universality Problem:
  given automaton $\mathcal{A}$, decide if $L(\mathcal{A}) = \mathbb{N}^*$
- State of the art in 2018:
- For register automata

# registers	DET	UNA	NONDET
1	PSPACE-complete	?	Ackermann-complete
> 1	PSPACE-complete	?	Undecidable

- For register automata with guessing

# registers	UNA	NONDET
1	?	Undecidable
> 1	?	Undecidable

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	EXPSPACE if 1 register [Mottet & Q, 2019]

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	EXPSPACE if 1 register [Mottet & Q, 2019]
2EXPTIME [Barloy & Clemente, STACS 2021]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	EXPSPACE if 1 register [Mottet & Q, 2019]
2EXPTIME [Barloy & Clemente, STACS 2021]	
EXPSPACE [Czerwinski, Mottet & Q, 2021]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

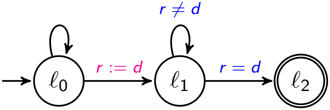
- Recall: the equivalence $L(\mathcal{A}) = \mathbb{N}^*$ iff $\overline{L(\mathcal{A})} \neq \emptyset$ cannot be used
- Table shows the upper bounds for universality (PSPACE-hardness follows from DET)

Unambiguous register automata	
without guessing	with guessing
2EXPSPACE [Mottet & Q, STACS 2019]	EXPSPACE if 1 register [Mottet & Q, 2019]
2EXPTIME [Barloy & Clemente, STACS 2021]	
EXPSPACE [Czerwinski, Mottet & Q, 2021]	
EXPTIME [Bojanczyk, Klin & Moelman, LICS 2021]	

- Reachability analysis on bounded graph (on-the-fly determinization)
- Counting the number of runs in the automaton à la Stearns and Hunt, 1981
- Algebraic approach à la Schützenberger, 1961

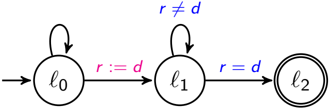
On-The-Fly Determinization

- We construct a reachability graph



- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

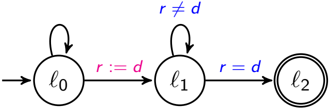
- We construct a reachability graph

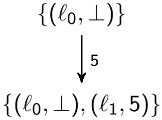


$\{(\ell_0, \bot)\}$

- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

- We construct a reachability graph

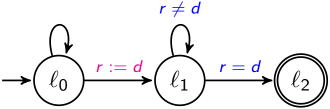


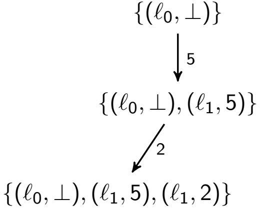


- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

On-The-Fly Determinization

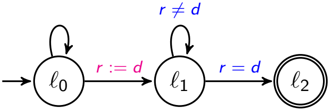
- We construct a reachability graph

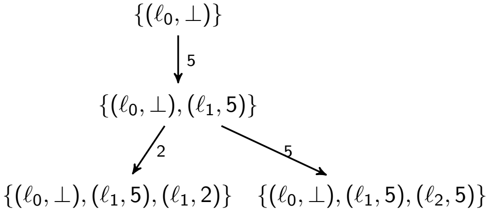




- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

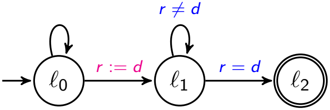
- We construct a reachability graph

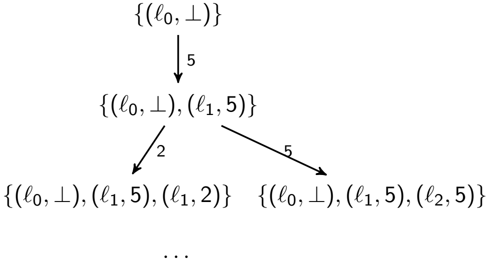




- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

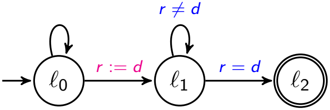
- We construct a reachability graph

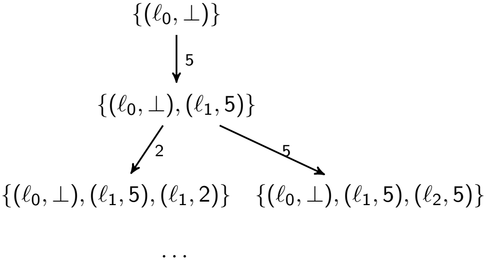




- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

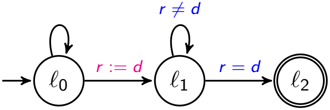
- We construct a reachability graph

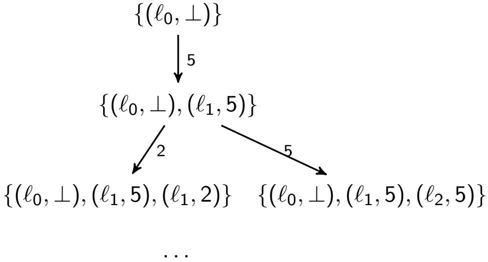




- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

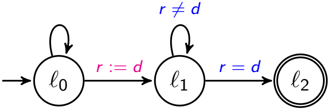
- We construct a reachability graph

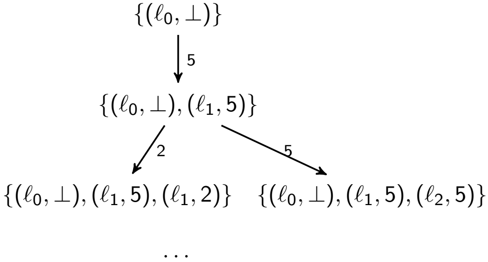




- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous automaton the size of a reachable set is bounded

- We construct a reachability graph





- $L(\mathcal{A}) = \mathbb{N}^*$ iff every set contains an accepting state
- Size of graph is a priori unbounded as the size of the sets is unbounded
- Idea is to prove that in an unambiguous universal automaton the size of a reachable set is even more bounded

Bounding the Size of Reachable Set

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$(\ell, 1)$

$(\ell, 2)$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$(\ell, 1) \longrightarrow$ acc

$(\ell, 2) \longrightarrow$ acc

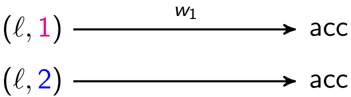
- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Bounding the Size of Reachable Set

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

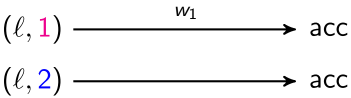


- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.



- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,2\,9\,3} \text{acc}$$

$$(\ell, 2) \longrightarrow \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,2\,9\,3} \text{acc}$$

$$(\ell, 2) \longrightarrow \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,3} \text{acc}$$

$$(\ell, 2) \qquad\qquad\qquad\qquad \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,3} \text{acc}$$

$$(\ell, 2) \xrightarrow{w_1 = 9\,8\,6\,9\,3} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

$$(\ell, 2) \qquad\qquad \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

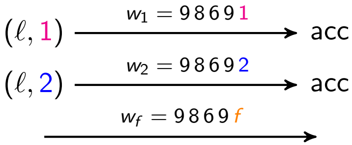
$$(\ell, 2) \xrightarrow{w_2 = 9\,8\,6\,9\,2} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.



- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Bounding the Size of Reachable Set

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

$$(\ell, 2) \xrightarrow{w_2 = 9\,8\,6\,9\,2} \text{acc}$$

$$(\ell', x) \xrightarrow{w_f = 9\,8\,6\,9\,f} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d, contradiction to unambiguity

Bounding the Size of Reachable Set

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

$$(\ell, 2) \xrightarrow{w_2 = 9\,8\,6\,9\,2} \text{acc}$$

$$(\ell', x) \xrightarrow{w_f = 9\,8\,6\,9\,f} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

$$(\ell, 2) \xrightarrow{w_2 = 9\,8\,6\,9\,2} \text{acc}$$

$$(\ell', x) \xrightarrow{w_f = 9\,8\,6\,9\,f} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

Bounding the Size of Reachable Set

Two assumptions on $\mathcal{A}$

- every config in a reachable set reaches an accepting state, and
- $\mathcal{A}$ is universal, i.e., $L(\mathcal{A}) = \mathbb{N}^*$

Let $\mathcal{A}$ have a single register, let $(\ell, 1), (\ell, 2)$ be in a reachable set.

$$(\ell, 1) \xrightarrow{w_1 = 9\,8\,6\,9\,1} \text{acc}$$

$$(\ell, 2) \xrightarrow{w_2 = 9\,8\,6\,9\,2} \text{acc}$$

$$(\ell', x) \xrightarrow{w_f = 9\,8\,6\,9\,f} \text{acc}$$

- First we prove that w_1 must contain 1 (and wlog does not contain 2)
- Take $d \in \{1, 2\}$ such that $x \neq d$: (ℓ', x) also accepts w_d , contradiction to unambiguity

- Similar situation for any number k of registers, e.g. for
  - $k = 2$, we cannot have $(\ell, 1, 2), (\ell, 3, 4), (\ell, 5, 6)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 4, 5, 6), (\ell, 7, 8, 9), (\ell, 10, 11, 12)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 1, 4, 5), (\ell, 1, 6, 7)$

  in a reachable set
- There is a finite constant M_k that upper bounds the number of configurations in a reachable set with the same state
- M_k is exponential in k: $M_k \leq (k \cdot 4^k \cdot k!)^k$
- Graph's size is bounded doubly-exponential in k
- Applying standard NL reachability algorithm we obtain EXPSPACE upper bound

Bounding the Size of Reachable Set

- Similar situation for any number k of registers, e.g. for
  - $k = 2$, we cannot have $(\ell, 1, 2), (\ell, 3, 4), (\ell, 5, 6)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 4, 5, 6), (\ell, 7, 8, 9), (\ell, 10, 11, 12)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 1, 4, 5), (\ell, 1, 6, 7)$

  in a reachable set
- There is a finite constant M_k that upper bounds the number of configurations in a reachable set with the same state
- M_k is exponential in k: $M_k \leq (k \cdot 4^k \cdot k!)^k$
- Graph's size is bounded doubly-exponential in k
- Applying standard NL reachability algorithm we obtain EXPSPACE upper bound

Bounding the Size of Reachable Set

- Similar situation for any number k of registers, e.g. for
  - $k = 2$, we cannot have $(\ell, 1, 2), (\ell, 3, 4), (\ell, 5, 6)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 4, 5, 6), (\ell, 7, 8, 9), (\ell, 10, 11, 12)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 1, 4, 5), (\ell, 1, 6, 7)$

  in a reachable set
- There is a finite constant M_k that upper bounds the number of configurations in a reachable set with the same state
- M_k is exponential in k: $M_k \leq (k \cdot 4^k \cdot k!)^k$
- Graph's size is bounded doubly-exponential in k
- Applying standard NL reachability algorithm we obtain EXPSPACE upper bound

- Similar situation for any number k of registers, e.g. for
  - $k = 2$, we cannot have $(\ell, 1, 2), (\ell, 3, 4), (\ell, 5, 6)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 4, 5, 6), (\ell, 7, 8, 9), (\ell, 10, 11, 12)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 1, 4, 5), (\ell, 1, 6, 7)$

  in a reachable set
- There is a finite constant M_k that upper bounds the number of configurations in a reachable set with the same state
- M_k is exponential in k: $M_k \leq (k \cdot 4^k \cdot k!)^k$
- Graph's size is bounded doubly-exponential in k
- Applying standard NL reachability algorithm we obtain EXPSPACE upper bound

- Similar situation for any number k of registers, e.g. for
  - $k = 2$, we cannot have $(\ell, 1, 2), (\ell, 3, 4), (\ell, 5, 6)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 4, 5, 6), (\ell, 7, 8, 9), (\ell, 10, 11, 12)$
  - $k = 3$, we cannot have $(\ell, 1, 2, 3), (\ell, 1, 4, 5), (\ell, 1, 6, 7)$

  in a reachable set
- There is a finite constant M_k that upper bounds the number of configurations in a reachable set with the same state
- M_k is exponential in k: $M_k \leq (k \cdot 4^k \cdot k!)^k$
- Graph's size is bounded doubly-exponential in k
- Applying standard NL reachability algorithm we obtain EXPSPACE upper bound

Future Research

- Besides $(\mathbb{N}; =)$ and $(\mathbb{Q}; <, =)$, UNA register automata over other relational structures
- Close gap between PSPACE lower bound and EXPTIME upper bound

  in general: new techniques for proving lower bounds for universality for UNA models
- Interesting techniques applicable to other infinite objects

Future Research

- Besides $(\mathbb{N};=)$ and $(\mathbb{Q};<,=)$, UNA register automata over other relational structures
- Close gap between PSPACE lower bound and EXPTIME upper bound

  in general: new techniques for proving lower bounds for universality for UNA models

- Interesting techniques applicable to other infinite objects

- Besides $(\mathbb{N}; =)$ and $(\mathbb{Q}; <, =)$, UNA register automata over other relational structures
- Close gap between PSPACE lower bound and EXPTIME upper bound

  in general: new techniques for proving lower bounds for universality for UNA models
- Interesting techniques applicable to other infinite objects

Future Research

- Besides $(\mathbb{N}; =)$ and $(\mathbb{Q}; <, =)$, UNA register automata over other relational structures
- Close gap between PSPACE lower bound and EXPTIME upper bound

  in general: new techniques for proving lower bounds for universality for UNA models
- Interesting techniques applicable to other infinite objects