Bounded Parikh Automata

M. Cadilhac[1], A. Finkel[2], and P. McKenzie[1]

[1]: Université de Montréal

[2]: ENS CACHAN

September 15th, 2011

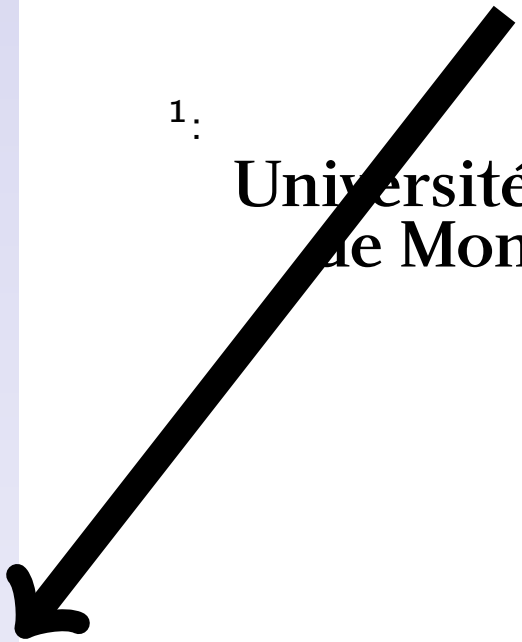
Bounded Parikh Automata

M. Cadilhac[1], A. Finkel[2], and P. McKenzie[1]

[1]: Université de Montréal

[2]: ENS CACHAN

September 15th, 2011



Bounded Parikh Automata

M. Cadilhac[1], A. Finkel[2], and P. McKenzie[1]

[1]: Université de Montréal

[2]: ENS CACHAN

Septe th, 2011

Bounded Parikh Automata

M. Cadilhac[1], A. Finkel[2], and P. McKenzie[1]

1: Université de Montréal

2: ENS CACHAN

Septe th, 201

Context

A practical problem:

- In model checking, properties described by automata

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism
- Finite automata offer poor expressiveness

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism
- Finite automata offer poor expressiveness
- Goal: provide large classes of "deterministic" languages

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism
- Finite automata offer poor expressiveness
- Goal: provide large classes of "deterministic" languages

Natural models of automata allow for a study of:

- Small complexity classes (within NC^2)

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism
- Finite automata offer poor expressiveness
- Goal: provide large classes of "deterministic" languages

Natural models of automata allow for a study of:

- Small complexity classes (within NC^2)
- Word logics (variants of MSO)

Context

A practical problem:

- In model checking, properties described by automata
- Efficiency depends, in particular, on determinism
- Finite automata offer poor expressiveness
- Goal: provide large classes of "deterministic" languages

Natural models of automata allow for a study of:

- Small complexity classes (within NC^2)
- Word logics (variants of MSO)
- Algebra (pseudovarieties of monoids associated with language varieties)

Outline

Outline

The result

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

  E.g., $\left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 2 \end{pmatrix} \cdot k_1 \right\}$ linear

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

  E.g., $\left\{\begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix} \cdot k_1\right\}$ linear, $\{2^n \mid n \in \mathbb{N}\}$ not linear

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

  E.g., $\left\{\begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix} \cdot k_1\right\}$ linear, $\{2^n \mid n \in \mathbb{N}\}$ not linear

- *Semilinear* set: finite union of linear sets (equiv. FO[+])

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

  E.g., $\left\{\begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix} \cdot k_1\right\}$ linear, $\{2^n \mid n \in \mathbb{N}\}$ not linear
- *Semilinear* set: finite union of linear sets (equiv. FO[+])

Why are they natural? One of many reasons:

Theorem ([Parikh, 1966])

With $\Sigma = \{a_1, \ldots, a_n\}$ *and* $w \in \Sigma^*$, *let*
$\text{Parikh}(w) = (|w|_{a_1}, \ldots, |w|_{a_n}) \in \mathbb{N}^n$ *be the Parikh image of* w.

The result

Definition: semilinear

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definitions

- *Linear* set: of the form $E = \{\vec{c}_0 + \sum_{i=1}^{m} \vec{c}_i \cdot k_i \mid k_i \in \mathbb{N}\}$

  E.g., $\left\{\begin{pmatrix}0\\0\end{pmatrix} + \begin{pmatrix}1\\2\end{pmatrix} \cdot k_1\right\}$ linear, $\{2^n \mid n \in \mathbb{N}\}$ not linear
- *Semilinear* set: finite union of linear sets (equiv. FO[+])

Why are they natural? One of many reasons:

Theorem ([Parikh, 1966])

With $\Sigma = \{a_1, \ldots, a_n\}$ *and* $w \in \Sigma^*$, *let*
$\text{Parikh}(w) = (|w|_{a_1}, \ldots, |w|_{a_n}) \in \mathbb{N}^n$ *be the Parikh image of* w.

L *context-free* $\Rightarrow$ $\text{Parikh}(L)$ *semilinear*

The result

Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$

The result

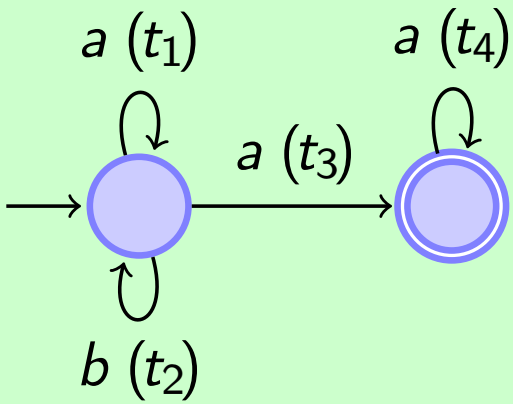
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



The result

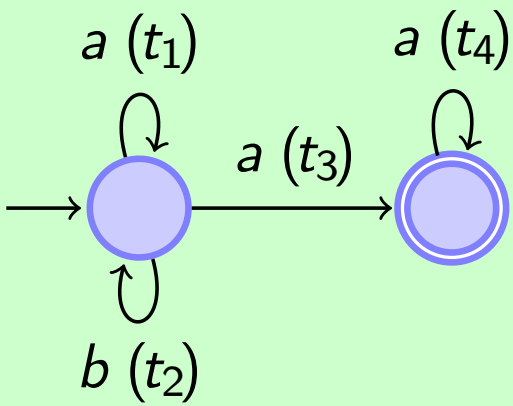
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

The result

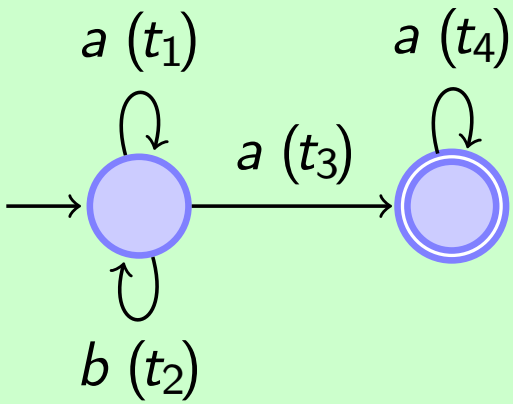
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \right.$$

The result

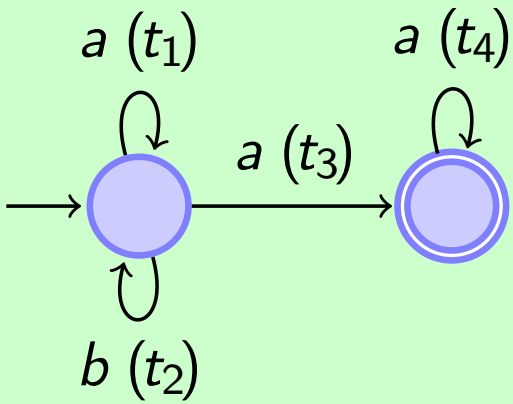
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \right.$$

The result

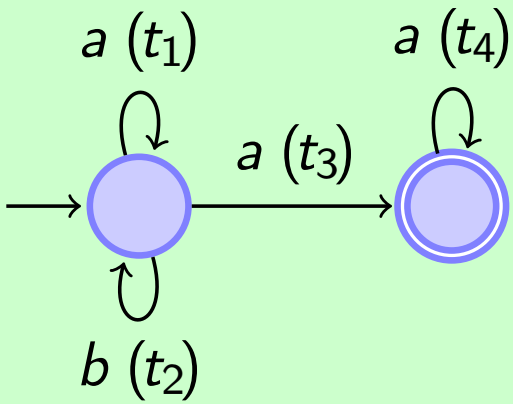
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \right.$$

The result

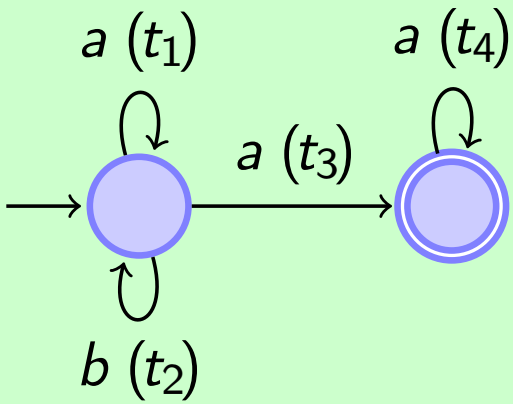
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

The result

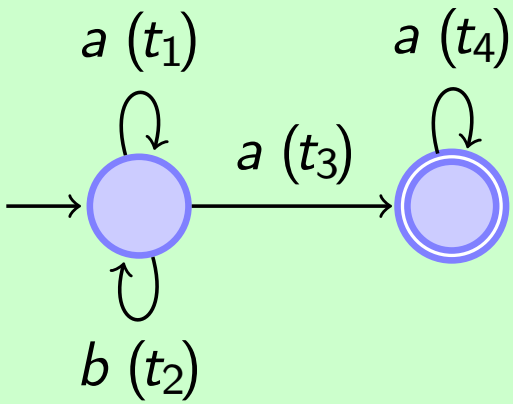
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*

The result

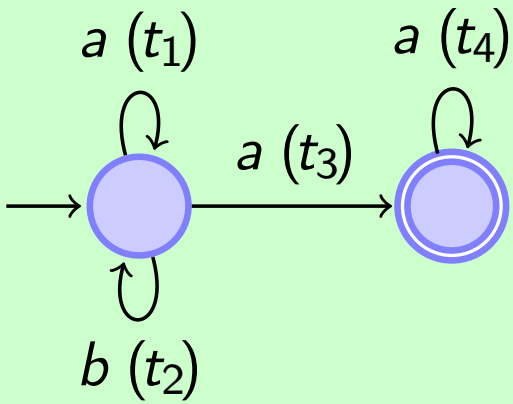
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*

Parikh(traced run):
$\pi =$

The result

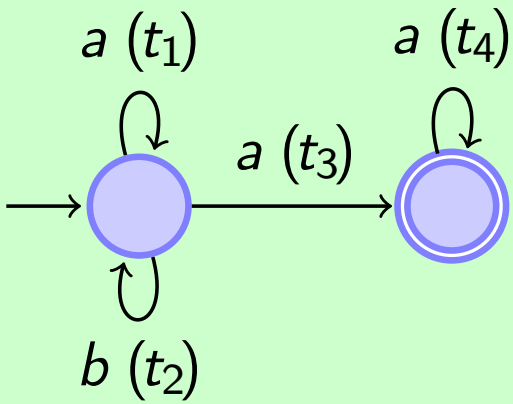
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*
▲

Parikh(traced run): (0, 0, 0, 0)
$\pi =$

The result

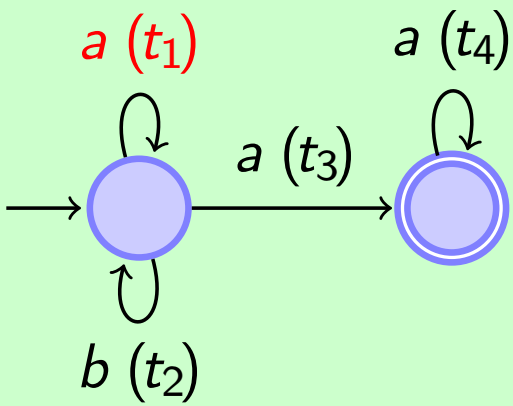
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*
▲

Parikh(traced run): (1, 0, 0, 0)
$\pi = t_1$

The result

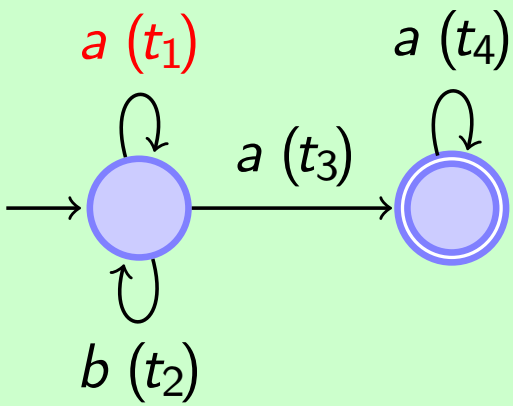
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*
▲

Parikh(traced run): (2, 0, 0, 0)
$\pi = t_1 t_1$

The result

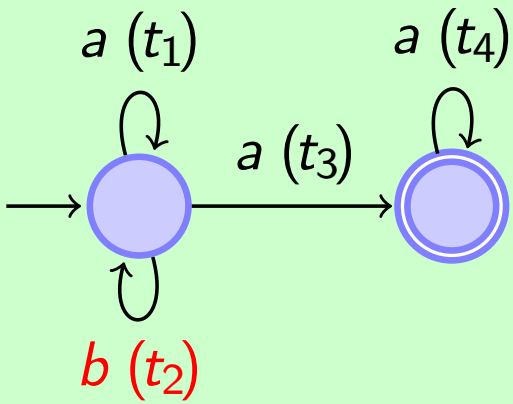
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*
▲

Parikh(traced run): (2, 1, 0, 0)
$\pi = t_1 t_1 t_2$

The result

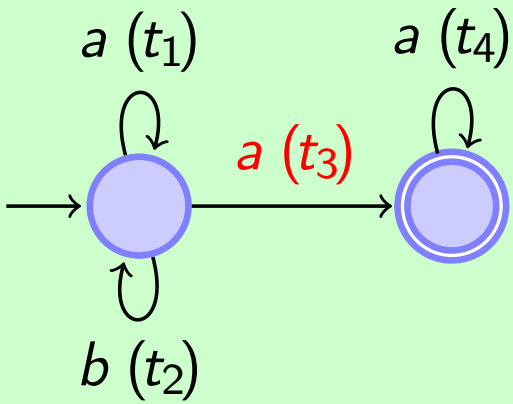
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*
▲

Parikh(traced run): (2, 1, 1, 0)
$\pi = t_1 t_1 t_2 t_3$

The result

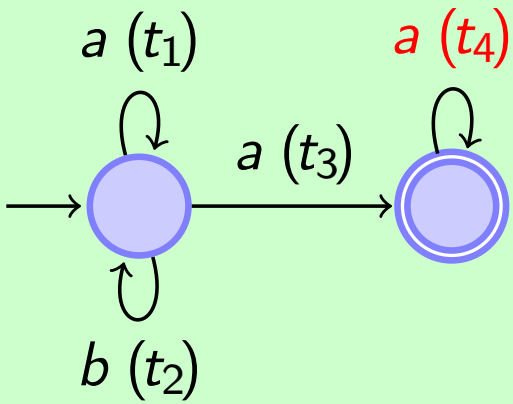
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*

▲

Parikh(traced run): (2, 1, 1, 1)

$\pi = t_1 t_1 t_2 t_3 t_4$

The result

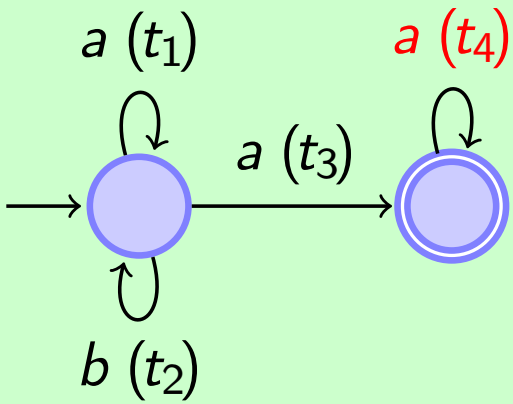
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa*

▲

Parikh(traced run): (2, 1, 1, 2)

$\pi = t_1 t_1 t_2 t_3 t_4 t_4$

The result

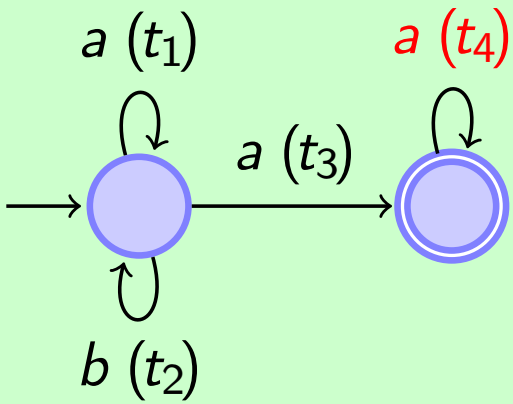
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{ w \cdot a^{|w|+1} \mid w \in \{a, b\}^* \} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa* ▲

Parikh(traced run): (2, 1, 1, 3)
$\pi = t_1 t_1 t_2 t_3 t_4 t_4 t_4$

The result

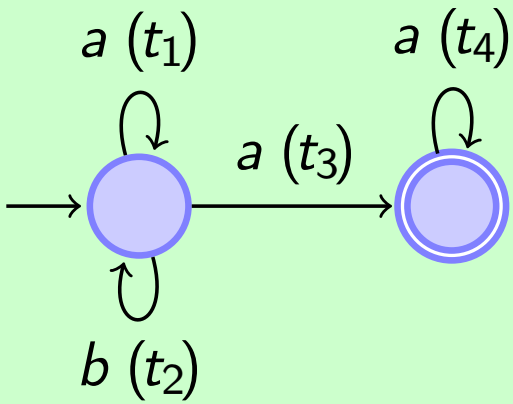
Example: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Example

$$L = \{w \cdot a^{|w|+1} \mid w \in \{a, b\}^*\} \in \text{PA}$$



$$\delta = \{t_1, t_2, t_3, t_4\}$$

$$C = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \cdot k_1 + \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \end{pmatrix} \cdot k_2 + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} \right\}$$

Word: *aabaaaa* ▲

Parikh(traced run): $(2, 1, 1, 3) \in C$

$\pi = t_1 t_1 t_2 t_3 t_4 t_4 t_4$

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear
- $L(A, C) = \{\text{Label}(\pi) \mid \pi \in \text{Runs}(A) \wedge \text{Parikh}(\pi) \in C\}$

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear
- $L(A, C) = \{\text{Label}(\pi) \mid \pi \in \text{Runs}(A) \wedge \text{Parikh}(\pi) \in C\}$
- *Deterministic Parikh automaton* (DetPA) if A is

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear
- $L(A, C) = \{\text{Label}(\pi) \mid \pi \in \text{Runs}(A) \wedge \text{Parikh}(\pi) \in C\}$
- *Deterministic Parikh automaton* (DetPA) if A is

- $L = \{w \cdot a^{|w|+1}\} \in$ PA but $\notin$ DetPA

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear
- $L(A, C) = \{\text{Label}(\pi) \mid \pi \in \text{Runs}(A) \wedge \text{Parikh}(\pi) \in C\}$
- *Deterministic Parikh automaton* (DetPA) if A is

- $L = \{w \cdot a^{|w|+1}\} \in$ PA but $\notin$ DetPA, same for $\overline{\text{PAL}}$

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition ([Klaedtke and Rueß, 2003])

- *Parikh automaton* (PA): a pair (A, C) with:
  - A a finite automaton of transition set δ
  - $C \subseteq \mathbb{N}^{|\delta|}$ semilinear
- $L(A, C) = \{\text{Label}(\pi) \mid \pi \in \text{Runs}(A) \wedge \text{Parikh}(\pi) \in C\}$
- *Deterministic Parikh automaton* (DetPA) if A is

- $L = \{w \cdot a^{|w|+1}\} \in$ PA but $\notin$ DetPA, same for $\overline{\text{PAL}}$
- PAL $\notin$ PA

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA
- Equivalent to reversal bounded counter machines (RBCM) of [Ibarra, 1978]

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA
- Equivalent to reversal bounded counter machines (RBCM) of [Ibarra, 1978]
- Equivalent to extended automata over $(\mathbb{Z}^k, +, 0)$ of [Mitrana and Stiebe, 2001]

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA
- Equivalent to reversal bounded counter machines (RBCM) of [Ibarra, 1978]
- Equivalent to extended automata over $(\mathbb{Z}^k, +, 0)$ of [Mitrana and Stiebe, 2001]
- Related deterministic models used for model checking

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA
- Equivalent to reversal bounded counter machines (RBCM) of [Ibarra, 1978]
- Equivalent to extended automata over $(\mathbb{Z}^k, +, 0)$ of [Mitrana and Stiebe, 2001]
- Related deterministic models used for model checking
- Logical characterization (WS1S + cardinalities)

The result

Definition: Parikh automata

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Why is PA a relevant model?

- Nice decidability properties (emptiness)
- Nice closure properties for DetPA
- Equivalent to reversal bounded counter machines (RBCM) of [Ibarra, 1978]
- Equivalent to extended automata over $(\mathbb{Z}^k, +, 0)$ of [Mitrana and Stiebe, 2001]
- Related deterministic models used for model checking
- Logical characterization (WS1S + cardinalities)
- Low complexity (PA $\subsetneq$ NL, DetPA $\subsetneq$ NC^1)

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- *L bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- *L bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- L *bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$
- $\mathrm{BSL} = \{L \subseteq w_1^* \cdots w_n^* \mid \mathrm{Iter}_{\vec{w}}(L) \text{ semilinear}\}$

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- L *bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$
- $\mathrm{BSL} = \{L \subseteq w_1^* \cdots w_n^* \mid \mathrm{Iter}_{\vec{w}}(L) \text{ semilinear}\}$

- $\{a^i b^{2i}\} \in \mathrm{BSL}$

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- L *bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$
- $\mathrm{BSL} = \{L \subseteq w_1^* \cdots w_n^* \mid \mathrm{Iter}_{\vec{w}}(L) \text{ semilinear}\}$

- $\{a^i b^{2i}\} \in \mathrm{BSL}$
- Σ^* not bounded

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- L *bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$
- $\mathrm{BSL} = \{L \subseteq w_1^* \cdots w_n^* \mid \mathrm{Iter}_{\vec{w}}(L) \text{ semilinear}\}$

- $\{a^i b^{2i}\} \in \mathrm{BSL}$
- Σ^* not bounded
- $\{a^{2^n}\}$ bounded $\notin \mathrm{BSL}$

The result

Definition: Bounded languages

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Definition

- *L bounded*: $L \subseteq w_1^* \cdots w_n^*$ for some words w_i's
- Define $\mathrm{Iter}_{\vec{w}}(L) = \{(i_1, \ldots, i_n) \mid w_1^{i_1} \cdots w_n^{i_n} \in L\} \subseteq \mathbb{N}^n$
- $\mathrm{BSL} = \{L \subseteq w_1^* \cdots w_n^* \mid \mathrm{Iter}_{\vec{w}}(L) \text{ semilinear}\}$

- $\{a^i b^{2i}\} \in \mathrm{BSL}$
- Σ^* not bounded
- $\{a^{2^n}\}$ bounded $\notin \mathrm{BSL}$

- BSL intensively studied, e.g., Ginsburg & Spanier, 60's

The result

The big picture

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

The result

The big picture

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Theorem, restated:

Theorem

$$\mathrm{PA} \cap \mathrm{BOUNDED} \quad \subseteq \quad \mathrm{BSL} \quad \subseteq \quad \mathrm{DetPA} \cap \mathrm{BOUNDED}$$

The result

The big picture

Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Theorem, restated:

Theorem

$$\mathrm{PA} \cap \mathrm{BOUNDED} \subsetneq \mathrm{BSL} \subseteq \mathrm{DetPA} \cap \mathrm{BOUNDED}$$

Parikh(*any L in PA*) *semilinear*
PA closed under h^{-1}, $\cap$

The result

The big picture

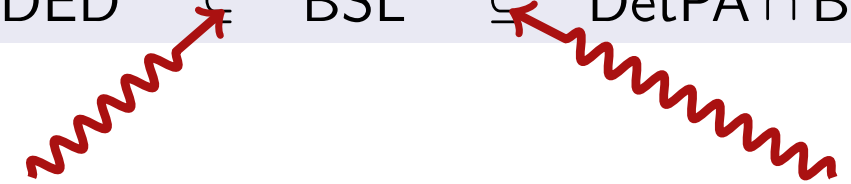
Theorem

Parikh automata and their deterministic variant recognize the same bounded languages: those with a semilinear iteration set

Theorem, restated:

Theorem

$$\mathrm{PA} \cap \mathrm{BOUNDED} \subsetneq \mathrm{BSL} \subseteq \mathrm{DetPA} \cap \mathrm{BOUNDED}$$



Parikh(*any L in PA*) *semilinear*
PA closed under h^{-1}, $\cap$

Rest of this talk

Outline

Preliminary

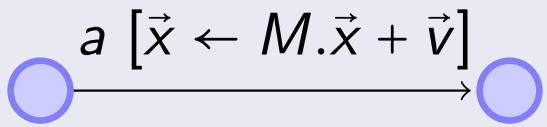
We make use of a related model:

Preliminary

We make use of a related model:

Definition

- *Affine Parikh automaton* given by:
  - A finite automaton
  - A labelling of the transitions by affine functions
  - A semilinear set

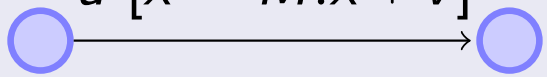


Preliminary

We make use of a related model:

Definition

- *Affine Parikh automaton* given by:
  - A finite automaton
  - A labelling of the transitions by affine functions
  - A semilinear set



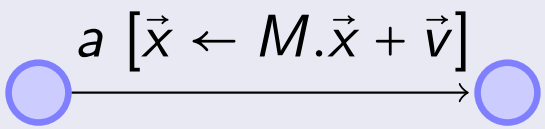
- Its language: accepted words which take $\vec{0}$ to some $\vec{x}$ in the semilinear set

Preliminary

We make use of a related model:

Definition

- *Affine Parikh automaton* given by:
  - A finite automaton
  - A labelling of the transitions by affine functions
  - A semilinear set



- Its language: accepted words which take $\vec{0}$ to some $\vec{x}$ in the semilinear set

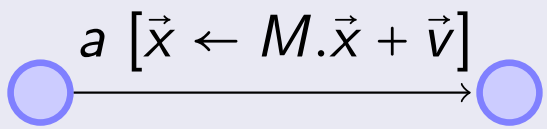
- PA: APA in which every M = Identity

Preliminary

We make use of a related model:

Definition

- *Affine Parikh automaton* given by:
  - A finite automaton
  - A labelling of the transitions by affine functions
  - A semilinear set



- Its language: accepted words which take $\vec{0}$ to some $\vec{x}$ in the semilinear set

- PA: APA in which every M = Identity
- Known facts: PAL, COPY, $\{a^{2^n}\} \in$ APA $\setminus$ PA
- Open: Dyck $\notin$ APA, PAL $\notin$ DetAPA

Outline

Result and definitions

BSL ⊆ DetPA

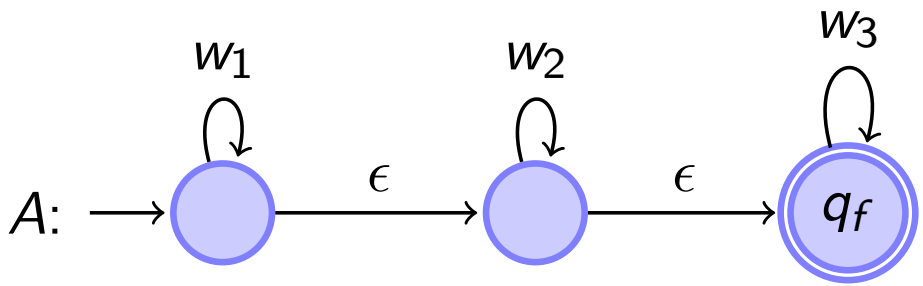
BSL ⊆ DetAPA with some property X

DetAPA with this property ⊆ DetPA

Corollaries and Further Work

BSL ⊆ DetAPA with some property X

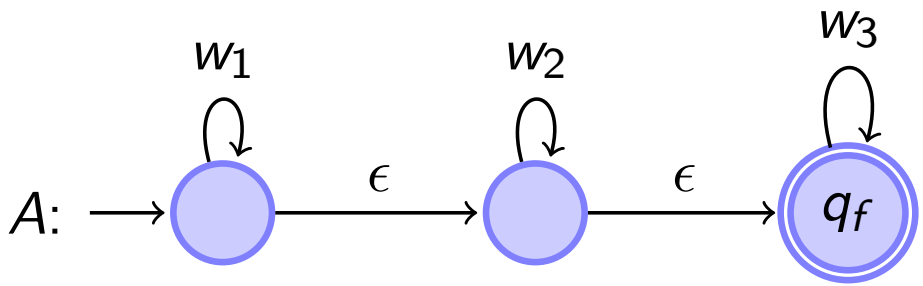
- Let $L \subseteq w_1^* w_2^* w_3^* \in$ BSL. Then a PA describes L:



and $C = \text{Iter}_{\vec{w}}(L)$

BSL ⊆ DetAPA with some property X

- Let $L \subseteq w_1^* w_2^* w_3^* \in \text{BSL}$. Then a PA describes L:



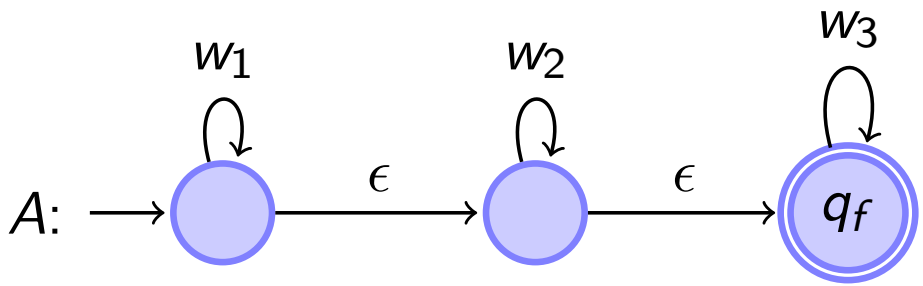
and $C = \text{Iter}_{\vec{w}}(L)$

- Let π_1, π_2 two accepting paths with same label:

$$\text{Parikh}(\pi_1) \in C \Leftrightarrow \text{Parikh}(\pi_2) \in C$$

BSL ⊆ DetAPA with some property X

- Let $L \subseteq w_1^* w_2^* w_3^* \in \mathrm{BSL}$. Then a PA describes L:



and $C = \mathrm{Iter}_{\vec{w}}(L)$

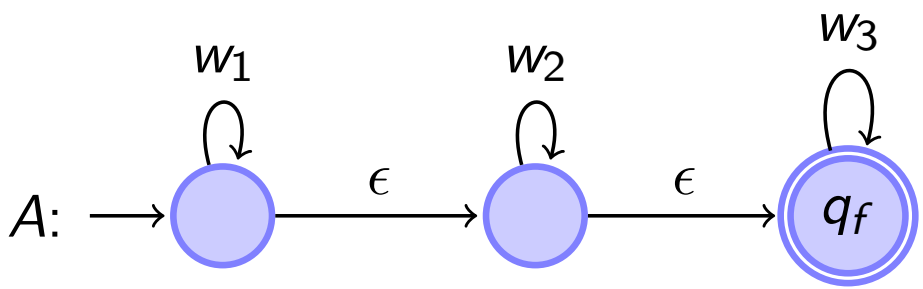
- Let π_1, π_2 two accepting paths with same label:

$$\mathrm{Parikh}(\pi_1) \in C \Leftrightarrow \mathrm{Parikh}(\pi_2) \in C$$

- In SubsetDeterminize(A), when reaching a final state $\{q_f, \ldots\}$, we need only recall the Parikh image of *one* possible run to q_f

BSL ⊆ DetAPA with some property X

- Let $L \subseteq w_1^* w_2^* w_3^* \in$ BSL. Then a PA describes L:



and $C = \text{Iter}_{\vec{w}}(L)$

- Let π_1, π_2 two accepting paths with same label:

$$\text{Parikh}(\pi_1) \in C \Leftrightarrow \text{Parikh}(\pi_2) \in C$$

- In SubsetDeterminize(A), when reaching a final state $\{q_f, \ldots\}$, we need only recall the Parikh image of *one* possible run to q_f
- In SubsetDeterminize(A), when reaching some state $\{q_1, \ldots, q_k\}$, we need only recall the Parikh image of *one* possible run to each q_i

BSL ⊆ DetAPA with some property X

Thus we do the following:

1. Let (A, C) be the PA for $L \in$ BSL

BSL ⊆ DetAPA with some property X

Thus we do the following:

1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction

BSL ⊆ DetAPA with some property X

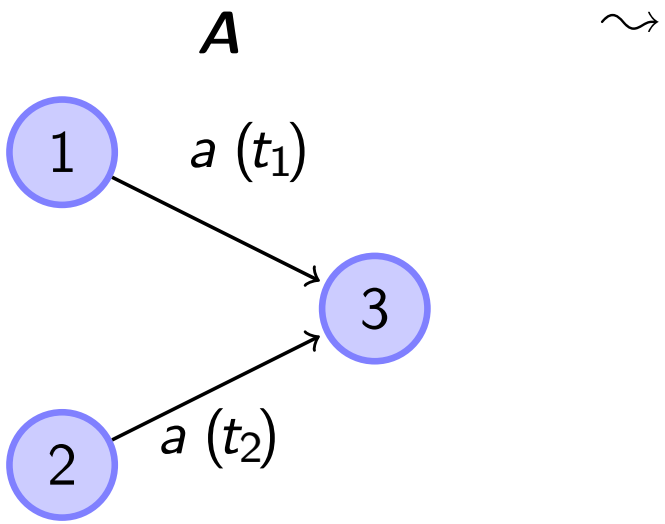
Thus we do the following:

1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:

BSL ⊆ DetAPA with some property X

Thus we do the following:

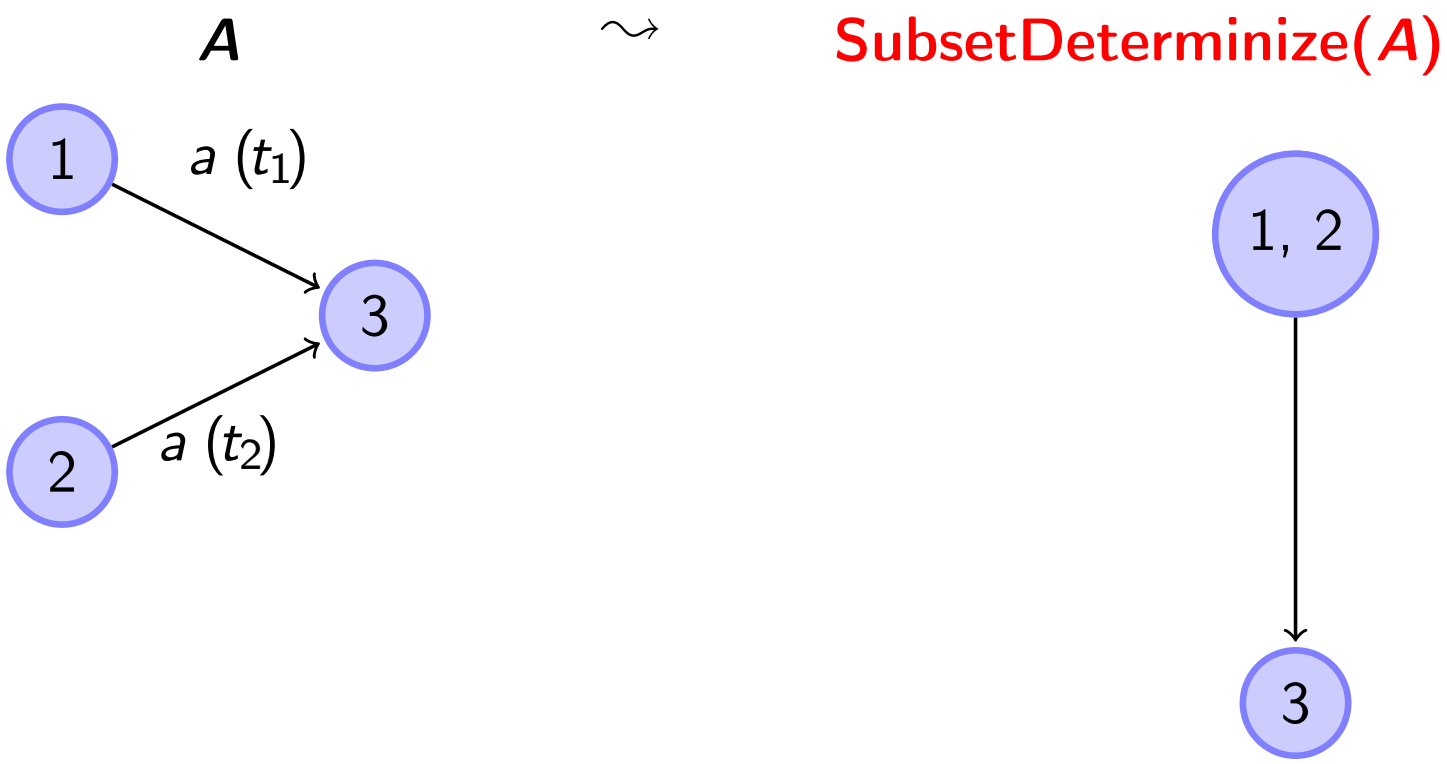
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

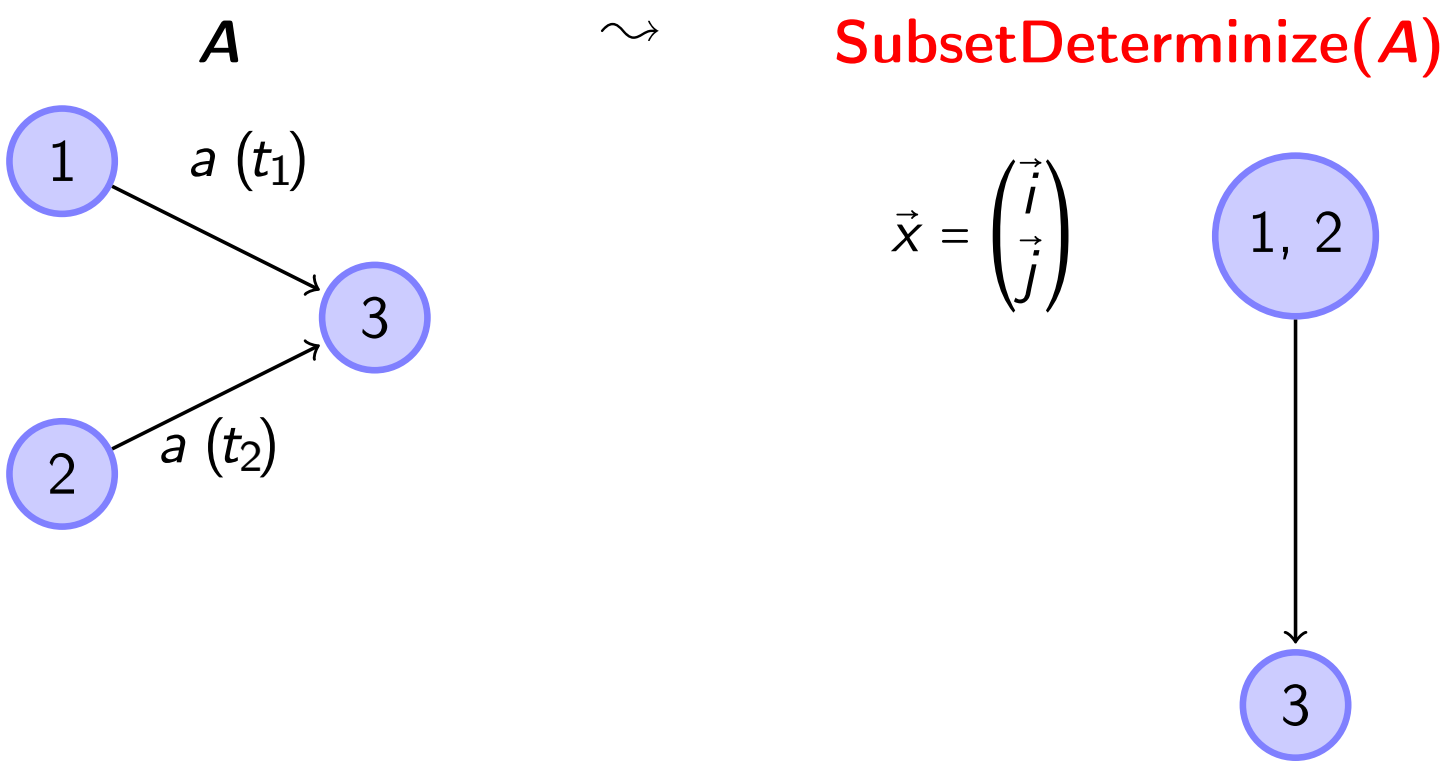
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

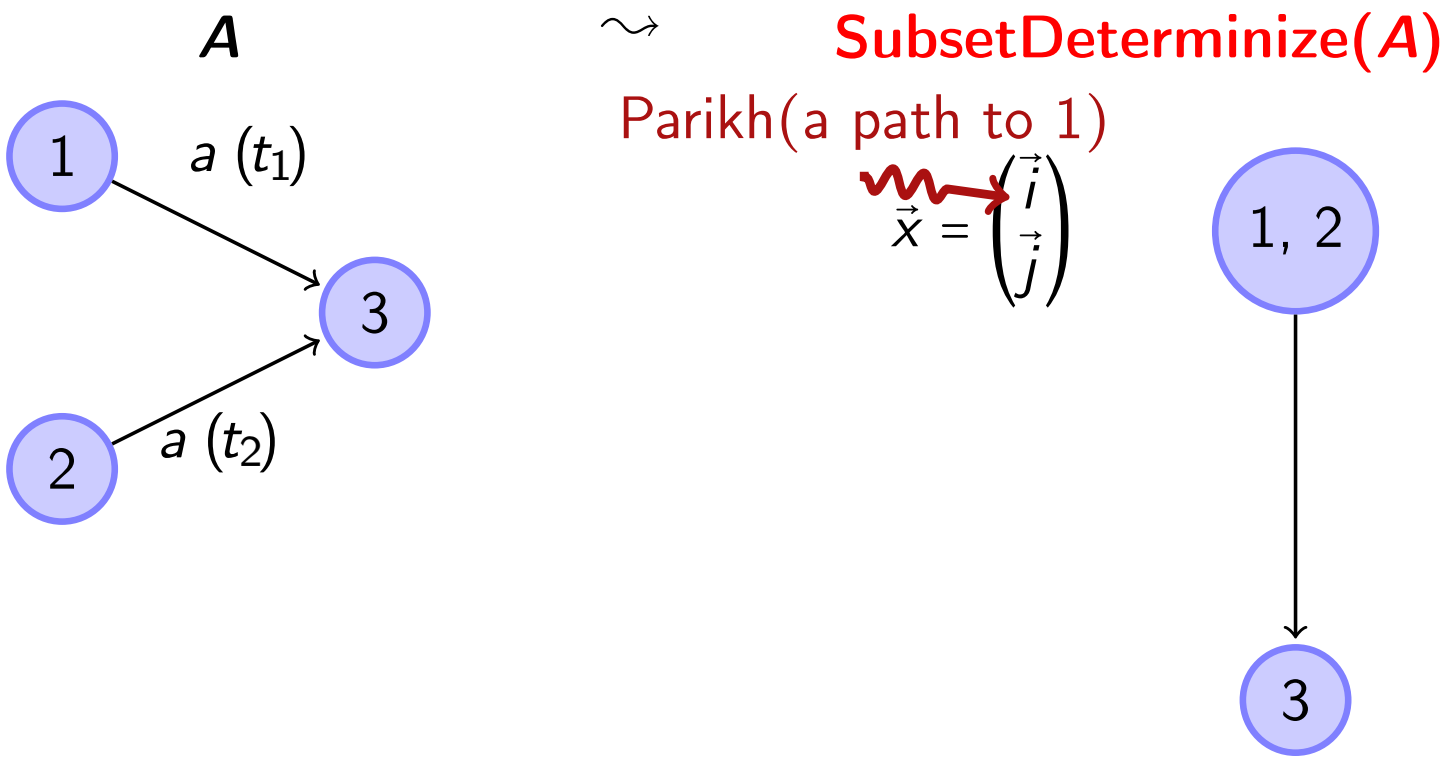
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

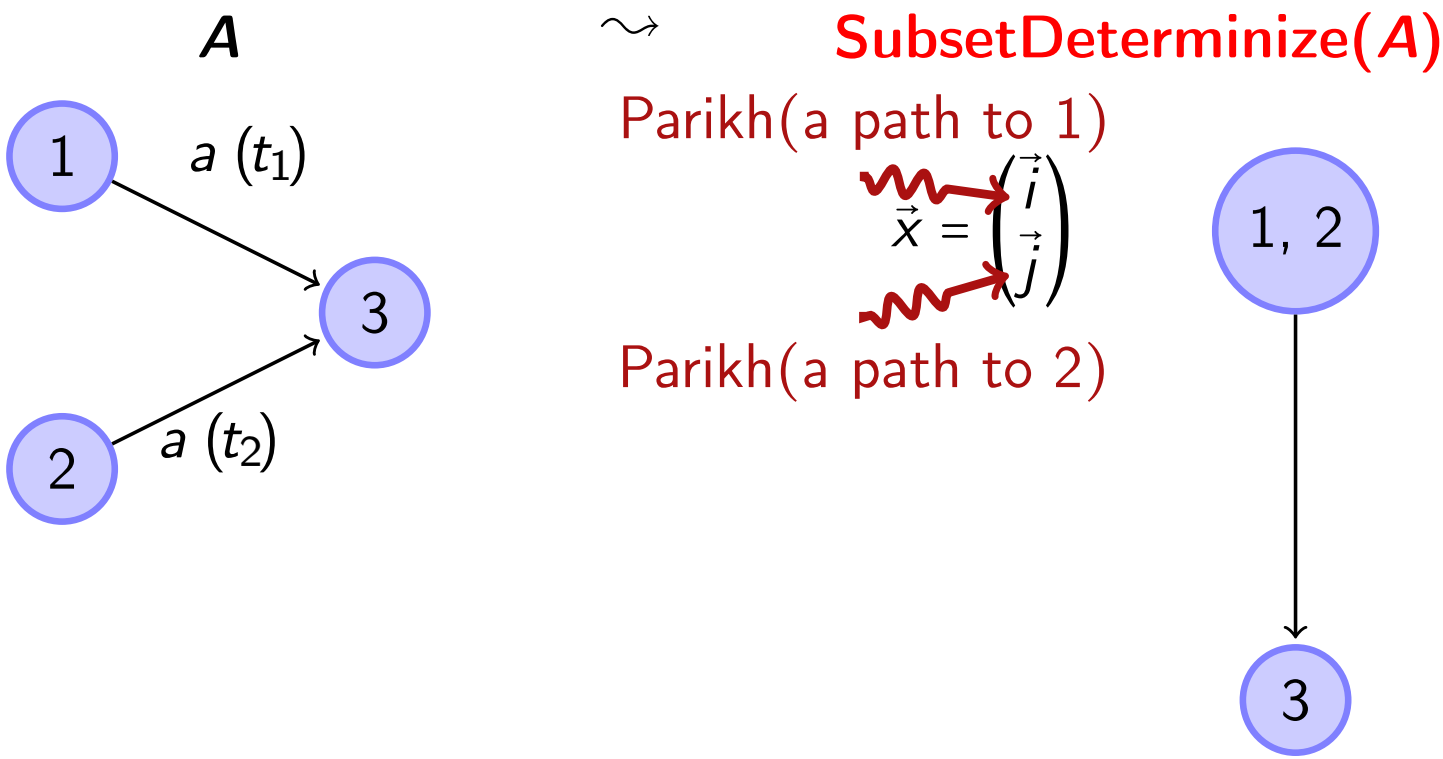
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

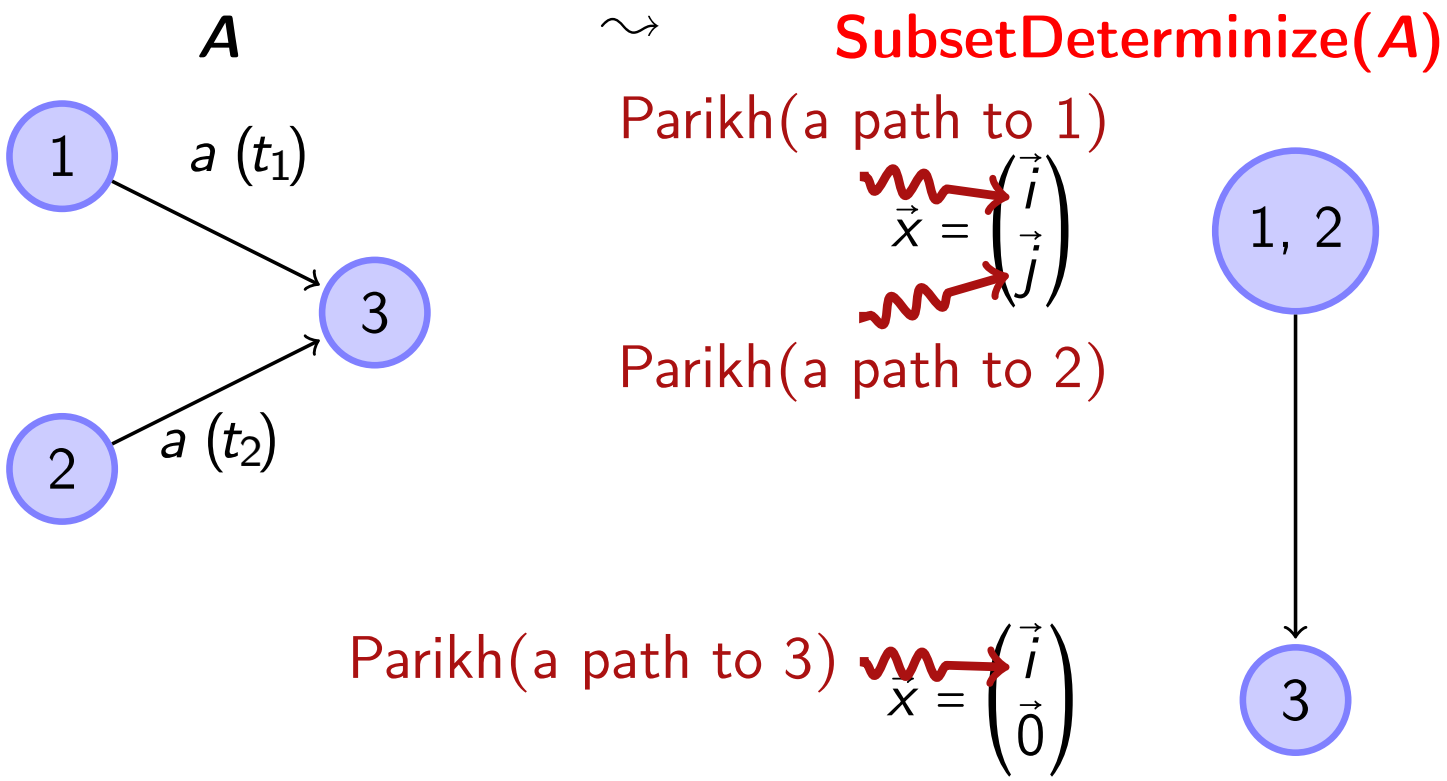
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

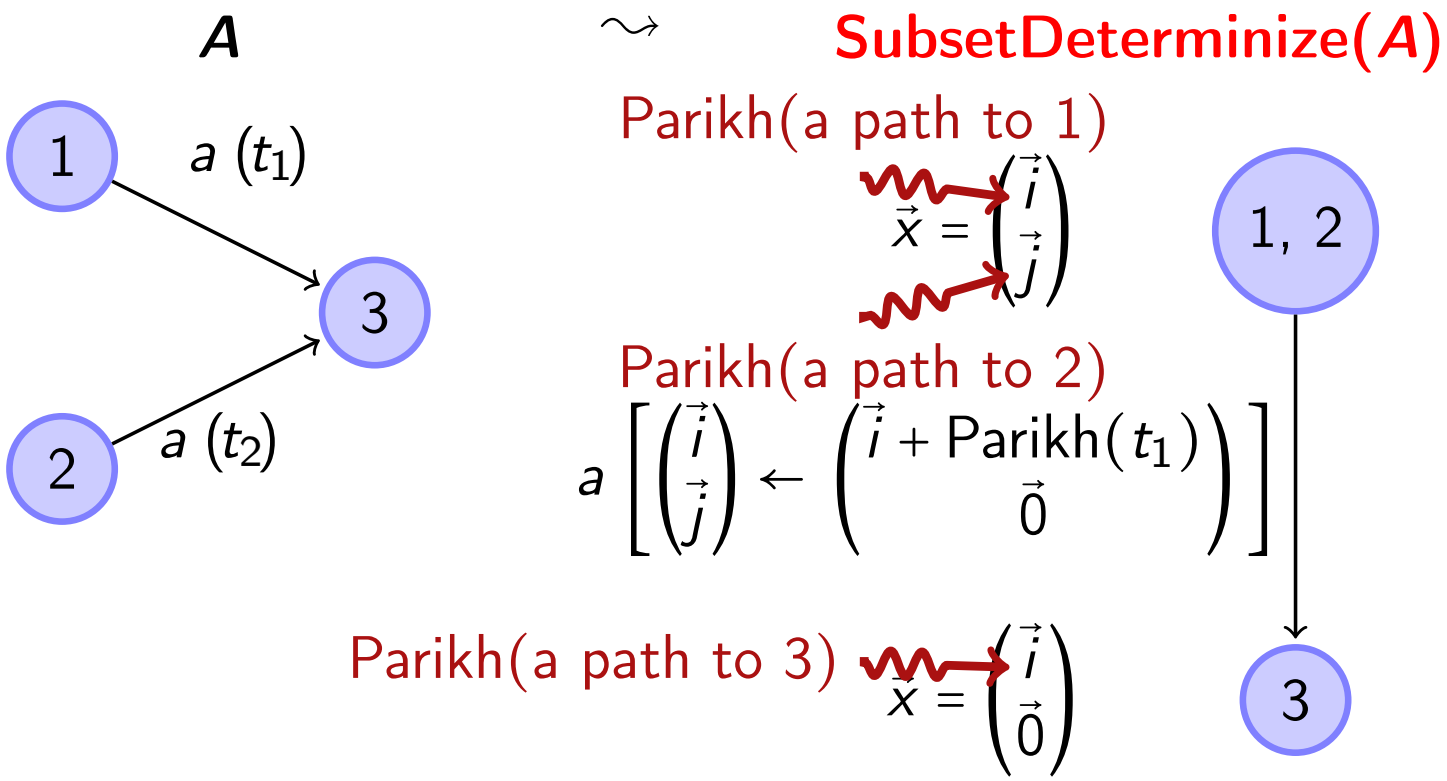
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

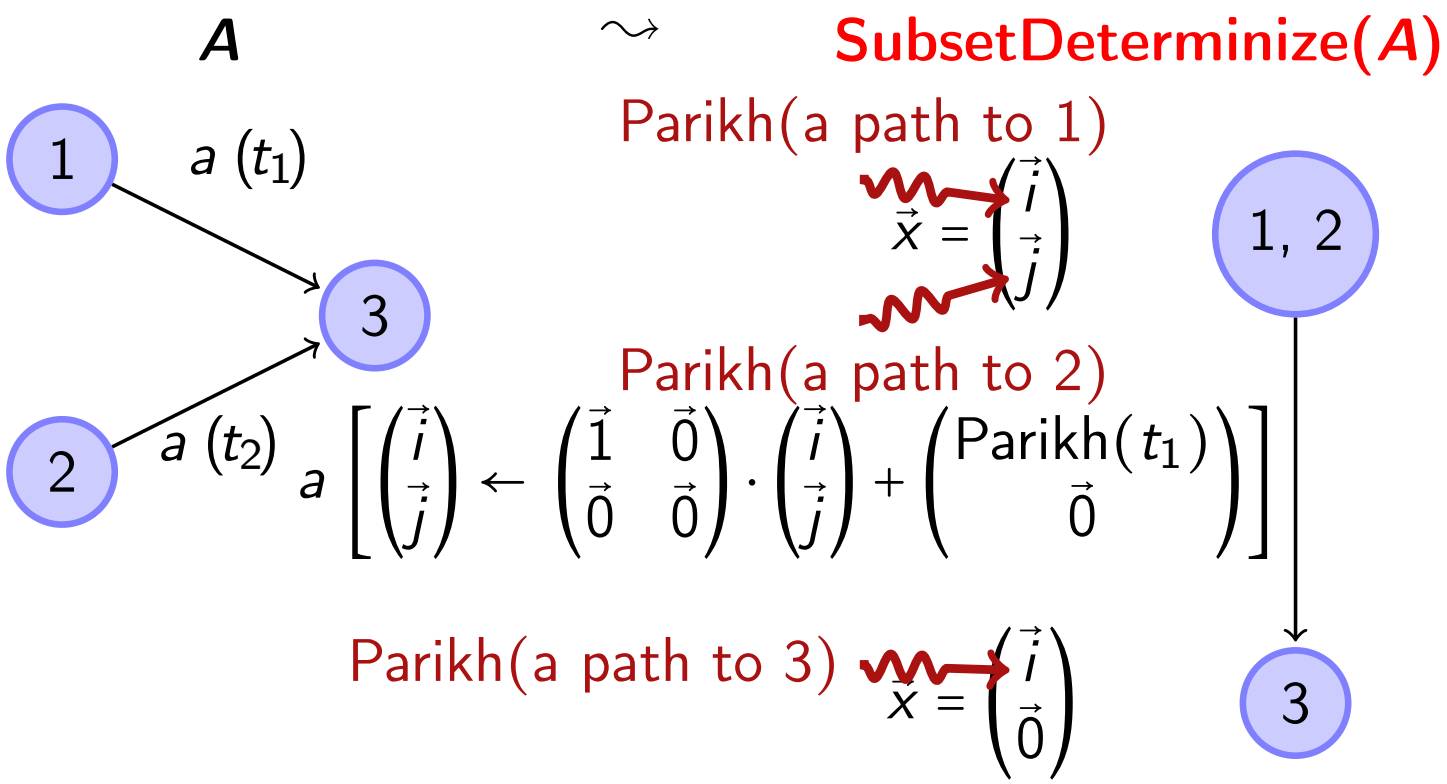
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

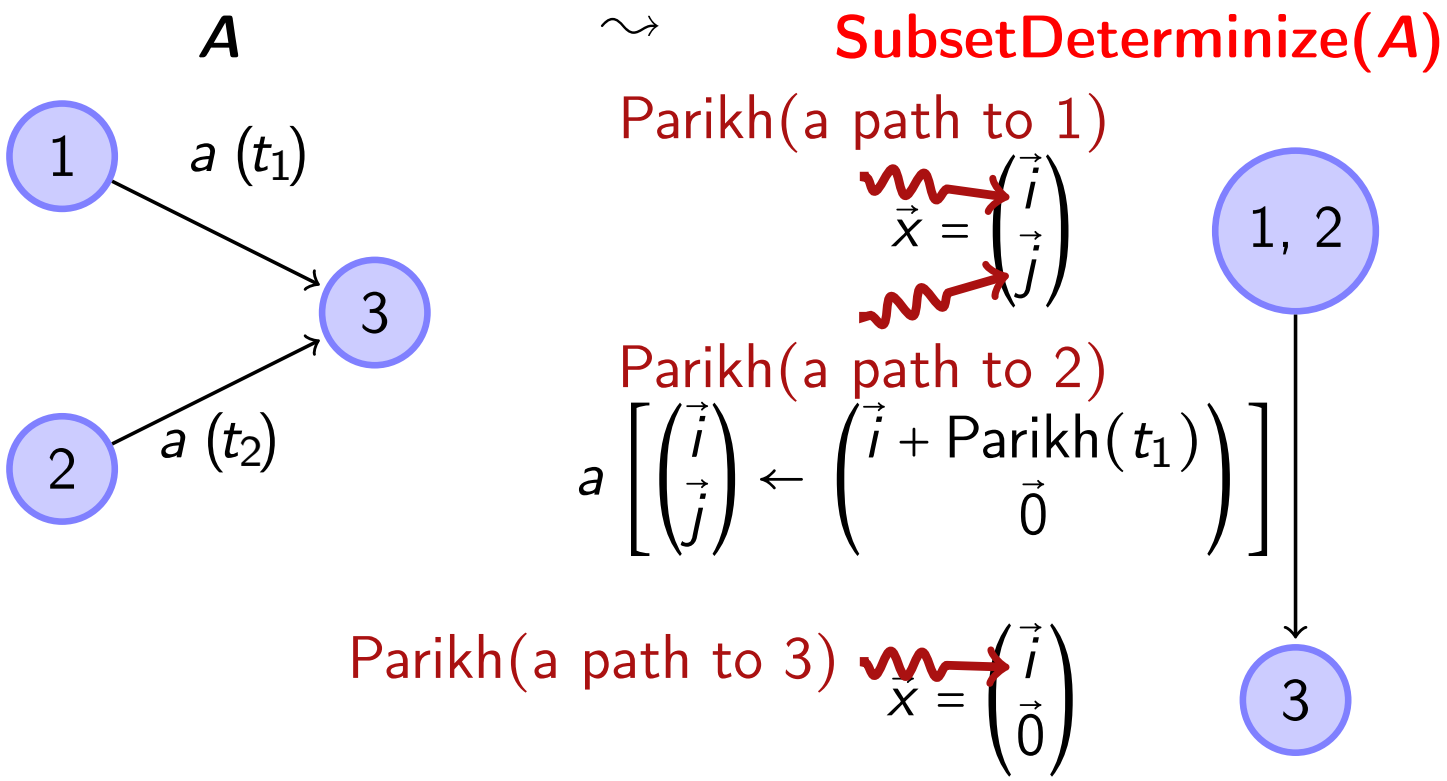
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

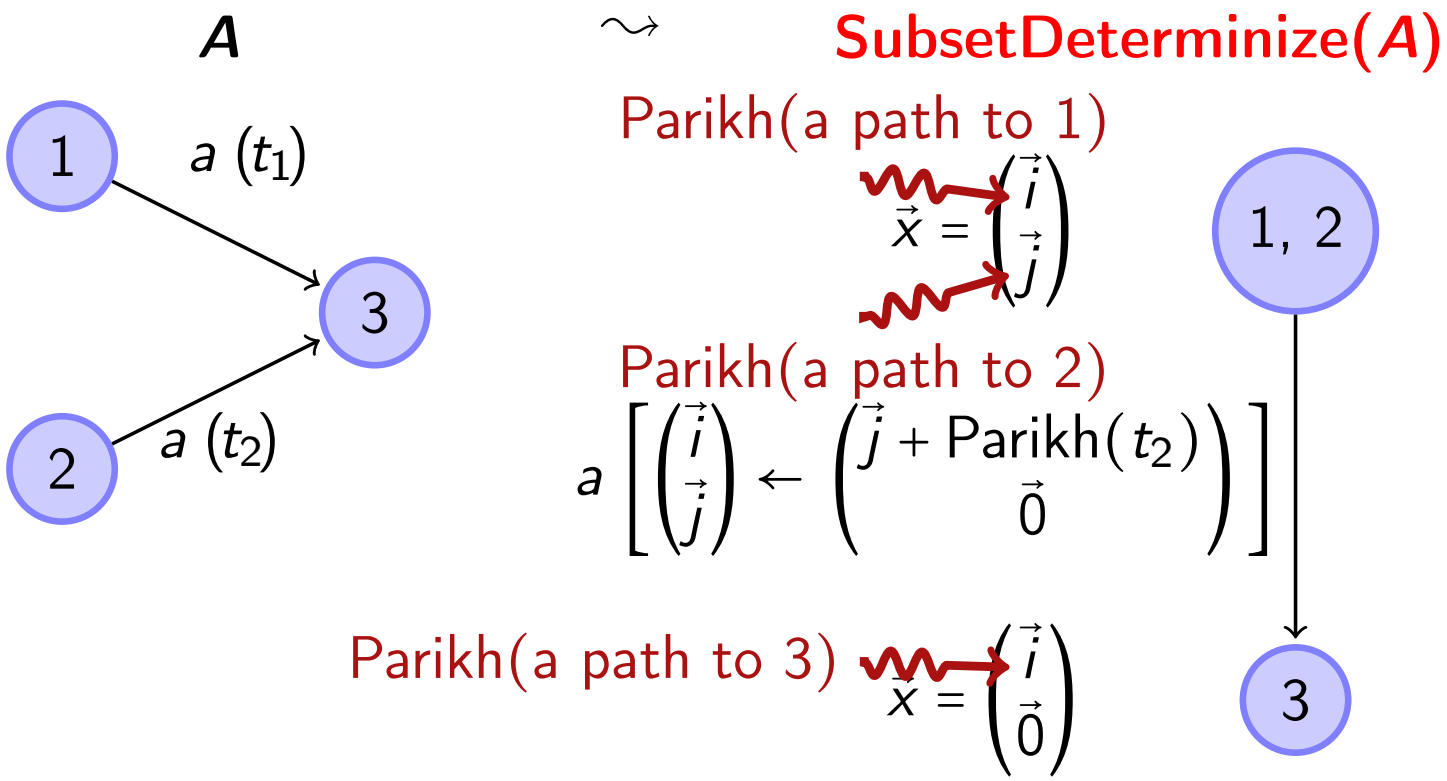
1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



BSL ⊆ DetAPA with some property X

Thus we do the following:

1. Let (A, C) be the PA for $L \in$ BSL
2. Determinize A by the subset construction
3. Associate functions to compute the Parikh image:



Outline

Result and definitions

BSL ⊆ DetPA

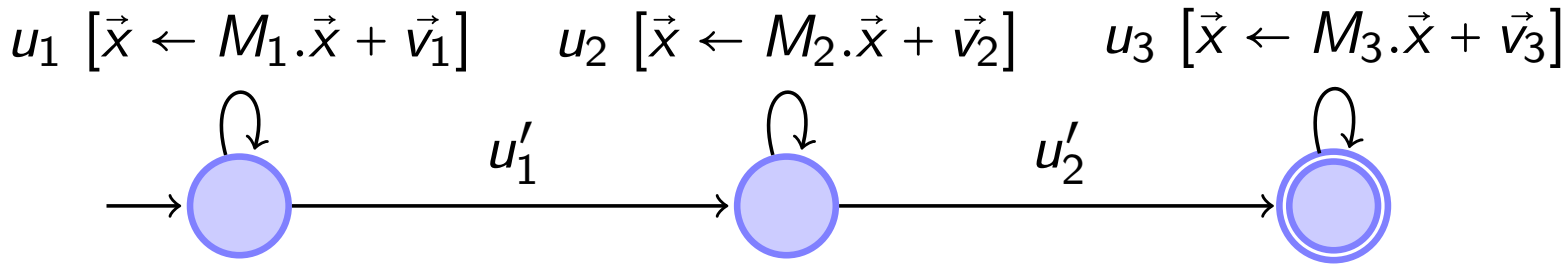
BSL ⊆ DetAPA with some property X

DetAPA with this property ⊆ DetPA

Corollaries and Further Work

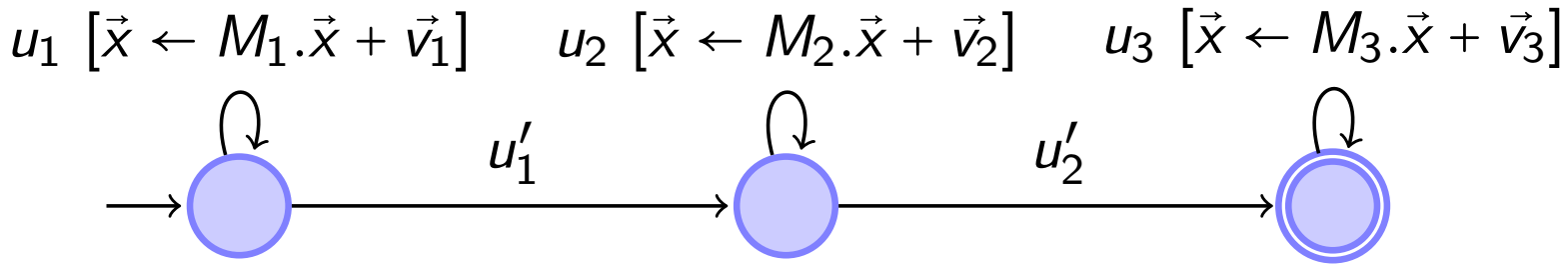
DetAPA with the property X ⊆ DetPA

- We can assume the DetAPA, of language L, is of the form:



DetAPA with the property X ⊆ DetPA

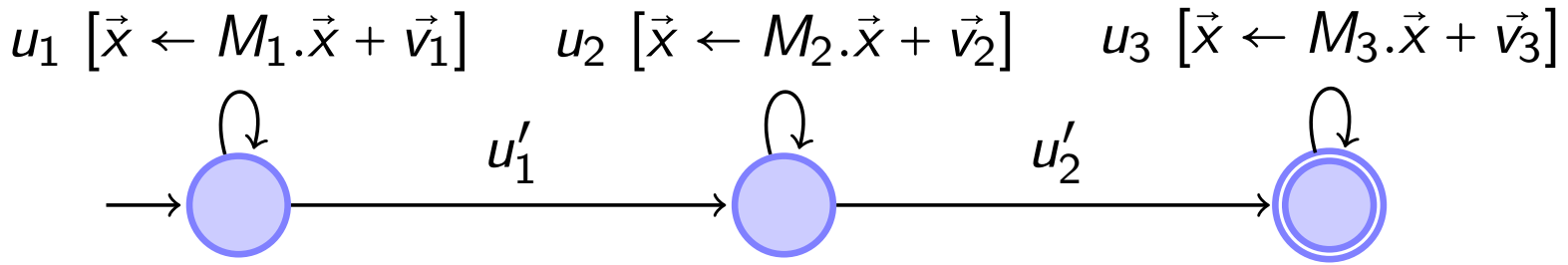
- We can assume the DetAPA, of language L, is of the form:



- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$

DetAPA with the property X ⊆ DetPA

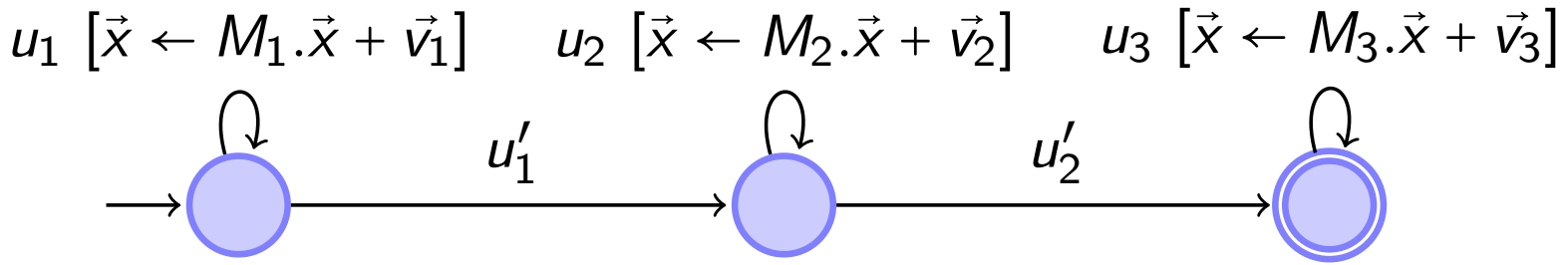
- We can assume the DetAPA, of language L, is of the form:



- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$
- For any $\vec{a} \in \{p_i, \ldots, p_i + k_i\}_{\{1,2,3\}}$, we give a DetPA for $L \cap (u_1^{a_1})(u_1^{k_1})^* \cdot u_1' \cdot (u_2^{a_2})(u_2^{k_2})^* \cdot u_2' \cdot (u_3^{a_3})(u_3^{k_3})^*$

DetAPA with the property X ⊆ DetPA

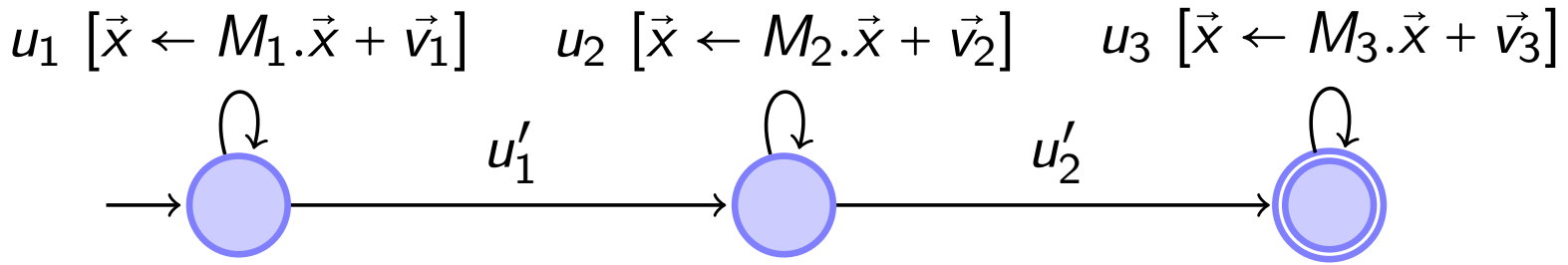
- We can assume the DetAPA, of language L, is of the form:



- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$
- For any $\vec{a} \in \{p_i, \ldots, p_i + k_i\}_{\{1,2,3\}}$, we give a DetPA for $L \cap (u_1^{a_1})(u_1^{k_1})^* \cdot u_1' \cdot (u_2^{a_2})(u_2^{k_2})^* \cdot u_2' \cdot (u_3^{a_3})(u_3^{k_3})^*$
- Construction main idea: suppose a word contains t times $u_2^{k_2}$. Final value of $\vec{x}$ contains:

DetAPA with the property X ⊆ DetPA

- We can assume the DetAPA, of language L, is of the form:

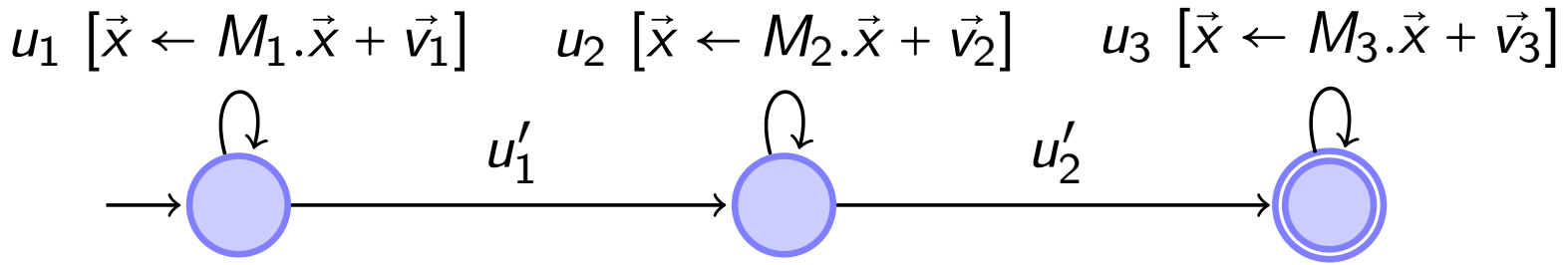


- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$
- For any $\vec{a} \in \{p_i, \ldots, p_i + k_i\}_{\{1,2,3\}}$, we give a DetPA for $L \cap (u_1^{a_1})(u_1^{k_1})^* \cdot u_1' \cdot (u_2^{a_2})(u_2^{k_2})^* \cdot u_2' \cdot (u_3^{a_3})(u_3^{k_3})^*$
- Construction main idea: suppose a word contains t times $u_2^{k_2}$. Final value of $\vec{x}$ contains:

$$\sum_{i=1}^{t} \cdots M_2^{i \times k_2} M_2^{a_2} \cdots$$

DetAPA with the property X ⊆ DetPA

- We can assume the DetAPA, of language L, is of the form:

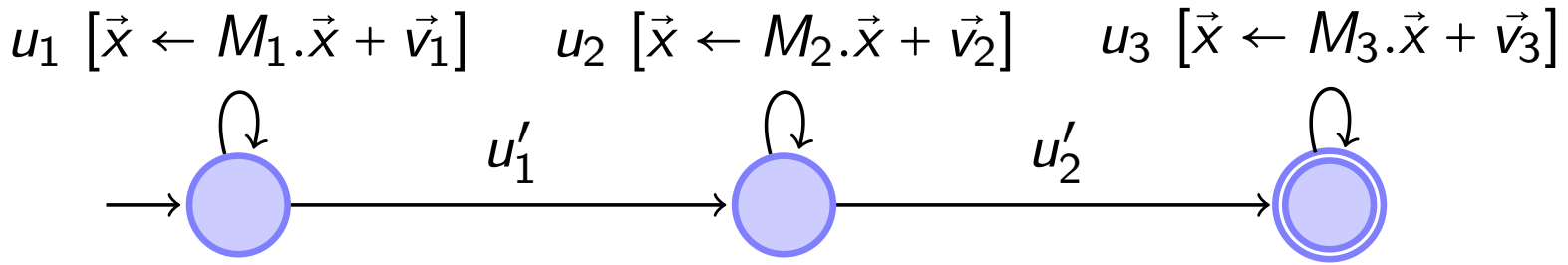


- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$
- For any $\vec{a} \in \{p_i, \ldots, p_i + k_i\}_{\{1,2,3\}}$, we give a DetPA for $L \cap (u_1^{a_1})(u_1^{k_1})^* \cdot u_1' \cdot (u_2^{a_2})(u_2^{k_2})^* \cdot u_2' \cdot (u_3^{a_3})(u_3^{k_3})^*$
- Construction main idea: suppose a word contains t times $u_2^{k_2}$. Final value of $\vec{x}$ contains:

$$\sum_{i=1}^{t} \cdots M_2^{i \times k_2} M_2^{a_2} \cdots = t \times \cdots M_2^{a_2} \cdots$$

DetAPA with the property X ⊆ DetPA

- We can assume the DetAPA, of language L, is of the form:



- Property X: For all i, there are p_i, k_i s.t. $M_i^{p_i} = M_i^{p_i+k_i}$
- For any $\vec{a} \in \{p_i, \ldots, p_i + k_i\}_{\{1,2,3\}}$, we give a DetPA for $L \cap (u_1^{a_1})(u_1^{k_1})^* \cdot u_1' \cdot (u_2^{a_2})(u_2^{k_2})^* \cdot u_2' \cdot (u_3^{a_3})(u_3^{k_3})^*$
- Construction main idea: suppose a word contains t times $u_2^{k_2}$. Final value of $\vec{x}$ contains:
$$\sum_{i=1}^{t} \cdots M_2^{i \times k_2} M_2^{a_2} \cdots = t \times \cdots M_2^{a_2} \cdots$$
- Contribution of $(u_2^{k_2})$ *constant*

Outline

Result and definitions

BSL ⊆ DetPA

Corollaries and Further Work

Corollaries and Further Work

Corollaries:

- Recall PA = RBCM; moreover DetPA $\subsetneq$ DetRBCM, thus RBCM accepting a bounded language can be determinized

Corollaries and Further Work

Corollaries:

- Recall PA = RBCM; moreover DetPA ⊊ DetRBCM, thus RBCM accepting a bounded language can be determinized
- In fact, the resulting DetPA has a special form: union of *flat* automata → CQDD
  [Bouajjani and Habermehl, 1999]; thus CQDD = BSL

Corollaries and Further Work

Corollaries:

- Recall PA = RBCM; moreover DetPA $\subsetneq$ DetRBCM, thus RBCM accepting a bounded language can be determinized
- In fact, the resulting DetPA has a special form: union of *flat* automata $\rightarrow$ CQDD
  [Bouajjani and Habermehl, 1999]; thus CQDD = BSL
- Properties of BSL: closure under concat, under $h^{-1} \cap$ BSL,

Corollaries and Further Work

Corollaries:

- Recall PA = RBCM; moreover DetPA $\subsetneq$ DetRBCM, thus RBCM accepting a bounded language can be determinized
- In fact, the resulting DetPA has a special form: union of *flat* automata $\rightarrow$ CQDD
  [Bouajjani and Habermehl, 1999]; thus CQDD = BSL
- Properties of BSL: closure under concat, under $h^{-1} \cap \mathrm{BSL}$, and
  $L \in \mathrm{BSL} \Rightarrow (\forall \vec{w})[L \subseteq w_1^* \cdots w_n^* \rightarrow \mathrm{Iter}_{\vec{w}}(L) \text{ is SL}]$

Corollaries and Further Work

Corollaries:

- Recall PA = RBCM; moreover DetPA $\subsetneq$ DetRBCM, thus RBCM accepting a bounded language can be determinized
- In fact, the resulting DetPA has a special form: union of *flat* automata $\rightarrow$ CQDD
  [Bouajjani and Habermehl, 1999]; thus CQDD = BSL
- Properties of BSL: closure under concat, under $h^{-1} \cap \mathrm{BSL}$, and
  $L \in \mathrm{BSL} \Rightarrow (\forall \vec{w})[L \subseteq w_1^* \cdots w_n^* \rightarrow \mathrm{Iter}_{\vec{w}}(L) \text{ is SL}]$

What implications to algebraic/logic characterizations?

Thank you

Result and definitions

BSL ⊆ DetPA

BSL ⊆ DetAPA with some property X

DetAPA with this property ⊆ DetPA

Corollaries and Further Work

Bibliography I


Bouajjani, A. and Habermehl, P. (1999).
Symbolic reachability analysis of FIFO-channel systems with nonregular sets of configurations.
In *Theoretical Computer Science*.


Ibarra, O. H. (1978).
Reversal-bounded multicounter machines and their decision problems.
J. ACM, 25(1):116–133.


Klaedtke, F. and Rueß, H. (2003).
Monadic second-order logics with cardinalities.
In *Proceedings of the 30th International Colloquium on Automata, Languages, and Programming (ICALP 2003)*, volume 2719 of *Lecture Notes in Computer Science*, pages 681–696. Springer-Verlag.

Bibliography II


Mitrana, V. and Stiebe, R. (2001).
Extended finite automata over groups.
Discrete Appl. Math., 108(3):287–300.


Parikh, R. J. (1966).
On context-free languages.
Journal of the ACM, 13(4):570–581.