Biautomata for k-Piecewise Testable Languages

Ondřej Klíma and Libor Polák

Institute of Mathematics and Statistics
Masaryk University, Brno
Czech Republic

MeLa 2012 (DLT 2012)

Topic of the Talk

- Biautomata

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).
- Piecewise testable languages

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).
- Piecewise testable languages
  - are studied in the algebraic theory of regular languages,

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).
- Piecewise testable languages
  - are studied in the algebraic theory of regular languages,
  - were characterized by Simon (via syntactic monoids),

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).
- Piecewise testable languages
  - are studied in the algebraic theory of regular languages,
  - were characterized by Simon (via syntactic monoids),
  - form level 1 in the Straubing-Thérien hierarchy of star-free languages.

Topic of the Talk

- Biautomata – a new notion (Klíma, Polák – NCMA'11).
- Piecewise testable languages
  - are studied in the algebraic theory of regular languages,
  - were characterized by Simon (via syntactic monoids),
  - form level 1 in the Straubing-Thérien hierarchy of star-free languages.

Note: An effective characterization of level 2 is an open problem for 40 years.

Outline

- Introduction
- Biautomata
- Our New Results

I. Introduction

Piecewise Testable Languages

Definition

A language L over an alphabet A is called piecewise testable if it is a Boolean combination of languages of the form

$$A^*a_1A^*a_2A^* \dots A^*a_\ell A^*, \text{ where } a_1,\dots,a_\ell \in A,\ \ell \geq 0\,. \qquad (*)$$

Piecewise Testable Languages

Definition

A language L over an alphabet A is called piecewise testable if it is a Boolean combination of languages of the form

$$A^*a_1A^*a_2A^* \dots A^*a_\ell A^*, \text{ where } a_1, \dots, a_\ell \in A, \ \ell \geq 0 \,. \qquad (*)$$

Theorem (Simon '72)

A regular language L is piecewise testable if and only if the syntactic monoid $\mathrm{M}(L)$ *of L is* $\mathcal{J}$*-trivial.*

Piecewise Testable Languages

Definition

A language L over an alphabet A is called piecewise testable if it is a Boolean combination of languages of the form

$$A^*a_1A^*a_2A^*\dots A^*a_\ell A^*, \text{ where } a_1,\dots,a_\ell \in A,\ \ell \geq 0\,. \quad (*)$$

Theorem (Simon '72)

A regular language L is piecewise testable if and only if the syntactic monoid $\mathrm{M}(L)$ of L is $\mathcal{J}$-trivial.

Definition

A language L is called k-piecewise testable if L can be written as a Boolean combination of languages of the form $(*)$ with $\ell \leq k$.

k-piecewise Testable Languages

- Question (for each k) : The k-piecewise testability.

k-piecewise Testable Languages

- Question (for each k) : The k-piecewise testability.
- Solution: The least k such that a given piecewise testable language L is k-piecewise testable can be found by brute-force algorithms.

k-piecewise Testable Languages

- Question (for each k) : The k-piecewise testability.
- Solution: The least k such that a given piecewise testable language L is k-piecewise testable can be found by brute-force algorithms.
  - For each fixed k and a fixed alphabet A, there are only finitely many k-piecewise testable languages over A.

k-piecewise Testable Languages

- Question (for each k) : The k-piecewise testability.
- Solution: The least k such that a given piecewise testable language L is k-piecewise testable can be found by brute-force algorithms.
  - For each fixed k and a fixed alphabet A, there are only finitely many k-piecewise testable languages over A.
  - A bit sophisticated algorithm: via Eilenberg's correspondence (using relatively free monoids).

k-piecewise Testable Languages

- Question (for each k) : The k-piecewise testability.
- Solution: The least k such that a given piecewise testable language L is k-piecewise testable can be found by brute-force algorithms.
    - For each fixed k and a fixed alphabet A, there are only finitely many k-piecewise testable languages over A.
    - A bit sophisticated algorithm: via Eilenberg's correspondence (using relatively free monoids).

Both methods are unrealistic in practice.

- Also using identies does not help (no finite bases in general).

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Instead of that, for a given piecewise testable language L, we would like to find a good estimate, i.e. a (possibly small) number k, such that L is k-piecewise testable.

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Instead of that, for a given piecewise testable language L, we would like to find a good estimate, i.e. a (possibly small) number k, such that L is k-piecewise testable.

- Simon: $k = 2n - 1$ where n is the maximal length of a $\mathcal{J}$-chain in the syntactic monoid of L.

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Instead of that, for a given piecewise testable language L, we would like to find a good estimate, i.e. a (possibly small) number k, such that L is k-piecewise testable.

- Simon: $k = 2n - 1$ where n is the maximal length of a $\mathcal{J}$-chain in the syntactic monoid of L.
- OK: $k = \ell + r - 2$ where ℓ and r are the maximal lengths of chains for the orderings $\leq_{\mathcal{L}}$ and $\leq_{\mathcal{R}}$.

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Instead of that, for a given piecewise testable language L, we would like to find a good estimate, i.e. a (possibly small) number k, such that L is k-piecewise testable.

- Simon: $k = 2n - 1$ where n is the maximal length of a $\mathcal{J}$-chain in the syntactic monoid of L.
- OK: $k = \ell + r - 2$ where ℓ and r are the maximal lengths of chains for the orderings $\leq_{\mathcal{L}}$ and $\leq_{\mathcal{R}}$.

Note: $\ell \leq n$ and $r \leq n$ and hence $\ell + r - 2 < 2n - 1$.

Basic Goal

Our ambition, in this contribution, is not to decide the k-piecewise testability in a reasonable computational time.

Instead of that, for a given piecewise testable language L, we would like to find a good estimate, i.e. a (possibly small) number k, such that L is k-piecewise testable.

- Simon: $k = 2n - 1$ where n is the maximal length of a $\mathcal{J}$-chain in the syntactic monoid of L.
- OK: $k = \ell + r - 2$ where ℓ and r are the maximal lengths of chains for the orderings $\leq_{\mathcal{L}}$ and $\leq_{\mathcal{R}}$.

Note: $\ell \leq n$ and $r \leq n$ and hence $\ell + r - 2 < 2n - 1$.

We found a different proof (OK+LP: NCMA'11) of Simon's result using a notion of biautomaton.

Biautomaton Characterization

Theorem (OK+LP '11)

A language L is piecewise testable if and only if its canonical biautomaton $\mathcal{C}_L$ of L is acyclic.

- Note: Loops are not considered as cycles.

Biautomaton Characterization

Theorem (OK+LP '11)

A language L is piecewise testable if and only if its canonical biautomaton $\mathcal{C}_L$ of L is acyclic.

- Note: Loops are not considered as cycles.
- The core of the proof was to show that if $\mathcal{C}_L$ has m states then L is $2m$-piecewise testable.

Biautomaton Characterization

Theorem (OK+LP '11)

A language L is piecewise testable if and only if its canonical biautomaton $\mathcal{C}_L$ of L is acyclic.

- Note: Loops are not considered as cycles.
- The core of the proof was to show that if $\mathcal{C}_L$ has m states then L is $2m$-piecewise testable.
- Here we improve this result in two directions:
  - We eliminate the coefficient 2,
  - We replace the size of $\mathcal{C}_L$ by the depth of the biautomaton.

A depth of an acyclic biautomaton $\mathcal{B}$ is the length of the longest simple path in $\mathcal{B}$.

Main Result

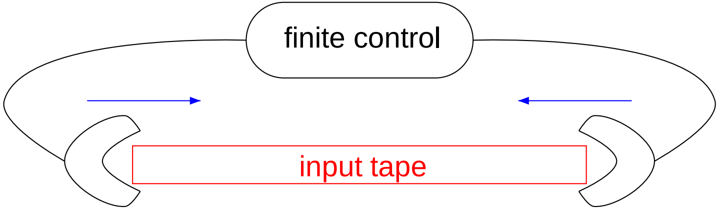
The main result of our contribution:

Theorem

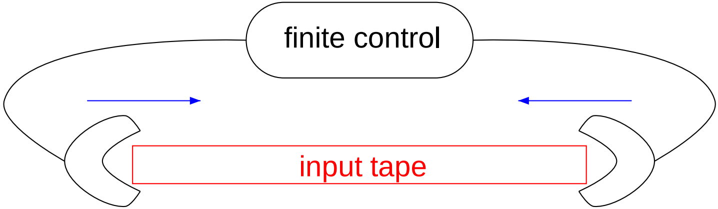
Let L be a piecewise testable language with an (acyclic) canonical biautomaton of depth k. Then L is k-piecewise testable.

II. Biautomata

Informal Description of Biautomata

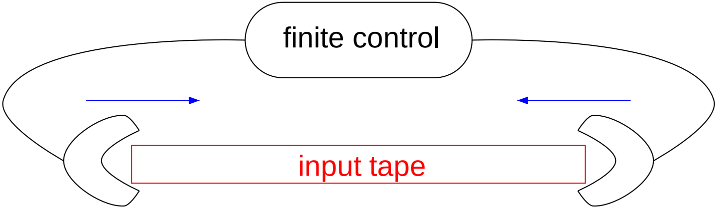


Informal Description of Biautomata



- The heads read symbols alternately, not depending on the current state of the finite control or on the symbol read from the tape.

Informal Description of Biautomata



- The heads read symbols alternately, not depending on the current state of the finite control or on the symbol read from the tape.
- The acceptance of a word depends neither on the position, in which the heads meet, nor on the sequence of states the finite control goes through.

Formal Definition of Biautomata

Definition (OK+LP – NCMA 2011)

A *biautomaton* is a sixtuple $\mathcal{B} = (Q, A, \cdot, \circ, i, T)$ where

(1) Q is a non-empty finite set of states,

(2) A is a finite alphabet,

(3) $\cdot : Q \times A \to Q$ is a left action,

(4) $\circ : Q \times A \to Q$ is a right action,

(5) $i \in Q$ is the initial state,

(6) $T \subseteq Q$ is the set of terminal states,

(7) $(q \cdot a) \circ b = (q \circ b) \cdot a$ for each $q \in Q$ and $a, b \in A$,

(8) $q \cdot a \in T$ if and only if $q \circ a \in T$ for each $q \in Q$ and $a \in A$.

Formal Definition of Biautomata

Definition (OK+LP – NCMA 2011)

A *biautomaton* is a sixtuple $\mathcal{B} = (Q, A, \cdot, \circ, i, T)$ where

(1) Q is a non-empty finite set of states,

(2) A is a finite alphabet,

(3) $\cdot : Q \times A \to Q$ is a left action,

(4) $\circ : Q \times A \to Q$ is a right action,

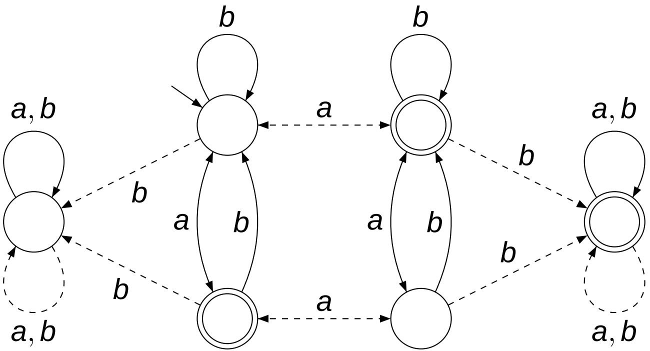
(5) $i \in Q$ is the initial state,

(6) $T \subseteq Q$ is the set of terminal states,

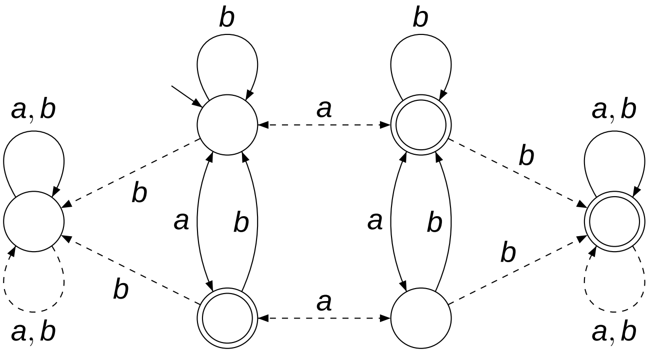
(7') $(q \cdot u) \circ v = (q \circ v) \cdot u$ for each $q \in Q$ and $u, v \in A^*$,

(8') $q \cdot u \in T$ if and only if $q \circ u \in T$ for each $q \in Q$ and $u \in A^*$.

Example 1

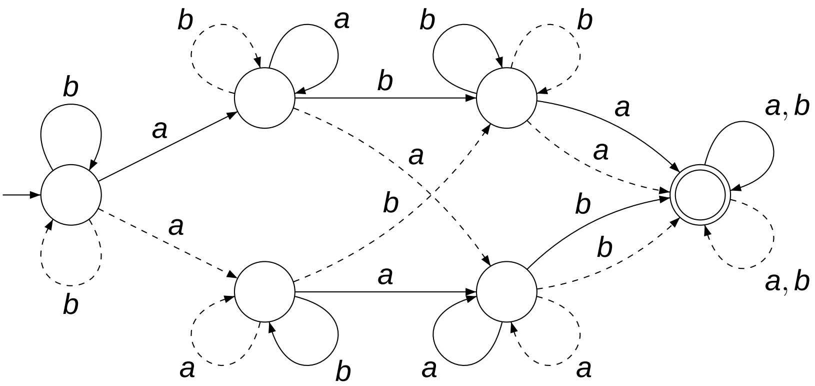


Example 1

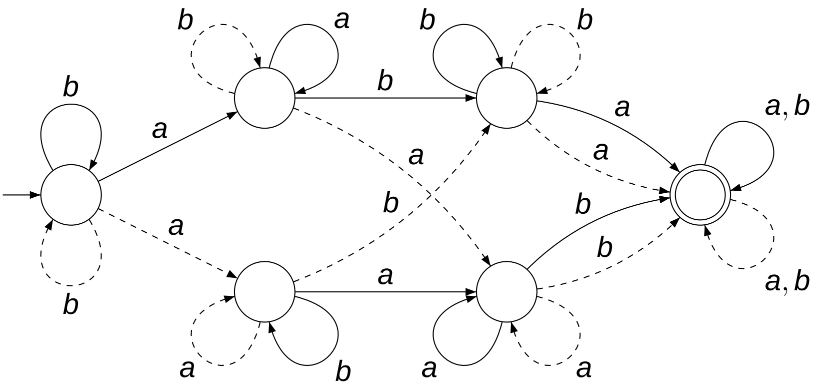


The diagram represents a biautomaton; it is not acyclic.

Example 2

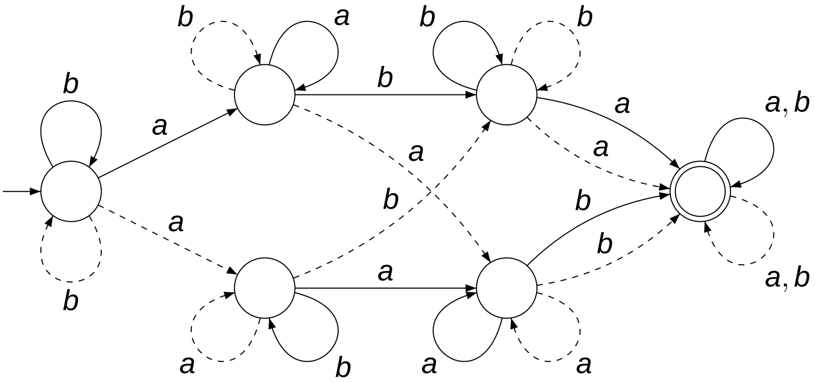


Example 2



The biautomaton is acyclic.

Example 2



The biautomaton is acyclic. Its depth is 3.

Acceptance of an Input Word

- The biautomaton $\mathcal{B}$ accepts a given word $u \in A^*$ if $i \cdot u \in T$.

Acceptance of an Input Word

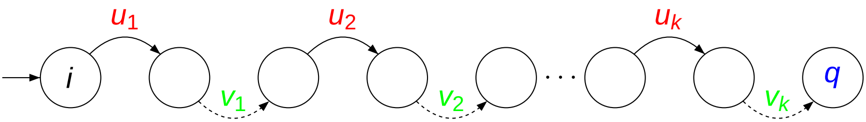
- The biautomaton $\mathcal{B}$ accepts a given word $u \in A^*$ if $i \cdot u \in T$.
- $\mathcal{B}$ accepts $u \in A^*$ iff $i \circ u \in T$.

Acceptance of an Input Word

- The biautomaton $\mathcal{B}$ accepts a given word $u \in A^*$ if $i \cdot u \in T$.
- $\mathcal{B}$ accepts $u \in A^*$ iff $i \circ u \in T$.
- But $\mathcal{B}$ can read the word u in many other ways:

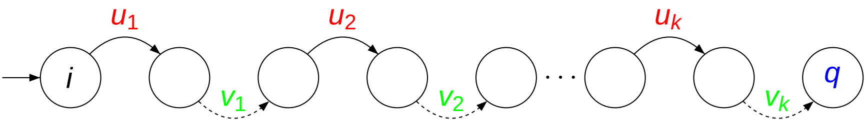
Acceptance of an Input Word

- The biautomaton $\mathcal{B}$ accepts a given word $u \in A^*$ if $i \cdot u \in T$.
- $\mathcal{B}$ accepts $u \in A^*$ iff $i \circ u \in T$.
- But $\mathcal{B}$ can read the word u in many other ways:
  We can divide $u = u_1 u_2 \dots u_k v_k \dots v_2 v_1$ arbitrarily, where $u_1, \dots, v_1 \in A^*$, and we read u_1 from left first, then v_1 from right, then u_2 from left, and so on.



Acceptance of an Input Word

- The biautomaton $\mathcal{B}$ accepts a given word $u \in A^*$ if $i \cdot u \in T$.
- $\mathcal{B}$ accepts $u \in A^*$ iff $i \circ u \in T$ iff $q \in T$.
- But $\mathcal{B}$ can read the word u in many other ways: We can divide $u = u_1 u_2 \dots u_k v_k \dots v_2 v_1$ arbitrarily, where $u_1, \dots, v_1 \in A^*$, and we read u_1 from left first, then v_1 from right, then u_2 from left, and so on.



$$q = ((\dots((((i \cdot u_1) \circ v_1) \cdot u_2) \circ v_2) \dots) \cdot u_k) \circ v_k \, .$$

Basic Properties of Biautomata

- The part $\{ i \cdot u \mid u \in A^* \} \subseteq Q$ together with the left actions is a DFA, which recognizes the same language.

Basic Properties of Biautomata

- The part $\{ i \cdot u \mid u \in A^* \} \subseteq Q$ together with the left actions is a DFA, which recognizes the same language.
- From a finite deterministic automaton one can construct a biautomaton which recognizes the same language.

Basic Properties of Biautomata

- The part $\{ i \cdot u \mid u \in A^* \} \subseteq Q$ together with the left actions is a DFA, which recognizes the same language.
- From a finite deterministic automaton one can construct a biautomaton which recognizes the same language.
- Biautomata recognize exactly regular languages.

Basic Properties of Biautomata

- The part $\{ i \cdot u \mid u \in A^* \} \subseteq Q$ together with the left actions is a DFA, which recognizes the same language.
- From a finite deterministic automaton one can construct a biautomaton which recognizes the same language.
- Biautomata recognize exactly regular languages.
- There is a minimal biautomaton recognizing a given regular language L and it is unique up to isomorphism.

Basic Properties of Biautomata

- The part $\{\, i \cdot u \mid u \in A^* \,\} \subseteq Q$ together with the left actions is a DFA, which recognizes the same language.
- From a finite deterministic automaton one can construct a biautomaton which recognizes the same language.
- Biautomata recognize exactly regular languages.
- There is a minimal biautomaton recognizing a given regular language L and it is unique up to isomorphism.

The so called canonical biautomaton is an analogy of Brzozowski's construction of a minimal complete deterministic automaton. It uses two-sided derivatives:

Canonical Biautomaton

For a language $L \subseteq A^*$ and $u, v \in A^*$, we define

$$u^{-1}Lv^{-1} = \{\, w \in A^* \mid uwv \in L \,\}, \quad C_L = \{\, u^{-1}Lv^{-1} \mid u, v \in A^* \,\}.$$

We define $\mathcal{C}_L = (C_L, A, \cdot, \circ, L, T)$, where

- $q \cdot a = a^{-1}q$,
- $q \circ a = qa^{-1}$,
- $u^{-1}Lv^{-1} \in T$ iff $\lambda \in u^{-1}Lv^{-1}$ (iff $uv \in L$).

Canonical Biautomaton

For a language $L \subseteq A^*$ and $u, v \in A^*$, we define

$$u^{-1}Lv^{-1} = \{\, w \in A^* \mid uwv \in L \,\}, \quad C_L = \{\, u^{-1}Lv^{-1} \mid u, v \in A^* \,\}.$$

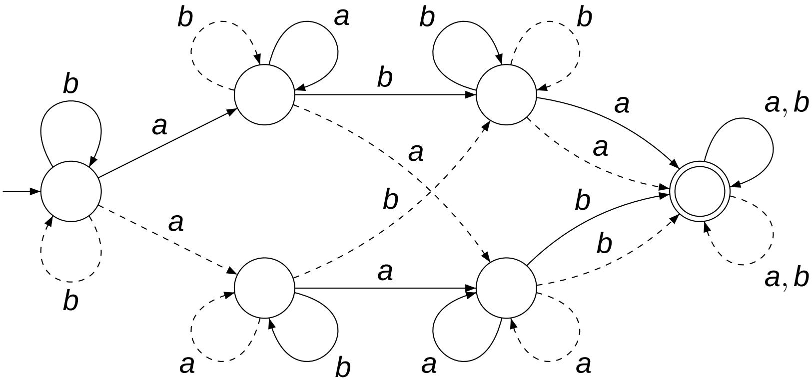
We define $\mathcal{C}_L = (C_L, A, \cdot, \circ, L, T)$, where

- $q \cdot a = a^{-1}q$,
- $q \circ a = qa^{-1}$,
- $u^{-1}Lv^{-1} \in T$ iff $\lambda \in u^{-1}Lv^{-1}$ (iff $uv \in L$).

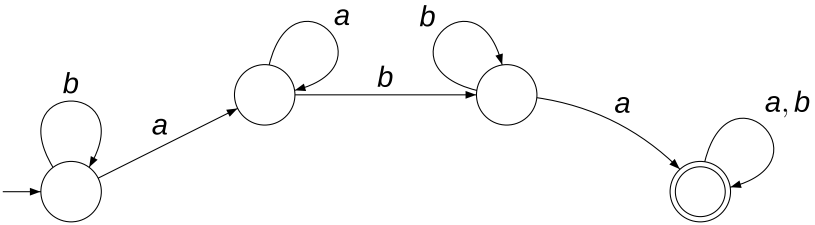
Lemma

For each regular language L over A, the structure $\mathcal{C}_L$ is a biautomaton, which recognizes the language L and it is minimal.

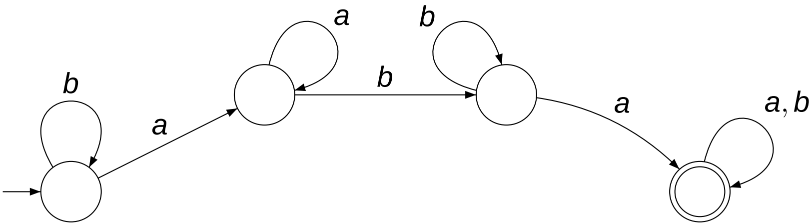
Example 2



Example 2

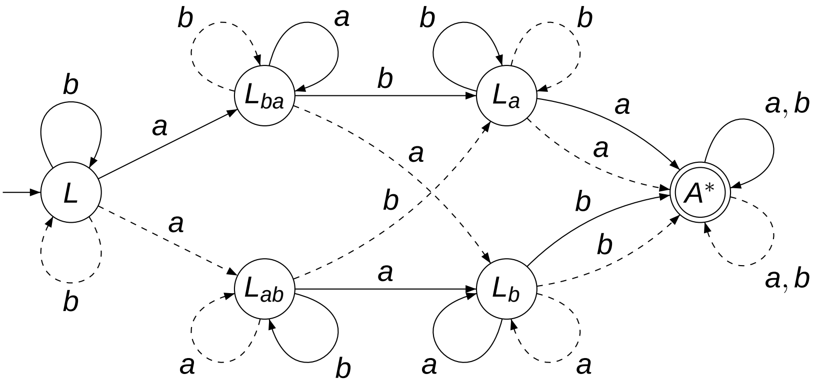


Example 2



The (bi)automaton recognizes $L = L_{aba} = A^*aA^*bA^*aA^*$.

Example 2



$L = L_{aba} = A^*aA^*bA^*aA^*$, $a^{-1}L = A^*bA^*aA^* = L_{ba}$,
$La^{-1} = A^*aA^*bA^* = L_{ab}$, $a^{-1}L_{ab} = L_b$, ...

III. Our New Results

Proof of the Main Result

Recall the formulation of the main result:

Theorem

Let L be a piecewise testable language with an (acyclic) canonical biautomaton of depth k. Then L is k-piecewise testable.

Proof of the Main Result

Recall the formulation of the main result:

Theorem

Let L be a piecewise testable language with an (acyclic) canonical biautomaton of depth k. Then L is k-piecewise testable.

First we need a characterization of k-piecewise testable languages:
For $u, v \in A^*$, the meaning $u \sim_k v$ is that u and v have the same subwords of lengths $\leq k$.

Proof of the Main Result

Recall the formulation of the main result:

Theorem

Let L be a piecewise testable language with an (acyclic) canonical biautomaton of depth k. Then L is k-piecewise testable.

First we need a characterization of k-piecewise testable languages:
For $u, v \in A^*$, the meaning $u \sim_k v$ is that u and v have the same subwords of lengths $\leq k$.

Lemma

A language L is k-piecewise testable if and only if L is a union of classes of the partition $A^*/\sim_k$.

Proof of the Main Result

The main result is a consequence on the following:

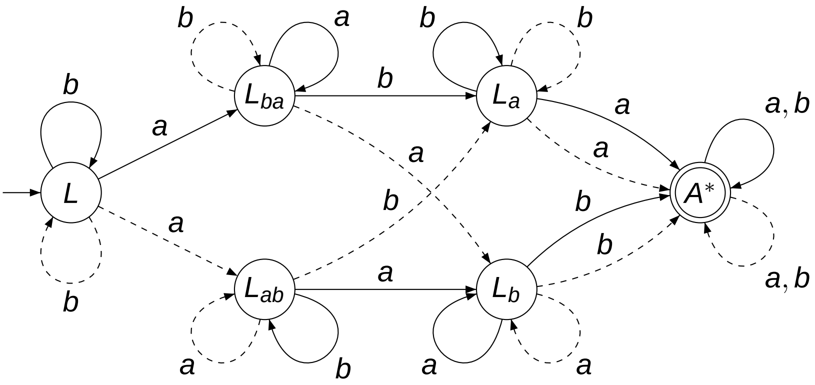
Proposition

Let $\mathcal{B} = (Q, A, \cdot, \circ, i, T)$ be an acyclic biautomaton with all states reachable and with depth $\mathcal{B} = \ell$. *Then, for every $u, v \in A^*$ such that $u \sim_\ell v$, we have*

$$u \in L_{\mathcal{B}} \quad \textit{if and only if} \quad v \in L_{\mathcal{B}} .$$

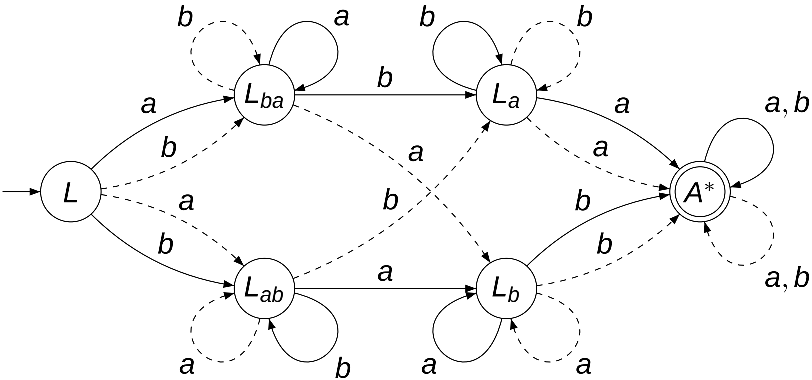
- The proof runs by induction with respect to ℓ.
- It is quite technically involved.
- A sketch could be find in Proceedings of DLT and a full version is placed in the first author's home page.

Example 2



The biautomaton is acyclic of depth 3. It recognizes the 3-piecewise language $L = L_{aba}$ which is not 2-piecewise testable.

Example 3



This biautomaton is acyclic of depth 3. It recognizes the 2-piecewise language $L = L_{ab} \cap L_{ba} = L_{aba} \cup L_{bab}$.

Comparing with the Previous Results

When comparing our result with the previous ones we can state the following

Proposition

Let L be a piecewise testable language and let $\mathrm{M}(L)$ *be its* ($\mathcal{J}$*-trivial) syntactic monoid where* ℓ *and* r *are the maximal lengths of chains for the orderings* $\leq_{\mathcal{L}}$ *and* $\leq_{\mathcal{R}}$*. Let* $\mathcal{C}_L$ *be (acyclic) canonical biautomaton of L. Then*

$$\text{depth}\,\mathcal{C}_L \leq \ell + r - 2\,.$$

Comparing with the Previous Results

When comparing our result with the previous ones we can state the following

Proposition

Let L be a piecewise testable language and let $\mathrm{M}(L)$ *be its* ($\mathcal{J}$*-trivial) syntactic monoid where* ℓ *and* r *are the maximal lengths of chains for the orderings* $\leq_{\mathcal{L}}$ *and* $\leq_{\mathcal{R}}$*. Let* $\mathcal{C}_L$ *be (acyclic) canonical biautomaton of L. Then*

$$\text{depth}\, \mathcal{C}_L \leq \ell + r - 2\,.$$

Proposition

For each n, there is a 3-piecewise language L such that depth $\mathcal{C}_L = 4$ *and the maximal length of a chain for the ordering* $\leq_{\mathcal{R}}$ *is* $r = n$.

Future Research

- Questions concerning k-piecewise testable languages:

Future Research

- Questions concerning k-piecewise testable languages:
  - More precise estimate on k. (E.g. using other characterisitics of the canonical biautomaton.)

Future Research

- Questions concerning k-piecewise testable languages:
  - More precise estimate on k. (E.g. using other characterisitics of the canonical biautomaton.)
  - A direct construction of a regular expression for a given biautomaton.

Future Research

- Questions concerning k-piecewise testable languages:
  - More precise estimate on k. (E.g. using other characterisitics of the canonical biautomaton.)
  - A direct construction of a regular expression for a given biautomaton.
- Other application of biautomata: Characterization of significant classes of regular languages.
  - For example, in NCMA'11(extended version) we showed: A language L is prefix-suffix testable (i.e. a Boolean combination of uA^*, A^*v, $u, v \in A^*$) if and only if the canonical biautomaton for the language L satisfies

$$(\text{for each } q \in Q,\, u, v \in A^+)\ q \cdot u = q \circ v = q$$
$$\text{implies that } q \text{ is absorbing}\,.$$

Future Research

- Questions concerning k-piecewise testable languages:
  - More precise estimate on k. (E.g. using other characterisitics of the canonical biautomaton.)
  - A direct construction of a regular expression for a given biautomaton.
- Other application of biautomata: Characterization of significant classes of regular languages.
  - For example, in NCMA'11(extended version) we showed: A language L is prefix-suffix testable (i.e. a Boolean combination of $uA^*,\ A^*v,\ u,v \in A^*$) if and only if the canonical biautomaton for the language L satisfies

$$(\text{for each } q \in Q,\ u,v \in A^+)\ q \cdot u = q \circ v = q$$
$$\text{implies that } q \text{ is absorbing}\,.$$

  - We have other characterizations, e.g. level 1/2 (for $L \neq \emptyset$: acyclic, a single final state and it is absorbing).

Future Research

- Questions concerning k-piecewise testable languages:
  - More precise estimate on k. (E.g. using other characterisitics of the canonical biautomaton.)
  - A direct construction of a regular expression for a given biautomaton.
- Other application of biautomata: Characterization of significant classes of regular languages.
  - For example, in NCMA'11(extended version) we showed: A language L is prefix-suffix testable (i.e. a Boolean combination of $uA^*,\ A^*v,\ u,v \in A^*$) if and only if the canonical biautomaton for the language L satisfies

$$\text{(for each } q \in Q,\ u,v \in A^+)\ q \cdot u = q \circ v = q$$
$$\text{implies that } q \text{ is absorbing}\,.$$

  - We have other characterizations, e.g. level 1/2 (for $L \neq \emptyset$: acyclic, a single final state and it is absorbing).
  - 3/2 ???

THANK YOU