k-block deterministic languages

Clément MIKLARZ

Université de Rouen
Laboratoire d'Informatique, du Traitement de l'Information et des Systèmes

June 30, 2017

History

- 1986: Publication of SGML language, using regular expressions with a property:
  - $(a + b)^* c$: OK
  - $(a + b)^* a$: KO

History

- 1986: Publication of SGML language, using regular expressions with a property:
  - $(a+b)^*c$: OK
  - $(a+b)^*a$: KO
- 1998: Brüggemann-Klein, Wood: One-unambiguous regular languages

History

- 1986: Publication of SGML language, using regular expressions with a property:
  - $(a+b)^*c$: OK
  - $(a+b)^*a$: KO
- 1998: Brüggemann-Klein, Wood: One-unambiguous regular languages
- 2001: Giammarresi, Montalbano, Wood: Block-deterministic regular languages

History

- 1986: Publication of SGML language, using regular expressions with a property:
  - $(a+b)^*c$: OK
  - $(a+b)^*a$: KO
- 1998: Brüggemann-Klein, Wood: One-unambiguous regular languages
- 2001: Giammarresi, Montalbano, Wood: Block-deterministic regular languages
- 2008: Han, Wood: Generalizations of 1-deterministic regular languages

History

- 1986: Publication of SGML language, using regular expressions with a property:
  - $(a+b)^*c$: OK
  - $(a+b)^*a$: KO
- 1998: Brüggemann-Klein, Wood: One-unambiguous regular languages
- 2001: Giammarresi, Montalbano, Wood: Block-deterministic regular languages
- 2008: Han, Wood: Generalizations of 1-deterministic regular languages
- 2015: Caron, Mignot, Miklarz: On the hierarchy of block deterministic languages

Plan

1. Basics
2. 1-unambiguous languages
3. k-block deterministic languages

1 Basics

2 1-unambiguous languages

3 k-block deterministic languages

Regular expressions and languages

Regular expression E	Language $\mathrm{L}(E)$ denoted by E
$\emptyset$	$\emptyset$
ε	$\{\varepsilon\}$
a	$\{a\}$
$F + G$	$\mathrm{L}(F) \cup \mathrm{L}(G)$
$F \cdot G$	$\mathrm{L}(F) \cdot \mathrm{L}(G)$
F^*	$\bigcup_{k \in \mathbb{N}} \mathrm{L}(F)^k$ with $\mathrm{L}(F)^0 = \{\varepsilon\}$

Regular expressions and languages

Regular expression E	Language $\mathrm{L}(E)$ denoted by E
$\emptyset$	$\emptyset$
ε	$\{\varepsilon\}$
a	$\{a\}$
$F + G$	$\mathrm{L}(F) \cup \mathrm{L}(G)$
$F \cdot G$	$\mathrm{L}(F) \cdot \mathrm{L}(G)$
F^*	$\bigcup_{k \in \mathbb{N}} \mathrm{L}(F)^k$ with $\mathrm{L}(F)^0 = \{\varepsilon\}$

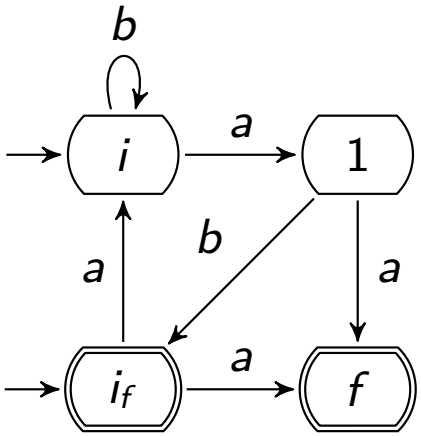
Examples

- $E = (a + bc)(d + e)$ with $\mathrm{L}(E) = \{ad, ae, bcd, bce\}$
- $F = (a + bc)^*$ with $\mathrm{L}(F) = \{\varepsilon, a, aa, bc, aaa, abc, bca, \ldots\}$

Automata

An *automaton* A is a 5-tuple $(\Sigma, Q, I, F, \delta)$ with:

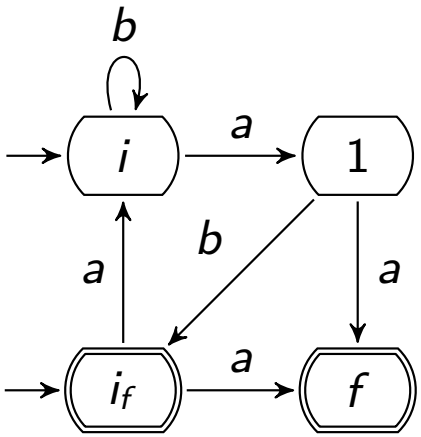
- Σ a finite alphabet
- Q a finite set of states
- $I \subset Q$ the set of initial states
- $F \subset Q$ the set of final states
- $\delta \subset Q \times \Sigma \times Q$ a set of labelled transitions



Automata

An *automaton* A is a 5-tuple $(\Sigma, Q, I, F, \delta)$ with:

- Σ a finite alphabet
- Q a finite set of states
- $I \subset Q$ the set of initial states
- $F \subset Q$ the set of final states
- $\delta \subset Q \times \Sigma \times Q$ a set of labelled transitions



Right language of a state

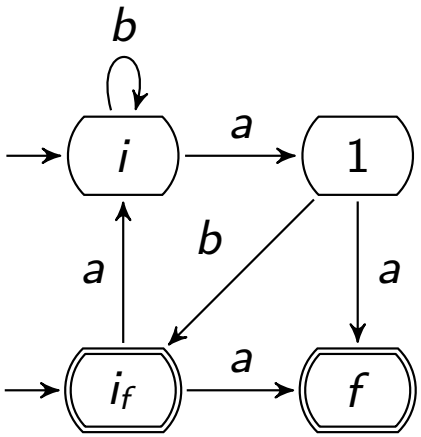
The set of words which can reach a final state from this state.

$$\mathrm{L}(f) = \{\varepsilon\}$$
$$\mathrm{L}(1) = \{a, b, ba, baaa, \ldots\}$$

Automata

An *automaton* A is a 5-tuple $(\Sigma, Q, I, F, \delta)$ with:

- Σ a finite alphabet
- Q a finite set of states
- $I \subset Q$ the set of initial states
- $F \subset Q$ the set of final states
- $\delta \subset Q \times \Sigma \times Q$ a set of labelled transitions



Right language of a state

The set of words which can reach a final state from this state.

$$\mathrm{L}(f) = \{\varepsilon\}$$
$$\mathrm{L}(1) = \{a, b, ba, baaa, \ldots\}$$

Language recognized by an automaton

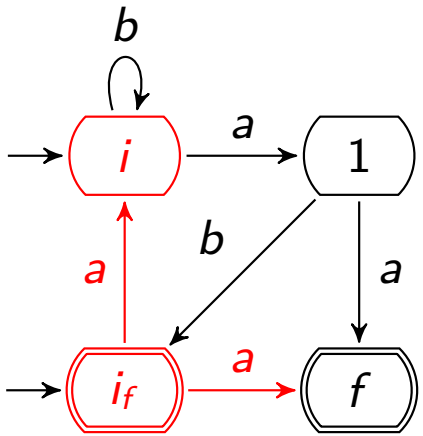
The union of the right languages of the initial states.

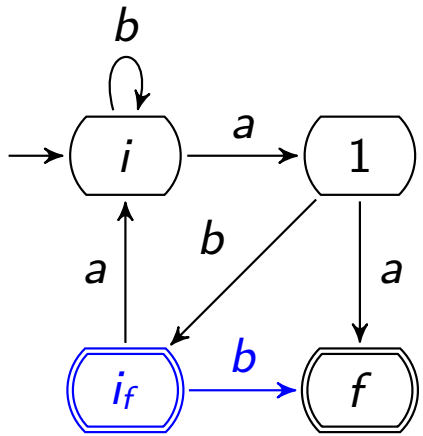
$$L = \{\varepsilon, a, aa, ab, aaa, aab, \ldots\}$$

Deterministic automata (DFA)

An automaton $A = (\Sigma, Q, I, F, \delta)$ is *deterministic* if:

- $|I| = 1$
- $\forall t_1 = (p, a, q_1), t_2 = (p, b, q_2) \in \delta, (t_1 \neq t_2) \Longrightarrow (a \neq b)$

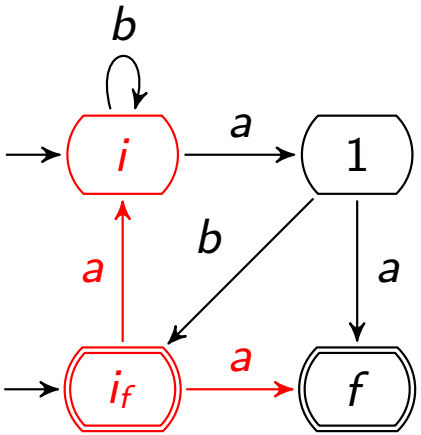


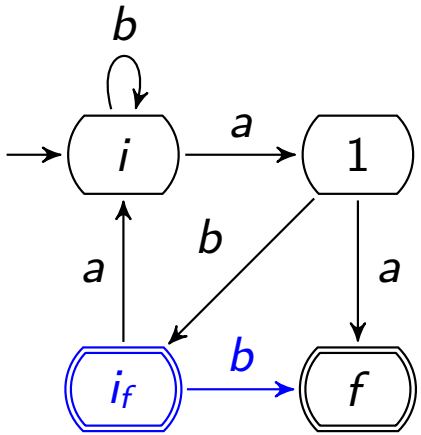


Deterministic automata (DFA)

An automaton $A = (\Sigma, Q, I, F, \delta)$ is *deterministic* if:

- $|I| = 1$
- $\forall t_1 = (p, a, q_1), t_2 = (p, b, q_2) \in \delta, (t_1 \neq t_2) \Longrightarrow (a \neq b)$





Minimality

A DFA is minimal if there does not exist an equivalent DFA with less states.

Properties of a minimal DFA

- no two equivalent states
- a canonical automaton of a language

The Glushkov automaton of a regular expression

$$E = (a + (bb)^*)ab^* + \varepsilon$$

Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 b_5^* + \varepsilon$$

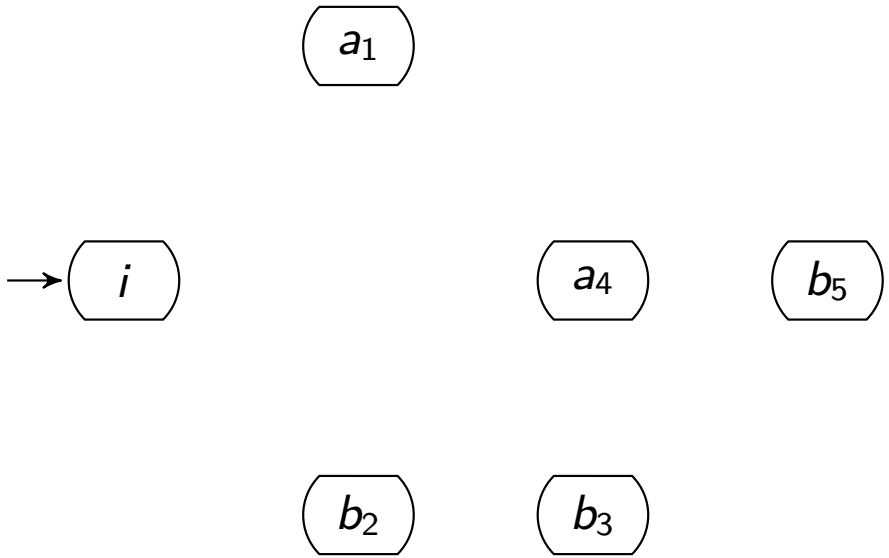


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 b_5^* + \varepsilon$$

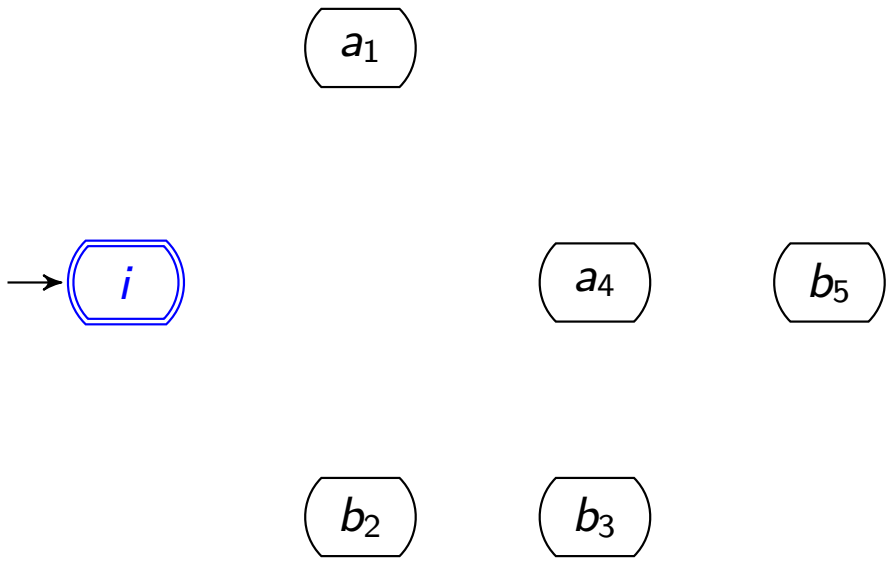


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 {b_5}^* + \varepsilon$$

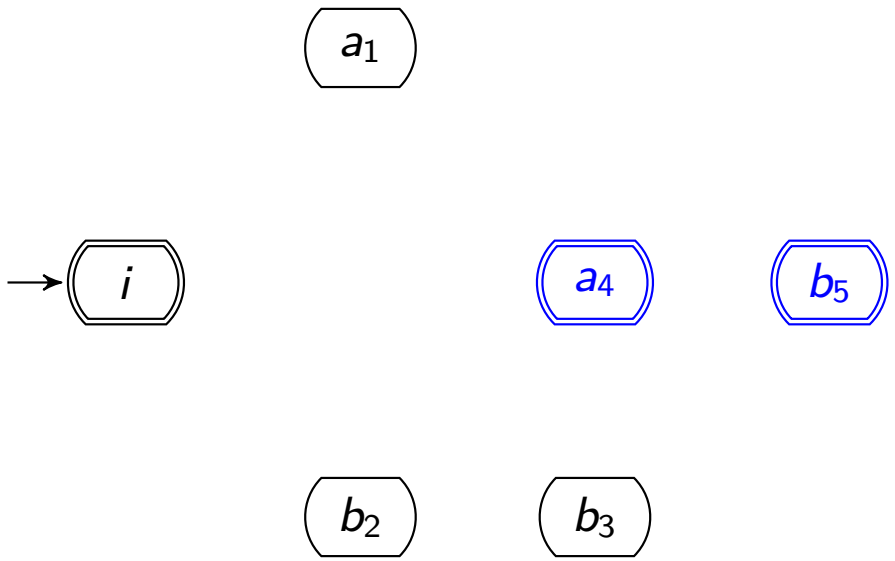


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 b_5^* + \varepsilon$$

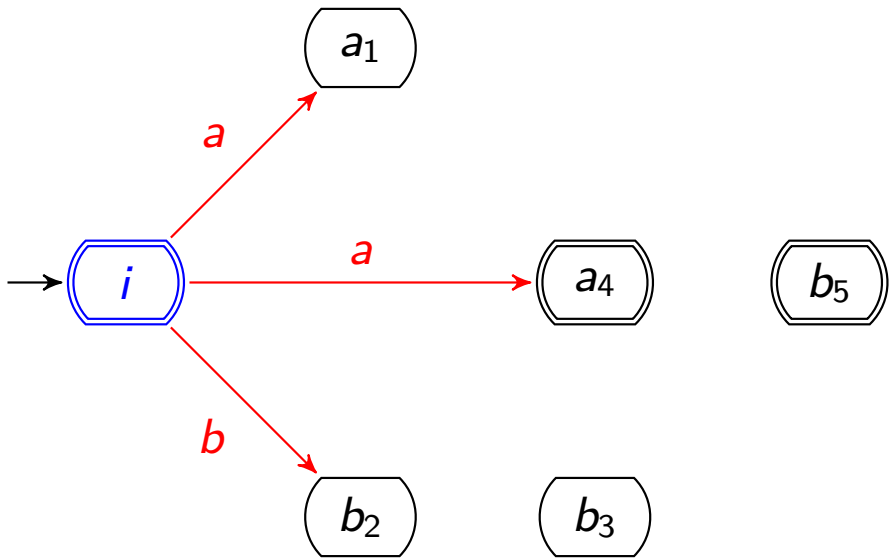


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 b_5^* + \varepsilon$$

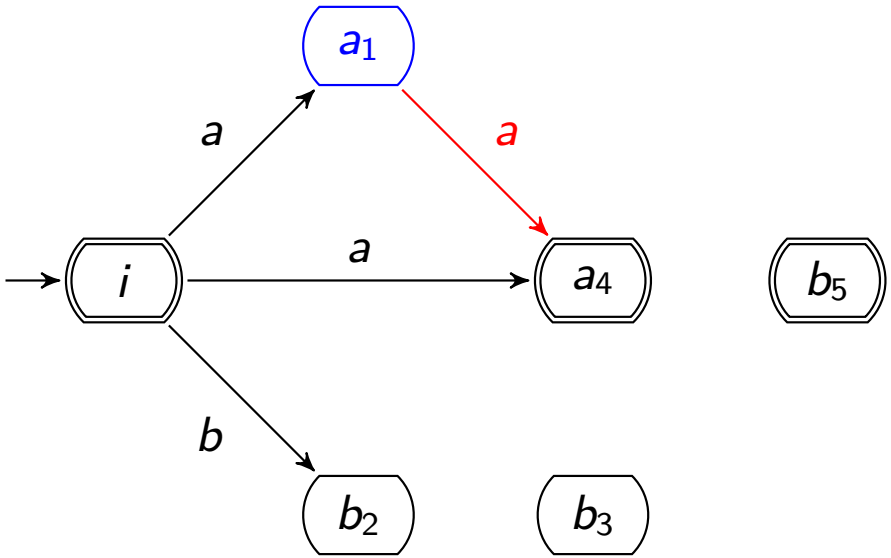


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2\, b_3)^*)\, a_4\, b_5^* + \varepsilon$$

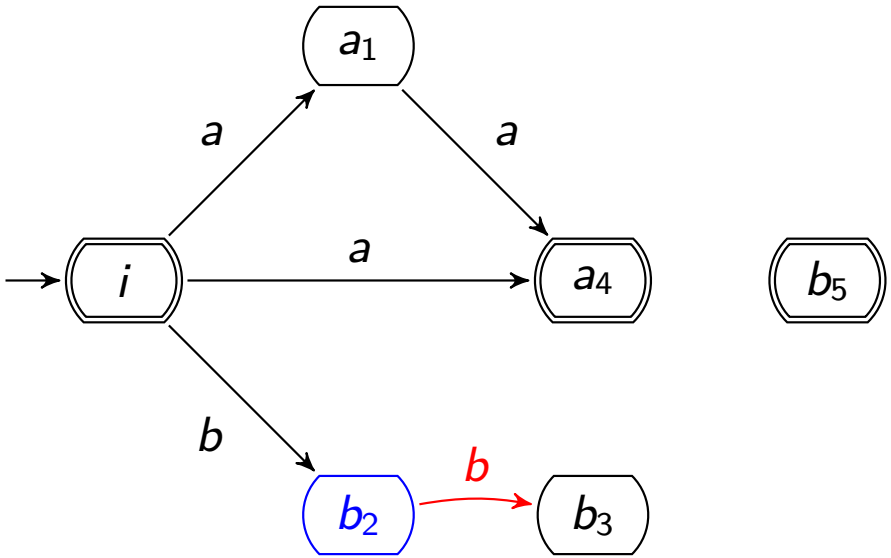


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 b_5^* + \varepsilon$$

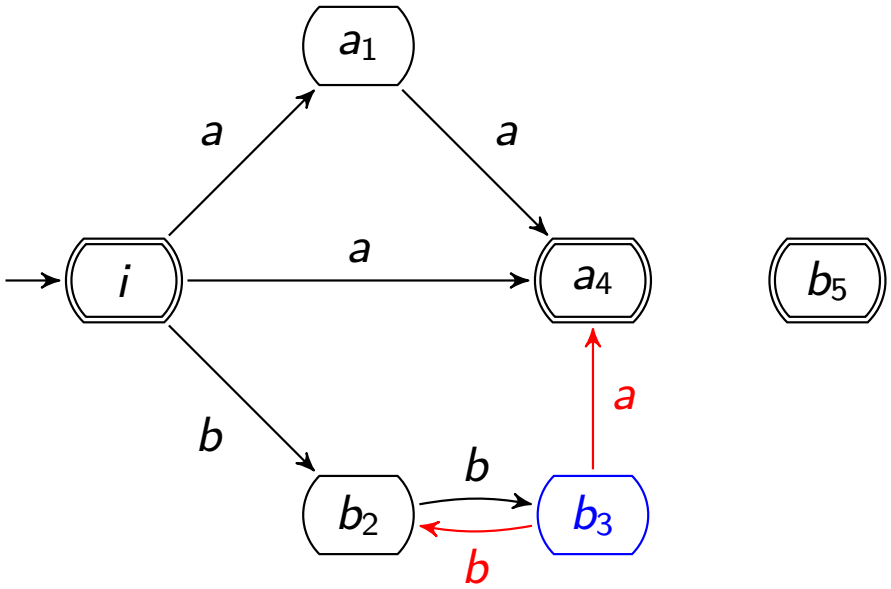


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 {b_5}^* + \varepsilon$$

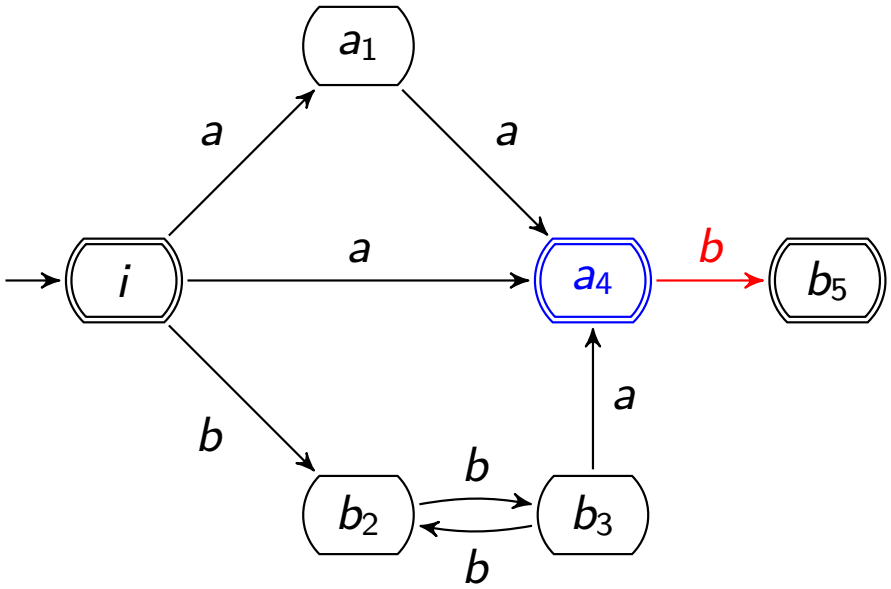


Figure: The Glushkov automaton of E

The Glushkov automaton of a regular expression

$$E^\sharp = (a_1 + (b_2 b_3)^*) a_4 {b_5}^* + \varepsilon$$

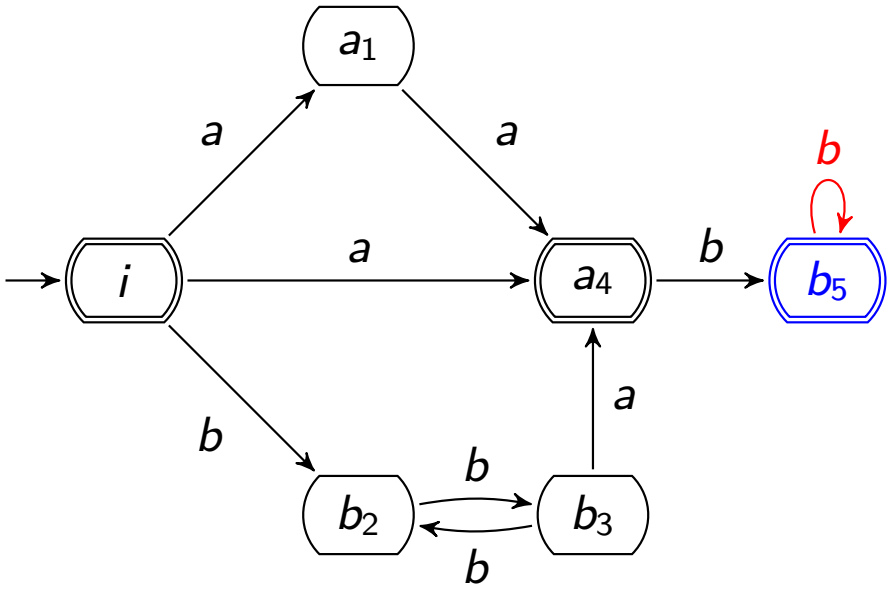


Figure: The Glushkov automaton of E

1 Basics

2 1-unambiguous languages

3 k-block deterministic languages

Definition

A language is 1-*unambiguous* if it can be denoted by a regular expression whose Glushkov automaton is deterministic.

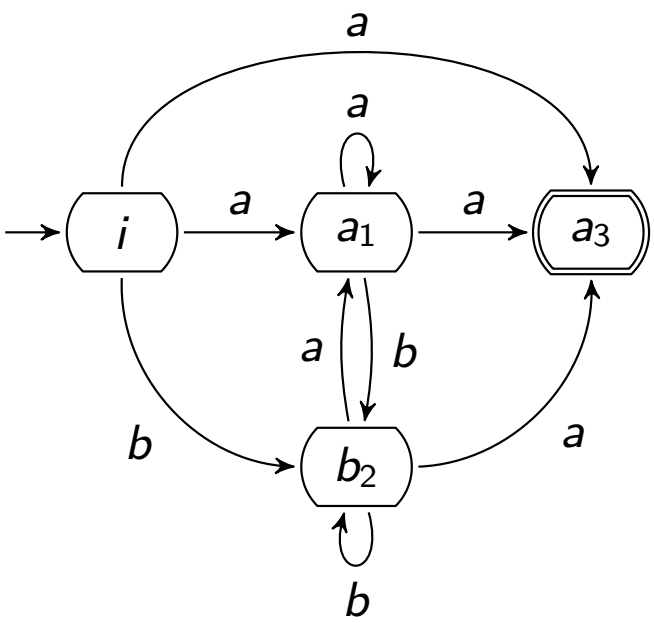


Figure: The Glushkov automaton of $(a+b)^*a$

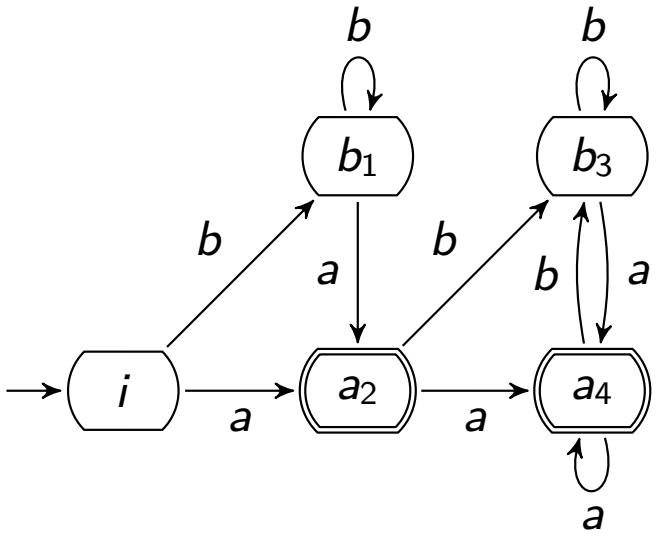


Figure: The Glushkov automaton of $b^*a(b^*a)^*$

Decidability

Does such a regular expression exist ?

Orbits

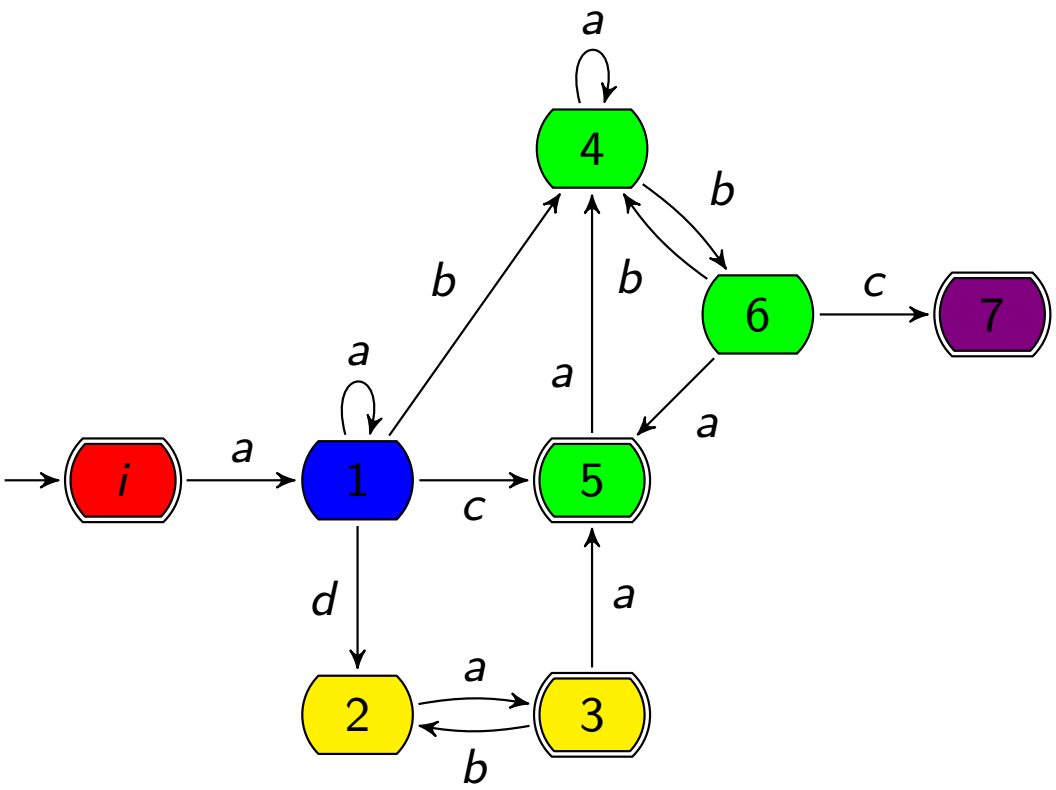


Figure: 5 orbits

Gates

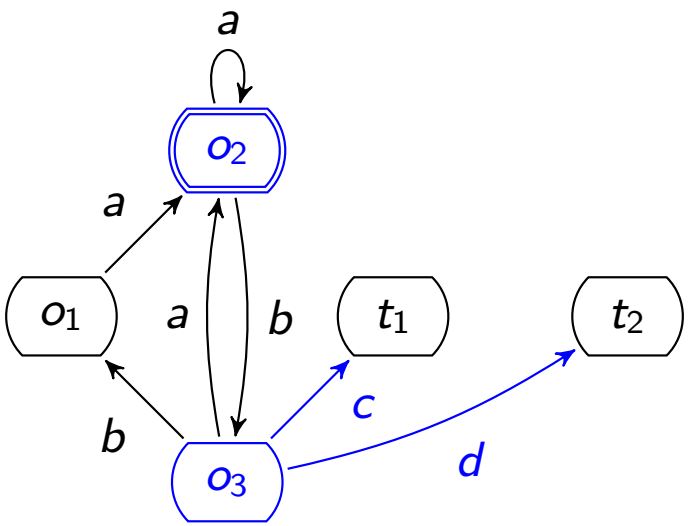
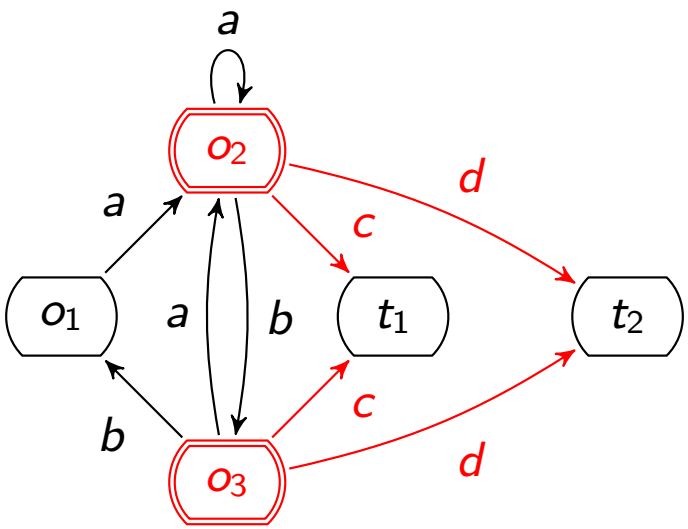
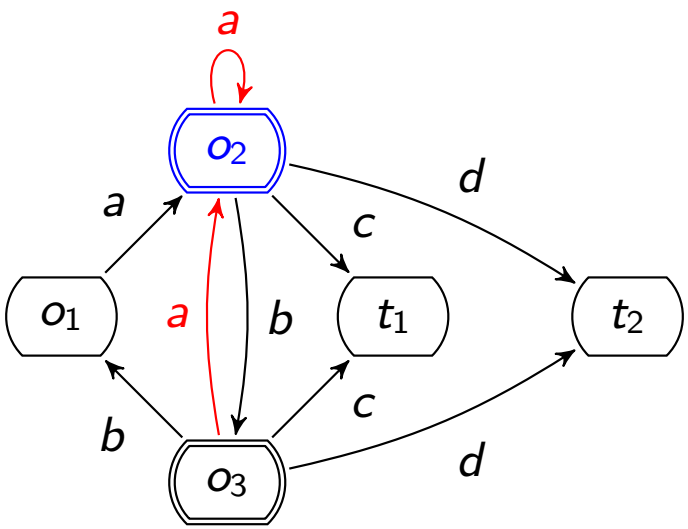


Figure: 2 gates

External synchronisation



States of internal synchronisation



Deciding 1-unambiguity

BW-test: necessary conditions

- every orbit is externally synchronized
- every non-trivial orbit has at least one state of internal synchronisation.

Characterisation [Brüggemann-Klein, Wood]

A language is 1-unambiguous if and only if its minimal DFA passes the BW-test.

Deciding 1-unambiguity

BW-test: necessary conditions

- every orbit is externally synchronized
- every non-trivial orbit has at least one state of internal synchronisation.

Characterisation [Brüggemann-Klein, Wood]

A language is 1-unambiguous if and only if its minimal DFA passes the BW-test.

Strict inclusion

There exist regular languages which are not 1-unambiguous.

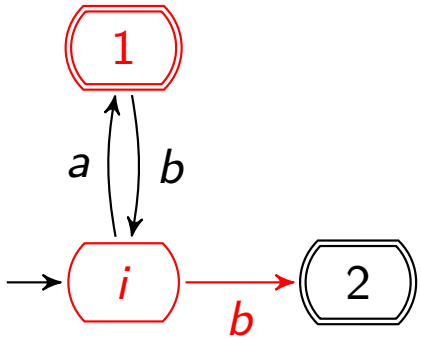


Figure: The minimal DFA of $\mathrm{L}(a(ba)^*(\varepsilon + bb) + b)$

1 Basics

2 1-unambiguous languages

3 k-block deterministic languages

Block regular expressions and automata

- block = non empty word
- k-block: all blocks have length at most k

Block regular expressions and automata

- block = non empty word
- k-block: all blocks have length at most k

A 2-block regular expression

$[aa]^*([ab]b + ba)b^*$

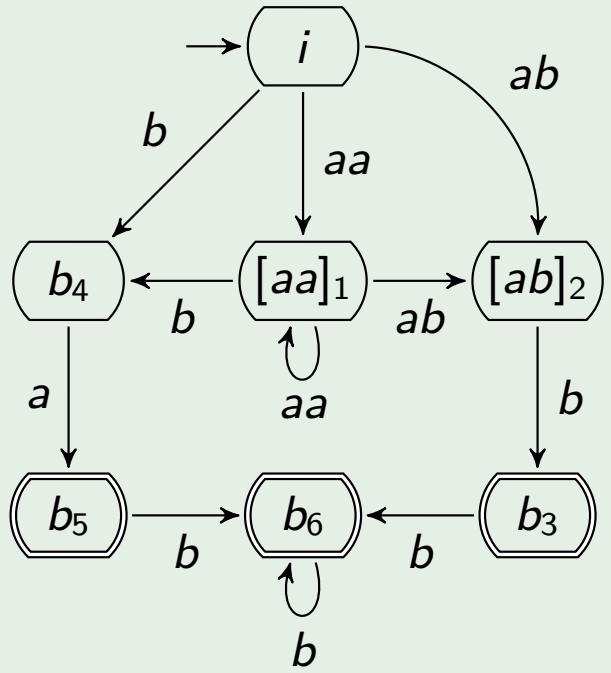
Block regular expressions and automata

- block = non empty word
- k-block: all blocks have length at most k

A 2-block regular expression

$[aa]^*([ab]b + ba)b^*$

A 2-block automaton



Deterministic block automaton

A block automaton $B = (\Gamma, Q, I, F, \delta)$ is *deterministic* if:

- $|I| = 1$
- $\forall t_1 = (p, b_1, q_1), t_2 = (p, b_2, q_2) \in \delta, (t_1 \neq t_2) \Longrightarrow (b_1 \notin \mathrm{Pref}(b_2))$

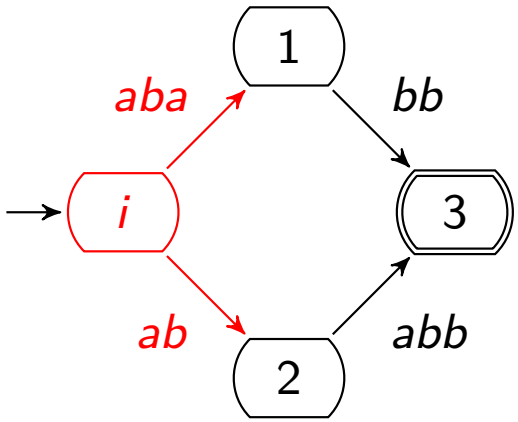


Figure: Non deterministic

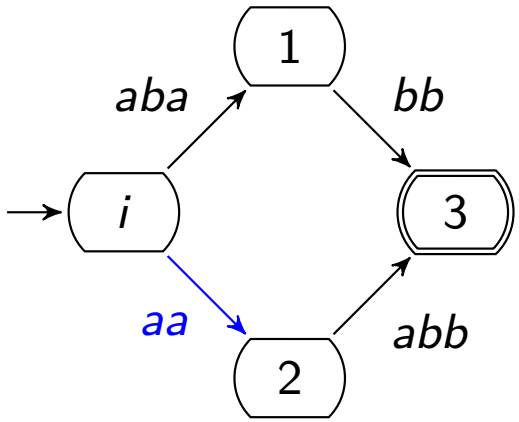


Figure: Deterministic

Deterministic block automaton

A block automaton $B = (\Gamma, Q, I, F, \delta)$ is *deterministic* if:

- $|I| = 1$
- $\forall t_1 = (p, b_1, q_1), t_2 = (p, b_2, q_2) \in \delta, (t_1 \neq t_2) \Longrightarrow (b_1 \notin \mathrm{Pref}(b_2))$

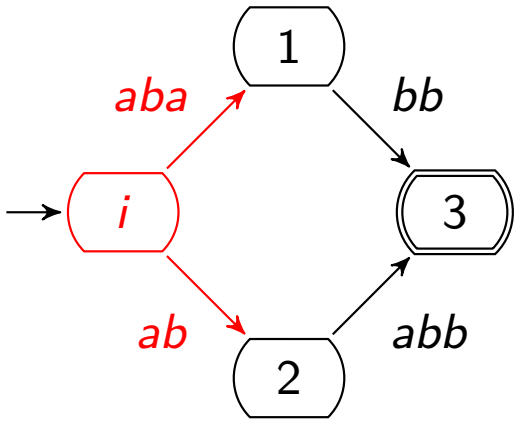


Figure: Non deterministic

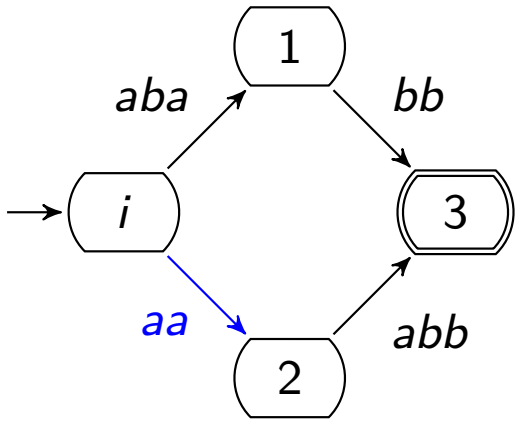


Figure: Deterministic

Block deterministic languages

- A language is k-block deterministic if it can be denoted by a k-block regular expression whose Glushkov automaton is deterministic.
- A language is block deterministic if there exists a k such that its k-block deterministic.

Studying the decidability of block determinism

Theorem

A language is k-block deterministic if and only if it is recognized by a k-block deterministic automaton which passes the BW-test.

Theorem

A language is k-block deterministic if and only if it is recognized by a k-block deterministic automaton which passes the BW-test.

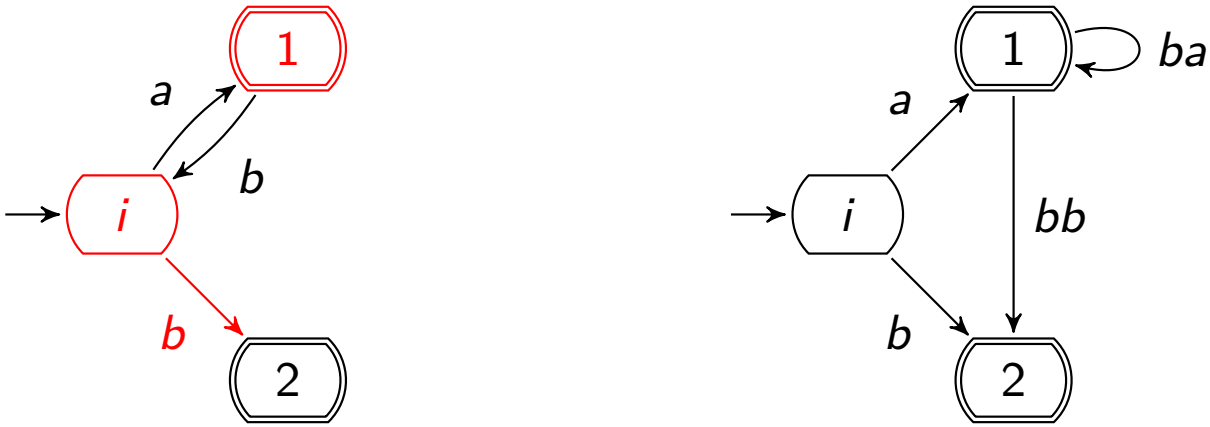


Figure: 2-block deterministic language without being 1-unambiguous

Studying the decidability of block determinism

Theorem

A language is k-block deterministic if and only if it is recognized by a k-block deterministic automaton which passes the BW-test.

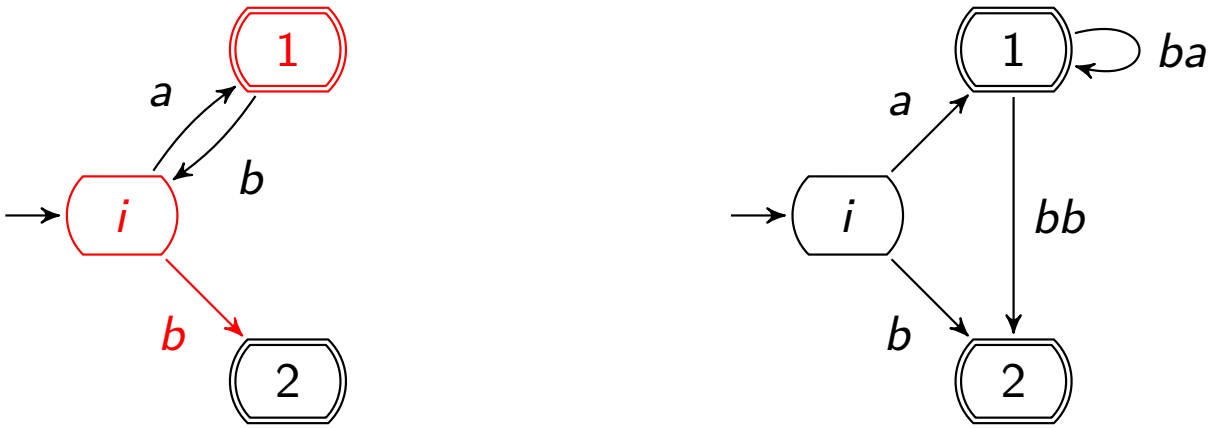


Figure: 2-block deterministic language without being 1-unambiguous

Problem

There can be an infinite number of deterministic k-block automata recognizing a language.

Compact block automata

A block automaton is *compact* if it has no two equivalent states.

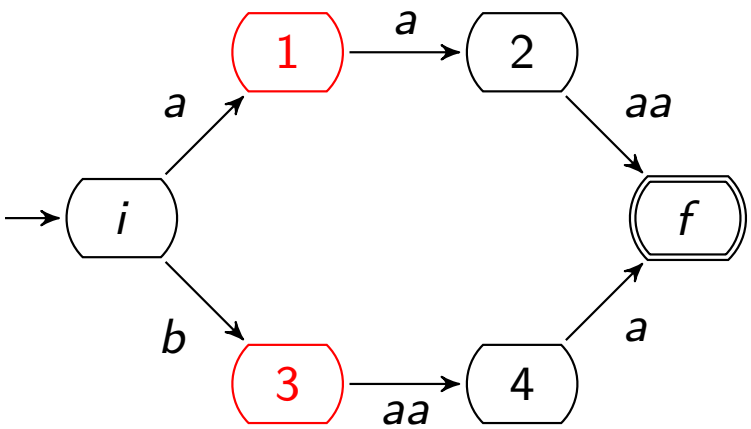


Figure: 1 and 3 are equivalent

Compact block automata

A block automaton is *compact* if it has no two equivalent states.

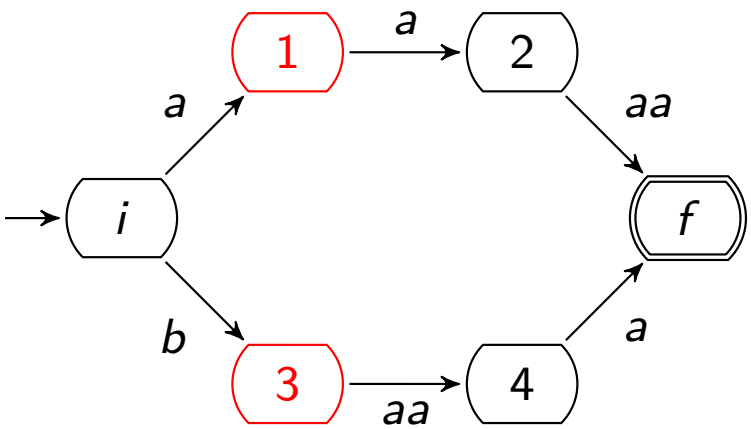


Figure: 1 and 3 are equivalent

Theorem

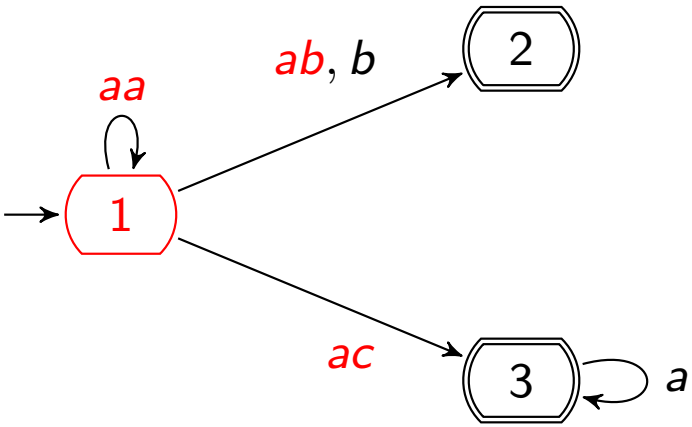
A language is k-block deterministic if and only if it is recognized by a k-block deterministic automaton which is compact and passes the BW-test.

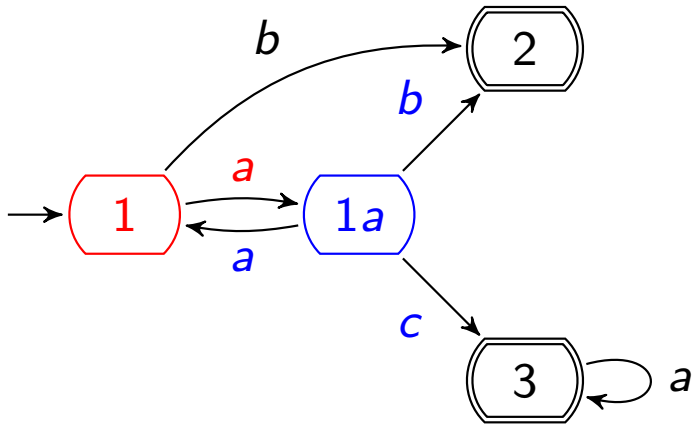
How to generate every compact deterministic block automata ?

Properties of the right languages of a deterministic block automaton

Let A be a deterministic block automaton and M be its equivalent minimal DFA:

- for every state of A, there exists an equivalent state of M.



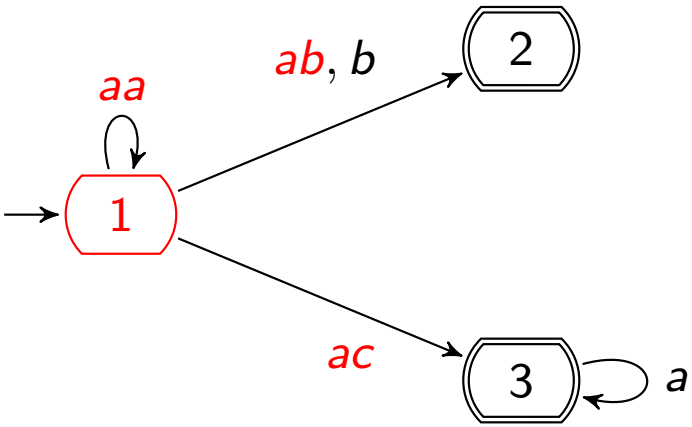


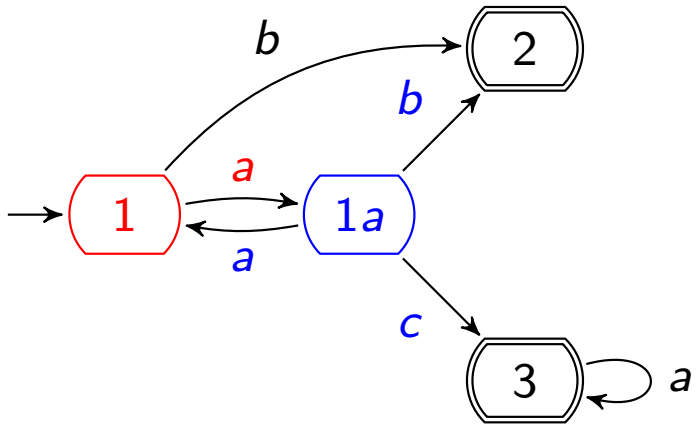
How to generate every compact deterministic block automata ?

Properties of the right languages of a deterministic block automaton

Let A be a deterministic block automaton and M be its equivalent minimal DFA:

- for every state of A, there exists an equivalent state of M.
- for every final state of M, there exists an equivalent state of A.



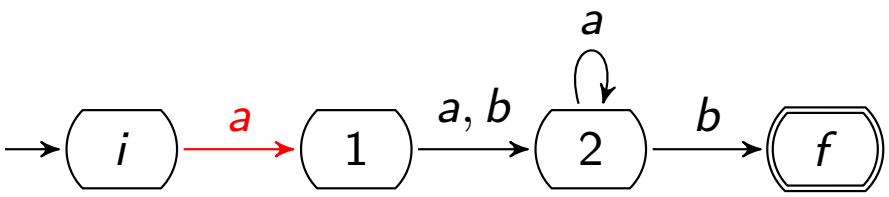


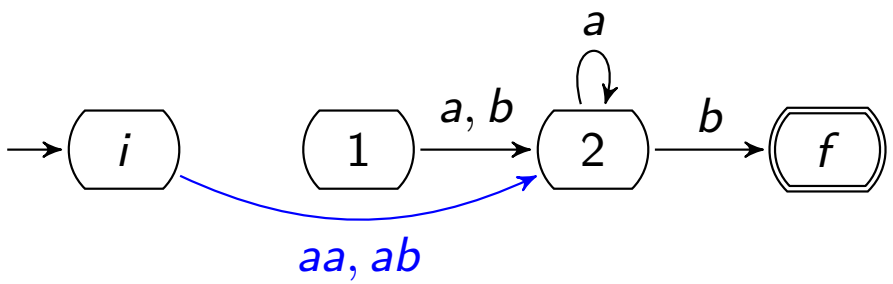
Generation of the compact deterministic block automata

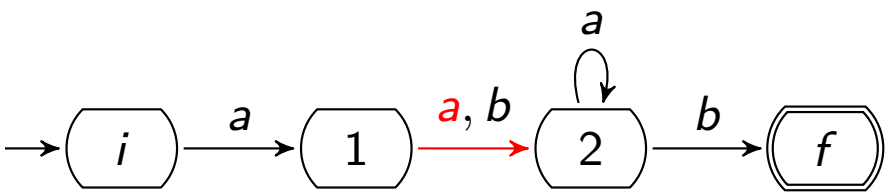
From the minimal DFA !

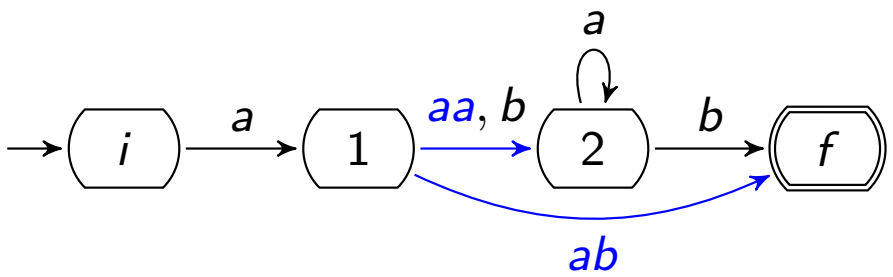
Extending transitions

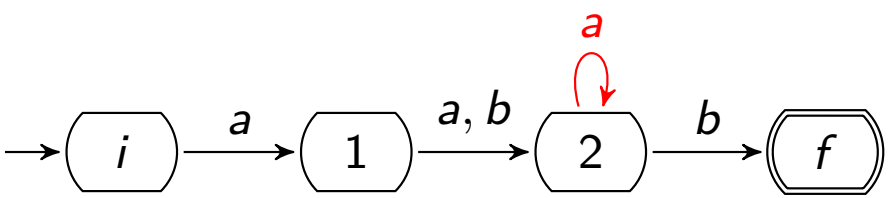
Lengthening a transition going out of a state while preserving the language recognized.

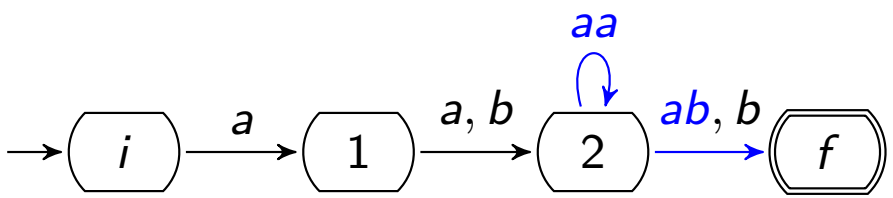




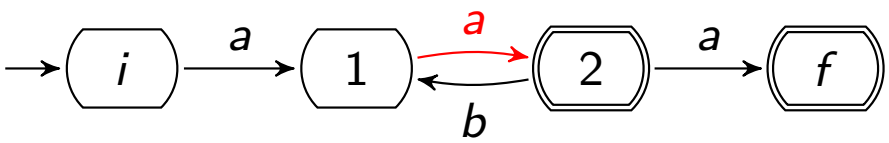


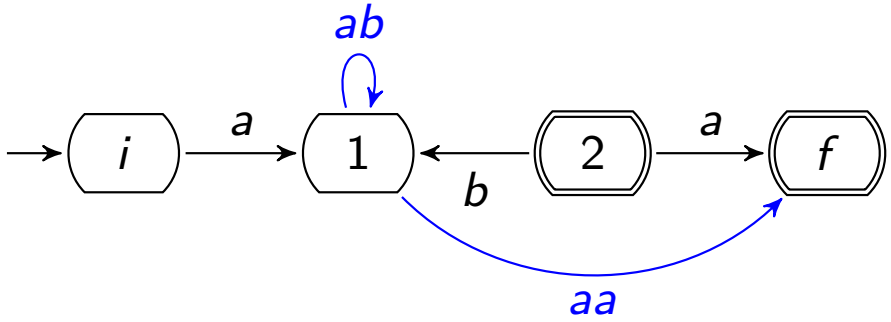


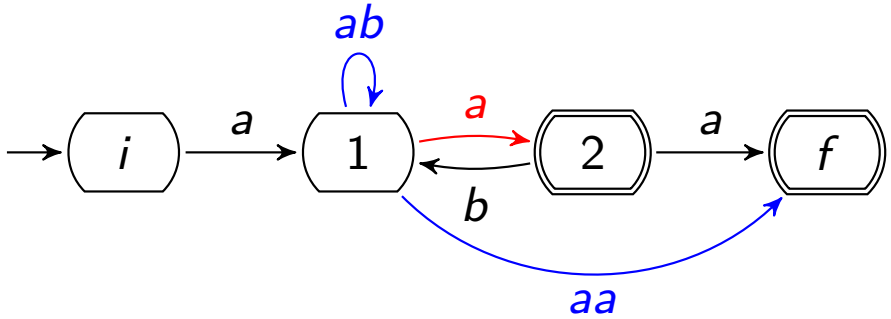




Constraint: preserving both the language and the determinism







Condition

The transition to lengthen must not go to a final state.

How to generate every possibility: the k-transitions automaton

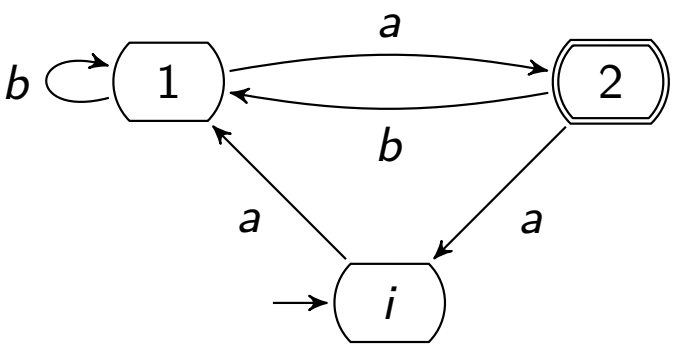


Figure: A minimal DFA M

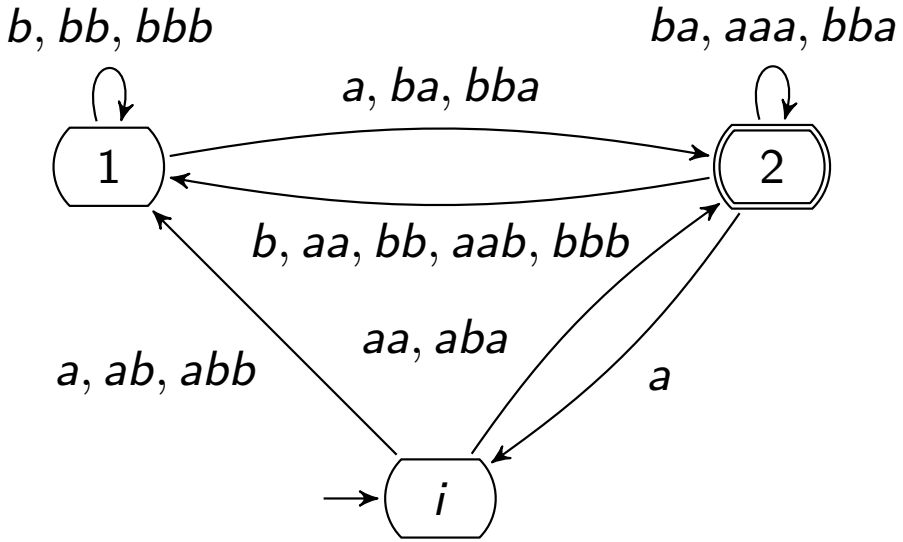


Figure: The 3-transitions automaton of $\mathrm{L}(M)$

How to generate every possibility: the k-transitions automaton

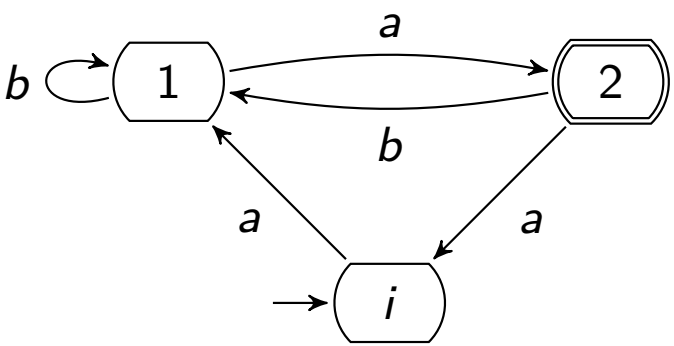


Figure: A minimal DFA M

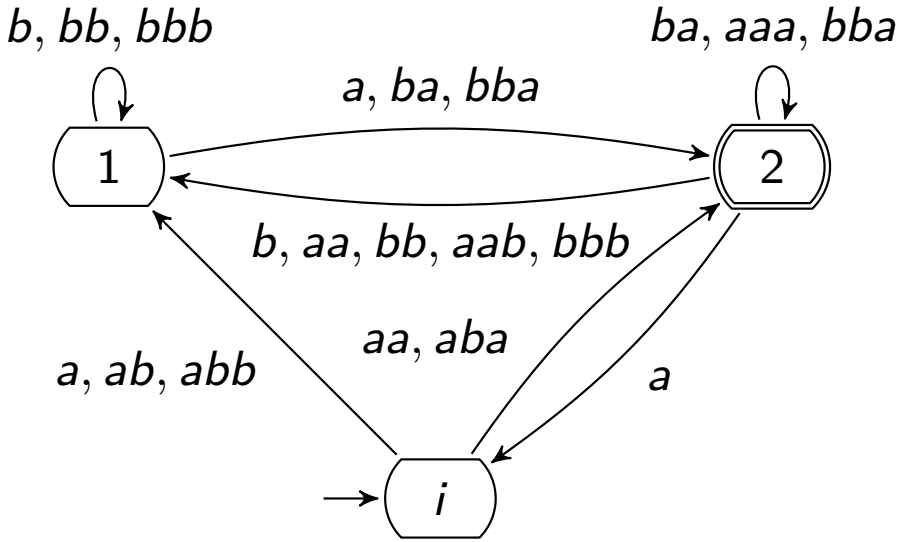
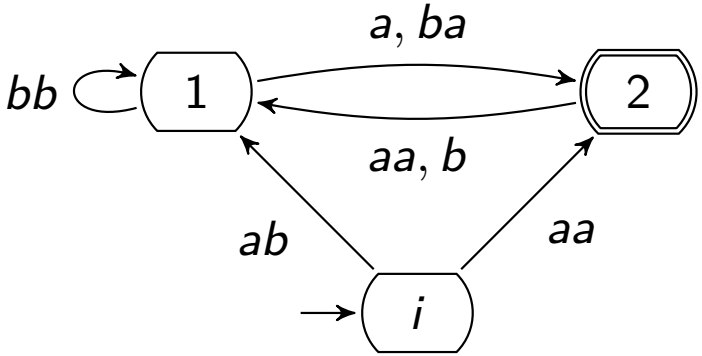


Figure: The 3-transitions automaton of $\mathrm{L}(M)$



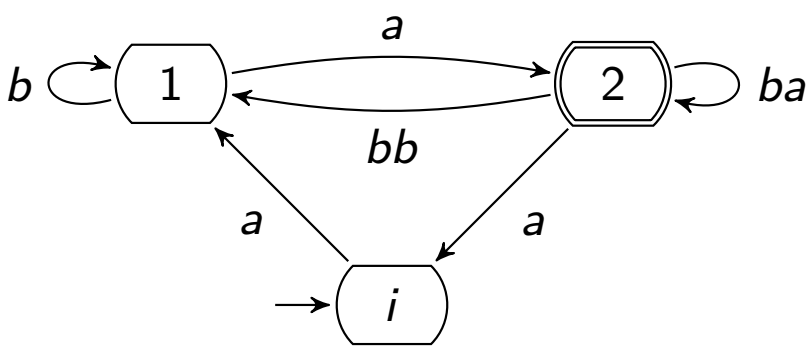


Figure: Two compact deterministic block automata recognizing $\mathrm{L}(M)$

Decidability

Theorem

The k-block determinism of a language is decidable.

Decidability

Theorem

The k-block determinism of a language is decidable.

Upper bound ?

The block determinism of a language is semi-decidable.

Internal hierarchy

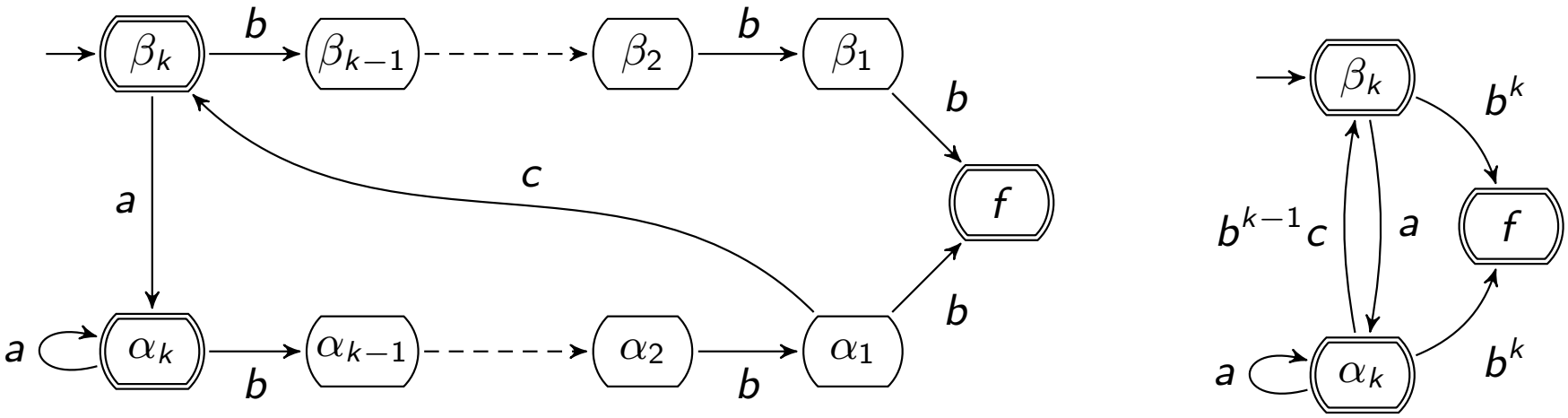


Figure: The family of languages denoted by $E_k = (a(\varepsilon + [b^{k-1}c]))^*(\varepsilon + [b^k])$

Internal hierarchy

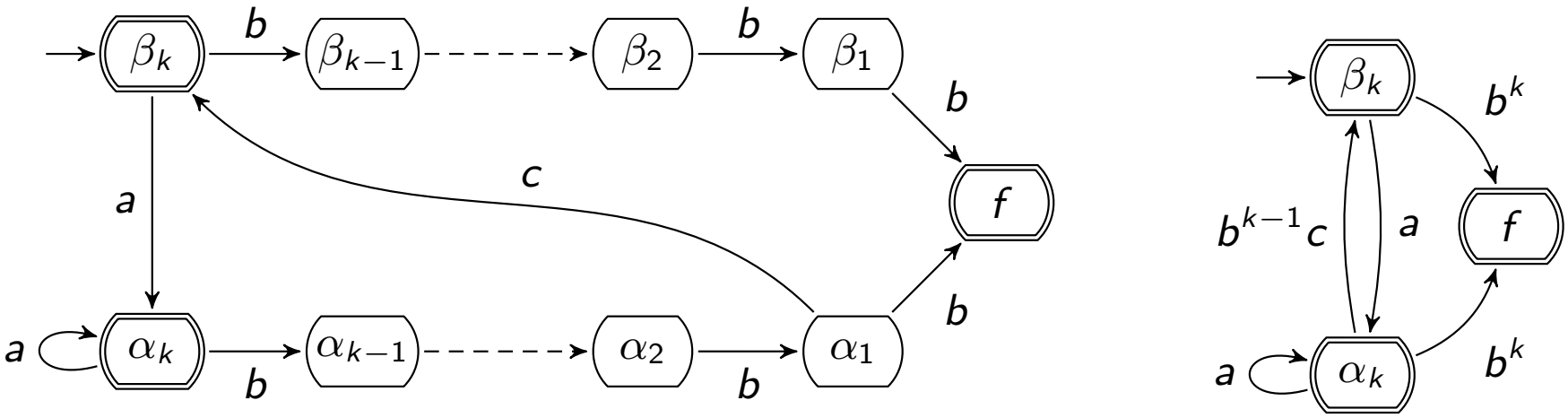


Figure: The family of languages denoted by $E_k = (a(\varepsilon + [b^{k-1}c]))^*(\varepsilon + [b^k])$

Why $\mathrm{L}(E_k)$ is not $(k-1)$-block deterministic:

- The sole non-trivial orbit has 3 gates: α_1, α_k and β_k.
- With blocks of length at most $k-1$, this orbit always has these 3 gates.

A proper subfamily of regular languages

Properties of unary block deterministic languages

If a unary language is block deterministic, then it is 1-unambiguous.

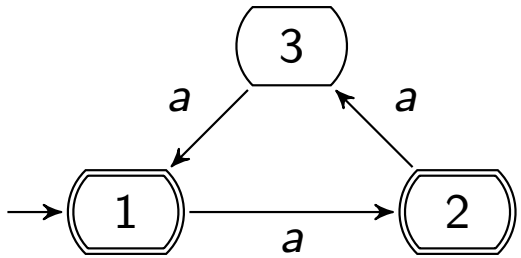


Figure: No synchronizing state

Theorem

There exist regular languages which are not block deterministic.

Thank you for your attention.

Do you have any question ?