Formal Languages and Linguistics

Pascal Amsili

Sorbonne Nouvelle, Lattice (CNRS/ENS-PSL/USN)

PSL Master's degree in Cognitive Science, February 2025

Overview

Back to “Natural” Languages

English as a formal language:

alphabet: morphemes (often simplified to words —depending on your view on flexional morphology)

⇒ Finite at a time t by hypothesis

words: well formed English sentences

⇒ English sentences are all finite by hypothesis

language: English, as a set of an infinite number of well formed combinations of “letters” from the alphabet

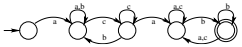
Good questions

Why would one consider natural language as a formal language?

- it allows to describe the language in a formal/compact/elegant way
- it allows to compare various languages (via classes of languages established by mathematicians)
- it give algorithmic tools to recognize and to analyse words of a language.

recognize u	: decide whether $u \in L$
analyse u	: show the internal structure of u

Final remarks

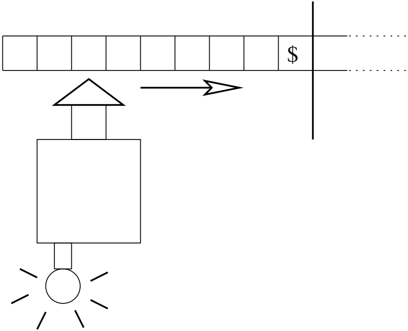
- We are only talking about syntax
- From now on, we'll mostly be looking for precise and efficient ways to **define** a language
  - L $= \{aa, ab, ba\}$
  - L $=$ { all the country names in English }
  - L $=$ { all the inflected forms of French *manger* }
  - L $= \{a^{2^k}$ with $k \geq 0\}$
  - L $= \{ww$ with $w \in \Sigma^*\}$
  - L $= (\{a\} \cup \{b\}.\{c\})^*$ — simplified notation $(a|bc)^*$
  - L $=$ the set of words recognized by this automaton: 
  - L $=$ the set of words engendered by this formal grammar

$$\mathcal{G}_3: \begin{array}{lcl} E & \longrightarrow & E+E \\ & | & E\times E \\ & | & (\,E\,) \\ & | & F \\ F & \longrightarrow & 0|1|2|3|4|5|6|7|8|9 \end{array}$$

Sorbonne Nouvelle

Overview

Metaphoric definition



Formal definition

Def. 9 (Finite deterministic automaton (FDA))

A finite state deterministic automaton $\mathcal{A}$ is defined by :

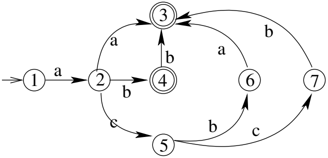
$$\mathcal{A} = \langle Q, \Sigma, q_0, F, \delta \rangle$$

- Q is a finite set of states
- Σ is an alphabet
- q_0 is a distinguished state, the initial state,
- F is a subset of Q, whose members are called final/terminal states
- δ is a mapping **fonction** from $Q \times \Sigma$ to Q. Notation $\delta(q, a) = r$.

Example

Let us consider the (finite) language $\{aa, ab, abb, acba, accb\}$.
The following automaton recognizes this langage: $\langle Q, \Sigma, q_0, F, \delta \rangle$, avec $Q = \{1,2,3,4,5,6,7\}$, $\Sigma = \{a, b, c\}$, $q_0 = 1$, $F = \{3,4\}$, and δ is thus defined:

δ :
(1,a) $\mapsto$ 2
(2,a) $\mapsto$ 3
(2,b) $\mapsto$ 4
(2,c) $\mapsto$ 5
(4,b) $\mapsto$ 3
(5,b) $\mapsto$ 6
(5,c) $\mapsto$ 7
(6,a) $\mapsto$ 3
(7,b) $\mapsto$ 3



	a	b	c
$\rightarrow$ 1	2		
2	3	4	5
$\leftarrow$ 3			
$\leftarrow$ 4		3	
5		6	7
6	3		
7		3	

Sorbonne Nouvelle

Recognition

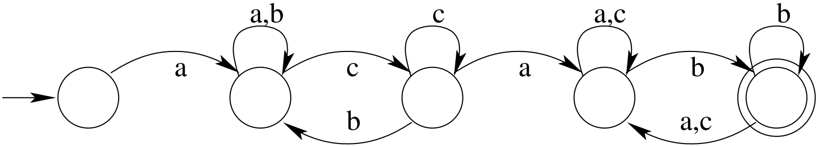
Recognition is defined as the existence of a sequence of states defined in the following way. Such a sequence is called a path in the automaton.

Def. 10 (Recognition)

A word $a_1 a_2 ... a_n$ is **recognized/accepted** by an automaton iff there exists a sequence $k_0, k_1, ..., k_n$ of states such that:

$$k_0 = q_0$$
$$k_n \in F$$
$$\forall i \in [1, n], \quad \delta(k_{i-1}, a_i) = k_i$$

Example

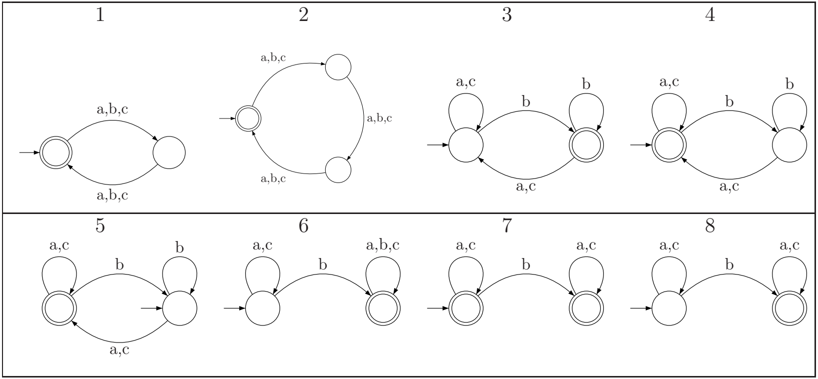


Exercices

Let $\Sigma = \{a, b, c\}$. Give deterministic finite state automata that accept the following languages:

1. The set of words with an even length.
2. The set of words where the number of occurrences of b is divisible by 3.
3. The set of words ending with a b.
4. The set of words not ending with a b.
5. The set of words non empty not ending with a b.
6. The set of words comprising at least a b.
7. The set of words comprising at most a b.
8. The set of words comprising exactly one b.

Answers


1
a,b,c
a,b,c
2
a,b,c
a,b,c
a,b,c
3
a,c
b
b
a,c
4
a,c
b
b
a,c
5
a,c
b
b
a,c
6
a,c
b
a,b,c
7
a,c
b
a,c
8
a,c
b
a,c

Overview

Ways of non-determinism

A word is recognized if there exists a path in the automaton. It is not excluded however that there be several paths for one word: in that case, the automaton is non deterministic.
What are the sources of non determinism?

- $\delta(a, S_1) = \{S_2, S_3\}$
- "spontaneous transition" = ε-transition

Equivalence theorems

For any non-deterministic automaton, it is possible to design a complete deterministic automaton that recognizes the same language.
Proofs: algorithms (constructive proofs)
First “remove” ε-transitions, then “remove” multiple transitions.

Closure (1)

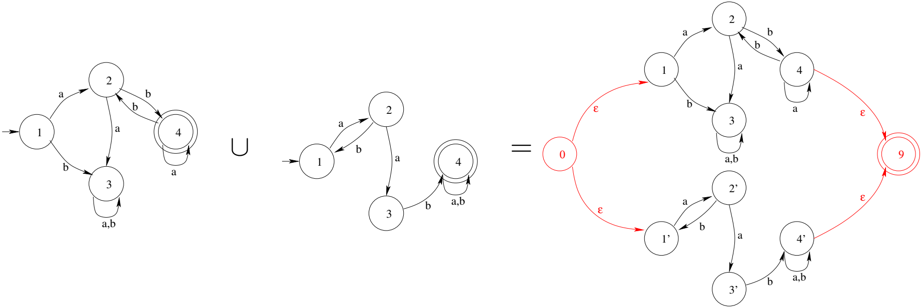
Regular languages are closed under various operations: if the languages L and L' are regular, so are:

- $L \cup L'$ (union); $L.L'$ (product); L^* (Kleene star)

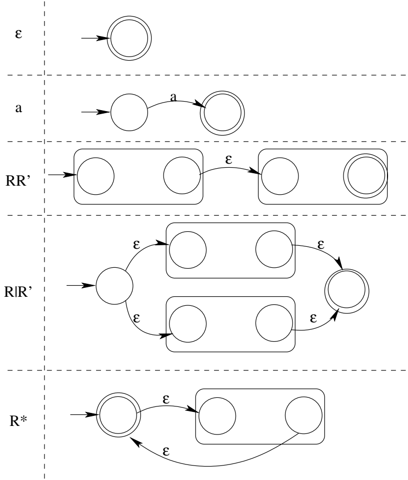
(rational operations)

Sorbonne Nouvelle

Union of regular languages: an example



Rational operations



Closure (2)

Regular languages are closed under various operations: if the languages L and L' are regular, so are:

- $L \cup L'$ (union); $L.L'$ (product); L^* (Kleene star)
  (rational operations)
  $\rightarrow$ for every rational expression describing a language , there is a FSA that recognizes L

Closure (2)

Regular languages are closed under various operations: if the languages L and L' are regular, so are:

- $L \cup L'$ (union); $L.L'$ (product); L^* (Kleene star) *(rational operations)*
  $\rightarrow$ for every rational expression describing a language , there is a FSA that recognizes L and vice-versa

Closure (2)

Regular languages are closed under various operations: if the languages L and L' are regular, so are:

- $L \cup L'$ (union); $L.L'$ (product); L^* (Kleene star)
  (rational operations)
  $\rightarrow$ for every rational expression describing a language , there is a FSA that recognizes L and vice-versa
- $L \cap L'$ (intersection); $\overline{L}$ (complement)
- ...

Intersection of regular languages

Algorithmic proof
Deterministic complete automata

L_1	a	b
$\rightarrow$ 1	2	4
2	4	3
$\leftarrow$ 3	3	3
4	4	4

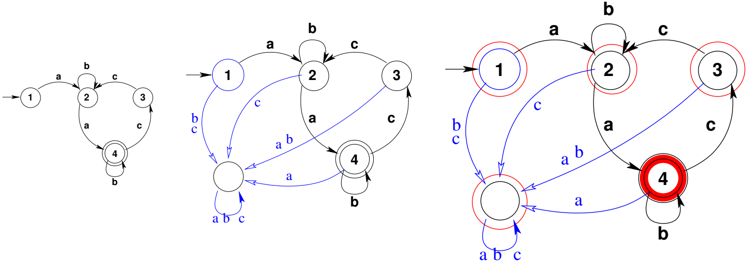
L_2	a	b
$\leftrightarrow$ 1	2	5
2	5	3
3	4	5
4	1	4
5	5	5

$L_1 \cap L_2$	a	b
$\rightarrow$ (1,1)	(2,2)	(4,5)
(2,2)	(4,5)	(3,3)
(4,5)	(4,5)	(4,5)
(3,3)	(3,4)	(3,5)
(3,4)	(3,1)	(3,4)
$\leftarrow$ (3,1)	(3,2)	(3,4)
(3,2)	(3,4)	(3,3)
(3,5)	(3,5)	(3,5)

Complement of a regular language

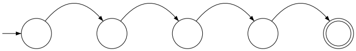
Deterministic complete automata

completed complemented



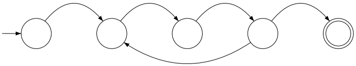
Pumping lemma (intuition)

Take an automaton A with k states.
If $\mathcal{L}(A)$ is infinite,
then $\exists w \in \mathcal{L}(A), |w| \geq k$.
Therefore, when accepting w, A goes through some state q at least twice.
That means that there is a loop $q \overset{w_{i:j}}{\rightarrow} q$.
Repeating the loop any number of times (even 0) always produces a word $(w_{1:i-1}\, {w_{i:j}}^n\, w_{j+1:|w|})$ in $\mathcal{L}(A)$.



Pumping lemma (intuition)

Take an automaton A with k states.
If $\mathcal{L}(A)$ is infinite,
then $\exists w \in \mathcal{L}(A), |w| \geq k$.
Therefore, when accepting w, A goes through some state q at least twice.
That means that there is a loop $q \overset{w_{i:j}}{\rightarrow} q$.
Repeating the loop any number of times (even 0) always produces a word $(w_{1:i-1}\, {w_{i:j}}^n\, w_{j+1:|w|})$ in $\mathcal{L}(A)$.



Pumping lemma (definition)

Pumping Lemma

Let L be a regular language.
$\exists k \in \mathbb{N}$ such that
$\forall w \in L$ such that $|w| \geq k$,
$\exists x, u, y$ such that $w = xuy$ and that

1. $|u| \geq 1$;
2. $|xu| \leq k$;
3. $\forall n \in \mathbb{N}$, $xu^n y \in L$.

$\rightarrow$ "L has the pumping property."

Is NL regular? Pumping lemma (example I)

a^*bc (i.e. $\{a^n bc \mid n \in \mathbb{N}\}$) is regular (there is a DFA).
So, it must have the pumping property.

It happens that $k = 3$ works.
For example, $w = abc \in L$ is long enough and can be decomposed:

$$\underbrace{\epsilon}_{x} \quad \underbrace{a}_{u} \quad \underbrace{b \quad c}_{y}$$

1. $|u| \geq 1$ ($u = a$);
2. $|xu| \leq k$ ($xu = a$);
3. $\forall n \in \mathbb{N}$, $xu^n y$ (i.e. $a^n bc$) belongs to the language.

Pumping lemma (consequences)

regular	$\Rightarrow$	pumping property satisfied
pumping property **NOT** satisfied	$\Rightarrow$	**NOT** regular
pumping property satisfied	$\not\Rightarrow$	regular

To prove that L is

- regular: provide a DFA;
- not regular: show that the pumping property is not satisfied.

Pumping lemma (example II)

Let's show that $L = \{a^n b^n \mid n \in \mathbb{N}\}$ is not regular.

- Consider any $k \in \mathbb{N}$.
- Consider $w = a^k b^k \in L$ ($|w| \geq k$).
- If $w = xuy$ with $|u| \geq 1$ and $|xu| \leq k$, then u contains no b.
- But then, $xu^0y = xy \notin L$ (strictly less as than bs).
- So no $k \in \mathbb{N}$ works; L does not have the pumping property.

A similar reasoning applies to $\{xu^n yv^n z \mid x, y, z, u, v \in \Sigma^*\}$.

Sorbonne Nouvelle

Overview

Sorbonne Nouvelle

Regular expressions

It is common to use the 3 *rational* operations:

- union
- product
- Kleene star

to characterize certain languages...

Regular expressions

It is common to use the 3 *rational* operations:

- union
- product
- Kleene star

to characterize certain languages...

$(\{a\} \cup \{b\})^*.\{c\} = \{c, ac, abc, bc, \ldots, baabaac, \ldots\}$
(simplified notation $(a|b)^*c$ — regular expressions)

Sorbonne Nouvelle

Regular expressions

It is common to use the 3 *rational* operations:

- union
- product
- Kleene star

to characterize certain languages...

$$(\{a\} \cup \{b\})^*.\{c\} = \{c, ac, abc, bc, \ldots, baabaac, \ldots\}$$

(simplified notation $(a|b)^*c$ — regular expressions)

... but not all languages can be thus characterized.

Def. 11 (Rational Language)

A rational language on Σ is a subset of Σ^* inductively defined thus:

- $\emptyset$ and $\{\varepsilon\}$ are rational languages ;
- for all $a \in X$, the singleton $\{a\}$ is a rational language ;
- for all g and h rational, the sets $g \cup h$, $g.h$ and g^* are rational languages.

Results: expressivity

- Any finite langage is regular
- $a^n b^m$ is regular
- $a^n b^n$ is not regular
- ww^R is not regular (^{R} : reverse word)

Decidable problems

- The "word problem" $w \stackrel{?}{\in} L(\mathcal{A})$ is decidable.

⇒ A computation on an automaton always stops.

Decidable problems

- The "word problem" $w \overset{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \overset{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where k is the number of states.

Decidable problems

- The "word problem" $w \overset{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \overset{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where k is the number of states.

- The "finiteness problem" $L(\mathcal{A})$ $\overset{?}{\text{is}}$ *finite* is decidable.

$\Rightarrow$ Test all possible words whose length is between k and $2k$. If there exists u s.t. $k < |u| < 2k$ and $u \in L(\mathcal{A})$, then $L(\mathcal{A})$ is infinite.

Decidable problems

- The "word problem" $w \stackrel{?}{\in} L(\mathcal{A})$ is decidable.

$\Rightarrow$ A computation on an automaton always stops.

- The "emptiness problem" $L(\mathcal{A}) \stackrel{?}{=} \emptyset$ is decidable.

$\Rightarrow$ It's enough to test all possible words of length $\leq k$, where k is the number of states.

- The "finiteness problem" $L(\mathcal{A}) \stackrel{?}{\text{is}}$ *finite* is decidable.

$\Rightarrow$ Test all possible words whose length is between k and $2k$. If there exists u s.t. $k < |u| < 2k$ and $u \in L(\mathcal{A})$, then $L(\mathcal{A})$ is infinite.

- The "equivalence problem" $L(\mathcal{A}) \stackrel{?}{=} L(\mathcal{A}')$ is decidable.

$\Rightarrow$ it boils down to answering the question:

$$\left(L(\mathcal{A}) \cap \overline{L(\mathcal{A}')}\right) \cup \left(L(\mathcal{A}') \cap \overline{L(\mathcal{A})}\right) = \emptyset$$

À quoi ça sert?

Why would you want to define (formally) a language?

- to formulate a request to a search engine (`mang.*`)
- to associate actions to (classes of) words (e.g., transducers)
  - formal languages (math. expressions, programming languages...)
  - artificial (interface) languages
  - (subpart of) natural languages

Overview

Definition

1. a regular language can be defined by rational/regular expressions
2. a regular language can be recognized by a finite automaton
3. a regular language can be generated by a regular grammar