UNIVERSITÉ Grenoble Alpes

INF 332: Languages and Automata

Tutorials

Univ. Grenoble Alpes
Licence Sciences et Technologies
Département Licence Sciences et Technologies

`www.univ-grenoble-alpes.fr`
`dlst.univ-grenoble-alpes.fr`

Academic year 2020 - 2021

Contents

CHAPTER

1 Basic Mathematical Notions

This chapter is purposed to let you practice the basic mathematical notions that will be used during the course. The exercises in this chapter will not be corrected during the tutorial sessions unless you prepare the exercises and you ask your professor to do so.

Exercise 1 ()

Consider a finite set E and $\mathcal{P}(E)$, that is the powerset of E or the set of subsets of E.

1. Recall the definition of $\mathcal{P}(E)$.
2. For $E = \{1, 2, 3\}$, give $\mathcal{P}(E)$.

Exercise 2 ()

Prove the following propositions:

1. $\mathcal{P}(A) = \mathcal{P}(B)$ iff $A = B$.
2. $\mathcal{P}(A \cup B) = \{X \cup Y \mid X \in \mathcal{P}(A) \wedge Y \in \mathcal{P}(B)\}$.
3. $\mathcal{P}(A \cap B) = \mathcal{P}(A) \cap \mathcal{P}(B)$.
4. In general, $\mathcal{P}(A \cup B) = \mathcal{P}(A) \cup \mathcal{P}(B)$ does not hold.

Exercise 3 ()

1. Recall the formal mathematical definitions of the following elements: relation, function, application, reflexive relation, anti-reflexive relation, symmetric relation, anti-symmetric relation, transitive relation, equivalence relation, equivalence class.
2. Give an example for each of the elements mentioned in the previous question.

Exercise 4 ()

Consider the relation $R \subseteq \mathbb{Z} \times (\mathbb{Z} \setminus \{0\})$ defined as follows:

$$\forall (a, b), (c, d) \in \mathbb{Z} \times (\mathbb{Z} \setminus \{0\}) : (a, b)R(c, d) \Leftrightarrow ad - bc = 0$$

1. Prove that R is an equivalence relation.
2. Give the equivalence classes.

Exercise 5 ()

1. Prove the following proposition:

$$\forall n \in \mathbb{N} : \sum_{i=0}^{n} i = \frac{n(n+1)}{2}$$

2. Deduce that $1 + 3 + 5 + ... + (2n - 1) = n^2$.
3. Prove the following proposition:

$$\forall n \in \mathbb{N} : \sum_{i=0}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}$$

Exercise 6 ()

Let E be the set inductively defined by the following rules:

- **Base rule** : $0 \in E$
- **Induction rule** : if $x \in E$, then $s(x) \in E$

1. Propose a definition of a function $+$ that behaves as the addition on integers (where $\mathrm{s}(x)$ is the integer after x). The function should work by induction on its first argument.
2. Propose a definition of a function $*$ that behaves as the multiplication on integers. The function should work by induction on its first argument.
3. Prove the following properties:

   - $\forall x \in E : x * 0 = 0 = 0 * x$,
   - $\forall x, y \in E : x * y = y * x$,
   - $\forall x, y, z : (x * y) * z = x * (y * z)$,
   - $\forall x, y, z : (x + y) * z = x * z + y * z$.

Exercise 7 ()

Let $E(\Sigma)$ be the set of lists with elements in Σ inductively defined as follows:

- **Base rule** : $\mathrm{nil} \in E(\Sigma)$
- **Induction rule** : if $l \in E(\Sigma)$, $cons(a, l) \in E(\Sigma)$, for any $a \in \Sigma$

Let $\Sigma = \{a, b\}$

1. Give an inductive definition of the set of lists that contain the same number of a's and b's.
2. Give an inductive definition of the set of lists that contain the same number of a's and b's and that start with a's followed by b's. Between the a's, there should be no b's.

CHAPTER

2 Preliminary Notions

Exercise 8 (♠) — Word concatenation

Consider the examples used to illustrate concatenation, as seen in the course:

- Concatenating words 01 and 10 results in word 0110.
- Concatenating the empty word ϵ and word 101 results in word 101.

1. Give the formal definitions of the applications corresponding to these 5 words and prove that the two above claims are consistent with the definition of the concatenation operator as seen in the course.

Exercise 9 (♠♠) — Length of a word and number of occurrences of a symbol

Consider an alphabet Σ.

1. Using the non-inductive definition of words as an application, define the function giving the length of a word.
2. Using the non-inductive definition of words as an application, define the function giving the number of occurrences of a symbol in a word.
3. Same questions using the inductive definition of words.

Exercise 10 (♠♠) — Neutral element of concatenation

Consider Σ an alphabet and ϵ_Σ the empty word over Σ.

1. Considering the definition of words, prove that word ϵ_Σ is the neutral element of concatenation, that is $\forall u \in \Sigma^* : u \cdot \epsilon_\Sigma = \epsilon_\Sigma \cdot u = u$.

Exercise 11 (♠♠♠) — Power of an alphabet

Let us recall that for an alphabet Σ, Σ^k and $\Sigma^{\leq k}$ are the sets of words over Σ respectively of length equal to k and of length lesser than or equal to k.

1. Give the cardinal of Σ^k.
2. Prove that $\forall k \in \mathbb{N} : \Sigma^{k+1} = \Sigma^k \cdot \Sigma^1$.
3. Prove that $\forall k \in \mathbb{N} : \Sigma^{k+1} = \Sigma^1 \cdot \Sigma^k$.
4. Prove the result about the cardinality of Σ^k.

Exercise 12 (♠♠♠) — Cardinal and concatenation

Consider an alphabet Σ. We are interested in the concatenation $L_1 \cdot L_2$ of two languages L_1 and L_2 defined over Σ.

1. Give two languages L_1 and L_2, with $L_1 \neq \emptyset$ and $L_2 \neq \emptyset$, such that $|L_1 \cdot L_2| < |L_1| \times |L_2|$.
2. Prove that $\forall L_1, L_2 \subseteq \Sigma^* : |L_1 \cdot L_2| \leq |L_1| \times |L_2|$.
3. Give sufficient conditions ensuring that $|L_1 \cdot L_2| = |L_1| \times |L_2|$.

CHAPTER

3 Deterministic Finite-state Automata (DFAs)

Reminders:

- DFA: Deterministic Finite-state Automaton.
- An automaton is said to be reachable (resp. coreachable) if all its states are reachable (resp. coreachable).
- An automaton is said to be trimmed if it is reachable and coreachable.

Exercise 13 (♠) — Language recognized by an automaton

Consider the alphabet $\{a, b\}$ and the DFAs in Figure 3.1.

1. Describe in natural language, the languages recognized by this DFAs.
2. How could you adapt the descriptions found in the previous question if the alphabet is now $\{a, b, c\}$?

Exercise 14 (♠) — Tabular representation of an automaton

Consider the alphabet $\{a, b\}$. You can consider only 2-3 examples in this exercise, once you have understood the principle.

1. Give the tabular representation of the DFAs in Figure 3.1.
2. Give the tabular representation of the DFAs found in Exercise 15.

Exercise 15 (♠) — Automaton for a language with constraints over the number of symbols

Consider the alphabet $\{a, b\}$. For each of the following set, give a DFA that recognizes it, if it is possible.

1. The set of words that contain an even number of a.
2. The set of words that contain a number of a multiple of 3.
3. The set of words that contain exactly n occurrences of symbol a, for some $n \in \mathbb{N}$.
4. The set of words that contain less than n occurrences of symbol a, for some $n \in \mathbb{N}$.
5. The set of words that contain more than n occurrences of symbol a, for some $n \in \mathbb{N}$.
6. The set of words that contain as many a as b.
7. The set of words such that each block of 3 consecutive symbol contains (exactly) 2 occurrences of symbols a.

Exercise 16 (♠♠) — Automata for languages with some prefix, suffix, or factor

Consider the alphabet $\{a, b, c\}$. For each of the following set, give a DFA that recognizes it, if it is possible.

1. The set of words that start with $a \cdot b$ or $b \cdot c$.
2. The set of words that start neither with $a \cdot b$ nor $b \cdot c$.
3. The set of words that contain $a \cdot b$.
4. The set of words that do not contain $a \cdot b$.
5. The set of words that end with $a \cdot b \cdot c$.
6. The set of words that do not end with $a \cdot b \cdot c$.
7. The set of words of length greater than or equal to 2 such that the next-to-last symbol is b.

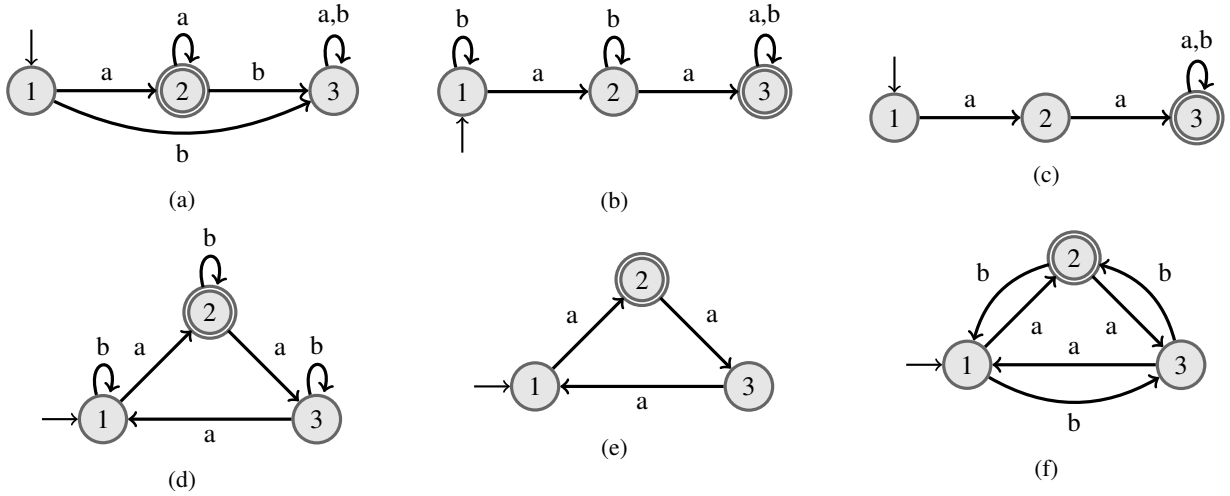


Figure 3.1: Some deterministic finite-state automata

Exercise 17 (♠♠♠) — Transition function and sink symbol

Consider the DFA $(Q, q_0, \Sigma, \delta, F)$ and let $a \in \Sigma$ be a particular symbol such that $\forall q \in Q : \delta(q, a) = q$.

1. Prove that: $\forall n \in \mathbb{N} : \delta^*(q, a^n) = q$ where a^n is the word formed by concatenating n a's (that is $\underbrace{a \cdot a \dots a}_{n \text{ fois}}$).
2. Prove that either $\{a\}^* \subseteq L(A)$ or $\{a\}^* \cap L(A) = \emptyset$.

Exercise 18 (♠♠♠) — Under and over automaton

Let A be an automaton. We call under-automaton of A any automaton obtained by removing one or several states from the set of states of A, except the initial state, or any transition from the transition function of A. We call over-automaton of A any automaton obtained by adding one or several states to the set of states of A, or a transition to the transition function of A. In the latter case, we assume that adding transitions is done while preserving the determinism of the automaton. An over-automaton of A has the same initial automaton as A and its set of accepting states is a super set of the set of accepting states of A.

1. Consider the DFA $A = (Q, q_0, \Sigma, \delta, F)$, formally define the condition at which $A' = (Q', q'_0, \Sigma, \delta', F')$ is an under-automaton of A.
2. Prove that the language recognized by any under-automaton of A is a subset of the language recognized by A.
3. Prove that the language recognized by any over-automaton of A is a superset of the language recognized by A.

CHAPTER

4 Composition Operators for DFAs

In this chapter, you can reuse the automata from the previous chapter. Reminder: the language accepted by a DFA A is denoted by $L(A)$.

Exercise 19 (♠) — Completeness of an automaton

Consider an alphabet $\{a, b\}$ and the DFAs in Figure 3.1.

1. Determine which automata are complete.
2. Complete the automata that are not complete.
3. Complete the automata by considering alphabet $\{a, b, c\}$.

Exercise 20 (♠) — Find the complement automaton

Consider an alphabet $\Sigma = \{a, b\}$. Give an automaton that recognizes the complement of the languages recognized by the following automata.

1. The automaton depicted in Figure 4.1a.
2. The automaton depicted in Figure 4.1b.
3. The automaton depicted in Figure 4.1c.

t

Exercise 21 (♠♠) — Obtain an automaton by computing the product of two automata

Consider an alphabet $\Sigma = \{a, b\}$ and the automata found in Exercise 15.

1. Give a DFA that recognizes the set of words that contain a k occurrences of symbol a, such that k is a multiple of 3 and a multiple of 2.
2. Give a DFA that recognizes the set of words that contain a k occurrences of symbol a, such that k is a multiple of 2 and not a multiple of 3.
3. For each of the preceding languages, give a DFA that recognizes the complement of the recognized language in Σ^*.

Exercise 22 (♠♠) — Obtain an automaton by applying the product operator

Consider an alphabet $\Sigma = \{a, b, c\}$. For each of the following language, by reusing the automata in Exercise 16, give an automaton that recognizes it.

1. The set of words that start by $a \cdot b$ or $b \cdot c$ and that do not end by $a \cdot b \cdot c$.

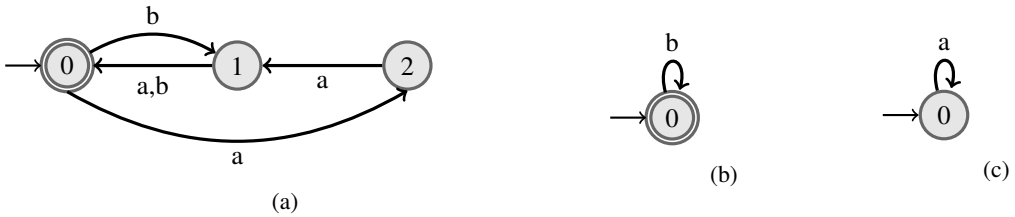


Figure 4.1: Some deterministic finite-state automata

2. The set of words that contain an even number of c and that do not contain $a \cdot b$.

Exercise 23 (♠♠♠) — Correctness of the completion operator

We want to prove that the operation/algorithm used to complete automata is correct, that is it does not change the language of the automaton to which it is applied. Given a DFA A, $C(A)$ denotes the automaton resulting from the application of the completion operator to A.

1. Prove that $L(C(A)) = L(A)$.

Exercise 24 (♠♠♠) — Correctness of the complementation operator

We want to prove that the operation/algorithm for complementing automata is correct, that is, it does produce an automaton that recognizes the complement of the automaton to which it was applied (if the starting automaton is commplete). We consider a complete DFA A over an alphabet Σ. We note A^C the automaton obtained by applying the complementation operator on A.

1. Prove that $L(A^C) = \Sigma^* L(A)$.

Exercise 25 (♠♠♠) — Correctness of the product operator

Let $A = (Q^A, \Sigma, q_0^A, \delta^A, F^A)$ and $B = (Q^B, \Sigma, q_0^B, \delta^B, F^B)$ be two DFAs. The objective of this exercise is to prove that $L(A) \cap L(B) \subseteq L(A \times B)$.

1. Prove that for any $n \in \mathbb{N}$, for any execution $(q_0^A, u_0) \cdots (q_n^A, u_n)$ of A and $(q_0^B, u_0) \cdots (q_n^B, u_n)$ of B on a common word u of length greater than or equal to n:

   $((q_0^A, q_0^B), u_0) \cdots ((q_n^A, q_n^B), u_n)$ is an execution of $A \times B$.

2. Use the previous result to prove $L(A) \cap L(B) \subseteq L(A \times B)$.

CHAPTER

5 Algorithms and Decision Problems for DFAs

Exercise 26 (♠) — Algorithms to determine completeness

Consider two alphabets Σ and Σ' such that $\Sigma' \subseteq \Sigma$. Recall that an automaton over an alphabet Σ is said to be complete if its transition function is defined for every symbol in Σ in all states.

1. Give an algorithm that determines whether an automaton over an alphabet Σ is complete.
2. We say that an automaton over the alphabet Σ is complete with respect to alphabet Σ' if its transition function is defined in every state and over every symbol in Σ'. Give an algorithm that determines whether an automaton over an alphabet Σ is complete over an alphabet Σ'.

Exercise 27 (♠) — Algorithms to complement

Consider an alphabet Σ.

1. From the definition of the complement automaton seen in the course, give an algorithm realizing the negation of an automaton with respect to Σ and producing an automaton recognizing the complement of the language recognized by the automaton passed as a parameter.

Exercise 28 (♠♠) — Algorithms to compute the product automaton

Consider an alphabet Σ.

1. From the definition of the product of automata seen in the course, give an algorithm producing an automaton recognizing the intersection of the languages of the automata passed as parameters. Constructing the state space of the automaton should be done “on the fly", that is by adding new state in a lazy fashion by simultaneously searching the state spaces of the parameter automata.
2. Let us recall that, according to the definition of the product automaton seen in the course, if the sets of states of the automata passed as parameters are Q_1 and Q_2, then the set of states of the product automaton is $Q_1 \times Q_2$ (with $|Q_1 \times Q_2| = |Q_1| \times |Q_2|$). Give examples of automata such that your algorithm produces an automaton such that the cardinal of its set of states is strictly less than $|Q_1 \times Q_2|$.

Exercise 29 (♠♠♠) — Prove the inclusion between languages

Consider two finite-state languages L and L' and the automata A_L and $A_{L'}$ recognizing L and L', respectively.

1. Using the intersection and complementation operators between languages, give a relation between languages equivalent to $L \subseteq L'$.
2. Deduce from the previous question an algorithm allowing to determine whether $L \subseteq L'$.
3. Deduce from the previous question an algorithm allowing to determine whether $L = L'$.
4. Let L_1 be the language of words that do not contain $abaa$ and that contain an even number of a. Give a complete DFA that recognizes L_1. Let L_2 be the language of words that do not contain aba and that contain a number of a multiple of 4. Give a DFA that recognizes L_2. Prove that $L_2 \subseteq L_1$ using the previous results from Exercise 29.

Exercise 30 (♠♠♠) — Determine whether an automaton recognizes a prefix-closed language

We want to show that the problem of determining whether the language recognized by an automaton is prefix-closed is decidable.

1. Give examples and counter-examples of automata that recognize prefix-closed languages.
2. Characterize with a condition the automata that recognize prefix-closed languages.
3. Give an algorithm that permits to decide whether the language recognized by a DFA is prefix closed.
4. Test your algorithm on the automata in the first question.

CHAPTER

6 Equivalence, distinguishability, and minimisation

This chapter contains some exercises on minimisation and equivalence between DFAs. The two following chapters will come back on the notion of minimisation.

Exercise 31 (♠♠) — Minimize automata

We consider the automata in Figure 6.2 over alphabet $\Sigma = \{a, b\}$.

1. Minimise the DFA in Figure 6.2a.
2. Minimise the DFA in Figure 6.2b.

Exercise 32 (♠♠) — Determine the equivalence or the non-equivalence between automata

Consider the two automata in Figure 6.1a and Figure 6.1b. prove that these two automata are equivalent:

1. by using the procedure based on distinguishability between states;
2. by using the procedure based on minimisation;
3. by using the procedure based on the product of automata.

Exercise 33 (♠♠) — Determine whether an automaton is minimal or not

We consider the two automata in Figure 6.3. We want to determine whether these automata are minimal. For each of these automata (Figure 6.3a and Figure 6.3b), do the following questions.

1. Use the algorithm computing the distinguishable states to determine minimality.
2. Use the algorithm computing the equivalent states to determine minimality. Show clearly the computation steps. List the equivalent classes.
3. Verify that you obtain compatible results and thus the same conclusions with the two methods.
4. Represent the minimal automaton in tabular or graphic format.

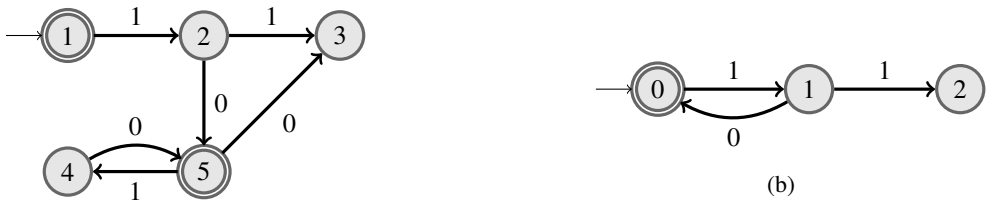


Figure 6.1: DFAs for which we want to determine equivalence.

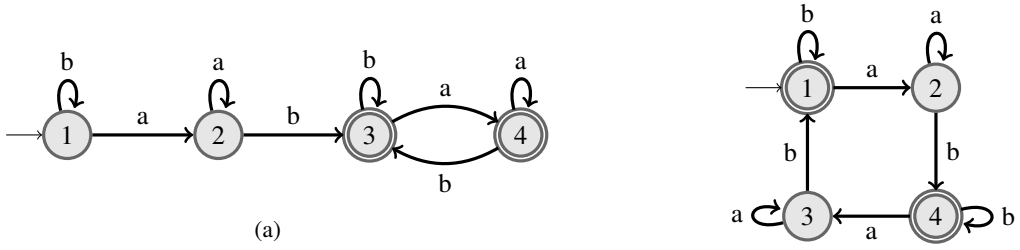


Figure 6.2: DFAs in graphical representation to minimise.

Figure 6.3: DFAs in graphical representation to minimise.

Table 6.1: DFAs in tabular representation to minimize. States are in line, symbols in columns. The arrow indicates the initial state, stars indicate accepting states.

		a	b
→	0	0	1
	1	2	3
	2	2	3
*	3	2	4
	4	0	1

(a)

		a	b
→	1	3	8
*	2	3	1
	3	8	2
*	4	5	6
	5	6	2
	6	7	8
	7	6	4
	8	5	8

(b)

		a	b
→	* 0	1	3
	1	2	3
*	2	5	2
	3	4	1
*	4	5	4
	5	5	5

(c)

CHAPTER

7 Non-deterministic Finite-state Automata

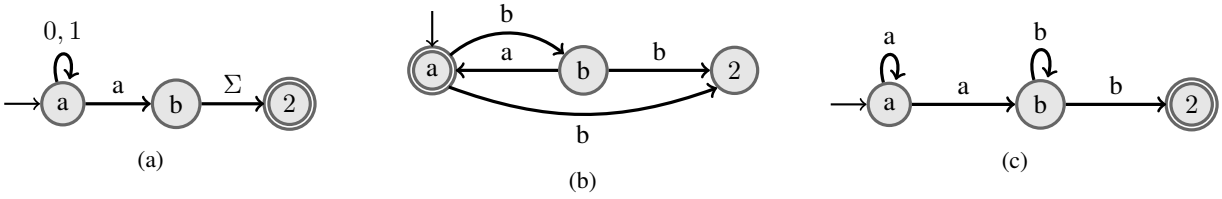


Figure 7.1: Non-deterministic finite-state automata

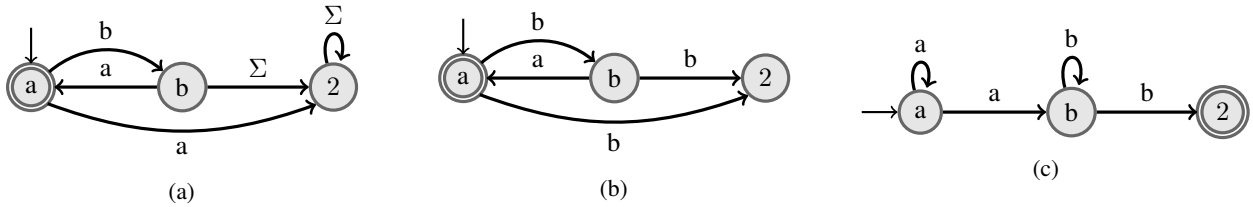


Figure 7.2: Non-deterministic finite-state automata

Exercise 34 (♠) — Describe the language recognized by an NFA

Describe in natural language the language recognized by the automaton depicted in Figure 7.1.

Exercise 35 (♠) — Give an NFA that recognizes a language

We define the language L_i as the set of words in Σ^* such that the i-th symbol from the right is symbol a. (The first symbol of a word starting from the end is the last symbol.)

1. Give a formal definition of language L_i.
2. Give an NFA that recognizes L_1.
3. Give an NFA that recognizes L_2.
4. Give an NFA that recognizes L_3.

One can use a tool such as Aude to find this last result.

Exercise 36 (♠♠) — Determinize

Consider the two following automata:

- A the NFA defined as $(\{0,1,2,3,4,5\}, \{a,b\}, 0, \Delta, \{4\})$ such that the transition relation is defined by Table 7.1a.
- B the NFA defined as $(\{0,1,2,3,4,5\}, \{a,b\}, 0, \Delta, \{0,3,4\})$ such that the transition relation is defined by Table 7.1b.
  C the NFA defined as $(\{0,1,2,3,4,5\}, \{a,b\}, 0, \Delta, \{0,3,4\})$ such that the transition relation is defined by

$$\Delta = \{(1,a,2),(2,a,3),(3,a,2),(2,a,4),(4,b,2),(2,b,5),(5,a,2),(2,b,6),(6,b,2)\}.$$

Table 7.1: Non-deterministic finite-state automata represented in tabular form. The initial state is indicated by an arrow. The states with a star are final states.

	→ 0	1	2	3	4*	5
a	1,2,3,4,5	2,3	0,1,4	0	1	2
b		4	1,2,3	1,2,5		2,3,5

(a)

	→ 0*	1	2	3*	4*	5
a	1,2			5		
b		3	4			0

(b)

1. Determinize A and minimize the DFA obtained after determinization.
2. Same question for automaton B.
3. Same question for automaton C.

Exercise 37 (♠♠) — Determining equivalence

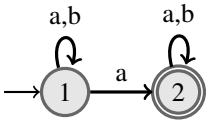
Let $\Sigma = \{0, 1\}$. Consider the two first NFAs depicted in Figure 7.2.

1. What is the language recognized by the automata depicted in Figure 7.2a?
2. What is the language recognized by the automata depicted in Figure 7.2b ?
3. Prove that these automata are equivalent.

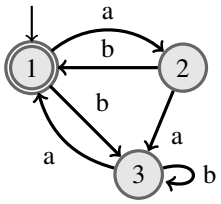
Exercise 38 (♠♠♠) — Number of accepted executions

We are interested in computing the number $\mathcal{N}(A, u)$ of executions accepted by an automaton A for a word u given as input. Recall that $|u|_a$ denotes the number of occurrences of a in word u.

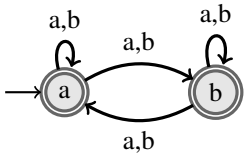
1. Consider the automaton A_1 represented in Figure 7.3a, list the accepted executions associated to $abaa$. Deduce $\mathcal{N}(A_1, abaa)$.
2. Show that $\mathcal{N}(A_1, u) = |u|_a$.
3. Consider the automaton A_2 represented in Figure 7.3b, what is the value of $\mathcal{N}(A_2, u)$ for a word u. Justify.
4. Consider an automaton A_3 represented in Figure 7.3c, what is the value of $\mathcal{N}(A_3, u)$ for a word u. Justify.
5. Consider the two DFAs $Auto_1$ and $Auto_2$ over alphabet Σ and DFA $Auto$ resulting from the product for intersection between $Auto_1$ and $Auto_2$. Show that $\forall u \in \Sigma^* : \mathcal{N}(Auto, u) = \mathcal{N}(Auto_1, u) \times \mathcal{N}(Auto_2, u)$.
6. Give an automaton A such that $\forall u \in \Sigma^* : \mathcal{N}(A, u) = (|u|_a)^2$.



(a) AENFD A_1



(b) AENFD A_2



(c) AENFD A_3

Figure 7.3: Some NFAs

CHAPTER

8 Non-deterministic Finite-State Automata with ϵ-transitions

Reminder: ϵ-NFA: Non-deterministic Finite-state Automaton with ϵ-transitions.

Exercise 39 (♠♠♠) — Prove that a language is a finite-state language

Let L be a finite-state language over an alphabet Σ.

1. Prove that $\{w \mid s \cdot w \in L\}$, the set of words of L starting with a symbol $s \in \Sigma$ and suppressing it, is a finite-state language.
2. Prove that $\{w \mid w \cdot s \in L\}$, the set of non-empty words of L ending with a symbol $s \in \Sigma$ and suppressing it, is a finite-state language.
3. Prove that $\{w \cdot s \cdot s \mid w \cdot s \in L\}$, the set of words obtained by doubling the last letter of non-empty words of L, is a finite-state language.
4. Prove that $\{w_1 \cdot s \cdot w_2 \mid w_1, w_2 \in L\}$, the set of words obtained by inserting an occurrence of a symbol $s \in \Sigma$ in a word of L, is a finite-state language.

Exercise 40 (♠) — Composing ϵ-NFAs

Let $A_1 = (Q_1, q_1^0, \Sigma, \Delta_1, F_1)$ and $A_2 = (Q_2, q_2^0, \Sigma, \Delta_2, F_2)$ two ϵ-NFAs and L_1, L_2 the languages recognized by A_1 and A_2 respectively.

1. Define the automaton $A_\cup = (Q_\cup, q_\cup^0, \Sigma, \Delta_\cup, F_\cup)$ which recognizes $L_1 \cup L_2$.
2. Define the automaton $A_\cdot = (Q_\cdot, q_\cdot^0, \Sigma, \Delta_\cdot, F_\cdot)$ which recognizes $L_1 \cdot L_2$.
3. Define the automaton $A_* = (Q_*, q_*^0, \Sigma, \Delta_*, F_*)$ which recognizes L_1^*.
4. Define the automaton $A_\cap = (Q_\cap, q_\cap^0, \Sigma, \Delta_\cap, F_\cap)$ which recognizes $L_1 \cap L_2$.
5. Prove that the compositions of automata defined in the previous questions are correct, that is $L(A_\cup) = L_1 \cup L_2$, $L(A_\cdot) = L_1 \cdot L_2$, $L_* = L_1^*$ et $L(A_\cap) = L_1 \cap L_2$.

Exercise 41 (♠♠) — Suppressing ϵ-transitions and determinization

Consider the ϵ-NFA defined as $(\{0,1,2,3,4\}, \{a,b\}, 0, \Delta, \{4\})$ such that the transition relation Δ is defined in Table 8.1a. For the following question, use the tabular representation of automata.

1. Eliminate ϵ-transitions.
2. Determinize the obtained automaton.

Table 8.1: NFAs with ϵ-transitions

	0	1	2	3	4
ϵ	$1,3$		4		
a		2		4	
b			1		3

(a)

	0	1	2	3	4
ϵ	$1,3$	3	1		3
a		2			
b			4		

(b)

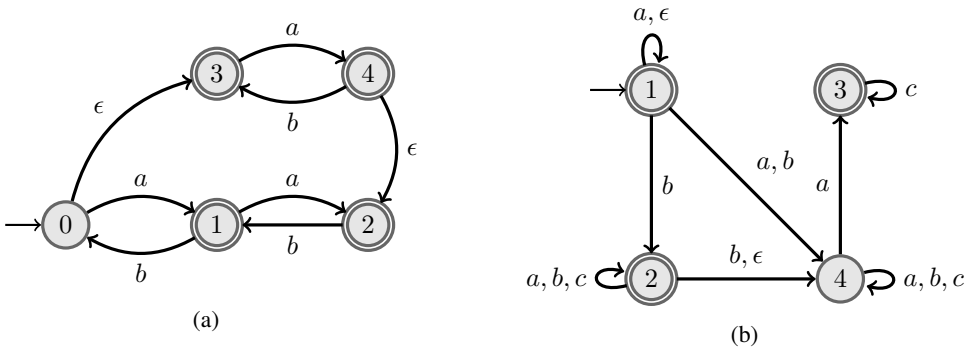


Figure 8.1: NFAs with ϵ-transitions

3. Determinize the initial automaton using the direct method (combination of elimination of the ϵ-transitions and determinization).
4. Check that the automata obtained at the two previous questions (that is using the two methods) are equivalent.

Exercise 42 (♠♠) — Suppressing ϵ-transitions and determinization

Consider the ϵ-NFA defined as $(\{0,1,2,3,4\},\{a,b\},0,\Delta,\{4\})$ such that the transition relation Δ is defined in Table 8.1b. For the following question, use the tabular representation of automata.

1. Eliminate ϵ-transitions.
2. Determinize the obtained automaton.
3. Determinize the initial automaton using the direct method (combination of elimination of the ϵ-transitions and determinization).
4. Check that the automata obtained at the two previous questions (that is using the two methods) are equivalent.

Exercise 43 (♠♠) — Suppressing ϵ-transitions and determinization

Consider the alphabet $\Sigma = \{a, b\}$ and the ϵ-NFAs defined in Figure 8.1. For each automaton, in the following questions, use the graphical representation of automata.

1. Give an accepted and non-accepted word.
2. Eliminate ϵ-transitions.
3. Determinize the obtained automaton.
4. Minimize the automaton obtained at the previous question.

Exercise 44 (♠♠♠) — Mirror langage

Let Σ be an alphabet. The mirror image $R(u)$ of a word u is the word that one obtains by reading from right to left (as in Arab or Hebrew). More precisely: $R(\epsilon) = \epsilon$ and $R(u \cdot a) = a \cdot R(u)$. Let L be a finite-state language.

1. Prove that $R(L) = \{R(u) \mid u \in L\}$ is a finite-state language.

Exercise 45 (♠♠♠) — Kleene closure, union and inclusion

Let L_1 and L_2 be languages.

1. Prove that if $L_1 \subseteq L_2$, then $L_1^* \subseteq L_2$.
2. Deduce that $L_1^* \cup L_2^* \subseteq (L_1 \cup L_2)^*$.
3. Prove that, in general, $L_1^* \cup L_2^* \neq (L_1 \cup L_2)^*$.
4. Find languages L_1 and L_2 such that $L_1 \not\subseteq L_2$, $L_2 \not\subseteq L_1$ and $L_1^* \cup L_2^* = (L_1 \cup L_2)^*$

CHAPTER

9 Regular Expressions

Exercise 46 (♠) — Simplify regular expressions

Consider an alphabet $\Sigma = \{a, b\}$. Simplify each of the following regular expression, that is find an equivalent regular expression that contains least symbols.

1. $\epsilon + a \cdot b + a \ \cdot b \cdot a \ \cdot b \cdot (a \ \cdot b)^*$.
2. $a \cdot a(b^* + a) + a \cdot (a \cdot b^* + a \cdot a)$
3. $a \cdot (a + b)^* + a \cdot a \cdot (a + b)^* + a \cdot a \cdot a \cdot (a + b)^*$.

Exercise 47 (♠) — Find regular expressions

Consider an alphabet $\Sigma = \{a, b, c\}$. For each of the following language over Σ, give a regular expression that denotes it.

1. The set of words that start with a and end with b.
2. The set of words that contain at least three occurrences of symbol b.
3. The set of words that contain at least three consecutive occurrences of symbol b.
4. The set of words that contain an even number of a.
5. The set of words that contain an odd number of a.
6. The set of words that contain an number of a multiple of 3.
7. The set of words that do not contain the factor $a \cdot a$.
8. The set of words that do not contain the factor $a \cdot a \cdot b$.
9. The set of words that contain at least 2 non-consecutive occurrences of symbol a.
10. The set of words that contain at least 3 symbols and the third symbol is symbol a.
11. The set of words that start and end by the same symbol.
12. The set of words of odd length.
13. The set of words of length at least one and at most 3.

Exercise 48 (♠♠♠) — Equivalence between regular expressions

Give a proof or a counter-example for each of the following algebraic law over regular expressions:

1. $(\epsilon + R)^* = R^*$.
2. $(\epsilon + R) \cdot R^* = R^*$.
3. $\emptyset \cdot R = R \cdot \emptyset = \emptyset$.
4. $\emptyset + R = R + \emptyset = R$.
5. $(R + S)^* = R^* + S^*$.
6. $(RS + R)^* R = R(SR + R)^*$.
7. $(RS + R)^* RS = (RR^* S)^*$.
8. $(R + S)^* S = (R^* S)^*$.
9. $S(RS + S)^* R = RR^* S(RR^* S)^*$.

Exercise 49 (♠♠♠) — Arden's Lemma

Let $A, B, X \subseteq \Sigma^*$ be languages.

1. prove that language $A^* B$ is a solution of equation $X = AX + B$.
2. Prove that if $\epsilon \notin A$, then $A^* B$ is the unique solution of $X = AX + B$ (Arden's Lemma).

CHAPTER

10 Kleene's Theorem

Exercise 50 (♠♠) — Automaton to regular expression

We want to compute regular expressions associates to the automata in Figure 10.2.

1. Compute the regular expressions using the method associating regular expressions to paths.
2. Compute the regular expressions using the method associating linear equations to states.

Exercise 51 (♠♠) — Regular expressions to automata

Let $\Sigma = \{a, b\}$. Give ϵ-NFAs associated to the following regular expressions:

1. $a \cdot b$,
2. $a^* \cdot b$,
3. $(a+b)^* \cdot a^* \cdot b^* \cdot a$,
4. $\big(a^* \cdot b + d \cdot c\big)^* \cdot \big(b^* \cdot d + a \cdot d\big)^*$,
5. $\big(a \cdot b + a^* \cdot b + c \cdot d\big) \cdot \big((c \cdot a^* + b \cdot d) \cdot (a \cdot b^* + a \cdot b \cdot d)\big)^*$.

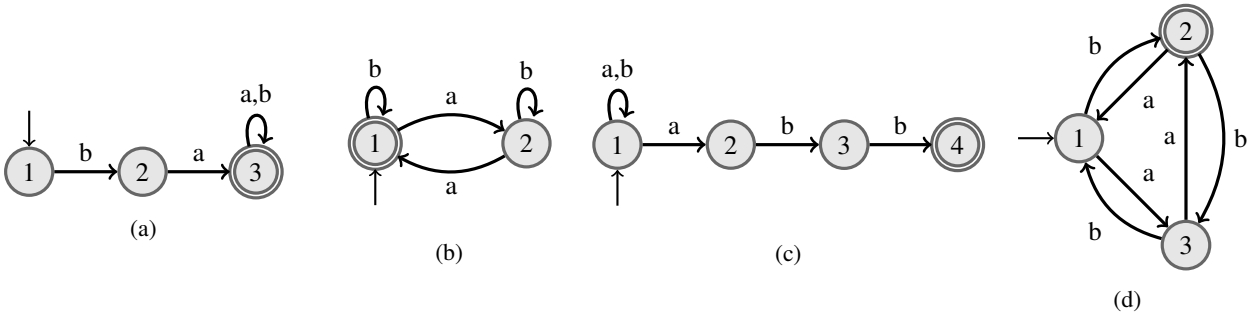
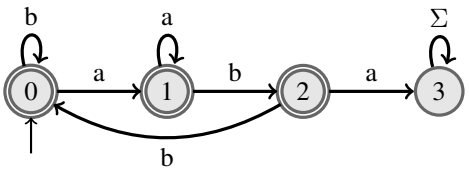
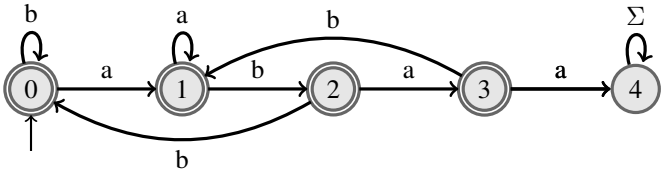


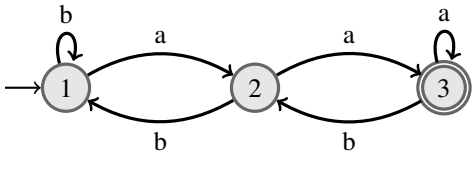
Figure 10.1: Automata to compute regular expressions



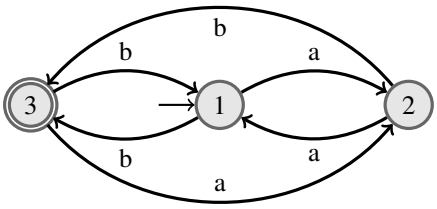
(a) Automaton recognizing the words that do not contain aba.



(b) Automaton recognizing the words that do not contain $abaa$.



(c)



(d)

Figure 10.2: Automata to compute regular expressions

CHAPTER

11 Grammars

Exercise 52 (♠) — Derivation of grammars

Consider grammar $G = (\{a, b, c\}, \{S\}, S, P)$ where P is given by the rules in Figure 11.1. Give a derivation of G for the following words:

1. $bcbba$
2. $bbbcbba$
3. $bcabbbbcb$

Exercise 53 (♠) — Words generated by a grammar

Consider grammar $G = (V_T, V_{\mathrm{NT}}, S, P)$, where :

- V_T the following set of terminal symbols

  $$\{un, une, l', etudiante, etudiant, enseignante, enseignant, dutennis, duski, descours, faitdu, etudie, donne\}$$

- P the following set of production rules

$$\begin{array}{lcllcllcl} PH & \to & GNGV & GN & \to & AFNF \mid AMNM & GV & \to & VC \\ V & \to & fait \mid etudie \mid donne & NF & \to & etudiante \mid enseignante & NM & \to & etudiant \mid enseignant \\ AF & \to & une \mid l' & AM & \to & un \mid l' & C & \to & dutennis \mid duski \mid descours \end{array}$$

1. Give some sentences generated by the grammar.
2. Give some sentences not generated by the grammar.
3. Give the number of sentences generated by this grammar. It is not required that sentences be meaningful.

Exercise 54 (♠♠) — Language generated by a grammar

What are the languages generated by the grammars of the form $(\{a, b\}, \{S, A, B\}, S, P)$ with the following sets for P, the set of production rules:

1. $\left\{ \begin{array}{lcl} S & \to & AB, \\ A & \to & ab, \\ B & \to & BB \end{array} \right\}$;
2. $\left\{ \begin{array}{lcl} S & \to & AB \mid aA, \\ A & \to & a, \\ B & \to & b \end{array} \right\}$;
3. $\left\{ \begin{array}{lcl} S & \to & AB \mid AA, \\ A & \to & aB, \\ B & \to & b \end{array} \right\}$;
4. $\left\{ \begin{array}{lcl} S & \to & AA \mid B, \\ A & \to & aaA \mid aa, \\ B & \to & bB \mid B \end{array} \right\}$;
5. $\left\{ \begin{array}{lcl} S & \to & AB \mid aAb, \\ B & \to & bBa \mid \epsilon, \\ A & \to & \epsilon \end{array} \right\}$.

$$
\begin{aligned}
S &\rightarrow abS,\\
S &\rightarrow bcS,\\
S &\rightarrow bbS,\\
S &\rightarrow a,\\
S &\rightarrow cb
\end{aligned}
$$

Figure 11.1: Production rules for the grammar in Exercise 52.

CHAPTER

12 Grammars and Regular Grammars

Exercise 55 (♠♠) — Composing grammars

Let L and L' be regular languages generated by grammars G and G' respectively. We want to prove that there exist regular grammars that generate each of the following grammars.

$$L \cup L' \qquad\qquad L \cdot L' \qquad\qquad (L)^*$$

1. From example grammars, show how to construct these grammars.
2. Generalize. Give the grammars for the requested languages from some input regular grammars. Explain the construction of these grammars.

Exercise 56 (♠) — Language generated by a grammar

What are the languages generated by the grammars of the form $(\{a, b\}, \{S, T, U\}, S, P)$ with the following sets for P, the set of production rules:

1. $\left\{ \begin{array}{lcl} S & \to & T \mid bSb, \\ T & \to & aT \mid \epsilon \end{array} \right\}$;

2. $\left\{ \begin{array}{lcl} S & \to & T \mid bS, \\ T & \to & aT \mid \epsilon \end{array} \right\}$

Exercise 57 (♠♠) — From DFAs and NFAs to Grammars

Give grammars generating the languages recognized by the DFAs given in the following figures.

1. The DFAs in Figure 3.1.
2. The DFAs in Figure 10.1.
3. The non-minimal DFAs in Figure 6.2.
4. The NFAs in Figure 7.1.
5. The NFAs in Figure 7.2.

CHAPTER

13 Pumping Lemma and Non-regular Languages

Exercise 58 (♠♠) — Find the minimal iteration constant

Consider an alphabet $\Sigma = \{0, 1\}$. Find the minimal iteration constant of the following regular languages over Σ:

1. 0001^*.
2. 0^*.
3. 0^*1^*.
4. 1011.
5. $(01)^*$.
6. ϵ.
7. $(0 + 1)^*$.
8. 10^*1.

Exercise 59 (♠♠) — Find the minimal iteration constant

Consider an alphabet $\Sigma = \{0, 1\}$. Find the minimal iteration constant of the following regular languages over Σ:

1. $10(11^*0)^*0$.
2. $011 + 0^*1^*$.
3. $0^*1^+0^+1^*$.
4. $0^*1^+0^+1^* + 10^*1$.
5. 0^*101^*.
6. $0^*101^* + 101^*$.

Exercise 60 (♠♠♠) — Relation between two languages

We consider an alphabet $\Sigma = \{a, b\}$ and the languages

- $L_1 = \{u \cdot v \cdot w \mid u, w \in \Sigma^*, v \in \{aaa, bbb\}\}$, et
- $L_2 = \{(aab)^n \cdot (abb)^n \mid n \in \mathbb{N}\}$.

1. Is language L_1 a state language? If yes, give an automaton recognizing L_1.
2. Remark a property on the length of words in L_2 ?
3. Give the factors not appearing in L_2.
4. Deduce from the previous question a relation between L_1 and L_2.
5. Is language L_2 a state language? Give a proof.
6. Is language $L_1 \cup L_2$ a state language? Give a proof.

CHAPTER

14 Proving that a Language is not Regular

In this section, we Consider an alphabets $\Sigma_1 = \{a, b, c\}$ and $\Sigma_2 = \{a, b\}$. In the following exercises, one has to show that the proposed languages are not regular using the pumping lemma.

Exercise 61 (♠) — Using the pumping lemma

Prove that the following languages are not regular.

1. $\{a^n b^{n+1} \mid n \in \mathbb{N}\}$;
2. $\{a^n b^{2\times n} \mid n \in \mathbb{N}\}$;
3. $\{a^{2\times i} b^{2\times i} \mid i \in \mathbb{N}\}$;
4. $\{a^i b^j c^{i+j} \mid i, j \in \mathbb{N}\}$;

Exercise 62 (♠♠) — Using the pumping lemma

Prove that the following languages are not regular.

1. $\{w \in \Sigma_1^* \mid |w|_a = |w|_b + |w|_c\}$.
2. $\{a^{2\times i}(b \cdot c)^i \mid i \in \mathbb{N}\}$.
3. $\{w \cdot w \cdot w \mid w \in \Sigma_2^*\}$.

Exercise 63 (♠♠) — Regular or non regular?

Consider an alphabet $\Sigma = \{a, b\}$. Say if the following languages over Σ are regular or not. Justify your answer by either giving a recognizing automaton or using the pumping lemma.

1. $\{a^n 1 b^n \mid n \in \mathbb{N}\}$
2. $\{a^n b^n \mid n \in \mathbb{N}\}$
3. $\{w \cdot w^R \mid w \in \Sigma^*\}$
4. $\{w \cdot u \cdot w^R \mid w \in \Sigma^*, u \in \Sigma^+\}$.

Exercise 64 (♠♠) — Using the closure properties to prove non-regularity

Consider an alphabet Σ. In this exercise, one has to use the closure properties of regular languages. Prove that the following languages are not regular.

1. $\{u \in \Sigma^* \mid |u| \text{ is prime}\}$.
2. $\{u \in \Sigma^* \mid |u| \text{ is a square}\}$.
3. $\{0^i 1^j 2^{i+j} \mid i, j \in \mathbb{N}\}$.
4. $\{0^{2\times i+1} 1^{2\times i+1} \mid i \in \mathbb{N}\}$.
5. $\left\{a^m b^k a^n \mid, k, n \in \mathbb{N}, m \neq n\right\}$
6. $\left\{w \in \ \Sigma^* \mid w \neq w^R\right\}$

Exercise 65 (♠♠♠) — Correct or incorrect

For each of the following statements, say whether it is correct or not. If it is correct, give a proof. If not, give a counter example.

1. If $A \cup B$ is regular and A is regular, then B is regular.
2. If $A \cap B$ is regular and A is regular, then B is regular.

3. If $A \cup B$ is not regular and A is not regular, then B is not regular.
4. If $A \cap B$ is not regular and A is not regular, then B is not regular.
5. If $A \cup B$ is not regular and A is regular, then B is not regular.
6. If $A \cap B$ is not regular and A is regular, then B is not regular.
7. If A is regular and B is not regular, then $A \cup B$ is not regular.
8. If A is regular and B is not regular, then $A \cap B$ is not regular.

Exercise 66 (♠♠♠) — Using the pumping lemma

Prove that the following languages are not regular.

1. $\{a^i \mid i \text{ is a square}\}$;
2. $\{a^i \mid i \text{ is a cube}\}$;
3. $\{a^i \mid i \text{ is a factorial}\}$;
4. $\{a^i \mid i \text{ is prime}\}$.
5. $\{w \in \Sigma_2^* \mid |w|_a/|w|_b \in \mathbb{N}\}$.
6. $\{w \in \Sigma_2^* \mid |w|_a/|w|_b \in \mathbb{N} \text{ et est premier}\}$.

Exercise 67 (♠♠♠) — A non-regular language that satisfies the pumping lemma

Let X be a non-regular language over alphabet $\Sigma = \{a, b\}$. We consider the following languages over Σ :

- $A = \{aba\} \cdot \Sigma^*$
- $B = \overline{A}$
- $C = (\{aba\} \cdot X) \cup B$

1. Prove that C satisfies the pumping lemma.
2. Prove that C is not regular.

APPENDIX

A Modeling and Solving Problems with Automata

Exercise 68 (♠♠♠) — Code for access control

Consider the alphabet formed by digits in base 10: $\Sigma = \{0, \ldots, 9\}$. We want to model several forms of automata that allow to recognize a code in the form $x \cdot y \cdot z$ with $x, y, z \in \Sigma$. For the following questions, consider alternatives depending on the condition determining that a code has been entered. For example, it is possible to consider a button to validate one's input or that each sequence of three digits corresponds to an input.

1. The automaton only leaves one chance.
2. The automaton leaves only two chances.
3. The automaton allows to cancel its input using special button.
4. The automaton accepts as soon as the entered digits are entered correctly.

Exercise 69 (♠♠♠) — Tennis

In Tennis, points are counted as follows: 15, 30, 40. Up to 40, points are counted in an incremental fashion. When at least one of the players reach score 40, if the other player has a strictly lower score and the player with 40 points scores another point, he wins the game. If the two players reach score 40, then the player winning the next point takes the advantage. If the player with the advantage scores another point, he wins the game. Otherwise, if the other player wins the next point, they both come back to 40.

1. Give an automaton that counts points in tennis. One can use a state to contain the score and an alphabet with two symbols, each symbol corresponds to the victory of a point.

Exercise 70 (♠♠♠) — Coffee machine

We want to model a simplified coffee machine and its interaction with clients. The machine serves several types of drinks: coffee, tea, and chocolate. Alls drinks have the same price: 1 token. The machine should let the user choose the drink. It has to deliver a drink when the user has paid the price of the drink and done his choice. It has to offer to the user the possibility of retrieving his token if she has not confirmed her choice. The user can insert tokens, select a drink, retrieve his drink, retrieve his token, and asks for the cancellation of the service.

1. Determine the actions of the user and model his behavior with an automaton.
2. Determine the actions of the machine and model his behavior with an automaton.
3. How can we obtain the global (observable) behaviors of the machine? Describe some of its behavior.
4. Using the algorithms seen in the course, determine how to verify the following properties on the global behaviors of the model:
   - Can the client obtain a drink without inserting a token?
   - Can the client, after inserting a token and choose a drink, not obtain her drink?
   - Can the client obtain a drink when she has selected another drink?
   - Can the system block?
5. Change the behavior of the client and the machine and revisit the previous questions. For instance, for the machine, one can elaborate the behavior, make it more realistic, or introduce breaches. For the client, one can consider a malicious client.

Exercise 71 (♠♠♠) — A shepherd, a wolf, a goat, and a cabage

Mr Shepard S brings up the wolf W, the goat G, and the cabbage C near the river and he wants to get across using a small boat. The boat is so small that S enters in it alone or with one fellow. With no surveillance of S, the W eats G and G eats C. We want to determine with an automaton how S can get the fellowship across the river?

1. Determine the set of states of the automaton which could contain the possible situation on both sides of the river.
2. Build an automaton that models the situation.
3. Use the automaton to find how the river crossing situation can be solved.

Exercise 72 (♠♠♠♠) — Stange planet

We revisit the modeling of the strange planet seen in the course. On this planet, there are three species. We call these species the red, blue and orange ones. We want to model the evolution of the population of these species according to their reproduction rules;

- 2 individuals of two *different species* can mate ;
- mating kills the individuals ;
- mating generates two individuals of the third species.

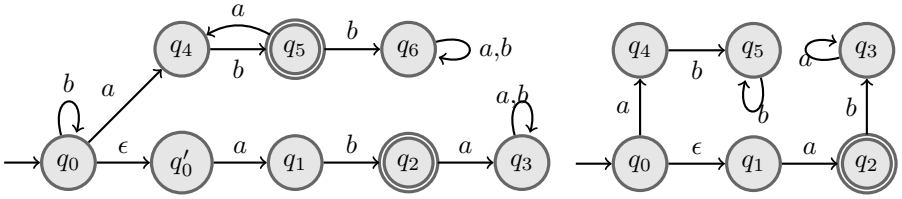
For example: 1 red and 1 blue $\rightarrow$ 2 orange. We also suppose that:

- individuals can mate just after being generated (no distinction child/adult);
- individuals can die only after mating.

1. Find a symmetry in the states when there is a fixed number of individuals.
2. Express a condition on the set of individuals (and not on the list of individuals).
3. Propose a new representation of the state based on the observations done in the two previous questions.
4. Revisit the automata of the course for a planet with 2, 3 and 4 individuals.
5. Propose an automaton for a planet with 5 individuals.
6. In general, how many states are there for a planet with n individuals?
7. For a given state, how can one determine if:
   a) The evolution will inevitably stop.
   b) The evolution cannot stop.
   c) The evolution can stop.

Exercise 73 (♠♠♠♠) — Opacity of a system

In this exercise, we are interested in an important property of information systems: *opacity*.
We suppose that an attacker observes a system which behavior is modeled by an ϵ-NFA. The accepting states of the automaton represent the “secret": during an execution of the system, the attacker should not be able to know for sure that the system is in a secret state. If during the execution of a system, the attacker is able to determine that the system is in a secret state, then we say that this execution *reveals the secret*. A system is said to be *opaque* if there does not exist an execution that reveals the secret. The attacker observes the system through an “observation window" that allows him/her to observe all transitions but ϵ-transitions. The attacker knows perfectly the structure of the automaton.



1. When we consider the system represented by the above left automaton. When the attacker observes a, b, ab, what are the current possible states of the system?
2. Say whether this system is opaque.
3. Same questions with the system modeled by the above right automaton.
4. Is it possible to construct an automaton that indicates the knowledge of an attacker according to its observation, from an automaton modeling the system?

Exercise 74 (♠♠♠) — Containers

We consider the situation where we have an input flow of water (supposed to be infinite) and two containers of 3 and 5 liter. We want to use an automaton allowing us to describe (and find) how to obtain precisely 4 liter in the 5-liter container. It is only possible to carry the following actions:

- complete the content of any container until its maximal capacity (hence ensuring that the quantity of water in the container is equal to its capacity);
- move the content of one container to the other.

These two operations are the only ones that allow to obtain a precise measure of the quantity of water in the containers.

1. Define the state space of the automaton that represents the evolving capacity of the two containers.
2. Define the set of accepting states of this automaton.
3. Define the alphabet of the automaton and the transition relation between states.
4. Suppose that this automaton is generated. How can we solve the initial problem?
5. How can we solve the initial problem if we cannot apriori generate completely the automaton.