Cursus: M1, computer-science
Code UE: JEIN8602
Solutions to the subject: Formal languages theory
Date: 20 March 2016
Duration: 3H
Documents: authorized
Lectures by: Mr Géraud Sénizergues

Exercice 1 [/4] We consider the finite automaton $\mathcal{A}$ described on figure 1.

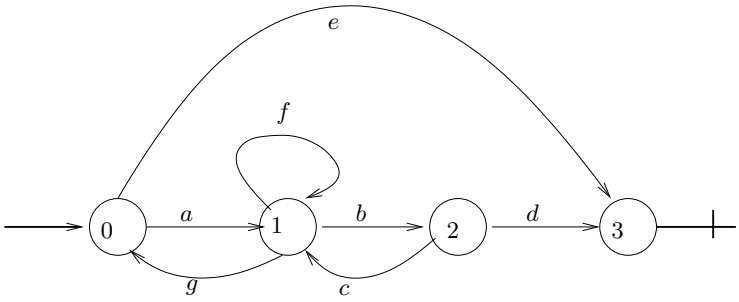

Figure 1: finite automaton $\mathcal{A}$

0- The following sequences are successful computations of $\mathcal{A}$:

$$0, a, 1, b, 2, d, 3$$
$$0, a, 1, g, 0, a, 1, b, 2, d, 3$$
$$0, a, 1, f, 1, b, 2, d, 3$$
$$0, a, 1, b, 2, c, 1, b, 2, d, 3$$
$$0, e, 3$$

1- We transform $\mathcal{A}$ into a normalized extended f.a. $\mathcal{A}_1$ where i (resp. t) is the initial (resp. terminal) state. We then eliminate successively states in the ordering: $0, 1, 2, 3$. We obtain the sequence of extended automata shown on figure 2.
It follows that:

$$e \cup a(f \cup (ga))^* ge \cup a(f \cup (ga))^* b[c(f \cup (ga))^* b]^*[d \cup c(f \cup (ga))^* ge]$$

is a regular expression for $L_{\mathcal{A}}$.
2- A regular expression for the language $h(L_{\mathcal{A}})$ is obtained by replacing each letter $x \in \{a, b, c, d, e, f, g\}$ by its image $h(x)$ in the above regular expression. We thus obtain the expression:

$$bb \cup a(aa \cup ba)^* bbb \cup a(aa \cup ba)^* ba[abb(aa \cup ba)^* ba]^*[b \cup abb(aa \cup ba)^* bbb]$$

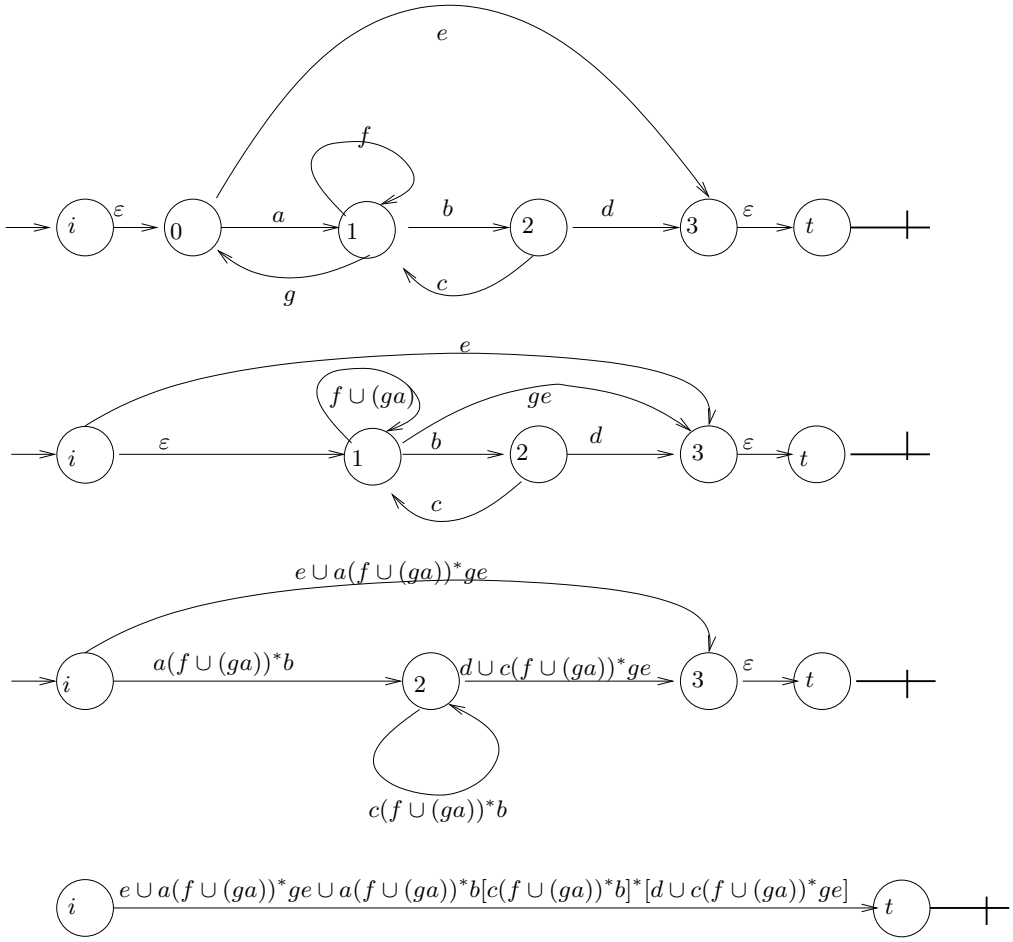

Figure 2: sequence of extended automata, ex. 2

3- A finite automaton recognizing the language $h(L_{\mathcal{A}})$ is depicted on figure 3

Exercice 2 [/4] Let us consider the regular expression:

$$e := (((ab) \cup (bc))^* a)^* c$$

Let us apply Glushkov's method.
The locally testable language associated to e is:

$$L' := (((a_1 b_1) \cup (b_2 c_1))^* a_2)^* c_2$$

We compute

$$Ini(L') = \{a_1, b_2, a_2, c_2\}, \quad Fin(L') = \{c_2\},$$

$$Dig(L') = \{a_1b_1, b_2c_1, b_1a_1, b_1b_2, c_1a_1, c_1b_2, b_1a_2, c_1a_2, a_2a_1, a_2b_2, a_2a_2, a_2c_2\}$$

This gives the finite automaton represented on figure 4, where the set of states is

$$\{0, a_1, a_2, b_1, b_2, c_1, c_2\}$$

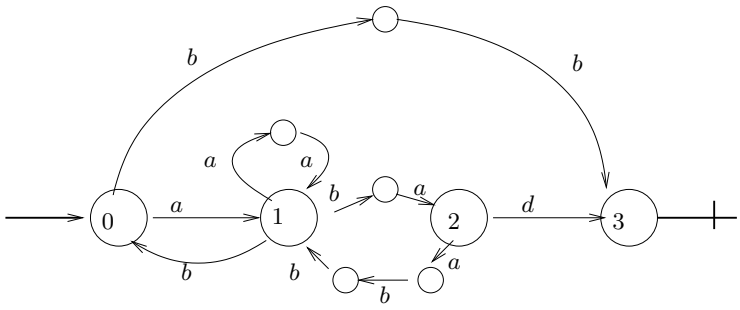

Figure 3: a finite automaton for $h(L_{\mathcal{A}})$

0 is the unique initial state and c_2 is the (unique) final state.

Exercice 3 [/4]

0-

$$\hat{\delta}(0, acaaa) = 5, \;\; \hat{\delta}(0, acbaa) = 5, \;\; \hat{\delta}(0, baa) = 5, \;\; \hat{\delta}(0, bcbaa) = 5.$$

hence all the words $acaaa$, $acbaa$, baa, $bcbaa$ belong to $L_{\mathcal{B}}$.
Concerning the four other words we obtain:

$$\begin{aligned} \hat{\delta}(0, ac) = 4, \quad & \text{and} \quad \delta(4, c) \text{ is undefined} \\ \hat{\delta}(0, bc) = 6, \quad & \text{and} \quad \delta(6, c) \text{ is undefined} \\ \hat{\delta}(0, bca) = 2, \quad & \text{and} \quad \delta(2, b) \text{ is undefined} \\ \hat{\delta}(0, acbaa) = 5, \quad & \text{and} \quad \delta(5, a) \text{ is undefined} \end{aligned}$$

hence the words acc, bcc, $bcab$, $acbaaa$, belong to the complement of $L_{\mathcal{B}}$.
1- The automaton is $\mathcal{B}$ *deterministic*: given a state q and a letter x, there exists at most one state q' such that $q \to^x_{\mathcal{B}} q'$.
2-

$$\hat{\delta}(0, \varepsilon) = 0, \;\; \hat{\delta}(0, a) = 1, \;\; \hat{\delta}(0, b) = 2, \;\; \hat{\delta}(0, aa) = 3, \;\; \hat{\delta}(0, ac) = 4, \;\; \hat{\delta}(0, aaa) = 5, \;\; \hat{\delta}(0, bc) = 6,$$

hence every state of $\mathcal{B}$ is accessible.
3- $\mathcal{B}$ is not complete: there is no transition reading letter b from state 2.
4- Let us add a new state P to the set of states of $\mathcal{B}$. We obtain the f.a. $\mathcal{B}'$, whose set of states is $Q' := \{, 0, 1, 2, 3, 4, 5, 6, P\}$, with initial state 0, final state 5 and transition table:

$x \backslash q$:	0	1	2	3	4	5	6	P
a	1	3	3	5	1	P	2	P
b	2	2	P	P	1	P	1	P
c	P	4	6	P	P	P	P	P

5- Let us compute the Nerode equivalence of $\mathcal{B}'$:
We apply the refinement algorithm exposed in the lectures:

$\equiv_0 = \{\{0, 1, 2, 3, 4, 6, P\}, \{5\}\}$; $\equiv_1 = \{\{0, 1, 2, 4, 6, P\}, \{3\}, \{5\}\}$; $\equiv_2 = \{\{0, 4, 6, P\}, \{1, 2\}, \{3\}, \{5\}\}$;

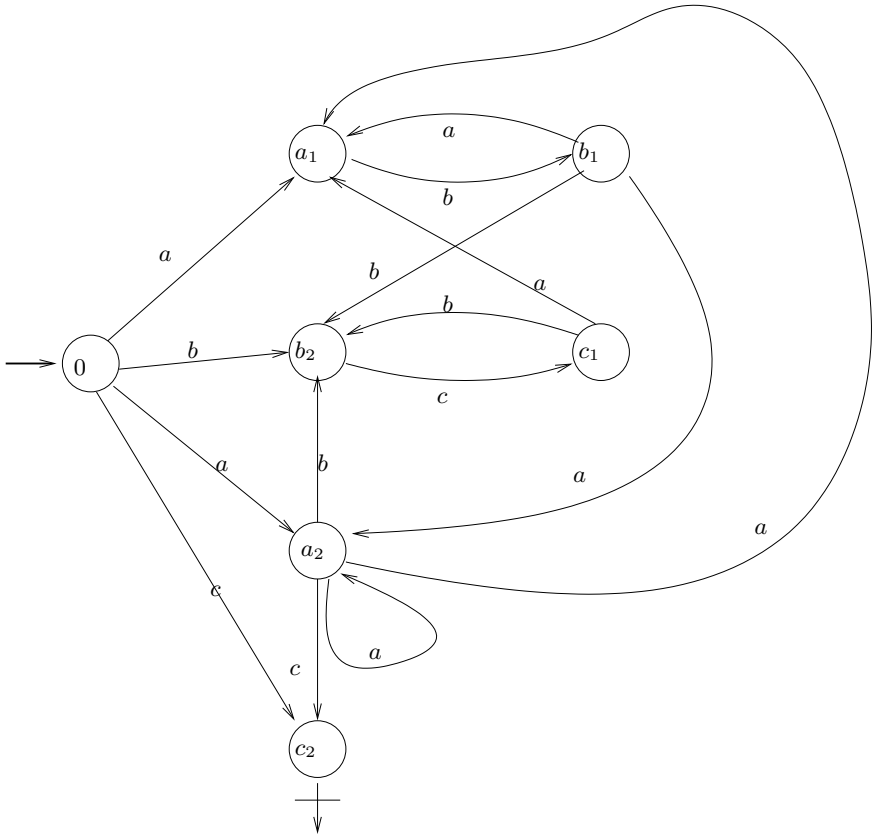

Figure 4: finite automaton for L_e

$$\equiv_3 = \{\{0,4,6\},\{P\},\{1,2\},\{3\},\{5\}\};\ \equiv_4 = \{\{0,4,6\},\{P\},\{1\},\{2\},\{3\},\{5\}\};$$
$$\equiv_5 = \{\{0\},\{4\},\{6\},\{P\},\{1\},\{2\},\{3\},\{5\}\};\ \equiv_6 = \equiv_5 .$$

Hence the Nerode equivalence over Q' is just the equality. This implies that $\mathcal{B}'$ is minimal (among all the *complete* deterministic finite automata recognizing $L_{\mathcal{B}}$).
It follows that $\mathcal{B}$ is minimal (among all the deterministic finite automata recognizing $L_{\mathcal{B}}$).

Exercice 4 [/5]

1- Let $A = \{a, b, c\}$ and $N = \{S\}$. For every $i \in \{1, 2, 3\}$, we define a set of rules R_i over A and N such that the c.f. grammar $G_i := (A, N, R_i)$ generates the language L_i.
R_1 is the set of rules:

$$S \to aSc,\ \ S \to b.$$

R_2 is the set of rules:

$$S \to abS,\ \ S \to abSc,\ \ S \to abc$$

R_3 is the set of rules:

$$S \to aSa,\ \ S \to bSb,\ \ S \to c.$$

2- We remark that: $a\tilde{b}a = aba \neq aaa$. Hence $abacaaa \in L_4$.
Similarly:

$$ba\tilde{b}a = abab \neq ab \Rightarrow babacab \in L_4$$

$$\tilde{ba} = ab \neq abab \Rightarrow bacabab \in L_4.$$

3-

3.1 Let us assume that

$$w = v_1 x v_2 c v_3 y v_4$$

for some $v_1, v_2, v_3, v_4 \in \{a,b\}^*, x, y \in \{a,b\}$ such that $v_4 = \tilde{v_1}, x \neq y$.
Since $v_1\tilde{x}v_2 = \tilde{v_2}x\tilde{v_1}$, we get that the word $x\tilde{v_1} = xv_4$ *is a suffix* of $v_1\tilde{x}v_2$. On the other hand, since $x \neq y$, this word xv_4 *is not a suffix* of v_3yv_4. It follows that

$$v_1\tilde{x}v_2 \neq v_3 y v_4$$

hence

$$w \in L_4.$$

3.2 Let us assume that

$$w = v_1 x v_2 c v_4$$

for some $v_1, v_2, v_4 \in \{a,b\}^*, x \in \{a,b\}$ such that $v_4 = \tilde{v_1}$.
Since $v_1\tilde{x}v_2 = \tilde{v_2}x\tilde{v_1}$ and $v_4 = \tilde{v_1}$ we get that the word v_4 *is a strict suffix* of $v_1\tilde{x}v_2$, hence $v_1\tilde{x}v_2 \neq v_4$, which shows that

$$w \in L_4.$$

4- Suppose that

$$u = v_1 c v_3 y v_4$$

for some $v_1, v_3, v_4 \in \{a,b\}^*, y \in \{a,b\}$ such that

$$v_4 = \tilde{v_1}.$$

(we denote by (3.3) this form of word w). By an argument similar to that used in question 3.2,

$$w \in L_4.$$

5- Let $A = \{a,b,c\}$ and $N_4 = \{S, T_1, T_2, L, R\}$. We build a c.f. grammar $G_4 := (A, N_4, R_4)$ by defining R_4 as the following set of rules:

$$S \to aSa, \;\; S \to bSb, \;\; S \to aT_1b, \;\; S \to bT_1a \tag{1}$$

$$T_1 \to aT_1, \;\; T_1 \to bT_1, \;\; T_1 \to T_2, \;\; T_2 \to T_2a, \;\; T_2 \to T_2b, \;\; T_2 \to c \tag{2}$$

$$S \to aL, \;\; S \to bL, \;\; L \to aL, \;\; L \to bL, \;\; L \to c \tag{3}$$

$$S \to Ra, \;\; S \to Rb, \;\; R \to Ra, \;\; R \to Rb, \;\; R \to c \tag{4}$$

One can check that, together with the set (1), the subset (2) of rules generates exactly all words w of form (3.1), the subset (3) of rules generates exactly all words w of form (3.2) and the subset (4) of rules generates exactly all words w of form (3.3).
A given word of L_4 fulfills exactly one of forms (3.1),(3.2),(3.3), hence a word $w \in L_4$ can be generated by the rules of set (1) augmented by only one of the sets of rules (2) or (3) or (4). We conclude that

$$L(G_4, S) = L_4.$$

For every such form, the decomposition given in the text (form (3.1) or(3.2)) or given in our solution of question 4 (form (3.3)) is also unique, implying that the derivation within the set

of rules (1, 2) [resp.(1, 3), (1, 4)] is unique.
Hence the grammar G_4 is non-ambiguous.

Exercice 5 [/5] We consider the context-free grammar $G := (A, N, R)$ where $A = \{a, b, c\}$, $N = \{S_1, S_2, S_3, S_4, S_5\}$ and R consists of the following 12 rules:

$$\begin{array}{ccc} S_1 \to aS_1S_1 & S_1 \to bS_3S_1 & S_1 \to S_2c \\ S_2 \to S_2S_1 & S_2 \to aS_3 & S_2 \to S_1S_2S_1 \\ S_3 \to a & S_3 \to S_1S_3 & S_4 \to cS_4 \\ S_4 \to aS_4S_5 & S_5 \to aS_5 & S_5 \to aS_4a \end{array}$$

The start symbol of G is S_1.
1- We compute the subset of *productive* non-terminals of G by the fixpoint technique explained in the lecture:

$$V_1 = \{S_4\}, V_2 = \{S_1, S_4\}. V_3 = \{S_1, S_4, S_5\}, V_4 = V_3.$$

Hence the set of productive non-terminals is $\{S_1, S_4, S_5\}$.
2- The c.f. grammar obtained by removing all the non-productive non-terminals is thus: $\hat{G} := (A, \hat{N}, \hat{R})$ where $A = \{a, b, c\}$, $\hat{N} = \{S_1, S_4, S_5\}$ and $\hat{R}$ consists of the following rules:

$$\begin{array}{ccc} S_1 \to aS_4 & & \\ S_4 \to bS_3S_4 & S_4 \to aS_4S_4 & S_4 \to b \\ S_5 \to aS_4S_5 & S_5 \to bS_1 & S_5 \to aS_5. \end{array}$$

We compute the subset of *useful* non-terminals of $\hat{G}$ by the fixpoint technique explained in the lecture:

$$N_1 = \{S_1\}, N_2 = \{S_1, S_4\}, N_3 = N_2$$

Hence the set of useful non-terminals is $\{S_1, S_4\}$.
3- We can thus transform the grammar $\hat{G}$ into an equivalent grammar G' where every non-terminal is productive and useful, just by restricting both the non-terminal alphabet and the rules to the subset N_2:
$G' := (A, N', R')$ where $A = \{a, b, c\}$, $N' = \{S_1, S_4\}$ and R' consists of the rules:

$$S_1 \to aS_4, \quad S_4 \to aS_4S_4, \quad S_4 \to b$$

4- We note that:

$$S_1 \to aS_4 \to ab,$$

hence the language $\mathrm{L}(G, S_1)$ is not empty.
(We can also invoke the fact that S_1 is productive, by question 1).
5- We observe that:

$$S_1 \to aS_4 \text{ and } S_4 \to^* aS_4b \text{ and } S_4 \to b.$$

hence $\{a^{n+1}b^{n+1} \mid n \geq 1\} \subseteq \mathrm{L}(G, S_1)$, which shows that the language $\mathrm{L}(G, S_1)$ is *infinite*.
6- One can prove, by induction over the length n of derivations that, for every $w \in \{a, b\}^*$:

$$\begin{array}{lcl} S_4 \xrightarrow{n} w & \rightarrow & |w|_a - |w|_b = -1 \\ S_1 \xrightarrow{n} w & \rightarrow & |w|_a - |w|_b = 0 \end{array} \tag{5}$$

Let us prove that the language $L := L(G, S_1)$ is not rational.
Suppose that L is recognized by some *f.a.* $\mathcal{A}$ with N states. Let $w := a^N b^N \in L = L_{\mathcal{A}}$. Let

$$q_0 \xrightarrow{a}_{\mathcal{A}} q_1 \cdots \xrightarrow{a}_{\mathcal{A}} q_i \cdots \xrightarrow{a}_{\mathcal{A}} q_N \xrightarrow{b^N}_{\mathcal{A}} q \in Q_f,$$

be some successful computation of $\mathcal{A}$ over w. Since $N + 1 > N$, $\exists 0 \leq i < j \leq N$ such that $q_i = q_j$. It follows that

$$q_0 \xrightarrow{a^i}_{\mathcal{A}} q_i = q_j \xrightarrow{a^{N-j}}_{\mathcal{A}} q_n \xrightarrow{b^N}_{\mathcal{A}} q,$$

is also a successful computation of $\mathcal{A}$. Hence $a^{N-(j-i)}b^N \in L_{\mathcal{A}} = L$, which contradicts implication (5). We have proved that such a f.a. $\mathcal{A}$ cannot exist.
Exercice 6 [/4] We consider the context-free grammar $G := (A, N, R)$ where $A = \{a, \bar{a}, b, \bar{b}\}$, $N = \{S\}$ and R consists of the following rules:

r1: $S \rightarrow aS\bar{a}$
r2: $S \rightarrow \bar{a}Sa$
r3: $S \rightarrow bS\bar{b}$
r4: $S \rightarrow \bar{b}Sb$
r5: $S \rightarrow SS$
r6: $S \rightarrow \varepsilon$

The start symbol of G is S.
1-

$$S \rightarrow aS\bar{a} \rightarrow abS\bar{b}\bar{a} \rightarrow abaS\bar{a}\bar{b}\bar{a} \rightarrow abaaS\bar{a}\bar{a}\bar{b}\bar{a} \rightarrow abaa\bar{a}\bar{a}\bar{b}\bar{a}.$$

This derivation is both rightmost and leftmost.
2-

$$S \rightarrow SS \rightarrow aS\bar{a}S \rightarrow a\bar{a}S \rightarrow a\bar{a}bS\bar{b} \rightarrow a\bar{a}b\bar{b},$$

is a leftmost derivation.

$$S \rightarrow SS \rightarrow SbS\bar{b} \rightarrow Sb\bar{b} \rightarrow aS\bar{a}b\bar{b} \rightarrow a\bar{a}b\bar{b},$$

is a rightmost derivation. A derivation-tree (corresponding to these two derivations) is depicted on figure 5.
3-Let us exhibit two different leftmost derivations for the word $a\bar{a}a\bar{a}$

$$S \rightarrow SS \rightarrow aS\bar{a}S \rightarrow a\bar{a}S \rightarrow a\bar{a}aS\bar{a} \rightarrow a\bar{a}a\bar{a}$$

$$S \rightarrow aS\bar{a} \rightarrow a\bar{a}Sa\bar{a} \rightarrow a\bar{a}a\bar{a}.$$

Their associated derivation-*trees* must be different since the correspondance between leftmost derivations and derivation-trees is *bijective*.
4- The result of question 3 proves that the c.f. grammar G is ambiguous.

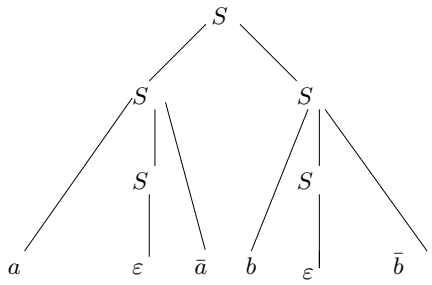

Figure 5: a derivation tree for $a\bar{a}b\bar{b}$

5- The grammar G' is non-ambiguous:it is linear and the first half of a word $w\bar{w}$ completely determines the sequence of rules that must be used in a leftmost derivation that generates $w\bar{w}$.
Every word in $L(G', S)$ has the same leftmost and rightmost letter. Hence $a\bar{a}b\bar{b} \in L(G, S) \setminus L(G', S)$.