Travaux dirigés
de
Langages formels

TD1 : Mots et langages

TD2 : Grammaires

TD3 : Automates

TD4 : ε-Transitions

TD5 : Déterminisation d'automates

TD6 : Expressions rationnelles

TD7 : Minimalisation d'automates

TD 1 : mots et langages

Décomposition des mots

Exercice 1

Soit $m = acba$. Donner tous ses
- ◇ préfixes
- ◇ suffixes
- ◇ facteurs
- ◇ sous-mots

Exercice 2

1. Soit $m = aba$. Donner toutes les factorisations de m en deux facteurs quelconques.
2. Soit $m = abab$. Donner toutes les factorisations de m en trois facteurs non vides.

Compréhension intuitive de * et de $^+$

Exercice 3

- ◇ Donner tous les éléments de $\{a, b\}^*$de longueur inférieure ou égale à 2.
- ◇ Donner tous les mots de longueur inférieure ou égale à trois de $\{a, b\}^*$. Quel ordre de classement avez-vous adopté ? Combien y a-t-il de mots de $\{a, b, c\}^*$de longueur 10 ?

Exercice 4

Que valent (on ne demande pas de démonstration)

$\emptyset^*$? $\emptyset^+$? $\{\epsilon\}^*$? $\{\epsilon\}^+$?

Exercice 5

Sous quelles conditions a-t-on (on ne demande pas de démonstration)
- ◇ $X^* = X^+ \cup \{\epsilon\}$?
- ◇ $X^+ = XX^*$?
- ◇ $X^+ = X^* - \{\epsilon\}$?

Formules algébriques sur les langages

Exercice 6

Prouver que pour tout langage L non vide : $\epsilon \in L \iff L \subseteq L^2$

Exercice 7

Soit un alphabet Σ et $A \subseteq \Sigma^*$. Montrer que :

$(A^* = \Sigma^*) \iff (\Sigma \subseteq A)$.

Exercice 8

1. Prouver $(XY)^*X = X(YX)^*$
2. Prouver $(XY)^* = \{\epsilon\} \cup X(YX)^*Y$

Exercice 9

1. Prouver $(X^*)^2 \subseteq X^*$ et en déduire que $(X^*)^2 = X^*$
2. Trouver un langage X tel que $(X^*)^2 \neq (X^2)^*$
3. Prouver $X^* = X^{*^*}$

Exercice 10

Pour tout mot m, on note $Prefs(m)$ l'ensemble des préfixes de m, et, si L est un langage, on note $Prefs(L) = \bigcup_{m \in L} Prefs(m)$.

1. Que vaut $Prefs(L)$ quand L est un langage infini sur un alphabet à une lettre ? (par exemple, tous les mots composés d'un nombre premier de a) .
2. A-t-on
   - ◇ $\forall L_1, L_2$ tels que $L_1 \neq \emptyset$ et $L_2 \neq \emptyset$:
     $Prefs(L_1)Prefs(L_2) = Prefs(L_1) \bigcup Prefs(L_2)$?
   - ◇ $\forall L_1, L_2 \qquad Prefs(L_1)Prefs(L_2) = Prefs(L_1L_2)$?

   Prouver ces résultats ou donner un contre exemple. Indiquer si l'on peut remplacer l'égalité par une inclusion.

Diverses propriétés : langages fermés pour la concaténation, combinatoire, code

Exercice 11

On considère les langages suivants sur le même alphabet $\Sigma = \{a, b\}$:

- ◇ $L_1 = \{m \in \Sigma^* \mid \ |m|_a = |m|_b + 1\}$
- ◇ $L_2 = \{m \in \Sigma^* \mid \ |m|_a \geq |m|_b\}$
- ◇ $D_1 = \{m \in \Sigma^* \mid |m|_a = |m|_b \text{ et } \forall i \in [1 \ldots |m| - 1] \ : \ |m[1..i]|_a > |m[1..i]|_b\}$
- ◇ $D'_1 = \{m \in \Sigma^* \mid |m|_a = |m|_b \text{ et } \forall i \in [1 \ldots |m| - 1] \ : \ |m[1..i]|_a < |m[1..i]|_b\}$
- ◇ $D_2 = \{m \in \Sigma^* \mid |m|_a = |m|_b \text{ et } \forall i \in [1 \ldots |m| - 1] \ : \ |m[1..i]|_a \geq |m[1..i]|_b\}$
- ◇ $L_L = \{m \in \Sigma^* \mid |m|_b = |m|_a + 1 \text{ et } \forall i \in [1 \ldots |m| - 1] \ : \ |m[1..i]|_a \geq |m[1..i]|_b\}$

Que peut-on dire du dessin d'un mot de chacun de ces langages ?
Expliquer pour chacun de ces langages s'il est fermé pour la concaténation ?
Montrer comment décomposer un mot de L_2 en un produit de mots de D_1, D'_1, et $\{a\}$.
Quels sont les mots premiers de D_2 ?

Exercice 12

Montrer par des dessins que pour tout mot $m \in L_L$ ([1]) :
$m \neq b \Rightarrow \exists m_1, m_2 \in L_L$ tels que $m = am_1m_2$
Prouver formellement le même résultat.

Exercice 13

Prouver le *lemme de Lévi*, à savoir que $\forall x, y, z, t \in A^* : \ xy = zt \Rightarrow \exists u \in A^*$ tel que

$$\begin{cases} x = zu \text{ et } uy = t \\ \qquad \text{ou} \\ xu = z \text{ et } y = ut \end{cases}$$

Exercice 14

Soit $\Sigma = \{a, \ b\}$ et $F = \{aa, \ ab, \ ba, \ bb\}$. F est-il un code ? F^* est-il égal à Σ^* ?
Rappel : F est un code si et seulement si $\forall m \in F^+, \exists! \, f_1, ... f_n \; tel \, que \, m = f_1 ... f_n \; avec \, f_i \in F$

1. tel qu'il est défini dans l'exercice précédent.

TD 2 : grammaires

Application de la définition des grammaires

Exercice 1

On définit les grammaires suivantes :

1. $G_D = < \{a, b\}, \{S\}, \{S \to aSb,\ S \to \epsilon\}, S >$
2. $G_{D_{e_1}} = < \{a, b\}, \{S\}, \{S \to aSb \,|\, SS | \epsilon\}, S >$
3. $G_{D_{e_2}} = < \{a, b\}, \{S\}, \{S \to aSbS \,|\, \epsilon\}, S >$
4. $G_{L_1} = < \{a, b\}, \{S\}, \{S \to aSS \,|\, b\}, S >$
5. $G_{L_2} = < \{a, b\}, \{S\}, \{S \to SSb \,|\, a\}, S >$

On donne d'autre part les mots suivants :

$m_1 = aabb$, $m_2 = aaabbb$, $m_3 = aaabbbb$, $m_4 = aabbab$, $m_5 = aaabaabbb$

Indiquer quels mots sont générés par quelles grammaires. Donner les chaînes de dérivations correspondantes.

Exercice 2

On définit les grammaires suivantes :

— $G_D = < \{a, b\}, \{S\}, \{S \to aSb,\ S \to \epsilon\}, S >$
— $G_3 = < \{a, b\}, \{S, T, U, V\},$
$\{S \to aT \,|\, aU \,|\, aV \,|\, \epsilon,\ T \to aU \,|\, aV \,|\, \epsilon,\ U \to aV \,|\, \epsilon,\ V \to bV \,|\, \epsilon\}, S >$

Quels sont les mots générés à la fois par G_D et G_3 ? Donner les chaînes de dérivations correspondantes pour chacune des deux grammaires.

Exercice 3

Soit la grammaire $G = < \{a, b\}, \{S\}, \{S \to aaS \,|\, b\}, S >$

Donner l'ensemble des mots du langage générés par cette grammaire.

Lemme fondamental

Exercice 4

Soit la grammaire $G_L = < \{a, b\}, \{S\}, \{S \to aSS \,|\, b\}, S >$. En admettant qu'il existe une chaîne de dérivations $S \to aSS \xrightarrow{*} aaabbbabb$, trouver deux chaînes de dérivations $S \xrightarrow{*} m_1$ et $S \xrightarrow{*} m_2$ telles que $am_1m_2 = aaabbbabb$

Pourquoi le lemme fondamental nous garantit-il l'existence de ces deux dérivations ?

Exercice 5

Soit la grammaire $G_L = < \{a, b\}, \{S\}, \{S \to aSbS \,|\, \epsilon\}, S >$. En admettant qu'il existe une chaîne de dérivations $S \to aSbS \xrightarrow{*} aaabbbab$, trouver deux chaînes de dérivations $S \to m_1$ et $S \to m_2$ telles que $m_1m_2 = aaabbbab$

Le lemme fondamental nous garantit-il l'existence de ces deux dérivations ?

Appliquer correctement le lemme.

Exercice 6

Soit une grammaire G d'axiome S sur un alphabet terminal Σ et un alphabet non terminal X.
On considère l'application L_G de $(\Sigma \cup X)^*$ dans $\mathcal{P}((\Sigma \cup X)^*)$ définie par :

$L_G(u) = \{v \in (\Sigma \cup X)^* \mid u \xrightarrow{*} v\}$.

Remarquons que $L_G(S)$ est le langage étendu de la grammaire G.

Montrer que $L_G(u_1u_2) = L_G(u_1)L_G(u_2)$

Arbre de dérivations

Exercice 7

Soit la grammaire $D' =< \{a,\ b\},\ \{S'\},\ S',\ \{S' \to aS'bS' \mid \epsilon\} >$.

1. Dessiner les arbres de dérivation de $aabbab$ et de $aabbabab$.
2. En déduire une justification graphique qui montre que $L_{D'}$ est fermé pour la concaténation.
3. Prouver la propriété suivante : $S' \xrightarrow{*} m \;\Rightarrow\; S' \xrightarrow{*} mS'$
4. En déduire que $L_{D'}$ est fermé pour la concaténation.

Exercice 8

Soit la grammaire $D =< \{a,\ b\},\ \{S\},\ S,\ \{S \to aSbS \mid aS \mid \epsilon\} >$.

Dessiner les arbres de dérivation de aab, de $aaabaab$ et de $aaaabaabaab$

Preuve de propriétés des mots d'un langage algébrique

Exercice 9

Soit L_L le langage associé à la grammaire (dite de Lukasiewicz) $G_L =< \{a, b\},\ \{S\},\ S,\ \{S \to aSS \mid b\} >$.

1. Dessiner l'arbre de dérivation de $aaabbbabb$.
2. Prouver que, pour tout $m \in L_L$, on a la propriété $\Gamma(m)$ suivante :
   $|m|_b = |m|_a + 1$ et $\forall i \in [1 \ldots |m| - 1]\,,\quad |m[1..i]|_a \geq |m[1..i]|_b$
3. Prouver que : $\{m \in \{a, b\}^* \mid \Gamma(m)\ \} \;=\; L_L$ (se souvenir du TD1).

Exercice 10

Soient la grammaire $G =< \Sigma,\ \{S\},\ S,\ \{S \to aSS \mid b\} >$. Prouver que les mots m du langage étendu $\hat{L_G}$ associé à la grammaire G vérifient la formule $|m|_S + |m|_b = |m|_a + 1$. En déduire une propriété sur les mots de L_G.

Exercice 11

Soit la grammaire (de Dyck) $D =< \{a, b\},\ \{S\},\ S,\ \{S \to aSb \mid SS \mid \epsilon\} >$ et L_D son langage associé.

1. Dessiner l'arbre de dérivation de $aabbab$.
2. Prouver que pour tout $m \in L_D$ on a la propriété $\Gamma(m)$ suivante :
   $|m|_b = |m|_a$ et $\forall i \in [1..|m| - 1]\,,\quad |m[1..i]|_a \geq |m[1..i]|_b$.
3. Soit le langage $L_\Gamma = \{m \in \{a,\ b\}^* \mid \Gamma(m)\ \}$.
   Montrer que $L_\Gamma =\ L_D$.
   Aide :
   (a) Soit la propriété $\overline{\Gamma}(m)$ suivante :
   $|m|_b = |m|_a$ et $\forall i \in [1..|m| - 1]\,,\quad |m[1..i]|_a > |m[1..i]|_b$.
   Montrer que :
   $\Gamma(m) \;\Rightarrow\; m = \epsilon \;\; ou \;\; \exists\, m = m_1.m_2\, tel\, que\, \overline{\Gamma}(m_1)\, et\, \Gamma(m_2)\, et\, m_1 \neq \epsilon$
   (b) Montrer que : $\overline{\Gamma}(m) \;\Rightarrow\; m = \epsilon \;\; ou \;\; \exists\, m = a.m_1.b\ tel\, que\, \Gamma(m_1)$
   (c) Prouver par récurrence sur $|m|$ que : $\Gamma(m) \;\Rightarrow\; S \xrightarrow{*} m$
4. Quel est le rapport entre le langage de Dyck (L_D) et le langage de Lukasiewicz (L_L) ?

Exercice 12

L'ensemble P des palindromes sur $\Sigma = \{a,\, b\}$ est défini par induction structurelle de la façon suivante :

1. $\{\epsilon,\, a,\, b\} \subseteq P$
2. Pour tout $m \in P$, on a $ama \in P$ et $bmb \in P$

a) Définir une grammaire G_P qui engendre l'ensemble des palindromes sur $\Sigma = \{a,\, b\}$.

b) Faire une preuve par induction structurelle pour prouver que P est inclus dans le langage associé à la grammaire G_P.

c) Prouver que le langage associé à la grammaire G_P est inclus dans P.

Exercice 13

Soient les grammaires $D' = < \Sigma,\ \{S'\},\ S',\ \{S' \rightarrow aS'bS' \mid \epsilon\} >$ et $D = < \Sigma,\ \{S\},\ S,\ \{S \rightarrow aSb \mid SS \mid \epsilon\} >$
En utilisant le fait que $L_{D'}$ est fermé pour la concaténation, prouver que $L_{D'} = L_D$.

Exercice 14

Soit la grammaire $H = < \Sigma,\ \{T\},\ T,\ \{T \rightarrow aTb \mid bTa \mid TT \mid \epsilon\} >$.
Prouver que $L_H \subseteq \{m \in \Sigma^* \mid |m|_a = |m|_b\}$.

Exercice 15

Soit la grammaire $G = < \Sigma,\ \{S, T, U\},\ S,\ \mathcal{R} >$, où $\mathcal{R}$ est l'ensemble suivant de règles : $\{\ S \rightarrow aU \mid bUUU,\ T \rightarrow aTb \mid bTa \mid TT \mid \epsilon,\ U \rightarrow aT \mid bS\}$.
Prouver que tout mot de L_G a deux a de plus que de b.

Ambiguïté

Exercice 16

Soit la grammaire d'axiome S dont les règles de production sont $\{S \rightarrow SOS \mid 9,\ O \rightarrow + \mid *\}$. Montrer sur un mot le plus court possible que cette grammaire est ambigüe.

Exercice 17

Soit la grammaire d'axiome S dont les règles de production sont
$\{S \rightarrow$ **si** T **alors** S **sinon** $S \mid$ **si** T **alors** $S \mid$ **a :=0** $\mid$ **a :=1** $\}$. Montrer sur un mot le plus court possible que cette grammaire est ambiguë. Comment a-t-on résolu ce problème dans les langages de programmation ?

Exercice 18

Soit la grammaire $G = < \Sigma = \{a,\ b,\ c,\ d\},\ N = \{S, T, U\},\ S,\ \mathcal{R} >$, où $\mathcal{R}$ est l'ensemble suivant de règles : $\{S \rightarrow aS \mid T,\ T \rightarrow Tb \mid U,\ U \rightarrow cSc \mid d\ \}$.

1. Montrer que d a une et une seule occurrence dans tout mot de L_G
2. Montrer que G n'est pas ambiguë, ou plus précisément que tout mot $m \in L_G$ s'obtient par une seule chaîne de dérivation.

TD 3 : Automates

Constructions d'automates déterministes

Exercice 1
Construire un automate déterministe qui accepte les mots sur $\{a\}^*$ qui ont un nombre pair de a. Justifier que tous les mots acceptés vérifient la propriété.

Exercice 2
Construire un automate déterministe qui accepte les mots sur $\{a,\ b\}^*$ qui ont un nombre pair de a. Justifier que tous les mots acceptés vérifient la propriété.

Exercice 3
Construire un automate déterministe qui accepte les mots sur $\{a\}^*$ qui ont un nombre de a congru à 3 modulo 5. Justifier que tous les mots acceptés vérifient la propriété.

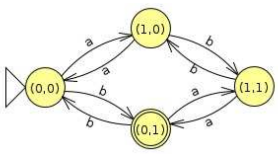
Exercice 4
Construire un automate déterministe qui accepte les mots sur $\{a,\ b\}^*$ qui ont un nombre de a congru à 3 modulo 5. Justifier que tous les mots acceptés vérifient la propriété.

Exercice 5
Dessiner un automate déterministe qui accepte les mots sur $\{a,\ b\}^*$ qui contiennent au moins un facteur aaa suivi ultérieurement d'un facteur bbb (tel $ababaaaababab bbbbba$). Justifier que tous les mots acceptés vérifient la propriété.

Exercice 6
Construire un automate déterministe qui accepte les mots sur $\{a,\ b\}^*$ qui ont un nombre pair de a et un nombre impair de b. Justifier que tous les mots acceptés vérifient la propriété.

Automates déterministes : preuve et théorie

Exercice 7
Soit l'automate $\mathcal{A} = (\Sigma,\ E,\ i,\ F,\ \delta)$, suivant :



1. Montrer que $\delta^*(\,(0,0)\,,\,m) = (|m|_a \, mod\, 2\,,\ |m|_b \, mod\, 2)$.
2. En déduire que le langage associé à cet automate est l'ensemble des mots sur $\{a, b\}$ qui ont un nombre pair de a et un nombre impair de b.

Exercice 8

Soit l'automate $\mathcal{A} = (\Sigma,\ E,\ i,\ F,\ \delta)$, défini par :
$\Sigma = \{a, b\}$ et $E = \{0,\ 1,\ -1,\ 3\}$ et $i = 0$ et $F = \{0\}$
et $\delta(e, \alpha) = \left\{ \begin{array}{lll} |\alpha|_a - |\alpha|_b + e & si & |\alpha|_a - |\alpha|_b + e \in \{0,\ 1,\ -1\} \\ 3 & sinon & \end{array} \right\}$

1. Construire une représentation graphique de cet automate.
2. En notant $p \leq_p m$ le fait que p soit un préfixe de m, prouver que δ^* vérifie :
$\delta^*(0, m) = \left\{ \begin{array}{lll} |m|_a - |m|_b & si & \forall\, p \leq_p m\,,\ |p|_a - |p|_b \in \{0,\ 1,\ -1\} \\ 3 & sinon & \end{array} \right\}$
3. En déduire que le langage associé à cet automate est :
$\{m \in \{a, b\},\ |m|_a = |m|_b$ et $\forall\, p \leq_p m\,,\ |\,|p|_a - |p|_b\,| \leq 1\ \}$

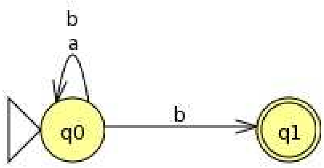
Exercice 9

Démontrer pour un automate déterministe complet $\mathcal{A} = (\Sigma,\ E,\ i,\ F,\ \delta)$ la propriété :
$\forall\, m_1, m_2 \in \Sigma^*\,,\ \forall\, e \in E\,,\ \ \delta^*(e, m_1 m_2) = \delta^*(\delta^*(e, m_1), m_2)$

Exercice 10

Soit un automate déterministe complet $\mathcal{A} = (\Sigma,\ E,\ i,\ F,\ \delta)$.

1. Définir l'automate $\mathcal{A}_c$ qui reconnaît le langage complémentaire de $L(\mathcal{A})$ sur Σ^*. Prouver le résultat.
2. En déduire comment trouver l'automate correspondant au langage complémentaire d'un langage reconnu par un automate déterministe quelconque donné.
3. Rendre complet l'automate **indéterministe** suivant et permuter les états terminaux et non terminaux :



4. Que peut-on en déduire ?

Construction d'automates indéterministes

Exercice 11

Dessiner un automate indéterministe qui accepte les mots sur $\{a,\ b,\ c\}^*$ qui contiennent au moins un facteur *aa* ou un facteur *bb* suivi ultérieurement d'un facteur *cc* (tel *aababaccab* ou *abbcabcca*). Justifier que tous les mots acceptés vérifient la propriété.

Exercice 12

Construire un automate qui accepte les mots sur $\{a,\ b\}^*$ dont au moins un facteur de longueur 5 ne contient pas deux fois la lettre a.
En déduire que le langage des mots sur $\{a,\ b\}^*$ dont tout facteur de longueur 5 contient au moins deux fois la lettre a est un langage rationnel.

Exercice 13

Construire un automate indéterministe complet à cinq états qui accepte les mots sur $\{a,\ b\}^*$ qui ont au moins un facteur *aaaa*

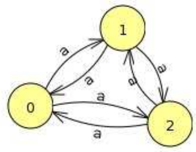
Exercice 14

Construire un automate indéterministe complet à cinq états qui accepte les mots sur $\{a,\ b\}^*$ qui ont au moins un facteur *abab*

Étude d'automates indéterministes

Exercice 15

Soit la partie suivante d'un automate :

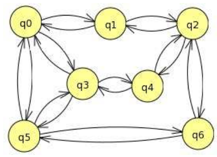


1. Quelle est la plus petite valeur de n telle que :
$\forall i \in \{0,\ 1,\ 2\}\ :\ \delta^*(i, a^n) = \{0,\ 1,\ 2\}$?
2. Plus généralement, pour :
   (a) $\Sigma = \{a\}$
   (b) $E = \{e_0,\ e_1, \ldots, e_{p-1}\}$ où $p \geq 2$
   (c) La fonction de transition δ telle que :
   $\forall i \in [0 \ldots p-1]\ :\ \delta(e_i, a) = \{e_{i+1\, mod\, p}\ ,\ e_{i+2\, mod\, p}\}$.

   A partir de quelle valeur de n a-t-on $\forall i \in [0 \ldots p-1]\ :\ \delta^*(e_i, a^n) = E$?
   On ne demande pas de preuve, seulement une justification.

Exercice 16

Soit l'automate suivant, où toutes les transitions sont toutes étiquetées par la lettre « a » :



Quelle est la plus petite valeur de n telle que $\forall e \in E\ :\ \delta^*(e, a^n) = E$?

Automates indéterministes : théorie

Exercice 17

Soit $\mathcal{A}_1 = (\Sigma,\ E_1,\ I_1,\ F_1,\ \delta_1)$ et $\mathcal{A}_2 = (\Sigma,\ E_2,\ I_2,\ F_2,\ \delta_2)$ deux automates indéterministes avec $E_1 \subseteq E_2$. Prouver par un raisonnement par récurence que :

$$\delta_1 \subseteq \delta_2 \implies \delta_1^* \subseteq \delta_2^*$$

En déduire que : $\delta_1 \subseteq \delta_2\ et\ E_1 \subseteq E_2\ \ et\ \ I_1 \subseteq I_2\ \ et\ \ F_1 \subseteq F_2 \implies L_{\mathcal{A}_1} \subseteq L_{\mathcal{A}_2}$

Exercice 18

Soit $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$ un automate indéterministe. Soit $\mathcal{A}_C = (\Sigma,\ E \cup \{p\},\ I,\ F,\ \delta_c)$ l'automate complété issu de $\mathcal{A}$, où p est un état qui n'existe pas dans E, avec $\delta_c = \delta \cup \{(p, \alpha, p) \mid \alpha \in \Sigma\} \cup \{(e, \alpha, p) \mid \delta(e, \alpha) = \emptyset\}$

1. Prouver que $L_A \subseteq L_{A_c}$
2. Prouver que $L_{A_c} \subseteq L_A$ par une étude sur les chemins.

TD4 Automates avec ϵ-transitions

Construction

Exercice 1

Construire un automate qui reconnaît le langage constitué de l'unique mot $m = abaa$.

En n'ajoutant que des ϵ-transitions à l'automate précédent, sans modifier les états initiaux et terminaux, construire les automates qui reconnaissent les langages suivants :

- ◇ Les préfixes de m.
- ◇ Les suffixes de m.
- ◇ Les facteurs de m.
- ◇ Les sous-mots de m.

Exercice 2

Construire un automate indéterministe avec ϵ-transitions qui reconnaisse le langage $L = \{(ab)^{n_1}(ba)^{n_2}\,,\, n_1, n_2 \in \mathbb{N}\}$ et qui n'ait qu'un seul état terminal.

Raisonnement sur les chemins

Exercice 3

Soit un automate indéterministe $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$. Définir l'automate standard qui reconnaît le langage $L_{\mathcal{A}}$. Prouver votre résultat.

Exercice 4

Soient deux automates indéterministes $\mathcal{A}_1 = (\Sigma,\ E_1,\ I_1,\ F_1,\ \delta_1)$ et $\mathcal{A}_2 = (\Sigma,\ E_2,\ I_2,\ F_2,\ \delta_2)$ (avec $E_1 \cap E_2 = \emptyset$). Définir formellement les automates qui reconnaissent les langages :

- ◇ $L(A_1) \cup L(A_2)$
- ◇ $L(A_1)\ .\ L(A_2)$

Prouver votre résultat.

Exercice 5

Soit $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$ un automate indéterministe.

1. Soit l'automate $\mathcal{A}_* = (\Sigma,\ E,\ I,\ I,\ \delta_*)$ où $\delta_* = \delta \cup \{(f,\, \epsilon,\, e) \,|\, e \in I \,et\, f \in F\}$. Montrer que $L(\mathcal{A})^* = L(\mathcal{A}_*)$ est vrai si l'automate $\mathcal{A}$ est standard.
2. Trouver un contre-exemple pour montrer que $L(\mathcal{A})^* = L(\mathcal{A}_*)$ est faux en général.

Exercice 6

Soit un automate indéterministe $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$. Pour un sous ensemble quelconque $E' \subseteq E$, on définit $\delta_{E'} = \delta \cup \{e'\epsilon e' \mid e' \in E'\}$.
Prouver que $\forall E' \subseteq E$, $\mathcal{A}$ et $\mathcal{A}' = (\Sigma,\ E,\ I,\ F,\ \delta_{E'})$ reconnaissent le même langage.

Exercice 7

Soit un automate indéterministe $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$. Pour tout état $e \in E$, on définit les automates indéterministes $\mathcal{C}_e = (\Sigma,\ E,\ \{e\},\ F,\ \delta)$ et $\mathcal{B}_e = (\Sigma,\ E,\ I,\ \{e\},\ \delta)$.
Pour tout couple d'états $(e', e'') \in E^2$, on définit $\delta_{e'e''} = \delta \cup \{e'\epsilon e''\}$ et $\mathcal{A}_{e'e''} = (\Sigma,\ E,\ I,\ F,\ \delta_{e'e''})$
Montrer que l'on peut ne pas avoir $L(\mathcal{A}_{e'e''}) = L(\mathcal{A}) \cup L(\mathcal{B}_{e'}).L(\mathcal{C}_{e''})$. Quelle inclusion est-elle valide ?

Suppression des ϵ-transitions.

Exercice 8

Soit $\mathcal{A}_\epsilon = (\Sigma,\ E,\ I,\ F_\epsilon,\ \delta_\epsilon)$ un automate indéterministe avec ϵ-transitions et $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$ l'automate indéterministe sans ϵ-transitions déduit de $\mathcal{A}_\epsilon$ et qui reconnaît le même langage.

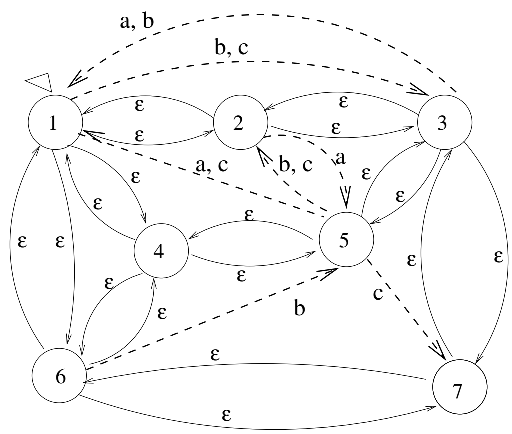
1. Rappeler quelle relation y a-t-il entre les fonctions δ et δ_ϵ et quel est l'ensemble des états terminaux de $\mathcal{A}$? On ne demande pas de preuve.
2. Donner une définition récursive de la fonction δ^* de $\mathcal{P}(E) \times \Sigma^* \ \rightarrow\ \mathcal{P}(E)$ qui à tout ensemble d'états Y et tout mot m fait correspondre l'ensemble des états que l'on peut atteindre grâce au mot m en partant de l'un (au moins) des états de Y.

Exercice 9

Soit un automate indéterministe avec ϵ-transition $\mathcal{A} = (\Sigma,\ E,\ I,\ F,\ \delta)$. Définir l'$\epsilon$-fermeture $\hat{\epsilon}$.

Exercice 10

Combien d'itérations faut-il pour calculer les ϵ-fermetures des états de l'automate suivant ?



Exercice 11

Éliminer les ϵ-transitions dans l'automate du deuxième exercice.

Exercice 12

Calculer un automate (indéterministe) sans ϵ-transition qui reconnaît le même langage que l'automate de la figure .

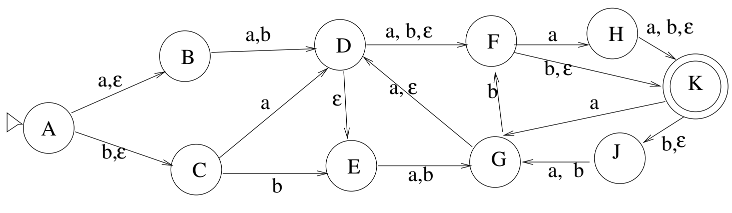


FIGURE 1 – Supprimer les ϵ-transitions

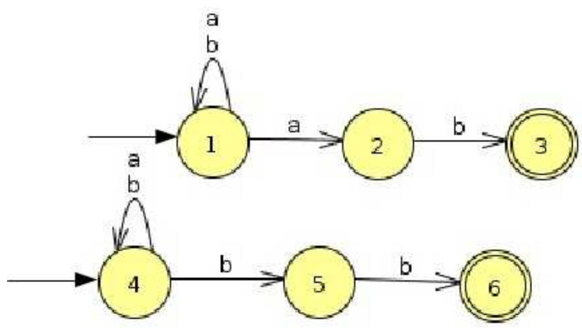
TD 5 : déterminisation d'automates

Exercice 1

1. Construire l'automate indéterministe associé au langage, défini sur l'alphabet $\{a, b\}$, des mots qui terminent par la lettre b.
2. Construire l'automate déterministe associé au langage précédent.
3. Construire l'automate déterministe associé au langage, défini sur l'alphabet $\{a, b\}$, des mots qui ne se terminent pas par la lettre b.

Exercice 2

Déterminiser l'automate suivant :



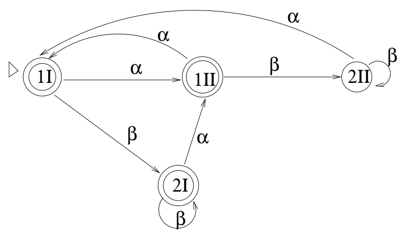
Exercice 3

On veut construire un automate déterministe complet qui accepte tous les mots qui ont au moins un facteur aaa **et** au moins un facteur bbb (peu importe l'ordre respectif de ces deux facteurs). Appliquer les trois approches différentes :

1. Construire directement l'automate en le justifiant par le commentaire de quelques états judicieusement choisis.
2. Construction l'automate à partir d'un automate indéterministe très simple.
3. Construire d'abord l'automate complémentaire avec une construction mêlant unions et complémentaires d'automates.

Exercice 4

L'automate déterministe suivant est le résultat de la déterminisation d'un automate :



Retrouver l'automate indéterministe à 4 états, qui est à l'origine de cette déterminisation.

Exercice 5

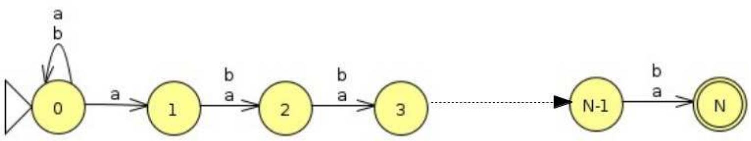
Soit un automate indéterministe sans ϵ-transition $\mathcal{A} = (\Sigma = \{a,\ b,\ c\},\ E,\ I,\ F,\ \delta)$.
On définit la fonction Φ par $\Phi(\mathcal{A}) = (\{a,\ b\},\ E,\ I,\ F,\ \delta_\Phi)$ avec $\forall e \in E\ :\ \delta_\Phi(e,a) = \delta(e,a)$ et $\delta_\Phi(e,b) = \delta(e,b) \cup \delta(e,c)$ et la fonction Det qui, à un automate indéterministe sans ϵ-transition $\mathcal{A}$ associe l'automate déterministe équivalent.

A-t-on $\forall \mathcal{A}\ \Phi(Det(\mathcal{A})) = Det(\Phi(\mathcal{A}))$?

Si oui prouvez le, sinon donnez un contre exemple.

Exercice 6

1. Soit un automate indéterministe à N états. Quel est le nombre d'états de l'automate déterministe correspondant (tel qu'il a été défini en cours) ?
2. Soit $\mathcal{A}$ l'automate suivant :



Et soit $\mathcal{A}_d = (\Sigma = \{a, b\},\ \mathcal{P}(E),\ I,\ F,\ \delta)$ l'automate déterministe construit à partir de $\mathcal{A}$ selon la méthode vue en cours.

1. Notations :
   $\overline{m}$ est le mot m écrit dans l'ordre inverse ($\overline{\alpha.m} = \overline{m}.\alpha$ et $\overline{\epsilon} = \epsilon$).
   $@_m = \{i \mid m[i] = a\}$
   $succ(E') = \{i + 1 \mid i \in E'\}$
2. Justifier les égalités suivantes :

$$\forall\, m \in \Sigma^*,\ \forall\, \alpha \in \Sigma,\ @_{\alpha.m} = @_\alpha \cup succ(@_m)$$
$$\forall\, E' \in \mathcal{P}(\{1, \cdots, N-1\}),\ \forall\, \alpha \in \Sigma,\ \ \delta(E', \alpha) = succ(E')$$
$$\forall\, \alpha \in \Sigma,\ \ \delta(0, \alpha) = \{0\} \cup @_\alpha$$

3. Prouver pour tout entier n que :

$$\forall\, m\ ,\ tel\,que\, |m| \leq n < N,\ \delta^*(\{0\}, m) = \{0\} \cup @_{\overline{m}}$$

4. En déduire un minorant du nombre d'états accessibles générés par la méthode de déterminisation de $\mathcal{A}$.

TD6 : Expressions rationnelles

Construction d'expressions rationnelles

Exercice 1

Sur l'alphabet $\{0, 1\}$, construire les expressions rationnelles correspondant aux langages contenant tous les mots m (et seulement eux) ayant la propriété $P(m)$ suivante :

1. $P(m)$: l'avant dernier symbole de m est 1.
2. $P(m)$: m commence par 0 et finit par 1.
3. $P(m)$: m contient exactement trois occurrences de 0.
4. $P(m)$: m contient au moins trois occurrences de 0.
5. $P(m)$: m contient au plus deux occurrences de 0.
6. $P(m)$: m est l'écriture binaire d'un nombre, sans 0 inutiles en tête.

Exercice 2

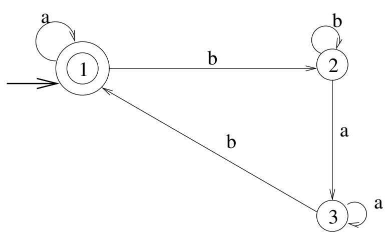
1. Sur l'alphabet $\{0, 1\}$, construire l'automate déterministe complet qui reconnaît les mots contenant le facteur 01.
2. En déduire l'automate qui reconnaît les mots m qui ne contiennent pas le facteur 01.
3. Construire l'expression rationnelle correspondant au langage des mots qui ne contiennent pas le facteur 01.
4. Construire l'expression rationnelle et l'automate correspondant au langage des mots qui contiennent une et une seule fois le facteur 01.

Exercice 3

Deviner une expression rationnelle correspondant au langage de tous les mots (et seulement eux) qui ont au moins un facteur `aba`.
Dessiner un automate déterministe qui reconnaît ce langage. Justifier.
Calculer de deux façons une expression rationnelle correspondant à cet automate.

Exercice 4

Calculer de deux façons une expression rationnelle correspondant au langage complémentaire (sur $\{a,\ b\}^*$) de celui reconnu par l'automate suivant :



Equivalences entre expressions rationnelles

Les égalités entre expressions régulières données ci-après sont à comprendre comme l'égalité des langages associés.

Exercice 5

Montrer que : $(a + b)^* = (a^*b)^*a^* = a^*(ba^*)^*$

Exercice 6

En utilisant la notation $e^+ \underset{déf}{=} e.e^*$:

- ◇ Montrer que $e^* = \epsilon + e^+$ pour toute expression rationnelle e (sachant que $X^+ = X.X^*$ pour tout langage X).
- ◇ Montrer que : $(ab)^+ = a(ba)^*b$.
- ◇ En déduire que : $(ab)^*a = a(ba)^*$ et $(ab+a)^*a = a(ba+a)^*$.

Exercice 7

Démontrer $(a+b)^*b(a+b)^* = (a+b)^*ba^* = a^*b(a+b)^*$

Résiduel

Soit L un langage sur un alphabet Σ et $u \in \Sigma^*$ (u peut ou non appartenir à L). On appelle résiduel de L par rapport à u, et on note $u^{-1}L$, le langage $u^{-1}L = \{m \in \Sigma^* \mid um \in L\}$. D'autre part, on note $L(r)$ le langage correspondant à une expression rationnelle r.

Exercice 8

Prouver que $\forall a \in \Sigma,\ \forall r, s$ deux expressions rationnelles sur Σ^*

- ◇ $a^{-1}L(r+s) = a^{-1}L(r) \cup a^{-1}L(s)$
- ◇ $[\epsilon \notin L(r)] \Rightarrow [a^{-1}L(rs) = (a^{-1}L(r))L(s)]$
- ◇ $[\epsilon \in L(r)] \Rightarrow [a^{-1}L(rs) = (a^{-1}L(r))L(s) \cup a^{-1}L(s)]$
- ◇ $a^{-1}L(r^*) = (a^{-1}L(r))L(r^*)$

Exercice 9

Prouver que $\forall u,\ v \in \Sigma^*,\ \forall L \subseteq \Sigma^*\ :\ (uv)^{-1}L = v^{-1}(u^{-1}L)$.

Exercice 10

Soit l'expression rationnelle $r = (a+b)^*b(a+b)^*$. Pour simplifier l'écriture, on notera $L = (a+b)^*b(a+b)^*$. Calculer $a^{-1}L$ et $b^{-1}L$, en déduire $(aa)^{-1}L$ et $(ab)^{-1}L$.

TD 7 : Minimisation d'automates

Exercice 1

Construire l'automate déterministe minimal correspondant à l'expression régulière $(aa + bb + (ab + ba)(aa + bb)^*(ab + ba))^*$. Caractériser chacun de ses quatre états en indiquant quel serait le langage reconnu par l'automate si c'était cet état qui était un état final.

Exercice 2

En éclatant l'état q_{34} en deux états q_3 et q_4 et l'état q_{125} en trois états q_1, q_2 et q_5, fabriquer un automate **déterministe** complet de **six** états qui reconnaisse le même langage que l'automate de la figure 1.

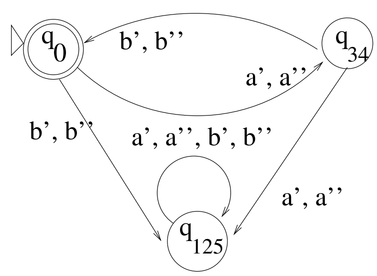


Figure 1 – Rendre non minimal cet automate !

Exercice 3

Soit L un langage sur un alphabet Σ.
Définissons alors, pour tout $\omega \in \Sigma^*$: $L_{/\omega} = \{m \in \Sigma^* \mid \omega m \in L\}$.
Comment a déjà été dénoté $L_{/\omega}$?
Que vaut $L_{/\epsilon}$?
On définit sur Σ^* la relation $\equiv_L$ par : $\omega_1 \equiv_L \omega_2 \iff L_{/\omega_1} = L_{/\omega_2}$.
Cette relation est (évidemment ?) une relation d'équivalence et l'on notera E_L l'ensemble de ses classes d'équivalence.

Soit L le langage sur $\{a, b\}$ des mots finissant par aaa.

1. Prouver que $\forall m \in \Sigma^* \;:\; L_{/mb} = L$.
2. que vaut $L_{/a}$? $L_{/aa}$? $L_{/aaa}$?
3. calculer $\forall m \in \Sigma^* \;:\; L_{/mba}$
4. calculer $\forall m \in \Sigma^* \;:\; L_{/mbaa}$
5. calculer $\forall m \in \Sigma^* \;:\; L_{/mbaaa}$
6. calculer $\forall m \in \Sigma^*,\ \forall n > 3 \in \mathbb{N} \;:\; L_{/ma^n}$

En déduire E_L l'ensemble des classes d'équivalences de $\equiv_L$

Exercice 4

Application de l'exercice précédent.
Soit $L_n = \{m \in \{a,b\}^* \mid m$ a un a comme n^{ieme} lettre en partant de la fin $\}$.
Soient $F = \{w_1, \ldots, w_{2^n}\}$ l'ensemble des 2^n mots de longueur n sur $\{a,b\}^*$.
Montrer que $\forall f_1, f_2 \in F \ : \ f_1 \neq f_2 \Rightarrow f_1 \not\equiv_{L_n} f_2$.

Exercice 5

Suite de l'avant dernier exercice
Supposons maintenant que L soit reconnu par un automate déterministe $\mathcal{A} = (\Sigma,\ E,\ q_0,\ F,\ \delta)$ possédant e états accessibles.
a) Prouver que $\delta(q_0,\, w_1) = \delta(q_0,\, w_2) \implies \omega_1 \equiv_L \omega_2$.
b) Prouver que $|E_L| \leq e$.

Exercice 6

Suite des deux derniers exercices
Montrer qu'il existe un automate indéterministe de $n+1$ états qui ne peut être déterminiser en moins de 2^n états.

Exercice 7

Soit sur un alphabet Σ un langage L dont l'ensemble des résiduels $E = \{m^{-1}L \mid m \in \Sigma^*\}$ est fini.
Pourquoi L est-il un élément de E ?
Posons $\delta = \{(m^{-1}L\,,\, \alpha\,,\, (m\alpha)^{-1}L) \mid m \in \Sigma^*,\ \alpha \in \Sigma\}$.
Pourquoi δ est-il un ensemble fini ?
Posons $F = \{m^{-1}L \mid m \in L\}$. On va prouver que l'automate déterministe $\mathcal{R} = (\Sigma,\ E,\ L,\ F,\ \delta)$ reconnaît L, en prouvant par récurrence sur l la propriété :
$\Pi(l) \ : \ \ \forall p \in L, \forall m \in \Sigma^* \ [|m| \leq l] \ \ \Rightarrow \ \ [\delta^*(p^{-1}L,\, m) \in F \iff m \in p^{-1}L]$.
Démontrer $\forall\, l \in \mathbb{N} \ : \ \Pi(l)$.