

Langages et automates : Polycopié des exercices

1 TD n°1

A désigne un alphabet fini. Le mot vide est noté ε .

1.1 Mots et langages

Exercice 1 : Généralités

1. Compter les occurrences des lettres a et b dans les mots suivants : a^3cbbca , $aabgjdd$, $titi$, $babc$.
2. Donner l'ensemble des couples (u, v) tels que $u \cdot v = abaac$.
3. Un mot u est un *facteur* d'un mot v si u apparaît à l'intérieur de v : v s'écrit $w_1 \cdot u \cdot w_2$ pour certains mots w_1 et w_2 . Un mot u est un *sous-mot* d'un mot v si on peut obtenir u à partir de v par 'effacement' de certaines lettres (pas forcément consécutives) de v . Le nombre d'*occurrences* d'un facteur (resp. sous-mot) u dans le mot v est le nombre de façons de voir u comme facteur (resp. sous-mot) de v .
Donner le nombre d'occurrences du facteur aba dans le mot $v = ababab$. Donner le nombre d'occurrences du sous-mot aba dans le même mot v .

Exercice 2 : Opérations sur les langages

1. Calculer $\mathcal{L} \cdot \mathcal{M}$ pour les ensembles suivants :
 - $\mathcal{L} = \{a, ab, bb\}$ et $\mathcal{M} = \{\varepsilon, b, a^2\}$;
 - $\mathcal{L} = \emptyset$ et $\mathcal{M} = \{a, ba, bb\}$;
 - $\mathcal{L} = \{\varepsilon\}$ et $\mathcal{M} = \{a, ba, bb\}$;
 - $\mathcal{L} = \{aa, ab, ba\}$ et $\mathcal{M} = A^*$.
2. Montrer que le produit est une opération distributive par rapport à l'union, c'est-à-dire que, pour tous langages $\mathcal{L}$, $\mathcal{M}$ et $\mathcal{N}$, on a : $\mathcal{L} \cdot (\mathcal{M} \cup \mathcal{N}) = (\mathcal{L} \cdot \mathcal{M}) \cup (\mathcal{L} \cdot \mathcal{N})$. Montrer que le produit n'est pas distributif par rapport à l'intersection.
3. Parmi les égalités suivantes, lesquelles sont correctes (prouvez ou donnez un contre-exemple) ?
 - $\mathcal{M}^* = \mathcal{M}^* \cdot \mathcal{M}^*$
 - $\mathcal{M}^* = (\mathcal{M} \cdot \mathcal{M})^*$
 - $\mathcal{M}^* = \mathcal{M} \cdot \mathcal{M}^*$
 - $\mathcal{M}^* = (\mathcal{M}^*)^*$
 - $\mathcal{M} \cdot (\mathcal{N} \cdot \mathcal{M})^* = (\mathcal{M} \cdot \mathcal{N})^* \cdot \mathcal{M}$
 - $(\mathcal{M} \cup \mathcal{N})^* = \mathcal{M}^* \cup \mathcal{N}^*$
 - $(\mathcal{M} \cap \mathcal{N})^* = \mathcal{M}^* \cap \mathcal{N}^*$
 - $(\mathcal{M} \cup \mathcal{N})^* = (\mathcal{M}^* \cdot \mathcal{N}^*)^*$
 - $(\mathcal{M} \cup \mathcal{N})^* = (\mathcal{M}^* \cdot \mathcal{N})^* \cdot \mathcal{M}^*$

Exercice 3 : Commutation (pour aller plus loin)

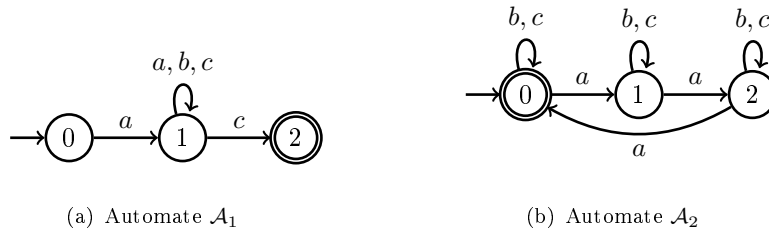
Soient u et v deux mots. On dit que u et v *commutent* si $u \cdot v = v \cdot u$.

Montrer que u et v commutent si et seulement s'il existe un mot w et deux entiers positifs ou nuls m et n tels que $u = w^m$ et $v = w^n$. Pour le sens $\Rightarrow$, on pourra procéder par récurrence sur $|u| + |v|$.

Exercice 4 : Conjugaison (pour aller plus loin)

Deux mots u et v sont dits *conjugués* s'il existe deux mots w_1 et w_2 tels que $u = w_1 \cdot w_2$ et $v = w_2 \cdot w_1$. En d'autres termes, v s'obtient à partir de u par permutation cyclique de ses lettres.

- Montrer que la conjugaison est une relation d'équivalence, c'est-à-dire :
 - tout mot u est conjugué à lui-même ;
 - si u est conjugué à v , alors v est conjugué à u ;
 - si u est conjugué à v et v est conjugué à w , alors u est conjugué à w .
- Montrer que u et v sont conjugués si et seulement s'il existe un mot w tel que $u \cdot w = w \cdot v$.

1.2 Automates**Exercice 5 : Langages reconnus par des automates**

- Décrire pour chacun des deux automates les ensembles d'états initiaux et finaux, et l'ensemble des transitions.
- Les mots abc , $abbc$ et $abacabcc$ sont-ils reconnus par l'automate $\mathcal{A}_1$? Sont-ils reconnus par l'automate $\mathcal{A}_2$?
- Décrire les langages reconnus par chacun des automates.

Exercice 6 : Construction d'automates

Montrer que les langages suivants sont reconnaissables en donnant pour chaque langage un automate qui le reconnaît :

- $\mathcal{L}_1 = \{u \in A^* : \text{toute occurrence de } b \text{ dans } u \text{ est immédiatement suivie d'au moins deux occurrences de } a\}$,
- $\mathcal{L}_2 = \{u \in A^* : u \text{ ne contient pas deux } a \text{ successifs}\}$,
- $\mathcal{L}_3 = \{u \in A^* : \text{le nombre d'occurrences de } a \text{ dans } u \text{ est pair}\}$,
- $\mathcal{L}_4$ l'ensemble des mots sur l'alphabet $\{/, \star, a, LF\}$ correspondant à des commentaires en C^1 . Tout commentaire soit :
 - Commence par $//$, finit par un saut de ligne LF .
 - Commence par $/\star$ et fini par $\star/$. Le commentaire peut contenir des $/$ et des $\star$, mais il ne peut pas contenir le motif $\star/$.

Exemples :

$/\star\star aaaaa/\star LF aaa \star/$

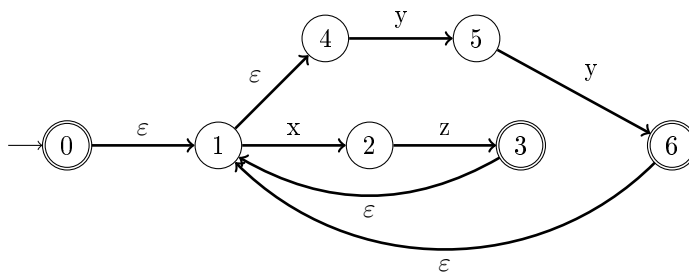
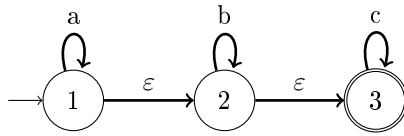
$//aa/\star\star aaa \star/aaa \star/LF$

- $\mathcal{L}_5 = \{u \in A^* : \text{les blocs de } a \text{ dans } u \text{ sont alternativement de longueur paire et impaire}\}$.

1. La lettre a représentera tout autre caractère ; LF est un seul caractère : le saut de ligne (Line Feed), de code ASCII 10.

Exercice 7 : ε -transitions

Éliminer les ε -transitions des automates suivants en appliquant l'algorithme vu en cours. Émonder les automates obtenus.



2 TD n°2 : déterminisation

Exercice 8 : Déterminisation

Déterminiser les automates suivants :

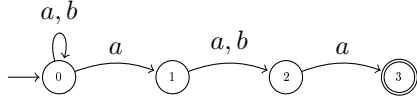


FIGURE 1 – Automate $\mathcal{A}_1$

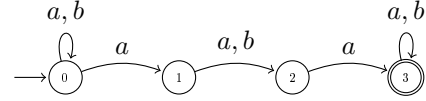


FIGURE 2 – Automate $\mathcal{A}_2$

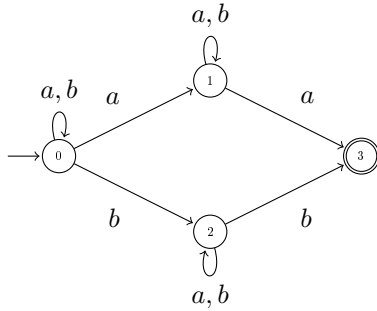


FIGURE 3 – Automate $\mathcal{A}_3$

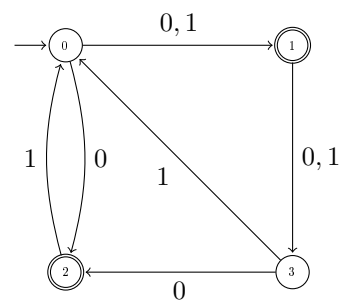


FIGURE 4 – Automate $\mathcal{A}_4$

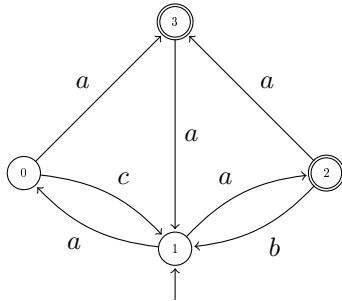


FIGURE 5 – Automate $\mathcal{A}_5$

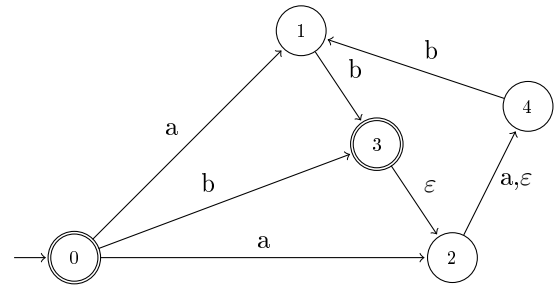


FIGURE 6 – Automate $\mathcal{A}_6$ (examen 2016)

Exercice 9 :

Construire un automate déterministe pour chacun des deux langages suivants :

- le langage $\mathcal{L}_1$ formé des mots contenant le facteur aba ;
- le langage $\mathcal{L}_2$ formé des mots terminant soit par le suffixe aa , soit par le suffixe abb .

Exercice 10 : Préfixes, suffixes, facteurs, sous-mots

Soit le langage fini composé d'un seul mot $\mathcal{L} = \{\text{maman}\}$.

1. A partir de l'automate reconnaissant ce mot, construire des automates pour les langages suivants, et les déterminer :
 - (a) le langage $\mathcal{L}_{pref}$ des préfixes de $\mathcal{L}$.
 - (b) le langage $\mathcal{L}_{suff}$ des suffixes de $\mathcal{L}$.

- (c) le langage $\mathcal{L}_{fact}$ des facteurs de $\mathcal{L}$.
 - (d) le langage $\mathcal{L}_{sous}$ des sous-mots de $\mathcal{L}$.
2. Les constructions pour les préfixes et suffixes peuvent être généralisées, mais ne marchent pas pour tous les automates. Donner un exemple d'automate pour lequel la construction faite pour le langage des préfixes ne marche pas (donne un automate qui reconnaît un langage différent de celui des préfixes).

Exercice 11 : Digicode

On veut écrire 2 automates déterministes qui reconnaissent l'entrée du "mot de passe" d'un digicode. Il n'y a que des chiffres possibles en entrée. Le code est 11654.

1. Construire un automate qui arrive dans un état final pour toute séquence tapée qui finit par le bon code.
2. Construire un automate qui lit un code de taille 5, l'accepte si c'est le bon, refuse sinon, et permet ensuite de retenter sa chance.

Exercice 12 : Questions supplémentaires (*pour s'entraîner*)

1. Reprendre l'exercice 3 avec le langage $\mathcal{L}' = \{\text{papa, maman}\}$.
2. Montrer que si un langage $\mathcal{L}$ est reconnu par un automate émondé, alors l'automate obtenu en rendant tous les états acceptants reconnaît l'ensemble des préfixes des mots de $\mathcal{L}$. Trouver un exemple d'automate non émondé pour lequel cette propriété est fausse.

3 TD n°3

Exercice 13 : Union, intersection, complémentaire ...

Soit $\Sigma = \{a, b\}$, et soient deux langages

$$\mathcal{L}_1 = \{u \in \Sigma^* : |u| \equiv 0 \pmod{3}\}$$

$$\text{et } \mathcal{L}_2 = \{u \in \Sigma^* : u \text{ ne contient pas le facteur } a^2\}.$$

En utilisant les constructions vues en cours, construire des automates reconnaissant les langages suivants :

$$\mathcal{L}_1 \cap \mathcal{L}_2 = \{u \in \Sigma^* : |u| \equiv 0 \pmod{3} \text{ et } u \text{ ne contient pas le facteur } a^2\}$$

$$\mathcal{L}_1 \cup \mathcal{L}_2 = \{u \in \Sigma^* : |u| \equiv 0 \pmod{3} \text{ ou } u \text{ ne contient pas le facteur } a^2\}$$

$$\mathcal{L}_2 \setminus \mathcal{L}_1 = \{u \in \Sigma^* : |u| \not\equiv 0 \pmod{3} \text{ et } u \text{ ne contient pas le facteur } a^2\}$$

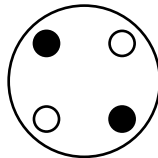
Exercice 14 : Langage Miroir

Le langage miroir d'un langage est le langage $\tilde{\mathcal{L}} = \{\tilde{u}, u \in \mathcal{L}\}$, où $\tilde{u} = x_n \dots x_1$ si $u = x_1 \dots x_n$.

1. Décrire un procédé permettant de construire l'automate reconnaissant $\tilde{\mathcal{L}}$ connaissant celui de $\mathcal{L}$.
2. Calculer ainsi le langage miroir du langage $\mathcal{L}$ formé des mots commençant par *baa*

Exercice 15 : Problème du barman aveugle

On dispose 4 verres sur un plateau, disposés aux quatre sommets d'un carré. Chaque verre peut être soit à l'endroit, soit à l'envers. Par exemple avec deux verres à l'endroit, deux verres à l'envers, on pourra avoir la configuration :



On joue au jeu suivant avec un barman aveugle, avec des gants de boxe : il ne peut pas savoir/sentir dans quel sens sont les verres.

- A chaque tour, le barman retourne 1 ou 2 verres de son choix.
- Ensuite, on fait tourner le plateau de 0, 1/4, 1/2 ou 3/4 de tours, sans que le barman sache quelle rotation on a fait.

Problème : quelle stratégie peut suivre le barman pour arriver à mettre tous les verres dans le même sens (soit tous retournés, soit tous à l'endroit) ?

Nous allons répondre à ce problème à l'aide d'automates.

1. Proposer des états pour modéliser les différentes situations. L'idée est d'avoir *le moins d'état possible* : en particulier, comme le barman ne maîtrise pas quelle rotation a été faite, on définira les situations *a rotation près*.
2. Quels sont les différents *coups* que peut jouer le barman ? Encore une fois, on ne définit les coups qu'à une rotation près.

Complétez l'automate en ajoutant, pour chaque état, la liste de toutes les transitions possibles pour chaque coup.

3. Déterminer l'automate obtenu. En déduire une stratégie gagnante pour le barman aveugle.

Exercice 16 :

Donner des expressions rationnelles décrivant les langages ci-dessous :

- $_1 = \{u \in A^* : \text{toute occurrence de } b \text{ est immédiatement suivie de deux occurrences de } a\}$,
- $_2 = \{u \in A^* : u \text{ ne contient pas deux } a \text{ successifs}\}$,
- $_3 = \{u \in A^* : \text{le nombre d'occurrences de } a \text{ dans } u \text{ est pair}\}$,
- $_4 = \{u \in A^* : \text{les blocs de } a \text{ dans } u \text{ sont alternativement de longueur paire et impaire}\}$.

3.1 Passage d'une expression rationnelle à un automate**Exercice 17 : Expression Rationnelle vers Automate**

Répondre aux questions suivantes pour les expressions rationnelles $(a^*b^*)^*$, $b(ab)^* + (ba)^*b$ et $(a+bb)^*(b+aa)^*$:

1. Donner l'automate obtenu par les constructions de Thompson correspondant à cette expression rationnelle,
2. Supprimer les ε -transitions. On pourra utiliser l'algorithme vu en cours, ou éliminer les ε -transitions une à une.
3. Déterminer l'automate obtenu.
4. L'automate obtenu après détermination vous semble-t-il être minimal ?

3.2 Lemme de l'étoile**Exercice 18 : Lemme de l'étoile (pour aller plus loin)**

On rappelle le lemme de l'étoile :

Lemme de l'étoile

Soit L un langage rationnel. Alors il existe une constante N telle que tout mot w de longueur supérieure à N peut se décomposer en $w = xyz$ tel que

1. $0 < |y|$
2. $|xy| \leq N^2$
3. $\forall i \in \mathbb{N}, xy^iz \in L$.

Parmi les langages suivants, lesquels sont reconnaissables ? Justifier.

Méthode :

- **Pour montrer qu'un langage est reconnaissable**, on peut donner un automate (déterministe ou non) ou une expression rationnelle, puisque c'est équivalent (même si ce n'est pas encore démontré en cours).
- **Pour montrer qu'un langage n'est pas reconnaissable**, on peut montrer qu'il ne vérifie pas le lemme de l'étoile, ou montrer que si on le reconnaissait, on pourrait reconnaître un langage dont on sait déjà qu'il n'est pas reconnaissable.

1. $\mathcal{L}_1 = \{a^n b^n \mid n \in \mathbb{N}\}$
2. $\mathcal{L}_2 = \{a^p b^q \mid (p, q) \in \mathbb{N}^2\}$
3. $\mathcal{L}_3 = \{a^p b^q \mid p \geq q\}$
4. $\mathcal{L}_4 = \{a^p b^q \mid p \geq q \text{ et } q \leq 5\}$

2. On peut aussi supposer $|xy| \geq N$. Pourquoi ?

5. $\mathcal{L}_5 = \{a^p b^q \mid p \neq q\}$
6. $\mathcal{L}_6 = \{u \in A^* \mid |u|_a = |u|_b\}$
7. $\mathcal{L}_7 = \{u \in A^* \mid |u|_a < |u|_b\}$
8. $\mathcal{L}_8 = \{u \in A^* \mid |u|_a \equiv |u|_b \pmod{7}\}$
9. $\mathcal{L}_9 = \{a^n \mid n \text{ est un nombre premier}\}$
10. $\mathcal{L}_{10} = \{uu \mid u \in A^*\}$
11. $\mathcal{L}_{11} = \{u \in A^* \mid u \text{ est un palindrome}\}$
12. $\mathcal{L}_{12}$ l'ensemble des mots sur l'alphabet $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ représentant des nombres divisibles par 3^3 .
13. $\mathcal{L}_{13}$ l'ensemble des expressions bien parenthésées sur l'alphabet $\{a, (,), \}$.

3. on rappelle qu'un nombre est divisible par 3 si la somme de ses chiffres est divisible par 3 ; par exemple, 273 est divisible par 3 car $2 + 7 + 3 \equiv 12 \pmod{3}$. Question tirée de l'examen 2017

4 TD n°4 : équivalence langages reconnus/langages rationnels

4.1 Passage d'un automate à une expression rationnelle

On rappelle le lemme d'Arden

Lemme d'Arden

Soient A et B deux langages, tels que $\varepsilon \notin A$. L'équation $L = AL + B$ d'inconnue L a pour unique solution le langage $L = A^*B$.

Exercice 19 : Echauffement

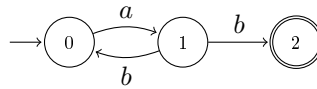
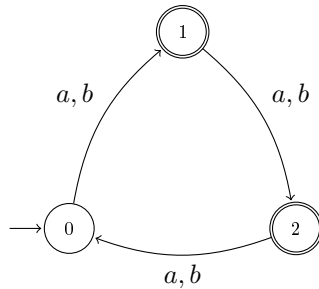


FIGURE 7 – Automate $\mathcal{A}_0$

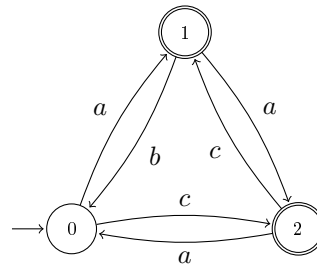
1. Ecrire les équations des langages L_0 , L_1 , L_2 reconnus par $\mathcal{A}_0$ en commençant dans l'état 0, 1 et 2 respectivement.
2. Résoudre de deux façons ce système d'équations : en appliquant le lemme d'Arden au langage L_0 , puis au langage L_1 . En déduire deux expressions rationnelles pour le langage reconnu par $\mathcal{A}_0$.

Exercice 20 : Automate vers Expression Rationnelle

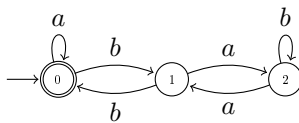
Grâce au Lemme d'Arden, donnez une expression régulière représentant les langages reconnus par les automates suivants.



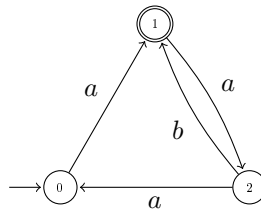
(a) Automate $\mathcal{A}_1$



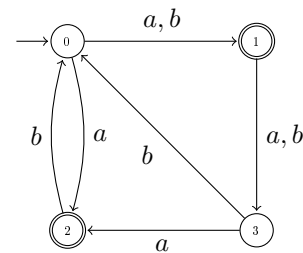
(b) Automate $\mathcal{A}_2$



(c) Automate $\mathcal{A}_3$



(d) Automate $\mathcal{A}_4$



(e) Automate $\mathcal{A}_5$

$\mathcal{A}_6$	0	1	2
a	$\emptyset$	$\{0\}$	$\{1, 2\}$
b	$\{2\}$	$\{0, 1\}$	$\emptyset$
initial	oui	non	non
terminal	oui	non	non

$\mathcal{A}_7$	0	1	2	3	4	5
a	$\{4, 3\}$	$\{4, 2\}$	$\{0\}$	$\{1, 5\}$	$\emptyset$	$\{4\}$
b	$\{1\}$	$\{5\}$	$\{1, 2\}$	$\emptyset$	$\{4\}$	$\{3\}$
initial	oui	non	non	non	oui	non
terminal	non	non	non	oui	non	oui

4.2 Propriétés

Exercice 21 : Conséquences

Pourquoi peut-on dire que

1. L'intersection de deux langages rationnels est rationnelle ?
2. L'ensemble des préfixes d'un langage rationnel forme un langage rationnel ?
3. Le complémentaire d'un langage rationnel est rationnel ?

5 TD n°5 : Minimisation

Exercice 22 : Minimisation d'un automate

Pour chacun des automates suivants :

1. Calculer l'équivalence de Nérde.
2. En déduire l'automate minimal équivalent.

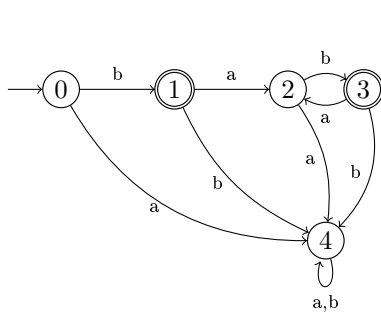


FIGURE 8 – Automate $\mathcal{A}_1$

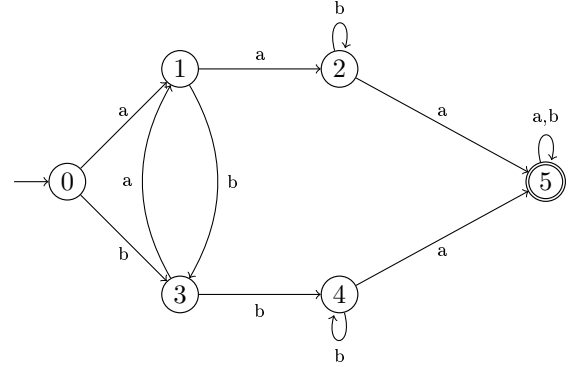


FIGURE 9 – Automate $\mathcal{A}_2$

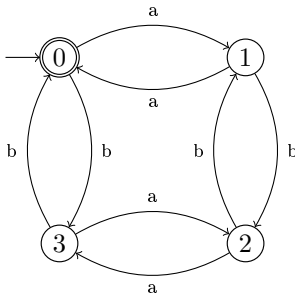


FIGURE 10 – Automate $\mathcal{A}_3$

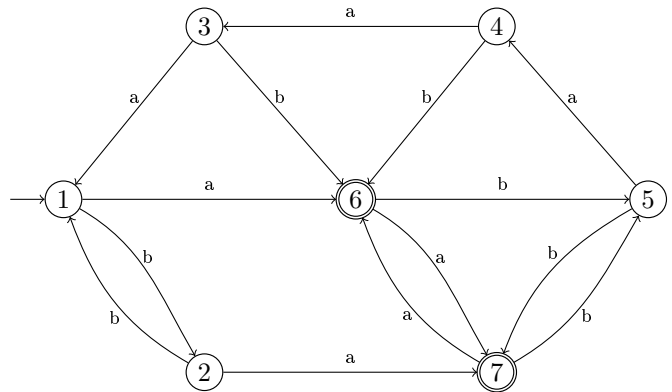


FIGURE 11 – Automate $\mathcal{A}_4$

Exercice 23 : Application : égalité d'expressions rationnelles

On veut vérifier "automatiquement" que les langages $\mathcal{L}_1 = (ab)^*a$ et $\mathcal{L}_2 = a(ba)^*$ sont égaux.

1. Construire des automates déterministes complets pour $\mathcal{L}_1$ et $\mathcal{L}_2$. On pourra construire les automates de thompson, les synchroniser, les déterminer et les compléter.
2. Minimiser les automates obtenus.
3. Conclure.
4. Montrer similairement que les langages $\mathcal{L}_3 = (a^*ba^*)^*$ et $\mathcal{L}_4 = (a+b)^*$ sont différents

6 TD n°6 : minimisation des automates et résiduels

Exercice 24 : Résiduels et automate minimal : échauffement

Le résiduel $u^{-1}.L$ d'un langage L par rapport à un mot u est le langage $\{v \mid uv \in L\}$. Les résiduels des langages rationnels vérifient les propriétés suivantes :

- $a^{-1}.\emptyset = \emptyset$
- $a^{-1}.\varepsilon = \emptyset$
- $a^{-1}.b = \emptyset$, si $b \neq a$
- $a^{-1}.a = \varepsilon$
- $(au)^{-1}.L = u^{-1}.(a^{-1}.L)$
- $u^{-1}.(L_1 + L_2) = (u^{-1}.L_1) + (u^{-1}.L_2)$
- $a^{-1}.(L_1 \cdot L_2) = \begin{cases} (a^{-1}.L_1) \cdot L_2 & \text{si } \varepsilon \notin L_1 \\ (a^{-1}.L_1) \cdot L_2 + a^{-1}.L_2 & \text{si } \varepsilon \in L_1 \end{cases}$
- $a^{-1}.(L^*) = (a^{-1}.L) \cdot L^*$

On considère le langage $L = ba^* + ab$:

1. Calculer les résiduels $a^{-1}.L$ et $b^{-1}.L$.
2. Calculer $(ab)^{-1}.L$, $(aa)^{-1}.L$, $(bb)^{-1}.L$ et $(ba)^{-1}.L$.
3. Calculer les résiduels de L par rapport aux mots sur $\{a, b\}$ de longueur 3. Est-il nécessaire de calculer les résiduels par rapport aux mots de longueur 4 ?
4. Construire l'automate des résiduels à gauche A_L suivant les règles suivantes :
 - Les états de A sont les *différents* résiduels non vides de L par rapport aux mots sur $\{a, b\}$.
 - L'état associé à $\varepsilon^{-1}.L = L$ est l'état initial.
 - Il y a une transition $(u)^{-1}.L \rightarrow (v)^{-1}.L$ étiqueté avec x si et seulement si $v = ux$.
 - Un état de A est final si le résiduel associé contient ε .

Exercice 25 : Résiduels et automate minimal

- Construire l'automate minimal des langages suivants en utilisant le calcul de résiduels :
 1. $L_1 = b(ab)^* + (ab)^*a$
 2. $L_2 = a(b + ab)^* + b^*(a + bb)$

Exercice 26 : Minimisation d'un automate : autre méthode

Pour chacun des automates à minimiser du TD précédent, appliquer successivement les transformations

1. miroir
2. déterminisation
3. miroir
4. déterminisation

Vérifier qu'à chaque fois, $\det(\text{mir}(\det(\text{mir}(A))))$ donne l'automate déterministe minimal équivalent à A .

Exercice 27 : Langages reconnaissables : caractérisation par les résiduels

Les résiduels peuvent aussi permettre de déterminer quels langages sont reconnaissables ou pas : un langage est reconnaissable par automate si et seulement si il a un nombre fini de résiduels.

Parmi les langages suivants, lesquels sont reconnaissables ?

1. $\mathcal{L} = \{a^n b^n \mid n \in \mathcal{N}\}$
2. $\mathcal{L} = \{a^p b^q \mid (p, q) \in \mathcal{N}^2\}$
3. $\mathcal{L} = \{a^p b^q \mid p \geq q\}$
4. $\mathcal{L} = \{a^p b^q \mid p \geq q \text{ et } q \leq 5\}$
5. $\mathcal{L} = \{a^p b^q \mid p \neq q\}$
6. $\mathcal{L} = \{u \in A^* \mid |u|_a = |u|_b\}$
7. $\mathcal{L} = \{u \in A^* \mid |u|_a < |u|_b\}$
8. $\mathcal{L} = \{u \in \Sigma^* \mid |u|_a \equiv |u|_b \pmod{7}\}$
9. $\mathcal{L} = \{a^n b a^n \mid n \in \mathcal{N}\}$
10. $\mathcal{L} = \{a^n \mid n \text{ est un nombre premier}\}$
11. $\mathcal{L} = \{a^{2^n} \mid n \in \mathcal{N}\}$
12. $\mathcal{L} = \{b a b a^2 b a^3 \dots b a^n \mid n \geq 0\}$
13. $\mathcal{L} = \{u u \mid u \in A^*\}$
14. $\mathcal{L} = \{u \in A^* \mid u \text{ est un palindrome}\}$

Exercice 28 : Automate pas si minimal (pour aller plus loin)

On considère le langage $\mathcal{L} = (a + b)^* a (a + b)^{n-1}$.

1. Expliquer (en une phrase) quel est le langage $\mathcal{L}$.
2. Construire un automate fini (non déterministe) à $n + 1$ états reconnaissant $\mathcal{L}$.
On se propose de montrer que tout automate fini complet déterministe reconnaissant $\mathcal{L}$ possède au moins 2^n états.
3. Déterminer, pour tout $u \in A^n$, le résiduel à gauche $u^{-1}\mathcal{L}$. Vérifier que ces résiduels sont tous différents.
4. Conclure que l'automate déterministe minimal de $\mathcal{L}$ a au moins 2^n états.

7 TD n°7 : grammaires algébriques

Exercice 29 : Échauffement

- On considère la grammaire suivante :
 - Terminaux : $\{1, 2, \dots, 9, +, *, (,)\}$
 - Non terminaux : $\{S, N\}$
 - Axiome : S
 - Productions :

$$S \rightarrow (S + S) | (S * S) | N$$

$$N \rightarrow 0N | 1N | \dots | 9N | 0 | \dots | 9.$$

- Décrire le langage généré par cette grammaire, donner quelques exemples de construction.
- Modifier cette grammaire pour que les nombres soient bien formés : soit 0, soit un premier chiffre différent de 0.
- Donner des grammaires algébriques pour le langage suivant :
- $\{a^n b^n : n \in \mathbb{N}\}$.

Exercice 30 : Retour sur les langages rationnels ou non

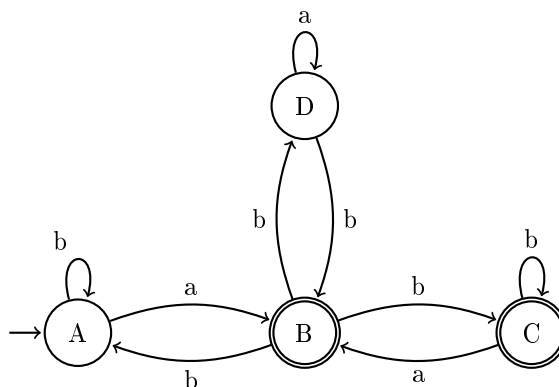
- Parmi les langages du dernier exercice du TD 6 :
 - rappeler si ils sont rationnels.
 - proposer une grammaire algébrique pour ces langages si cela vous semble possible.

Attention, ce n'est pas le cas de tous les langages ! Certains ne sont pas algébriques.

Exercice 31 : Rationnel $\subset$ Algébrique

Nous avons sous-entendu jusqu'ici que les langages algébriques formaient une classe plus large que les langages rationnels. Nous allons le prouver.

- Partons d'un automate exemple :



- Donner une grammaire algébrique engendrant le langage reconnu par un automate.
- A partir de n'importe quel automate, décrire une grammaire qui engendre le langage reconnu par cet automate.
- La grammaire en question est de forme assez particulière, que l'on pourra préciser : une telle grammaire est appelée grammaire rationnelle. Les grammaires rationnelles permettent de reconnaître exactement les langages rationnels.*

7.1 Ambiguïté

Une grammaire est *ambigüe* si il existe deux arbres de construction différents donnant le même mot de terminaux.

Par exemple, $S \rightarrow aB|Ab$

$A \rightarrow a$

$B \rightarrow b$

est ambigüe, et a deux constructions pour le mot ab :

$S \rightarrow aB \rightarrow ab$

et

$S \rightarrow Ab \rightarrow ab$

Exercice 32 : Aucune ambiguïté

1. Montrer que la grammaire de règles de production $S \rightarrow SS|(S)|\varepsilon$ est ambigue.
2. Vérifier (sans preuve formelle) que la grammaire d'axiôme B , et de productions

$$\begin{aligned} B &\rightarrow (RB|\varepsilon \\ R &\rightarrow)|(RR \end{aligned}$$

vérifie le même langage, et n'est pas ambigue.

7.2 Et au-delà des langages algébriques ? (pour aller plus loin)

Une **grammaire contextuelle** a le droit à des productions du type : $uXv \rightarrow u\alpha v$, où α, u, v sont des mots sur $N \cup T$.

Exercice 33 :

1. Remarquer qu'avec des symboles intermédiaires, il est possible de "simuler" des règles du type : $AB \rightarrow BA$.
2. Proposer une grammaire contextuelle pour $\{a^{2^n} : n \in \mathbb{N}\}$.
★ On pourra imaginer une production permettant de doubler le nombre de A d'un mot.
3. Proposer une grammaire contextuelle pour $\{a^n b^n c^n : n \in \mathbb{N}\}$

Nous verrons par la suite que ces langages ne sont pas algébriques !

8 TD n°8 : grammaires, simplifications

8.1 Constructions de grammaires

Exercice 34 :

Proposer (sans trop justifier) des grammaires algébriques pour les langages suivants :

1. (Rappel TD 1) le langage des palindromes sur $\{a, b\}$.
2. $\{a^n b^m c^m d^p : (n, m, p) \in \mathbb{N}^3 \wedge n \geq p\}$.
3. Le langage des mots qui ne sont pas des palindromes sur $\{a, b\}$.
4. $L = \{ww' \in \{a, b\}^* : |w| = |w'| \wedge w \neq w'\}$
On pourra utiliser que L est constitué de mots ww' avec w et w' qui diffèrent au moins d'une lettre : il existe k_1, k_2 tel que

$$w \in \Sigma^{k_1} a \Sigma^{k_2}, w' \in \Sigma^{k_1} b \Sigma^{k_2},$$

ou le contraire :

$$w \in \Sigma^{k_1} b \Sigma^{k_2}, w' \in \Sigma^{k_1} a \Sigma^{k_2}.$$

Mais alors, par exemple dans le premier cas,

$$ww' \in \Sigma^{k_1} a \Sigma^{k_2} \cdot \Sigma^{k_1} b \Sigma^{k_2} = \Sigma^{k_1} a \Sigma^{k_1} \cdot \Sigma^{k_2} b \Sigma^{k_2}$$

qui est facile à générer.

5. Sur l'alphabet $\{a, +, =\}$, l'ensemble des égalités correctes entre deux suites de caractères a : par exemple, $aa + aaa = aaaaa$ est dans le langage.

Exercice 35 : Expressions arithmétiques sans ambiguïté

1. Donnez une grammaire sur $\{(\,, \,), \, +, \, *, \, 0, \, \dots, \, 9\}$ pour les expressions arithmétiques, avec parenthèses ou pas.
2. Montrez que cette grammaire est ambiguë.
3. Proposez une grammaire sur $\{(\,, \,), \, +, \, *, \, 0, \, \dots, \, 9\}$ sans ambiguïté, telle que $*$ est toujours prioritaire sur $+$.
Par exemple, $1 + 1 * 1$ et $1 * 1 + 1$ ont tous deux été construits avec $+$ avant $*$ dans l'arbre : il ne sera pas possible de produire $1 + 1 * 1$ en "commençant" par une règle faisant apparaître $*$, et donc de créer cette expression avec le sens : $(1+1)*1$.

On pourra introduire des symboles S , P et T pour des sommes, produits et termes.

8.2 Simplification des grammaires algébriques

Exercice 36 : Symboles inaccessibles/inactifs

Débarrasser la grammaire suivante de ses symboles non terminaux inutiles (ie dans aucune dérivation qui produit un mot du langage).

- $S \rightarrow A|B$
- $A \rightarrow aB|bS|b$

- $B \rightarrow AB|BaC$
- $C \rightarrow AS|b$.

Exercice 37 : Forme normale de Chomsky

1. Transformer sous forme normale de Chomsky la grammaire suivante.

$$\begin{aligned}S &\rightarrow AB|abaS|a \\ A &\rightarrow Ab|\varepsilon \\ B &\rightarrow AS\end{aligned}$$

Pour cela, on :

- a) Supprimera les éventuels états inutiles
 - b) Supprimera les ε -productions (de la forme $A \rightarrow \varepsilon$)
 - c) Supprimera les productions unité (de la forme $A \rightarrow B$).
 - d) Passera en forme Normale de Chomsky.
2. Transformer sous forme normale de Chomsky la grammaire suivante.

$$\begin{aligned}S &\rightarrow CSC|aB \\ C &\rightarrow B|S \\ B &\rightarrow b|\varepsilon.\end{aligned}$$

9 TD n°9 : automates à pile

9.1 Constructions d'automates à pile

Exercice 38 : Automates à pile

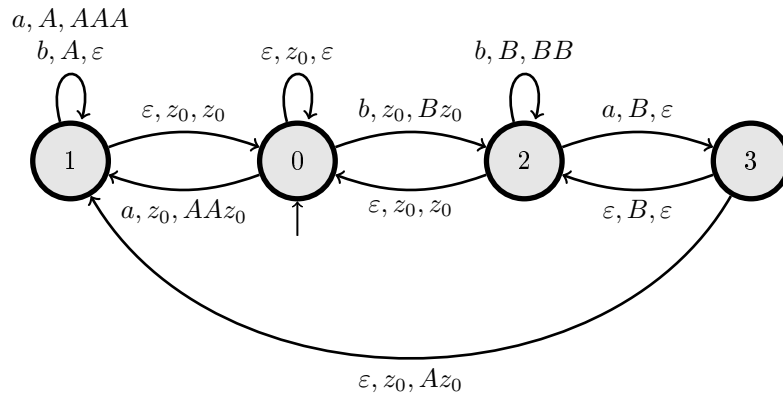
Pour chacun des langages suivants, donner un automate à pile le reconnaissant. Dire si l'automate est déterministe.

On pourra faire la trace de calculs sur quelques mots choisis pour vérifier l'exécution.

1. $\{w \in \{a, b, c\}^* : |w|_a = |w|_b\}$.
2. $\{va^n : n \in \mathbb{N}, v \in \{a, b\}^*, |v|_a = n\}$.
3. $\{a^n b^m : (n, m) \in \mathbb{N}^2, n \leq m \leq 2n\}$.
4. $\{w \in \{a, b\}^* : w = \text{miroir}(w)\}$ le langage des palindromes.
5. $\{w \in \{a, b\}^* : w \neq \text{miroir}(w)\}$ le langage des non palindromes.
6. Le langage des mots bien parenthésés avec $[,], (,), ,$.

Exercice 39 : Automate mystère

On considère l'automate suivant.



1. Suivre la trace d'un calcul sur l'entrée $abbbba$, sur l'entrée $baaa$. Ces mots sont-ils acceptés ?
2. Décrire le langage des mots acceptés par cet automate.

Exercice 40 : Traduction automatique

1. Donner une grammaire algébrique reconnaissant le langage :

$$\{a^n b^{2n} : n \in \mathbb{N}\} \cup \{a^{2n} b^n : n \in \mathbb{N}\}$$

2. Passer la grammaire sous forme normale de Greibach (on pourra enlever ε du langage reconnu).

On rappelle qu'une grammaire est sous forme normale de Greibach si toutes les productions sont de la forme $A \rightarrow bCD\dots : b \in T, A, C, D, \dots \in N$.

3. Suivre la méthode vue en CM pour construire un automate à pile pour le même langage.

9.2 Retour sur les grammaires (si il y a le temps)

Exercice 41 : Retour sur les grammaires (*si il y a le temps*)

On considère la grammaire $\{N, T, S, P\}$ définie par $N = \{S, A, B, C, D\}$, $T = \{a, b, d\}$, P contenant les productions :

$$\begin{aligned} S &\rightarrow AAB|BB|C \\ A &\rightarrow Aa|\varepsilon \\ B &\rightarrow Bb|BC|\varepsilon \\ C &\rightarrow CD \\ D &\rightarrow Dd|DC|d. \end{aligned}$$

Soit L le langage reconnu.

1. Réduire la grammaire : supprimer les non terminaux inutiles.
2. Le mot vide est-il dans L ?
3. Trouver une grammaire équivalente sans ε -productions (sauf $S_0 \rightarrow \varepsilon$).
4. Trouver une grammaire équivalente sans productions unité.
5. Trouver une grammaire équivalente sous Forme Normale de Chomsky.
On peut ajouter des non terminaux $X \rightarrow a$ et $Y \rightarrow b$, et on remplace les a et les b par X et Y dans les productions de longueur ≥ 2 .
6. En déduire l'ensemble des mots de longueur ≤ 3 reconnu par cette grammaire.

9.3 Simplifications d'automates (pour aller plus loin)

Exercice 42 : Variantes des automates à pile (*pour aller plus loin*)

On peut faire varier un peu la définition des automates à pile :

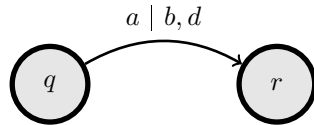
1. ε -transitions : transitions sans lecture de lettre du mot d'entrée, du type $p, \varepsilon, A \rightarrow q, W$.
Peut-on se passer de ces ε -transitions dans un automate :
 - (a) Pour un langage qui contient le mot vide ?
 - (b) Pour un langage qui ne contient pas le mot vide ?
2. Etats d'acceptation : certains états sont finaux, le calcul est acceptant si on finit dans un état final (peu importe le contenu de la pile).
Traduire certains des automates de l'exercice 1 en automates à pile avec états d'acceptation (éventuellement avec ε -transition).
3. autoriser les transitions sans dépilement, où on dépile ε : du type $p, a, \varepsilon \rightarrow q, W$.
Montrer que ces transitions sont toujours remplaçables par des transitions classiques.

10 TD n°10 : machine de Turing

10.1 Machines de Turing

Une machine de Turing est constituée :

- d'un automate,
- d'un ruban de cases infini des deux côtés,
- d'une tête de lecture/écriture sur le ruban : à chaque transition, la tête de lecture peut écrire un caractère, et se déplacer d'une case sur le ruban.

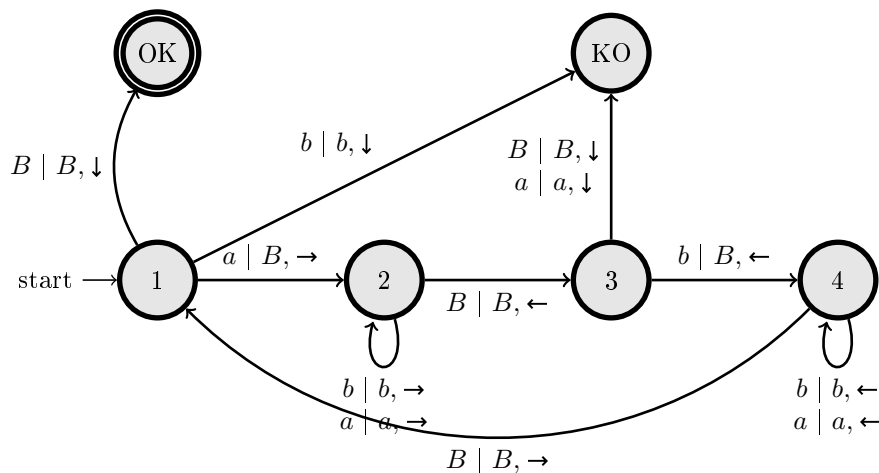


Une transition est étiquetée par :

- a : la lettre lue qui déclenche la transition,
- b : la lettre écrite à la place de a .
- d : le déplacement de la tête de lecture : aller à gauche ($\leftarrow$), à droite ($\rightarrow$) ou rester sur place ($\downarrow$).

Exercice 43 : Machine mystère

On définit la machine suivante sur l'alphabet $\{a, b, B\}$, où B désigne le blanc. L'état initial est 1, l'état acceptant est OK.



1. Simuler cette machine sur les entrées aab et $aabb$. On donnera la configuration de la machine tout au long du calcul.
2. Quel est le langage reconnu par la machine ? (pas de justification demandée).

10.2 TP sur simulateur

Rendez-vous sur <https://turingmachinesimulator.com/>. Cette page comporte un simulateur de machines de Turing à 1, 2 ou 3 bandes bi-infinies. Des exemples sont fournis.

Par convention, on définit des états acceptants ; tout état d'arrêt non acceptant est considéré comme rejetant.

Exemple de code d'une machine :

```
// nom de la machine :
name : Exemple de machine
//etat initial :
```

```
init: q0
//etat acceptant :
accept: qFinal

//Exemples de transitions: sur 2 lignes,

q0,0           // etat de départ, lettre lue
q0,1,>         // etat d'arrive, lettre écrite, déplacement (>,< ou -)

q0,1
q0,0,>

q0,_           // _ est le caractère blanc (case vide)
qFinal,_,-
```

Exercice 44 : Machines sur simulateur

Réaliser les codes des machines

1. à une bande, écrivant votre prénom sur le ruban sur l'entrée vide.
2. à une bande, ajoutant 1 à un nombre en binaire.
3. à une bande, ajoutant 1 en boucle à un nombre écrit en binaire.
4. à deux bandes, calculant le miroir d'un mot sur $\{a,b\}$. Par exemple, *abaab* donnera *baaba*.
5. à une bande, calculant le miroir d'un mot sur $\{a,b\}$ (on pourra utiliser des symboles supplémentaires au cours du calcul).
6. à deux bandes, acceptant les mots de la forme xx , où x est un mot de $\{a,b\}$.

Idée : recopier l'entrée sur la deuxième bande, puis il faut essayer de placer la tête de lecture du haut tout à gauche, et la tête de lecture du bas tout à droite. Pour cela, il faut trouver un moyen d'avancer vers la gauche deux fois plus vite en haut qu'en bas.