M2201

# Langages réguliers et automates finis

17 mai 2019 – B. Colombel

## 1 Langages

**Notation.** $\Sigma$ **:** un alphabet

$\Sigma^*$ **:** ensemble des mots construits avec $\Sigma$

$\varepsilon$ **:** le mot sans lettre

$|m|$ **ou** $\ell(m)$ **:** la longueur du mot $m$

$\emptyset$ **:** le langage vide

$L^n$ **:** la puissance $n^e$ du langage

$L^*$ **:** l'étoile du langage

$L^+$ **:** la puissance + du langage

**Exercice 1** : Soit $\Sigma$ un alphabet.

1. Qu'est-ce qu'une chaîne sur $\Sigma$ ?
2. Que désigne $\Sigma^*$ ?
3. Qu'est-ce qu'un langage sur $\Sigma$ ?
4. $\Sigma$ est-il un langage sur $\Sigma$ ?
5. $\Sigma^*$ est-il un langage sur $\Sigma$ ?
6. Toute partie de $\Sigma$ est-elle un langage sur $\Sigma$ ?
7. Qu'est-ce qu'un langage de $\Sigma^*$ ?
8. $\emptyset$ est-il un langage sur $\Sigma$ ?
9. $\{\epsilon\}$ est-il un langage sur $\Sigma$ ?

**Exercice 2** : Former la liste des mots de longueur 3 avec $\Sigma = \{a,b,c\}$.

**Exercice 3** : Sur l'alphabet $\Sigma = \{a,b,c\}$, on considère les langages $A = \{a,bc,\varepsilon\}$ et $B = \{abc,cb\}$.
Pour chacun des mots suivants, dire s'il appartient à $A \cdot B$, $A^* \cdot B$ et $(A \cdot B)^*$ :

$$abc; \quad aacb; \quad abccb; \quad cba$$

**Exercice 4** : Soit $\Sigma$ un alphabet dont le nombre de caractère est supérieur ou égal à 2. On appelle *retournement* l'application $\rho : \Sigma^* \to \Sigma^*$ telle que $\rho(\varepsilon) = \varepsilon$ et qui associe au mot $\sigma$ de longueur non nulle la mot $\tau = \rho(\sigma)$, appelé *retourné* de $\sigma$, défini par :

$$\tau(k) = \sigma(n - k + 1)$$

1. Déterminer $\rho(\sigma)$ lorsque $\sigma = aabcdea$. De manière générale, comment décrire l'effet de l'application retournement sur une chaine de caractères ?
2. Exprimer $\rho(\sigma\tau)$ à l'aide de $\rho(\tau)$ et de $\rho(\sigma)$.
3. On dit qu'un mot $\sigma$ est un palindrome si $\rho(\sigma) = \sigma$.
   (a) Montrer que tout mot de la forme $\rho(\sigma)\sigma$ est un palindrome.
   Est-ce-là tous les palindromes ?
   (b) Si le nombre d'éléments de $\Sigma$ est $N$, combien y-a-t-il de palindromes de longueur $p$ dans $\Sigma^*$ ?

**Exercice 5** : Soit $\Sigma = \{a; b\}$ et les langages :

$$L_1 = \{a^n b^p,\, n,p \in \mathbb{N}\}, \quad L_2 = \{a^n b^n,\, n \in \mathbb{N}\}, \quad ,L_3 = \{a^n,\, n \in \mathbb{N}\}, \quad L_4 = \{b^n,\, n \in \mathbb{N}\}$$

1. Donner des mots de chacun de ces langages.
2. Déterminer $L_1 \cap L_2$.
3. Déterminer $L_1 \backslash L_3$ puis $L_2 \backslash L_4$.
4. Pourquoi peut-on écrire $L_1 = L_3 \cdot L_4$, c'est-à-dire $L_1 = a^* b^*$ ?
   Peut-on aussi écrire $L_2 = a^* b^*$ ?

## 2 Expressions régulières

> Soit $\Sigma$ un alphabet.
>
> Une *expression régulière* est une formule obtenue en combinant les lettres de l'alphabet au moyen de l'addition, le produit de concaténation et de l'étoile.
>
> Plus précisément, l'ensemble des expressions régulières est un langage sur l'alphabet $\{+, *, (, ), \varepsilon, \emptyset\} \cup \Sigma$ avec les règles suivantes :
>
> **Règles de base** — $\varepsilon$ et $\emptyset$ sont des expressions régulières ;
> — si $a$ est une lettre de $\Sigma$, alors le mot $a$ est une expression régulière ;
>
> **Règles de combinaison** : si $\alpha$ et $\beta$ sont des expressions régulières, alors :
> — $(a)$ est une expression régulière ;
> — $(a) + (b)$ est une expression régulière ;
> — $(a)(b)$ est une expression régulière ;
> — $(a)^*$ est une expression régulière.
>
> Un langage est *régulier* lorsqu'il peut être défini par une expression régulière.
> On note $L_\alpha$ le langage défini par l'expression régulière $\alpha$.

**Exercice 6** : Soit $\Sigma = \{\%, \#, a, b, c\}$ un alphabet. Repérer les expressions régulières sur $\Sigma$ parmi les suites de symboles suivantes :

1. $(a + \#)^* \# b \% c^*$ ;
2. $((a\#)^* +) b^* c$ ;
3. $\%^* + {}^* \#^*$ ;
4. $((a^* b)^* \# + ca\%^*)$

**Exercice 7** : Sur l'alphabet $\Sigma = \{a, b, c\}$, on considère les expressions régulières :

$$\alpha = (a + cb)(ab)^*(c + \varepsilon) \quad \text{et} \quad \beta = a(a + b)^*$$

Les mots suivants sont-ils dans $L_\alpha$ ? dans $L_\beta$ ?

1. $aabab$
2. $aabc$
3. $cb$
4. $\varepsilon$
5. $a^3bc$
6. $ac$
7. $cbcbc$
8. $cbcba^4c$

**Exercice 8** : Soit $\Sigma = \{a; b\}$. Vrai ou Faux ?

1. $(a^*)^* = a^*$
2. $aa^* = a^*a$
3. $aa^* + \varepsilon = a^*$
4. $(ab)^*a = a(ab)^*$
5. $(a + b)^* = (a^* + b^*)^*$
6. $(a + b)^* = (a^*b^*)$
7. $(a + b)^* = (a + b)^* \cdot (a + b)^*$
8. $(a + b)^* = a^*b^*$
9. $b^* + ab^* = (\varepsilon + a)b^*$

**Exercice 9** : Soit l'alphabet $\Sigma = \{a, b, c\}$. Pour chacun des langages suivants, proposer une expression régulière :

1. Langage des mots sur $\Sigma$ commençant par $a$ ;
2. Langage des mots sur $\Sigma$ commençant et se terminant par $a$ ;
3. Langage des mots sur $\Sigma$ commençant et se terminant par la même lettre ;
4. Langage des mots sur $\Sigma$ commençant et se terminant par des lettres distinctes ;
5. Langage des mots sur $\Sigma$ contenant au moins un $a$ ;
6. Langage des mots sur $\Sigma$ contenant exactement un $a$ ;
7. Langage des mots sur $\Sigma$ ne contenant aucun $a$ ;
8. Langage des mots sur $\Sigma$ contenant un nombre pair de $a$ ;
9. Langage des mots sur $\Sigma$ contenant un nombre impair de $a$.

**Exercice 10** :

1. Décrire le langage dont les mots sont les éléments de $\mathbb{N}$ (plus précisément, on demande de donner une expression régulière du langage des écritures en base dix d'entiers naturels).
2. Décrire le langage dont les mots sont les éléments de $\mathbb{Z}$.

# 3 Automates finis déterministes complets

## 3.1 Langages reconnus par un automate

**Rappels** : Un automate est appelé :
- *déterministe* : un seul état initial et pour chaque état, au plus une transition par lettre ;
- *complet* : au moins une transition par lettre.

**Exercice 11** : On considère l'alphabet $\Sigma = \{a; b\}$.

| | $a$ | $b$ |
|---|---|---|
| 1 | 1 | 2 |
| 2 | 3 | 2 |
| 3 | 3 | 3 |

1. Déterminer l'automate fini $\mathcal{A}$ à trois états, d'état initial 1 et d'état acceptant 3 dont la table de transition est donnée ci-contre.
2. Cet automate fini est-il déterministe ? complet ?
3. Les mots suivants sont-ils acceptés ?

$$m_1 = ba, \quad m_2 = ab, \quad m_3 = \varepsilon, \quad m_4 = a, \quad m_5 = b, \quad m_6 = baba, \quad m_7 = abab, \quad m_8 = aabbaa$$

4. Quel langage $L(\mathcal{A})$ reconnait-il ?

**Exercice 12** : On considère l'alphabet $\Sigma = \{a; b; c\}$.

| | $a$ | $b$ | $c$ |
|---|---|---|---|
| 1 | 1 | 2 | |
| 2 | | 1 | 3 |
| 3 | | | |

1. Déterminer l'automate fini $\mathcal{A}_2$ à trois états, d'état initial 1 et d'état acceptant 1 dont la table de transition est donnée ci-contre.
2. Cet automate fini est-il déterministe ? complet ?
3. Les mots suivants sont-ils acceptés ?

$$m_1 = ab, \quad m_2 = abc, \quad m_3 = abb, \quad m_4 = abba, \quad m_5 = bcbc, \quad m_6 = bab, \quad m_7 = abbbc, \quad m_8 = bccbc$$

4. Quel langage $L(\mathcal{A}_2)$ reconnait-il ?

**Exercice 13** : Construire des automates finis reconnaissant les langages réguliers de l'exercice 9. Ces automates finis ne seront pas nécessairement déterministes ou complets.

## 3.2 Lemme d'Arden

On cherche dans cette partie à déterminer l'expression régulière du langage reconnu par un automate fini donné.

**Théorème 1 (Lemme d'Arden)**
Soit $U$ et $V$ deux langages sur l'alphabet $\Sigma$ et l'équation d'inconnue $X$ : $X = UX + V \quad (1)$.
1. le plus petit langage solution de (1) est $X = U^*V$
2. De plus, si $\varepsilon \notin U$, alors cette solution est unique

**Exercice 14** : On considère sur l'alphabet $\Sigma = \{a,b\}$ l'automate fini déterministe $\mathcal{A}$ dont l'état initial est 1, les états acceptants sont les états numérotés 3 et 6 et dont les transitions sont données ci-contre.

| | $a$ | $b$ |
|---|---|---|
| 1 | 2 | 5 |
| 2 | 3 | 2 |
| 3 | 3 | 4 |
| 4 | 3 | 2 |
| 5 | 5 | 6 |
| 6 | 7 | 6 |
| 7 | 5 | 6 |

1. Dessiner le diagramme de l'automate.
2. Vérifier si les mots suivants sont acceptés par l'automate.
   (a) $a^3ba$ (b) $a(ba)^3$ (c) $(ba)^3$
3. Soit $D_i$ le langage de départ associé à l'état $i$.
   (a) Écrire et résoudre le système de départ de l'automate.
   (b) En déduire une expression régulière du langage $L$ de l'automate.

## 3.3 Résiduel d'un langage

On sait déjà, à partir d'un automate fini déterministe complet, trouver l'expression régulière caractérisant le langage de l'automate. On cherche ici, à faire le chemin en sens inverse.

**Définition 1**
Soient $\Sigma$ un alphabet, $L$ un langage sur $\Sigma$ et $\sigma \in \Sigma^*$. On appelle *résiduel de L par rapport à $\sigma$* le langage formé de tous les mots $\tau$ tels que :

$$\sigma \cdot \tau \in L$$

On le note $\sigma^{-1}L$.

**Remarque.** D'un point de vue concret, $\sigma^{-1}L$ est le langage obtenu en prenant les mots de $L$ commençant par $\sigma$ et en effaçant ce $\sigma$ au début des mots.

**Exercice 15** : Soit $\Sigma = \{a,b\}$ et $L = a^*$.
Montrer que $L$ a deux résiduels : $\emptyset$ et $L$.

**Propriétés.**
1. $\varepsilon \in \sigma^{-1}L \iff \sigma \in L$
2. $\sigma^{-1}\emptyset = \emptyset$ et $\sigma^{-1}\Sigma^* = \Sigma^*$
3. $\varepsilon^{-1}L = L$
4. $\tau^{-1}(\sigma^{-1}L) = (\sigma\tau)^{-1}L$

**Méthode.** Pour calculer tous les résiduels d'un langage régulier $L$, on détermine :
1. $\varepsilon^{-1}L = L$ ;
2. les différents résiduels $R_i$ du type $x^{-1}L$ pour chacune des lettres $x$ de $\Sigma$ ;
3. les résiduels $x^{-1}R_i$ pour chacun des résiduels $R_i$ obtenus à l'étape précédente qui sont différents de ceux rencontrés et pour chacune des lettres $x$ de $\Sigma$ ;

On itère **3.** jusqu'à ce que l'on n'obtienne plus de résiduels nouveaux.

**Exercice 16** : Sur l'alphabet $\Sigma = \{a; b\}$, on note $L$ l'ensemble des mots contenant exactement deux occurrences de $a$ mais non consécutives.
1. Donner une expression régulière de $L$.
2. Déterminer les résiduels de $L$.

**Théorème 2**
Un langage $L$ est régulier si, et seulement si, il n'a qu'un nombre fini de résiduels.

**Exercice 17** : À partir de l'alphabet $\Sigma = \{a,b\}$, on construit le langage $L$ des mots de la forme $a^m b a^m$ avec $m$ entier positif ou nul.
1. Quel est le résiduel $\sigma^{-1}L$ lorsque $\sigma$ contient plusieurs lettres $b$ ?
2. Quel est le résiduel $\sigma^{-1}L$ lorsque $\sigma = a^{p+q}ba^p$ avec $p$ et $q$ entiers positifs ou nuls ?
3. Le langage $L$ est-il régulier ?

# 4 Automate minimal

## 4.1 Construction de l'automate minimal

Soit $L$ un langage dont le nombre de résiduels est fini ; $L$ est ainsi régulier. Il existe alors un automate régulier qui reconnait $L$.
À partir des résiduels de $L$, on construit un automate $\mathcal{M}$ comme suit :

**Méthode. Construction d'un automate à partir des résiduels**

① Les états sont les différents résiduels de $L$ ;

② L'état initial est $L$ ;

③ Les états acceptants sont les résiduels qui contiennent $\varepsilon$ ;

④ La flèche $a$ qui part du résiduel $R$ arrive au résiduel $a^{-1}R$.

**Exercice 18** : Sur l'alphabet $\Sigma = \{a; b\}$, on note $L$ l'ensemble des mots contenant exactement deux occurrences de $a$ mais non consécutives.

1. Construire l'automate déterministe en suivant la méthode décrite ci-dessus.
2. Vérifier que le langage de cet automate est bien $L$.
3. Construire l'automate déterministe du langage $\overline{L}$, complémentaire de $L$ dans $\Sigma^*$.

L'intérêt d'une telle construction est donnée par la théorème suivant :

> **Théorème 3**
>
> 1. L'automate $\mathcal{M}$ construit par cette méthode admet $L$ pour langage ;
> 2. Si $R$ est un résiduel de $L$, alors le langage de départ de l'état $R$ est $R$ ;
> 3. Parmi tous les automates qui admettent $L$ pour langage, c'est $\mathcal{M}$ qui possède le moins d'états.
>
> L'automate $\mathcal{M}$ s'appelle l'*automate minimal* de $L$.

**Exercice 19** : Soit $\Sigma = \{a,b\}$. On note $L$ le langage constitué des mots dont la première et la dernière lettre sont les mêmes.

1. Quels sont les mots de longueur 4 ?
2. Calculer les résiduels de $L$.
3. Le langage $L$ est-il régulier ? Si oui, construire son automate déterministe minimal qui l'accepte et déterminer une expression régulière de $L$.

**Exercice 20** : On note $\Sigma = \{a,b\}$ un alphabet.
Construire l'automate minimal correspondant aux expressions rationnelles suivantes :

1. $aab^*ab$ ;
2. $a^*b^*a^*$ ;
3. $a^*b(a+b)^*$ ;

## 4.2 Simplification d'un automate

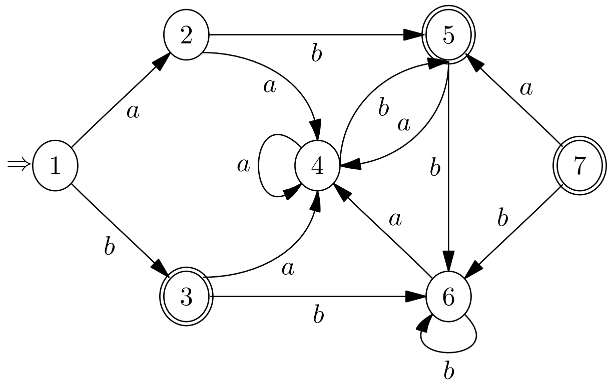
On sait maintenant déterminer, à partir de l'expression régulière d'un langage $L$ sur un alphabet $\Sigma$, l'automate fini *minimal* qui acceptant le langage $L$, c'est-à-dire celui comportant le moins d'états possibles.
On peut utiliser cette méthode pour simplifier des automates finis. Pour cela, regardons le lien entre résiduels et langages de départs.

**Notation.** Dans ce qui suit
— $\mathcal{A}$ désigne un automate dont les états sont numérotés de 1 à $n$ ;
— l'alphabet est $\Sigma$ ;
— le langage de $\mathcal{A}$ est $L$ ;
— le langage de départ de l'état $k$ est $D_k$.

> **Théorème 4**
>
> 1. Si le mot $\sigma$ fait passer de l'état $k$ à l'état $\ell$ alors :
>
> $$D_\ell = \sigma^{-1} D_k$$
>
> 2. En particulier, en partant de l'état initial 1, on obtient :
>
> $$D_\ell = \sigma^{-1} L$$

**Conséquences.** Les résiduels de $L$ sont les langages de départ des états accessibles (éventuellement répétés)

**Exercice 21** : On considère l'automate ci-contre. On cherche à déterminer le nombre de résiduels de cet automate.



1. Un des états n'est pas nécessaire. Dire lequel et expliquer pourquoi.
2. Écrire le système de départ de cet automate.
3. En déduire le nombre de résiduels de cet automate.
4. Construire alors l'automate minimal acceptant le même langage que l'automate initial.

On peut maintenant compléter le théorème vu précédemment :

**Théorème 5**

1. L'automate $\mathcal{M}$ construit par cette méthode admet $L$ pour langage ;
2. Si $R$ est un résiduel de $L$, alors le langage de départ de l'état $R$ est $R$ ;
3. Parmi tous les automates qui admettent $L$ pour langage, c'est $\mathcal{M}$ qui possède le moins d'états.

L'automate $\mathcal{M}$ s'appelle l'*automate minimal* de $L$.

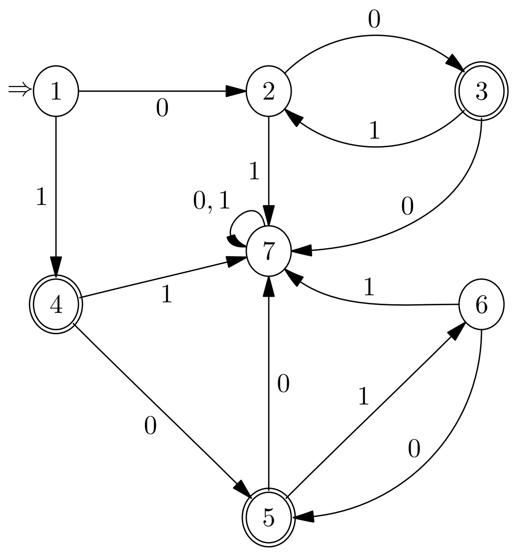
Partant d'un automate fini, on cherche à le remplacer par un automate qui a moins d'états mais qui admet le même langage : c'est la *simplification* de l'automate.

La méthode pour déterminer l'automate minimal est donc :

**Méthode pratique pour construire l'automate minimal :**

1. Éliminer les états inaccessibles et tous les arc qui y arrivent ou qui en partent ;
2. Regrouper tous les états de $\mathcal{A}$ qui ont le même langage de départ et les entourer d'un cercle ;
3. Doubler le cercle autour des regroupements contenant un état qui était acceptant dans l'automate $\mathcal{A}$.
4. Placer une grosse flèche à côté du regroupement contenant l'état initial de $\mathcal{A}$ ;
5. Joindre deux regroupements par un arc étiqueté $a$ lorsqu'ils contiennent deux états qui étaient joints par un arc étiqueté $a$.

**Exercice 22** : On considère l'automate $\mathcal{A}$ ci-contre.



1. Donner une expression régulière du langage $L$ de $\mathcal{A}$.
2. L'automate $\mathcal{A}$ est-il simplifiable ?
3. Déterminer les résiduels du langage $M = 1 + (00 + 10)(10)^*$. En déduire l'automate minimal de $M$.

# 5 Construction d'automates

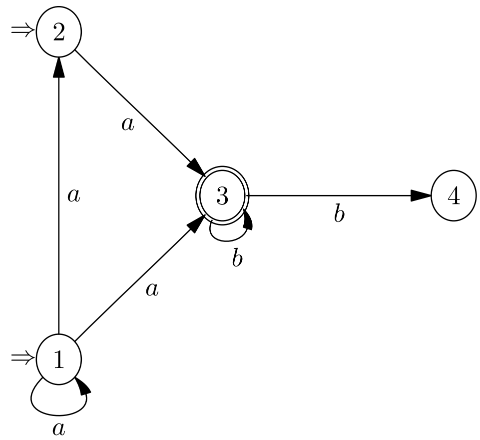
## 5.1 Déterminisation d'un automate

Partant d'un automate non déterministe, on peut construire un automate déterministe qui reconnait le même langage. Voici la méthode pratique de déterminisation d'un automate non déterministe $\mathcal{A}_n$.

**Méthode pratique pour déterminiser $\mathcal{A}_n$ en $\mathcal{A}_d$**

1. Regrouper tous les états initiaux dans un même cercle : c'est l'état initial de $\mathcal{A}_d$ ;
2. À chaque fois qu'on a dessiné un cercle représentant un état de $\mathcal{A}_d$ :
   — si son contenu est $\emptyset$, alors, c'est un piège refusant ;
   — sinon, son contenu est un regroupement d'états de $\mathcal{A}_n$. Pour chaque élément $x \in \Sigma$,
      — dessiner une flèche $x$ sortant de ce cercle pour aboutir à un cercle non rempli ;
      — remplir ce cercle avec les états de $\mathcal{A}_n$ que l'on peut atteindre en partant des états du regroupement(le résultat peut être vide) ;
      — si le cercle ainsi créé existe déjà le remplacer par ce cercle déjà connu ;
3. Continuer jusqu'à que l'on ne puisse plus construire d'état.
4. les états acceptants sont les états de $\mathcal{A}_d$ qui, contiennent au moins un état acceptant de $\mathcal{A}_n$.

**Exercice 23** : Appliquer la méthode de déterminisation à l'automate ci-contre ni déterministe ni complet.




**Exercice 24** : Soit l'alphabet $\Sigma = \{a,b\}$ et $L$ le langage sur $\Sigma$ des mots qui contiennent au moins une fois le facteur $aa$.

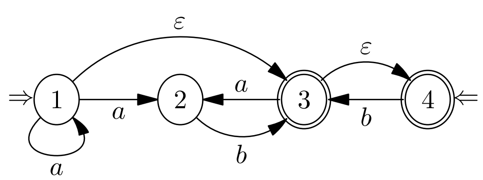
1. Écrire une expression régulière de $L$.
2. En déduire un automate fini $\mathcal{A}$ qui reconnaisse $L$. Cet automate devrait être ni déterministe ni complet.
3. Déterminer un automate fini déterministe complet équivalent à $\mathcal{A}$.
4. Construire l'automate minimal qui reconnait le langage $L$.

**Exercice 25** : Soit l'alphabet $\Sigma = \{a,b,c\}$. Construire un automate fini déterministe complet reconnaissant :

$$L = ab\Sigma^* + \Sigma^* ca\Sigma*$$

## 5.2 Transitions spontanées

On considère l'automate ci-dessous :




Un tel automate possède des *transition spontanées*.

1. Comment définiriez vous une *transition spontanées* ?
2. Compléter le tableau des transitions suivant :

| | $\varepsilon$ | $a$ | $b$ |
|---|---|---|---|
| 1 | | | |
| 2 | | | |
| 3 | | | |
| 4 | | | |

**Remarques.**

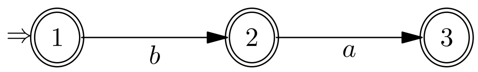
1. Un automate fini avec transitions spontanées peut être déterminisé comme les automates non déterministes et l'automate déterminisé admet le même langage que l'automate de départ.
2. Il faut néanmoins tenir compte des transitions spontanées : l'état initial est composé des états initiaux et de ceux que l'on peut atteindre uniquement en suivant les transitions spontanées.

3. Déterminiser cet automate.

**Exercice 26** : On cherche à construire un automate déterministe complet reconnaissant le langage $L = (ab)^* + (ba)^*$.

1. (a) Dessiner un automate non déterministe qui accepte uniquement le mot $ab$.
   (b) À partir de cet automate, construire un automate à 3 états et possédant une transition spontanée qui accepte le langage $(ab)^*$.
   (c) Déterminiser cet automate.
2. Quel automate déterministe complet reconnait le langage $(ba)^*$ ?
3. En déduire un automate déterministe complet qui reconnait le langage $L$.

**Exercice 27** : Soit l'alphabet $\Sigma = \{a,b\}$. On considère l'automate $\mathcal{A}$ ci-dessous :




1. Déterminer le langage $L$ accepté par $\mathcal{A}$.
2. En ajoutant deux transitions spontanées, mais sans ajouter d'état supplémentaire, construire un automate non déterministe $\mathcal{B}$ qui accepte le langage $L^*$. Dessiner son diagramme.
3. Déterminiser $\mathcal{B}$ afin d'obtenir un automate déterministe $\mathcal{C}$ qui reconnait $L^*$. Dessiner son diagramme (on notera 1 son état initial et si $\delta$ est la fonction de transition, on notera les autres états de manière à ce que $2 = \delta(1,b)$, $3 = \delta(2,a)$ et $4 = \delta(3,a)$.
4. Écrire le système de départ de $\mathcal{C}$.
   Vérifier que $D_1 = D_3$, exprimer $D_2$ en fonction de $D_1$ et en déduire $D_1$.
5. A-t-on $(bb^*a)^*b^* = (ba + b)^*$ ? Justifier la réponse.

**Exercice 28** : Sur l'alphabet $\Sigma = \{a,b\}$, on considère le langage $L = a^*b$.

1. Déterminer tous les résiduels de $L$. En déduire l'automate minimal $\mathcal{A}$ qui accepte $L$.
2. À partir de $\mathcal{A}$, construire un automate $\mathcal{B}$ avec transition(s) spontanée(s) qui accepte le langage $L^*$.
3. Déduire de $\mathcal{B}$ un automate déterministe complet $\mathcal{C}$ acceptant $L^*$.
4. Écrire les équations de départ pour l'automate $\mathcal{C}$ puis les résoudre. En déduire l'automate minimal $\mathcal{D}$ qui reconnait $L^*$.
5. Construire à partir de $\mathcal{D}$ un automate avec transition(s) spontanée(s) $\mathcal{F}$ qui accepte le langage $M = L^*a^*$.
6. Déterminiser $\mathcal{F}$. Que peut-on dire de $M$ ?