Langages

On rappelle que :
- un *alphabet* Σ est un ensemble fini de symboles, chiffres ou lettres ;
- l'ensemble des *mots finis* sur Σ est noté Σ^* ;
- cet ensemble contient, en particulier, le *mot vide* noté ϵ ;
- on dénote par $\cdot$ l'opération de *concaténation* sur les mots.

Un *langage* L est un sous-ensemble de Σ^*. On peut définir sur les langages :
- les opérations ensemblistes classiques : $\cup$ *(union)*, $\cap$ *(intersection)*, et $\setminus$ *(différence)* ;
- la concaténation $L_1 \cdot L_2 = \{ u \cdot v \mid u \in L_1 \text{ et } v \in L_2 \}$.

Exercice 1 :

Quel est l'élément neutre pour la concaténation $\cdot$ sur les mots ?
Trouver un élément neutre pour chacune des opérations $\cup$, $\cap$ et $\cdot$ sur les langages.

Exercice 2 :

Montrer que $L_1 \cdot (L_2 \cup L_3) = (L_1 \cdot L_2) \cup (L_1 \cdot L_3)$.
Montrer que $L_1 \cdot (L_2 \cap L_3) \subseteq (L_1 \cdot L_2) \cap (L_1 \cdot L_3)$. L'inclusion contraire est-elle vraie ?

Exercice 3 :

La concaténation des langages $\cdot$ est-elle commutative ?
Trouver une condition nécessaire et suffisante sur Σ pour que $\cdot$ soit commutative sur les langages définis sur Σ.

Langages rationnels

Une *expression rationnelle* est une expression fabriquée à partir :
- des éléments de Σ^* ;
- des opérateurs binaires $\cdot$ *(concaténation, souvent laissée implicite)* et $+$ *(union)* ;
- de l'opérateur unaire de *répétition* $*$.

Exercice 4 :

Rappeler la définition du *langage engendré* par une expression rationnelle (par récurrence sur la structure de l'expression). Un tel langage est dit rationnel.

Exercice 5 :

Quel est le langage engendré sur l'alphabet $\Sigma = \{a, b\}$ par l'expression rationnelle $(a + b)a^*(a + b)$?

Exercice 6 :

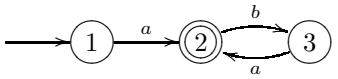
Soit l'alphabet $\Sigma = \{a, b\}$. Donner une expression rationnelle pour chacun des langages suivant :
- l'ensemble des mots qui se terminent par aba ;
- l'ensemble des mots de longueur paire ;
- l'ensemble des mots contenant le facteur ab au moins deux fois ;
- l'ensemble des mots contenant le facteur ab exactement une fois ;
- l'ensemble des mots où tout a est suivi immédiatement d'un b.

Automates non déterministes

On rappelle qu'un automate est un quadruplet (Q, I, F, Δ) composé de :
- un ensemble d'*états* Q ;
- un ensemble $I \subseteq Q$ d'*états initiaux* ;
- un ensemble $F \subseteq Q$ d'*états finaux* ;
- une *relation de transition* $\Delta \subseteq Q \times \Sigma \times Q$.

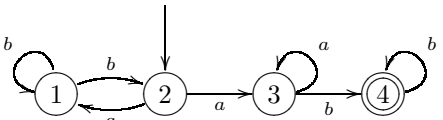
Exercice 7 :

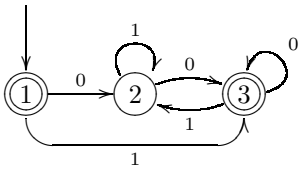
Soit l'automate suivant (deux cercles indiquent qu'un état est final, une flèche venue de nulle part indique qu'un état est initial) :



Donner sa relation de transition Δ.
Quel est le langage reconnu (donner une expression rationnelle) ?

Mêmes questions pour les automates suivants :





Exercice 8 :

Pour chacun des langages suivants sur l'alphabet $\Sigma = \{a, b\}$, donner un automate qui reconnaît exactement ce langage :
- l'ensemble des mots commençant et finissant par ab ;
- l'ensemble des mots qui n'ont pas trois a consécutifs ;
- l'ensemble des mots dont le nombre de a ou le nombre de b est pair.

Exercice 9 :

Donner une condition nécessaire et suffisante pour que le mot vide ϵ soit accepté par un automate.

Exercice 10 :

Un digicode a un clavier composé de 10 chiffres et ouvre la porte quand on tape le code 3234.
On veut modéliser le comportement du digicode par un automate. Pour cela, on modifie légèrement la condition d'acceptation qui devient : à chaque fois que les caractères 3,2,3 et 4 apparaissent à la suite dans un mot, l'automate est dans un état final.

Construire un tel automate.

Exercice 11 :

On peut coder un nombre en binaire par une suite finie de 0 et de 1 ; un langage sur $\Sigma = \{0, 1\}$ correspond alors à un ensemble d'entiers.

Donner un automate qui reconnaît les nombres divisibles par 4.

Donner un automate qui reconnaît les nombres divisibles par 3 (pour cela, déterminer le reste de la division de $2x$ et $2x + 1$ par 3 en fonction du reste de la division de x par 3).

Exercice 12 :

Étant donné un automate $\mathcal{A} = (Q, I, F, \Delta)$, construire un automate $\mathcal{A}'$ qui reconnaît exactement les mots reconnus par $\mathcal{A}$ écrits à l'envers.

Lemme de l'étoile

Exercice 13 :

Montrer que si le langage reconnu par un automate $\mathcal{A} = (Q, I, F, \Delta)$ n'est pas vide, alors ce langage contient au moins un mot de longueur strictement inférieure à $|Q|$.

Exercice 14 :

Montrer que si un langage L est reconnaissable par un automate, alors il existe un n tel que tout mot $u \in L$ de longueur supérieure ou égale à n peut se décomposer en $u = x \cdot y \cdot z$, avec $|x \cdot y| \leq n$, $|y| \geq 1$ et $\forall i,\ x \cdot y^i \cdot z \in L$ *(lemme de l'étoile).*

Exercice 15 :

Montrer que l'ensemble des mots contenant autant de a que de b n'est pas reconnaissable par un automate.

Exercice 16 :

Un palindrome est un mot symétrique : il se lit de la même façon de gauche à droite et de droite à gauche. Montrer que l'ensemble des palindromes sur un alphabet Σ d'au moins deux lettres n'est pas reconnaissable par un automate.

Exercice 17 :

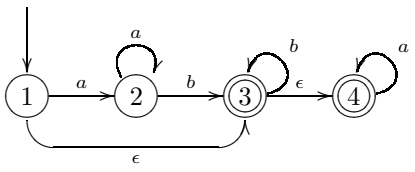
Montrer que le langage $L = \{\ a^n \mid n \text{ premier}\ \}$ n'est pas reconnaissable par un automate.

Automates avec ϵ-transitions

Dans un automate avec ϵ-transitions, la relation de transition Δ est un sous ensemble de $Q \times (\Sigma \cup \{\epsilon\}) \times Q$ (on suppose que ϵ n'est pas déjà dans Σ).
Un transition (q_1, ϵ, q_2) indique que l'on peut passer de l'état q_1 à l'état q_2 sans "manger" de caractère.

Exercice 18 :

Donner le langage reconnu par l'automate ci-dessous :



Exercice 19 :

Soit un automate (Q, I, F, Δ) avec ϵ-transitions.

Définir la fonction *post* qui à un état $q \in Q$ associe tous les états accessibles depuis q par *une* ϵ-transition.
Définir la fonction $post^*$ qui à un état q associe tous les états accessibles depuis q par *un nombre arbitraire* d'ϵ-transition(s) (c'est la *clôture transitive* de *post*).

Prouver (en utilisant $post^*$) qu'un langage reconnaissable par un automate avec ϵ-transitions est aussi reconnaissable par un automate sans ϵ-transition.

Donner un automate sans ϵ-transition qui reconnaît le même langage que celui de l'exercice précédent.

Exercice 20 :

Étant donnés deux automates avec ϵ-transitions $\mathcal{A}_1 = (Q_1, I_1, F_1, \Delta_1)$ et $\mathcal{A}_2 = (Q_2, I_2, F_2, \Delta_2)$, construire un automate avec ϵ-transitions qui reconnaît :
- l'union des langages reconnus par $\mathcal{A}_1$ et $\mathcal{A}_2$;
- la concaténation des langages reconnus par $\mathcal{A}_1$ et $\mathcal{A}_2$;
- l'étoile du langage reconnu par $\mathcal{A}_1$.

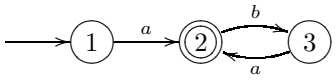
En déduire que tout langage rationnel est régulier.

Automates déterministes

Un automate est *déterministe* si et seulement si il n'a qu'un seul état initial i et pour chaque état $q \in Q$ et symbole $a \in \Sigma$, il y a *au plus* une transition étiquetée par a partant de q.
Un automate déterministe est *complet* si il y a *exactement* une transition partant de q et étiquetée par a. On note alors $a' = \delta(q, a)$ au lieu de $(a, q, a') \in \Delta$. δ est appelée *fonction de transition*.

Exercice 21 :

Montrer comment compléter formellement un automate déterministe (Q, i, F, Δ).
Exemple : compléter l'automate déterministe suivant :



Exercice 22 :

Proposer un algorithme qui prend en argument un automate déterministe complet $\mathcal{A} = (Q, i, F, \delta)$ et un mot w, et qui renvoie *vrai* si w est reconnu par $\mathcal{A}$, *faux* sinon.
Quelle est sa complexité en temps et en mémoire ?

Exercice 23 :

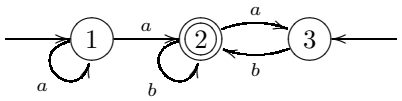
On veut maintenant déterminer si un mot est accepté par un automate *non déterministe*. Ce problème est similaire à celui du parcours d'un graphe et on peut opter pour deux stratégies :
- parcours en profondeur ;
- parcours en largeur.

Donner un algorithme pour chaque stratégie. Déterminer leurs complexités en temps et en mémoire.

Exercice 24 :

Décrire une procédure qui prend un automate non déterministe (Q, I, F, Σ) et qui construit un automate déterministe reconnaissant le même langage. Quelle est la taille de l'automate obtenu ?
Indice : considérer le parcours en largeur de l'automate non déterministe...

Exemple : appliquer la procédure à l'automate suivant :



Algèbre des langages réguliers

Exercice 25 :

Soit $\mathcal{A} = (Q, i, F, \delta)$ un automate déterministe complet.
Montrer que $(Q, i, Q \setminus F, \delta)$ reconnaît le langage complémentaire de $\mathcal{A}$.

En déduire une construction d'un automate déterministe complet reconnaissant les mots sur $\Sigma = \{a, b\}$ ne contenant pas le facteur *baba*.

Exercice 26 :

Montrer que si $\mathcal{A}' = (Q, I, F, \Delta)$ n'est pas déterministe et complet, alors $(Q, I, Q \setminus F, \Delta)$ ne reconnaît pas forcément le complémentaire de $\mathcal{A}'$.
Proposer une méthode pour construire un automate reconnaissant le complémentaire de $\mathcal{A}'$.

En déduire une méthode pour construire un automate reconnaissant l'intersection des langages reconnus par deux automates.

Exercice 27 :

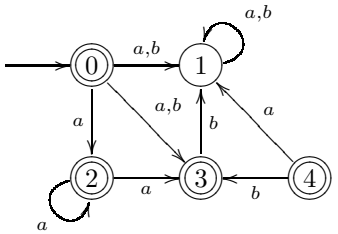
Proposer une méthode pour vérifier si le langage reconnu par un automate non déterministe est vide.

En déduire une méthode pour tester l'inclusion de deux langages réguliers ainsi qu'une méthode pour tester si deux automates non déterministes reconnaissent le même langage.

Automates émondés

Exercice 28 :

Soit l'automate suivant :



Déterminer les états *accessibles* A et les états *co-accessibles* C.
Simplifier l'automate en ne gardant que les états accessibles et co-accessibles.

Exercice 29 :

Étant donné un automate non déterministe $\mathcal{A} = (Q, I, F, \Delta)$, définir formellement l'ensemble des états accessibles et co-accessibles (s'inspirer de la fonction $post^*$ de l'exercice 19).
Définir formellement l'émondé de $\mathcal{A}$ (automate où seuls les états accessibles et co-accessibles sont gardés).

Systèmes d'équations

Exercice 30 :

Soit le système d'équations entre langages : $\begin{cases} L_1 = \{a, b\} \cdot L_1 \cup \{a\} \cdot L_2 \\ L_2 = \{c\} \cdot L_3 \cup \{\epsilon\} \\ L_3 = \{c\} \cdot L_2 \end{cases}$

Écrire un automate qui reconnaît exactement le langage L_1.

Résiduels

Étant donné un langage $L \subseteq \Sigma^*$ et un mot $u \in \Sigma^*$, le *résiduel* est défini par $u^{-1}L = \{v \mid u \cdot v \in L\}$.

Exercice 31 :

Calculer l'ensemble des résiduels des langages suivants :
- le langage engendré par l'expression rationnelle ab^*a^* ;
- $\{ a^n b^n \mid n \in \mathbb{N} \}$.

Exercice 32 :

Soient L_1 et L_2 deux langages et $a \in \Sigma$, déterminer $a^{-1}(L_1 \cup L_2)$, $a^{-1}(L_1 \cdot L_2)$ et $a^{-1}(L_1^*)$ en fonction de $a^{-1}L_1$ et $a^{-1}L_2$.
En déduire que les résiduels d'un langage rationnel sont rationnels et en nombre fini.

Exercice 33 :

Soit $\mathcal{A} = (Q, i, F, \delta)$ un automate déterministe complet reconnaissant le langage $L(\mathcal{A})$ et $u \in \Sigma^*$ un mot.
Définir formellement un automate qui reconnaît le résiduel $u^{-1}L(\mathcal{A})$.
En déduire qu'un langage régulier n'a qu'un nombre fini de résiduels.

Exercice 34 :

Montrer, par analyse des résiduels, que les langages $L_1 = \{u \in \{a, b\}^* \mid u \text{ contient autant de } a \text{ que de } b\}$ et $L_2 = \{ a^n b^n \mid n \in \mathbb{N} \}$ ne sont pas réguliers.

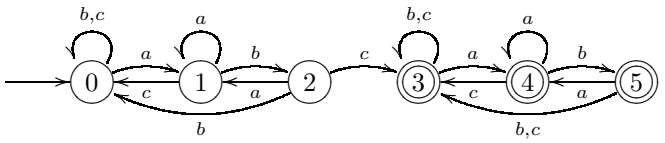
Minimisation

Exercice 35 :

Soit un langage L qui a un ensemble fini de résiduels $u^{-1}L$.
Construire formellement un automate déterministe qui reconnaît L (on étiquettera chaque état par un résiduel).

Montrer que cet automate est *minimal* (au sens de : a le plus petit nombre d'états possible).

Application : construire un automate déterministe reconnaissant les mots contenant le facteur ba.

Exercice 36 :

Minimiser l'automate déterministe suivant :