Université de Versailles – Saint-Quentin-en-Yvelines 2020 – 2021
Licence 2 Informatique

IN406 - Théorie des langages – TD

Version du 4 février 2021

Xavier Badin – xavier.badin-de-montjoye2@uvsq.fr
Pierre Coucheney – pierre.coucheney@uvsq.fr
Franck Quessette – franck.quessette@uvsq.fr
Yann Strozecki – yann.strozecki@uvsq.fr
Sandrine Vial – sandrine.vial@uvsq.fr

Table des matières

1 TD 1 – Mot, langage, automate 2

2 TD 2 – Manipulation d'automate 4

3 TD 3 – Manipulation d'automate (suite) 7

4 TD 4 – Expression régulière 9

5 TD 5 – Lemme de l'étoile 11

6 TD 6 – Automate à pile 12

7 TD 7 – Grammaire 13

8 TD 8 – Machines de Turing 15

9 TD 9 – Machines de Turing - 2 16

10 TD 10 – Décidabilité 17

11 TD 11 – Thèse de Church et reconnaissance 18

1 TD 1 – Mot, langage, automate

Exercice 1 *Alphabet et langage*
Soit $\Sigma = \{a, b\}$ un alphabet, comment peut-on définir les langages Σ^0, Σ^1, Σ^2 ? Comment définir Σ^* ?

Exercice 2 *Longueur*
Pour tout mot w sur l'alphabet Σ, $|w|$ est la longueur (nombre de lettres du mot). La longueur est définie par :

— $|\varepsilon| = 0$;
— $\forall a \in \Sigma$, $\forall w$ mot sur Σ, $|aw| = |wa| = 1 + |w|$;

Pour tout alphabet Σ, démontrez que :

$$\forall w_1, w_2 \in \Sigma^*, |w_1 w_2| = |w_1| + |w_2|$$

Exercice 3 *Concaténation*

1. Quel est la concaténation des mots $w_1 = abc$ et $w_2 = cba$?
2. La concaténation est-elle associative ?
3. La concaténation est-elle commutative ?
4. Donnez deux mots différents w_1 et w_2 de longueur non nulle tels que $w_1 w_2 = w_2 w_1$.

Exercice 4 *Lemme de Levi*
Lemme de Lévi

Soient w_1, w_2, w'_1 et w'_2 des mots sur un alphabet Σ avec $|w_1| \geq |w'_1|$. Si $w_1 w_2 = w'_1 w'_2$, alors il existe un mot z tel que $w_1 = w'_1 z$ et $z w_2 = w'_2$.

Lemme de Lévi (version 2)

Si deux mots u et v sont des préfixes d'un même mot, alors l'un deux est préfixe de l'autre.

Démontrez le lemme de Levi.

Exercice 5 *Lemme de Levi (application)*
En utilisant le lemme de Levi, montrez que deux mots w_1 et w_2 commutent, c'est à dire que $w_1 w_2 = w_2 w_1$, si et seulement si ils sont puissance d'un même mot.

Exercice 6
Soit un alphabet Σ. Soient a et b deux lettres de Σ et w un mot de Σ^*. Si $aw = wb$, que peut-on déduire sur, a, b et w ?

Exercice 7 *Premiers automates*
Soit $\Sigma = \{a, b, c\}$ un alphabet. Essayez de construire les automates reconnaissant les langages suivants :

— L_1, l'ensemble des mots qui commencent par a ;
— L_2, l'ensemble des mots qui ne contiennent pas de c ;
— L_3, l'ensemble des mots qui contiennent au moins un a ;
— L_4, l'ensemble des mots qui contiennent au plus un a ;
— $L_{51} = \Sigma^*$, $L_{52} = \emptyset$, $L_{53} = \{\epsilon\}$;
— L_6, l'ensemble des mots qui ont un nombre pair de a ;
— L_7, l'ensemble des mots qui ont un nombre impair de b ;
— $L_8 = L_6 \cup L_7$. Nommez les états de façon pertinente. Est-il facile d'avoir $L_6 \cap L_7$;

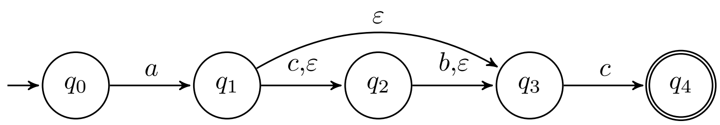
- L_9, l'ensemble des mots ayant un nombre de a multiple de 41. Donner une description formelle ;
- $L_{10} = L_6 \cup L_9$.

Exercice 8 *Automates plus complexes*

- Donner un automate qui reconnait le code `2341` sur un digicode à quatre touches : $\{1, 2, 3, 4\}$;
- même question pour le code `2232` ;
- construire un automate permettant de résoudre le problème du Loup, de la Chèvre et du Chou : http://fr.wikipedia.org/wiki/Problemes_de_passage_de_riviere ;
- construire un automate permettant de résoudre le problème des bidons de 3 gallons et de 5 gallons ; https://www.youtube.com/watch?v=pmk2mNf9iqE
- construire un automate reconnaissant les nombre multiples de 3 écrits en base 10 ;
- construire un automate reconnaissant les nombre multiples de 3 écrits en base 2 ;
- construire un automate reconnaissant les entiers signés en langage C.

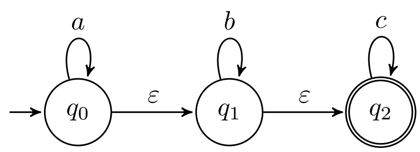
2 TD 2 – Manipulation d'automate

Exercice 9 *Suppression des ε-transitions*
Soit l'automate :



1. le langage reconnu par cet automate est-il fini ou infini ?
2. supprimer les ε-transitions de l'automate ;
3. déterminiser l'automate.
4. quel est le langage reconnu par l'automate ?
5. à partir des mots du langage, construire un automate en mettant un branche par mot.
6. déterminiser ce nouvel automate. Est-il le même que celui trouvé au point 3 ?

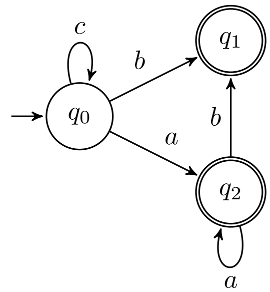
Exercice 10 *Suppression des ε-transitions*
Soit l'automate :



1. le langage reconnu par cet automate est-il fini ou infini ?
2. supprimer les ε-transitions de l'automate ;
3. déterminiser l'automate ;
4. quel est le langage reconnu par cet automate ?

Exercice 11 *AFD*
Soit l'automate :

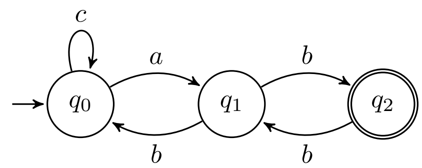
$\mathcal{A}$:



1. donnez sa description formelle $\mathcal{A} = (\Sigma, Q, q_0, F, T)$;
2. les mots $aaab$, cac, cc sont-ils reconnus par $\mathcal{A}$?
3. est-il déterministe, si non déterminisez-le ?
4. donner la description formelle de l'automate déterminisé ;
5. est-il complet, si non rendez-le complet ?
6. donner la description formelle de l'automate complet.

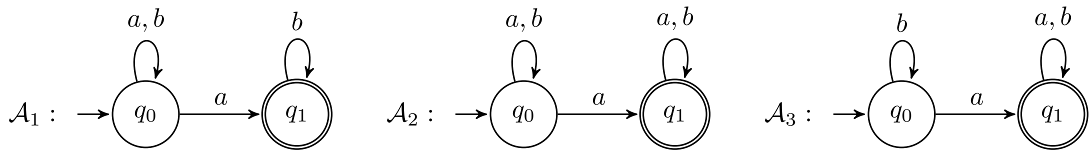
Exercice 12 *AFD*
Soit l'automate :

$\mathcal{A}$:



1. donnez sa description formelle $\mathcal{A} = (\Sigma, Q, q_0, F, T)$;
2. les mots *abbb*, *cabb* sont-ils reconnus par $\mathcal{A}$?
3. est-il déterministe, si non déterminisez-le ?
4. est-il complet, si non rendez-le complet ?

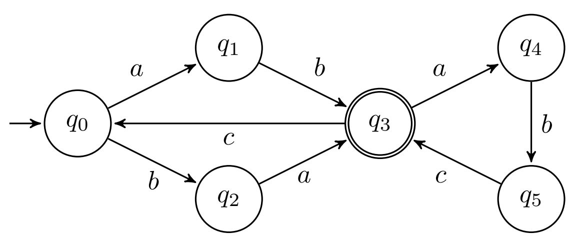
Exercice 13 *AFD*
Soit l'automate :



1. quelles différences entre les trois automates ?
2. quels automates sont déterministes ?
3. déterminisez ceux qui ne le sont pas ?
4. quel est le langage reconnu par chacun des automates

Exercice 14 *AFD*
Soit l'automate :

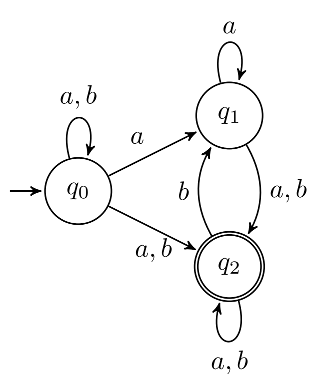
$\mathcal{A}$:



1. donnez sa description formelle $\mathcal{A} = (\Sigma, Q, q_0, F, T)$;
2. est-il déterministe, si non déterminisez-le ?
3. est-il complet, si non rendez-le complet ?

Exercice 15 *AFD*
Soit l'automate :

$\mathcal{A}$:

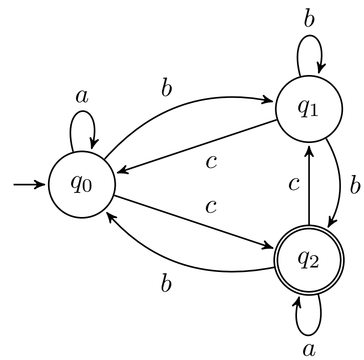


1. est-il déterministe, si non déterminisez-le ?

2. est-il complet, si non rendez-le complet ?
3. quel est le langage reconnu par l'automate ?

Exercice 16 *AFD*
Soit l'automate :

$\mathcal{A}$:



1. donnez sa description formelle $\mathcal{A} = (\Sigma, Q, q_0, F, T)$;
2. est-il déterministe, si non déterminisez-le ?
3. est-il complet, si non rendez-le complet ?

3 TD 3 – Manipulation d'automate (suite)

Exercice 17 *Langage complémentaire*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L_1 l'ensemble des mots ayant un nombre impair de a. Soit L_2 le langage des mots ayant au plus un a.

— donner un automate reconnaissant ces langages, déterminisez les et rendez les complets ;

— en notant $\overline{L_1} = \Sigma^* \backslash L_1$ et $\overline{L_2} = \Sigma^* \backslash L_2$ exprimer en français ces langages ;

— construire les automates reconnaissant $\overline{L_1}$ et $\overline{L_2}$ à partir de ceux reconnaissant L_1 et L_2 ;

— pour tout automate fini déterministe complet, donner une définition formelle de l'automate reconnaissant le langage complémentaire.

Exercice 18 *Union de langages*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L_1 l'ensemble des mots ayant un nombre impair de a et L_2 l'ensemble des mots ayant aba comme sous-mot.

— donner un automate pour chacun des deux langages ;

— on note $L = L_1 \cup L_2$. Construire l'automate reconnaissant L à partir des automates reconnaissant L_1 et L_2.

— pour tout couple d'automates, donner une définition formelle de l'automate reconnaissant l'union des deux langages.

Exercice 19 *Intersection de langages*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L_1 l'ensemble des mots ayant un nombre impair de a et L_2 l'ensemble des mots ayant aba comme sous-mot.

— donner une relation ensembliste permettant de définir l'intersection de deux ensembles à partir de l'union et du complémentaire.

Exercice 20 *Intersection de langages par produit*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L_1 l'ensemble des mots ayant un nombre impair de a. Soit L_2 le langage des mots ayant au plus un a.

— donnez un algo pour construire l'automate de l'intersection de deux langages en utilisant l'automate produit ;

— on note $L = L_1 \cap L_2$. Construire l'automate reconnaissant L à partir des automates reconnaissant L_2 et L_3 ;

— pour tout couple d'automates, donner une définition formelle de l'automate reconnaissant l'intersection des deux langages.

Exercice 21 *Étoile d'un langage*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L l'ensemble des mots ayant exactement deux a.

— construire l'automate reconnaissant L^* à partir de l'automate reconnaissant L.

— pour tout automate, donner une définition formelle de l'automate reconnaissant l'étoile du langage.

Exercice 22 *Étoile d'un langage*
Sur un alphabet $\Sigma = \{a, b\}$. Soit L l'ensemble des mots ayant ayant aba comme sous-mot.

— construire l'automate reconnaissant L^* à partir de l'automate reconnaissant L.

— pour tout automate, donner une définition formelle de l'automate reconnaissant l'étoile du langage.

Exercice 23 *Rationnel ⇒ Reconnaissable*

— donner un automate qui reconnait $L = \emptyset$;
— donner un automate qui reconnait $L = \{\epsilon\}$;
— donner un automate qui reconnait $L = \{a\}$;
— rappeler la définition d'un langage rationnel ;
— avec les résultats ci-dessus, montrer que si un langage est rationnel, il est reconnaissable.

Exercice 24 *Algorithme de Brzozowski de minimisation d'un automate fini*
Soit $\mathcal{A} = (\Sigma, Q, q_0, F, T)$ un automate. L'automate transposé de $\mathcal{A}$ noté $\mathrm{tr}(\mathcal{A}) = (\Sigma', Q', q'_0, F', T')$ est défini par :

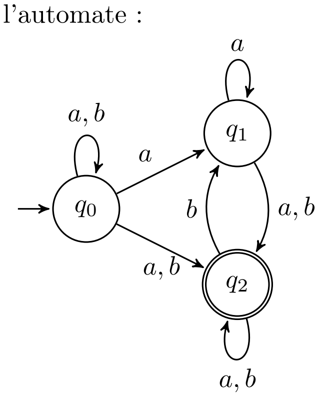
— $\Sigma' = \Sigma$;
— $Q' = Q \cup \{q'_0\}$;
— $F' = \{q_0\}$;
— $T' = \{(q, a, p)$ telle que $(p, a, q) \in T\} \bigcup \{(q'_0, \varepsilon, q)$ avec $q \in F\}$.

Pour tout automate $\mathcal{A}$, on note $\det(\mathcal{A})$ l'automate déterministe reconnaissant le même langage que $\mathcal{A}$.

L'algorithme de Brzozowski consiste à calculer $\det(\mathrm{tr}(\det(\mathrm{tr}(\mathcal{A}))))$, qui est alors l'automate minimum reconnaissant le même langage que $\mathcal{A}$.

Appliquer l'algorithme de Brzozowski sur l'automate :

$\mathcal{A}$:



4 TD 4 – Expression régulière

Exercice 25 *Expression régulière*
Soit l'alphabet $\Sigma = \{a, b, c\}$, donnez les expressions régulières pour les langages suivants :
— les mots ne commençant pas par un c ;
— les mots contenant deux a consécutifs ;
— les mots contenant à la fois les facteurs (sous-mots) aa et bb ;
— les mots dont la première et la dernière lettre sont les mêmes ;
— les mots ne contenant pas deux a consécutifs.

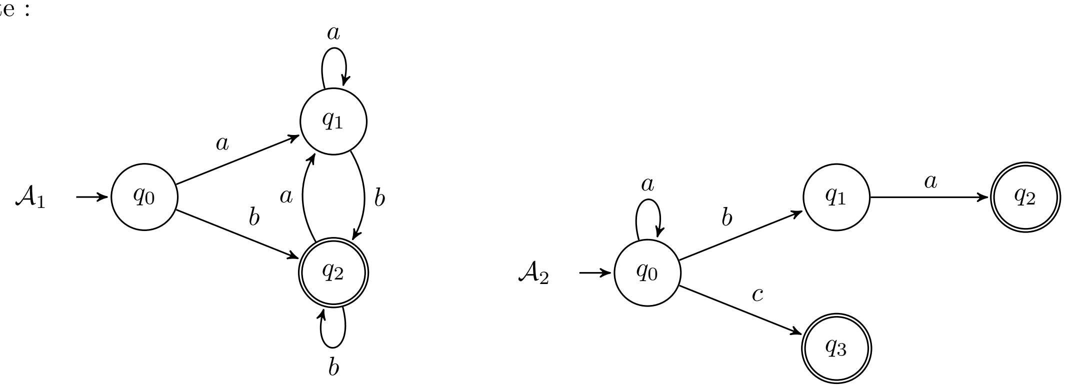
Exercice 26 *Expression régulière*
Donnez les expressions régulières pour les langages suivants :
— les entiers en écrits en binaire ;
— les entiers impairs en binaire ;
— les entiers divisibles par 5 en base 10 ;
— les entiers divisibles par 4 en base 10 ;
— les noms des variables en `C`.

Exercice 27 *Fini et régulier*
Montrer que tous les langages finis sont réguliers.

Exercice 28 *e.r. vers automate*
Donnez un automate reconnaissant les langages définis par les e.r. suivantes :
— $e_1 = (a + b)^*$;
— $e_2 = \varepsilon$;
— $e_3 = \emptyset$;
— $e_4 = a^*b$;
— $e_5 = ba^*$;
— $e_6 = a^*b + ba^*$;
— $e_7 = (a^*b + ba^*)^*$.

Exercice 29 *Égalité d'e.r.*
Montrer que les deux expressions régulières suivantes sont équivalentes : $e_1 = (a + b)^*$ et $e_2 = (ab^*)^* + (ba^*)^*$.

Exercice 30 *AFN vers e.r.*
Pour chaque automate, calculer les expressions régulières dont le langage est le même que celui reconnu par l'automate :



Exercice 31 *La totale*
Soit le langage qui est l'ensemble des mots contenant une seule occurence de aaa sur l'alphabet $\Sigma = \{a, b\}$:

1. donner une expression régulière de ce langage ;
2. construire l'AFN avec ε-transition reconnaissant ce langage ;
3. déterminiser, compléter et minimiser l'automate ;
4. calculer l'expression régulière de l'automate minimum ;
5. comparer avec l'expression régulière calculée au 1.

Exercice 32 *True Lies*
Pour les langages suivants, donner une expression régulière, l'AFN, l'AFD et calculer l'e.r. à partir de l'automate :

1. les mots de longueur paire sur l'alphabet $\{a, b\}$;
2. les mots ne contenant pas ba sur l'alphabet $\{a, b\}$;
3. les mots dont la dernière lettre est au moins deux fois dans le mot sur l'alphabet $\{a, b, c\}$;
4. les mots dont la dernière lettre n'est pas ailleurs dans le mot dans le mot sur l'alphabet $\{a, b, c\}$.

5 TD 5 – Lemme de l'étoile

Utilisation de lemme de l'étoile pour prouver qu'un langage n'est pas régulier :

Rappel du lemme

SI un langage L est régulier **ALORS** $\exists N > 0$, tel que $\forall z L, |z| > N$:

- $\exists u, v, w$ tels que $z = uvw$;
- $|v| > 0$;
- $|uv| \leq N$;
- $\forall i \geq 0$, $uv^i w \in L$.

Exemple

On va montrer que $L = a^n b^n$ n'est pas régulier par contradiction :

1. **Choix du mot z :** On suppose que L est régulier et on considère le mot $z = a^N b^N$, on a bien $|z| > N$ et la contrainte $|uv| \leq N$ fait que la décomposition z en uvw est telle que $u = a^x$, $v = a^y$ et $w = a^z b^N$ avec $y > 0$, $x + y < N$ et $x + y + z = N$.
2. **Choix du i :** D'après le lemme de l'étoile $uv^2 w$ doit appartenir au langage, or $uv^2 w = a^{N+y} b^N$ avec $y > 0$ et donc $uv^2 w \notin L$, ce qui est une contradiction.
3. **Conclusion :** Donc l'hypothèse de départ que L est régulier est fausse et donc L n'est pas régulier.

La technique de preuve consiste donc à trouver le mot z et la valeur de i astucieux pour obtenir une contradiction pour toutes les décompositions de z.

Exercice 33 *Langage non régulier*
Montrez que les langages suivants ne sont pas réguliers :

- $L_1 = \{a^n b^p, 0 \leq n < p\}$ (indication : $z = a^N b^{N+k}$) ;
- $L_2 = \{a^{n^2}, n \geq 0\}$;
- $L_3 = \{a^n b^p, 0 \leq n \neq p\}$ (indication : par l'absurde en utilisant le complémentaire de $a^n b^n$).

Exercice 34 *Régulier ou pas*
Pour les langages suivants prouver s'ils sont réguliers ou pas :

- $L_4 = \{a^p, p \text{ premier}\}$;
- $L_5 = \{a^n b^p, n \equiv p \pmod 2\}$;
- $L_6 = \{a^n b^p, n \geq p\}$;
- $L_7 = \{w, |w|_a = |w|_b\}$;

6 TD 6 – Automate à pile

Exercice 35 *Automate à pile*
Donner les automates à pile reconnaissant les langages suivants, en précisant si la reconnaissance s'effectue par pile vide ou par état final. Les automates ne doivent pas être nécessairement déterministes

— $L_1 = \{a^n b^n, n \geq 0\}$;
— $L_2 = \{a^i b^j c^k, i, j, k \geq 0 \text{ et } (i = j \text{ ou } j = k)\}$.

Exercice 36 *Automate à deux piles*
Pour $\Sigma = \{a, b\}$, donner les automates à une ou deux piles reconnaissant les langages suivants :

— L_3 : l'ensemble des palindromes ;
— $L_4 = \{w \cdot w, w \in \Sigma^*\}$;
— $L_5 = \{a^{n^2}, n \geq 0\}$.

Exercice 37 *Grammaire*
Construire les grammaires pour les langages suivants :

— L_6 : a^*b ;
— L_7 : ab^* ;
— L_8 : $(a^*b + ab^*)aa$;
— L_9 : $((a^*b + ab^*)aa)^*$;
— $L_{10} = \{a^n bc^n, n > 0\}$;
— $L_{11} = \{a^{n+1} b^n, n > 0\}$;
— $L_{12} = \{a^i b^j c^j b^i, i, j > 0\}$.

7 TD 7 – Grammaire

Exercice 38 *Grammaire*
Construire les grammaires pour les langages suivants en donnant à chaque fois la définition formelle :

— $L_1 : a^*b$;

— $L_2 : ab^*$;

— $L_3 : (a^*b + ab^*)aa$;

— $L_4 : ((a^*b + ab^*)aa)^*$;

— $L_5 = \{a^n bc^n, n > 0\}$;

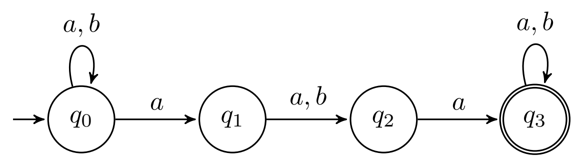
— $L_6 = \{a^{n+1} b^n, n > 0\}$;

— $L_7 = \{a^i b^j c^j b^i, i, j > 0\}$.

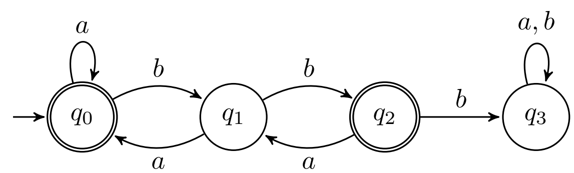
— $L_8 = \{a^n b^n c^n, n \geq 0\}$.

Exercice 39 *Automate vers grammaire*
Donner la grammaire engendrant le même langage que les automates suivants (en donnant à chaque fois la définition formelle) :

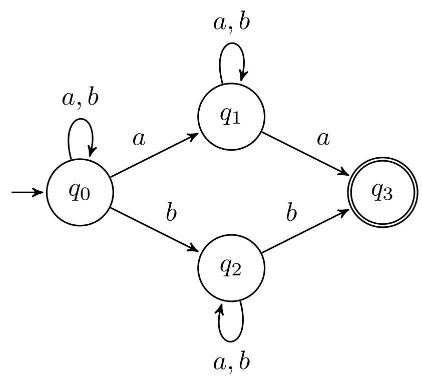
$\mathcal{A}_1$



$\mathcal{A}_2$



$\mathcal{A}_3$



Exercice 40 *Ambiguïté*

1. Montrer qu'une grammaire **régulière droite** est **contextuelle** de type 1.
2. Montrer qu'une grammaire **non ambiguë** est $LL(1)$.
3. Montrer qu'une grammaire est $LL(1)$ si elle n'est **pas ambiguë**.
4. Pour chacune des grammaires des deux exercices précédents, préciser sont type, si elle est **régulière droite**, **ambiguë**

Exercice 41 *Formes normales*

Les deux liens suivants sont à consulter et à étudier pour cet exercice :

— https ://fr.wikipedia.org/wiki/Forme_normale_de_Greibach

— https ://fr.wikipedia.org/wiki/Forme_normale_de_Chomsky

Soit la grammaire $G = (\Sigma, V, S, \mathcal{P})$ avec :

— $\Sigma = \{a, b\}$

— $V = \{S\}$

— $\mathcal{P} = \{$

$S \to aSbS,$

$S \to a,$

$S \to b,$

$S \to \varepsilon$

$\}$

1. Mettre G sous forme normale de Greibach, ce qui donne la grammaire G_1. L'algorithme est donné dans la section Construction de la page : https ://fr.wikipedia.org/wiki/Forme_normale_de_Greibach
2. Mettre G sous forme normale de Chomsky, ce qui donne la grammaire G_2 ; L'algorithme est donné dans la section Conversion de la page : https ://fr.wikipedia.org/wiki/Forme_normale_de_Chomsky
3. Mettre la grammaire G_1 sous forme normale de Chomsky, ce qui donne la grammaire G_3 ;
4. Comparer G_2 et G_3, que constatez-vous ?

8 TD 8 – Machines de Turing

Exercice 42 *Premières machines*
Ecrire des machines de Turing suivantes sur l'alphabet $\Sigma = \{0, 1\}$:

1. définir les nombres en base un (unaire) ;
2. donner une machine de Turing réalisant l'addition en base un ;
3. reconnaître si un mot (nombre) en binaire est pair ;
4. renverser un mot ;
5. ajouter 1 à un nombre en binaire.

Plusieurs rubans

On peut définir une machine de Turing à k rubans de façon similaire à une machine de Turing à un ruban. Si Q est l'ensemble d'états de la machine et Γ l'alphabet des rubans, la fonction de transitions est alors de la forme

$$\delta : Q \times \Gamma^k \to Q \times (\Gamma \times \{\mathrm{G}, \mathrm{D}\})^k .$$

On remarquera que chacune des têtes de lecture bouge indépendemment et que la transition dépend conjointement des lettres lues sur l'ensemble des rubans. La configuration initiale d'une telle machine est l'entrée écrite sur le premier ruban et tous les autres rubans vides.

Exercice 43 *Machines plus dures*
Ecrire des machines de Turing reconnaissant les langages suivants :

1. $\{a^n b^n c^n, n \in N^*\}$ écrire la machine avec un, puis deux rubans ;
2. $\{wcw^R, w \in \{a, b\}^*\}$ (w^R est le mot miroir de w) utiliser un seul ruban ;
3. $\{a^n b^{n^2}, n \in N\}$ utiliser trois rubans.

Exercice 44 *Machines de Turing qui calculent*

1. $a^n \to a^n b^n, n \geq 0$;
2. $w \to ww^R$ (w^R est le mot miroir de w) ;
3. $w \to ww$;
4. addition en binaire ;
5. soustraction en binaire ;
6. $n \to n^2$.

Exercice 45 *Quelques exemples*
Donner les machines de Turing reconnaissant les langages suivants

— $L_1 = \{a^i b^j a^i b^j, i, j \in \mathbb{N})\}$;
— $L_2 = \{w.w \mid w \in \{a, b\}^*\}$ (attention, pas de délimiteur entre les deux copies du mot) ;
— $L_3 = \{1^n \# 1^{n^2} | n > 0\}$.

Exercice 46 *Plusieurs rubans*
Donner des machines de Turing reconnaissant les langages suivants

— $L_4 = \{i\#j\#k | i, j, k$ entiers codés en binaire avec $k = i + j\}$ avec trois rubans ;
— $L_5 = \left\{a^{n^2} \mid n \in \mathbb{N}\right\}$ avec trois rubans ;
— $L_6 = \{i\#j\#k | i, j, k$ entiers codés en binaire avec $k = i \times j\}$ avec quatre rubans.

Exercice 47 *Automates et machines de Turing*
Montrer que tout langage régulier est décidable par une machine de Turing. Montrer que tout langage reconnaissable par un automate à pile déterministe est reconnaissable par une machine de Turing.

9 TD 9 – Machines de Turing - 2

Exercice 48 *Ruban bi-infini*
Une machine de Turing à ruban bi-infini est une machine de Turing dont le ruban est infini vers la gauche et vers la droite – par opposition à la définition standard où le ruban est infini seulement vers la droite. Montrer que tout langage reconnu par une machine de Turing à ruban bi-infini est reconnu par une machine de Turing standard.

Calcul de fonction
On dit qu'un machine de Turing M calcule la fonction f si, sur l'entrée x, M accepte avec $f(x)$ écrit sur son (premier dans le cas de plusieurs rubans) ruban.

Exercice 49 *Calculs*
Donner les machines de Turing calculant les fonctions suivantes :

— $f_1 : a^n \mapsto a^n b^n$;

— $f_2 : w \in \{a,b\}^* \mapsto w.\overline{w}$;

— $f_3 : w \in \{a,b\}^* \mapsto w.w$;

— $f_4 : (n,m) \mapsto n - m$ où n, m sont des entiers codés en binaire.

Exercice 50 *Composition*
Soient une machine de Turing M_f calculant la fonction f et une machine de Turing M_g calculant la foncction g. Donner une machine de Turing calculant $f \circ g$.

Exercice 51 *Composition et reconnaissance*
Soient M_L une machine de Turing reconnaissant le langage $L \subseteq \{a,b\}^*$ et M_f une machine de Turing calculant la fonction $f : \{a,b\}^* \to \{a,b\}^*$.

— Donner une machine de Turing qui reconnaît $f^{-1}(L) = \{x | f(x) \in L\}$;

— Donner une machine de Turing qui reconnaît $f(L) = \{y | \exists x \in L. f(x) = y\}$.

10 TD 10 – Décidabilité

Exercice 52 *Questions existentielles*
Rappelez ce que veut dire décidable, non décidable, semi-décidable.
L'un des deux langages suivants est décidable, lequel ? Justifiez.

— $L_1 = \{n |$ les décimales de π contiennent la séquence $1^n\}$;
— $L_2 = \{n |$ les décimales de π contiennent une sous séquence maximale de 1 de longueur $n\}$.

Exercice 53 *Clôtures*
Soient L_1 et L_2 des langages décidables, montrer que

— $L_1 \cup L_2$ est décidable ;
— $L_1 \cap L_2$ est décidable ;
— $\overline{L}_1$ est décidable.

Exercice 54 *Arrêt universel*
Donner une manière d'encoder une machine de Turing sur un ou plusieurs rubans.
Soit L_U le langage des codes de machines de Turing qui s'arrêtent sur toute entrée.

$$L_U = \{\langle M\rangle \mid \forall w \in \Sigma^* \ M \text{ s'arrête sur } w\}$$

Montrer que L_U est indécidable.

Exercice 55 *Arrêt existentiel*
Soit L_E le langage des codes de machines de Turing qui s'arrêtent sur au moins une entrée.

$$L_E = \{\langle M\rangle \mid \exists w \in \Sigma^* \ M \text{ s'arrête sur } w\}$$

Montrer que L_E est indécidable.

Exercice 56 *Arrêt en temps n*
Soit L_B le langage des codes de machines de Turing qui s'arrêtent sur une certaine entrée en temps n.

$$L_B = \{\langle M, w, n\rangle \mid \ M \text{ s'arrête sur } w \text{ en au plus } n \text{ étapes}\}$$

L_B est-il décidable ?

11 TD 11 – Thèse de Church et reconnaissance

Exercice 57 *Programmation*
Soient f_1, f_2, f_3 des fonctions calculables par une machine de Turing. Montrer que les fonctions suivantes sont calculable par une machine de Turing :

— $x \mapsto$`if` $f_1(x) = 1$ `then` $f_2(x)$ `else` $f_3(x)$

— $x \mapsto$ `while`$(f_1(x) = 1)\{x = f_2(x)\}$`return` $f_3(x)$

Exercice 58 *Automate à pile*
Montrer que tout langage reconnaissable par un automate à pile (pas forcément déterministe), est reconnaissable par une machine de Turing.

Exercice 59 *Automates à deux piles*
Montrer que tout langage reconnaissable par un automate à deux piles est reconnaissable par une machine de Turing. *Attention, l'automate n'est pas forcément déterministe.*

Exercice 60 *Automate à deux piles - bis*
Montrer que tout langage reconnaissable par une machine de Turing est reconnaissable par un automate à deux piles.

Exercice 61 *Décidabilité*
Soit $\langle \cdot \rangle$ un encodage des automates. Montrer que les langages suivants sont décidables :

$$A_{AFD} = \{\langle B \rangle \# w \mid B \text{ est un AFD reconnaissant } w\}$$

$$EQ_{AFD} = \{\langle B \rangle \# \langle C \rangle \mid \mathcal{L}(B) = \mathcal{L}(C)\}$$