POLYTECHNIQUE MONTRÉAL

UNIVERSITÉ D'INGÉNIERIE

LOG1810

STRUCTURES DISCRÈTES

TD 12 : MODÉLISATION COMPUTATIONNELLE

A2024

SOLUTIONNAIRE

Exercice 1 :

Pour chacune des grammaires ci-dessous, déterminez leur type en justifiant vos réponses, en commençant par les grammaires de type 3 et en progressant vers celles moins restrictives.

a) Considérez la grammaire $G_1 = (V_1, T_1, S, P_1)$ où $V_1 = \{a, b, S, A, B\}$ et $T_1 = \{a, b\}$. L'axiome est S, et l'ensemble des règles de production P_1 est le suivant :

$$\begin{aligned} S &\rightarrow BA \mid AB \mid CCC \\ AB &\rightarrow aBBa \\ BA &\rightarrow abccccC \\ CCC &\rightarrow cc \end{aligned}$$

Solution

- Type 3 : Elle n'est pas de type 3 car aucune des règles de production n'est de la forme $w_1 \rightarrow a|aA \; ou \; S \rightarrow \epsilon.$

- Type 2 : Elle n'est pas de type 2, à cause de la présence des règles de production $AB \rightarrow aBBa, BA \rightarrow abaaaaaC \; et \; CCC \rightarrow cc$ où la partie gauche n'est pas un symbole unique non terminal.

- Type 1 : Elle n'est pas de type 1, à cause de la présence de la règle de production $CCC \rightarrow cc$, la règle de production est telle que $l(CCC) > l(cc)$.

- Type 0 : Comme la grammaire n'est pas de type 1,2 ou 3, alors elle est de type 0.

Conclusion : G_1 est de type 0.

b) Considérez la grammaire $G_2 = (V_2, T_2, S, P_2)$ où $V_2 = \{a, b, S, A, B\}$ et $T_2 = \{a, b\}$. L'axiome est S, et l'ensemble des règles de production P_2 est le suivant :

$$
\begin{aligned}
S &\rightarrow aA \mid \epsilon \\
A &\rightarrow bB \mid \epsilon \\
B &\rightarrow aA \mid b
\end{aligned}
$$

Solution

- Type 3 : Elle n'est pas de type 3 à cause de la présence de la règle de production $A \rightarrow \epsilon$. Dans une grammaire de Type 3, seule l'axiome est autorisé à produire une chaîne vide.

- Type 2 : Elle est de type 2, car tous les symboles à gauche dans les productions sont des symboles uniques non terminaux.

Conclusion : G_2 est de type 2.

c) Considérez la grammaire $G_3 = (V_3, T_3, S, P_3)$ où $V_3 = \{a, b, S, A, B\}$ et $T_3 = \{a, b\}$. L'axiome est S, et l'ensemble des règles de production P_3 est le suivant :

$$
\begin{aligned}
S &\rightarrow aA \\
A &\rightarrow bB \\
B &\rightarrow b
\end{aligned}
$$

Solution

- Type 3 : Elle est de type 3 car toutes les règles de production sont de la forme $w_1 \rightarrow a|aA \; ou \; S \rightarrow \epsilon$.

Conclusion : G_3 est de type 3.

Exercice 2 :

Considérez la grammaire $G = (V, T, S, P)$ où $V = \{a, b, S, A, B, C, D\}$ et $T = \{a, b\}$. L'axiome est S, et l'ensemble des règles de production P est :

$$S \rightarrow aA$$
$$A \rightarrow bB \mid aA \mid bC \mid a$$
$$B \rightarrow bB \mid cC \mid b$$
$$C \rightarrow cC \mid \epsilon$$

Déterminez si les chaines suivantes peuvent être générés pas la grammaire G.

a) $w_1 = aabcc$

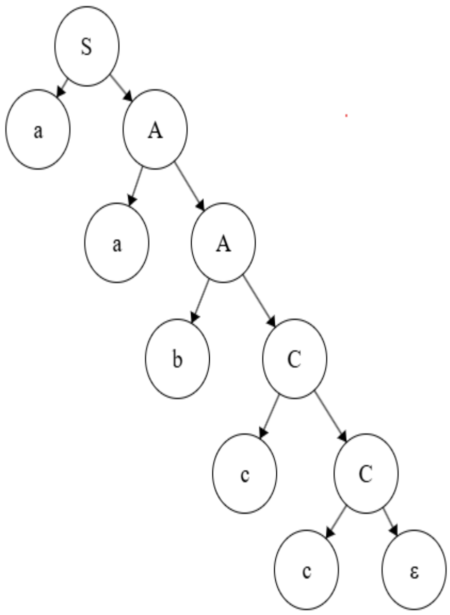
<u>Solution</u>

La grammaire G génère bien le mot w_1. Il est possible de procéder par la chaîne de dérivation ou l'arbre de dérivation pour le montrer.

Chaine de dérivation

$S \rightarrow aA$
$S \rightarrow aaA$ $(A \rightarrow aA)$
$S \rightarrow aabC$ $(A \rightarrow bC)$
$S \rightarrow aabcC$ $(C \rightarrow cC)$
$S \rightarrow aabcC$ $(C \rightarrow cC)$
$S \rightarrow aabc$ $(C \rightarrow \epsilon)$

Arbre de dérivation



b) $w_2 = ababcc$

Solution

La grammaire G ne génère pas le mot w_2. Pour montrer cela, nous analyserons les contraintes imposées par la grammaire.

La grammaire G impose des blocs cohérents pour a, b et c. Effectivement, la grammaire impose qu'après une séquence de a on peut uniquement :

- Avoir d'autres a.
- Avoir une séquence de b.

Après une séquence de b :

- Avoir d'autres n
- Avoir une séquence de c.

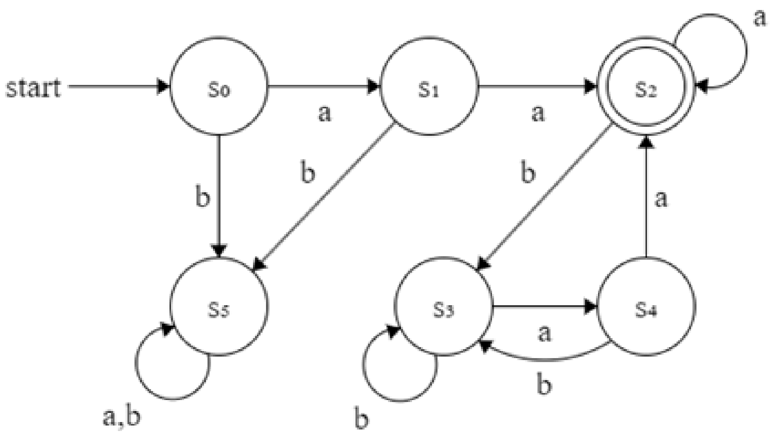
Après une séquence de c :

- Avoir une séquence de c.

Dans le mot $w_2 = ababcc$, on observe un a après un b (aba), ce qui est impossible avec les règles de production de la grammaire. Ainsi, le mot w_2 n'est pas dérivable par la grammaire G.

Exercice 3

Considérez l'ensemble des symboles terminaux $I = \{a, \mathrm{b}\}$. Donnez le langage reconnu par l'automate suivant. Justifiez votre réponse.



<u>Solution</u>

On remarque d'abord qu'il n'est pas possible d'atteindre l'état final à partir de l'état s_5, ainsi, si une séquence mène vers l'état s_5, un mot contenant cette séquence ne sera pas reconnu par l'automate. Les deux seules séquences qui mènent vers l'état s_5 sont les suivantes :

- Un mot commençant par b
- Un mot commençant par ab

Ainsi, nous notons que pour qu'un mot soit reconnu par l'automate, il doit commencer par la séquence aa.

Ensuite, étudions comment rejoindre l'état terminal s_2. Après la première séquence aa (première fois à l'état s_2), il est possible d'ajouter un nombre quelconque de a. Cependant, si de l'état s_2 on ajoute un b, nous allons à l'état s_3. Pour rejoindre l'état s_2 depuis l'état s_3, il faut avoir une séquence aa ($s_3 \rightarrow s_4 \rightarrow s_2$). Ainsi, les mots reconnus par l'automate doivent se terminer par la séquence aa. Entre les deux séquences de aa (début et fin), il est possible d'avoir n'importe quelle combinaison de a et b. Ainsi, mots reconnus sont d'une des deux formes suivantes :

- aa
- aaa
- aa (n'importe quelle séquence de a et b) aa

Ainsi, l'automate reconnait les mots commençants et terminant par la séquence aa. Cela correspond au langage suivant :

$$L = aa + \mathrm{aaa} + (aa(a+b)^*aa)$$

Exercice 4 :

Pour chacun des langages suivants, construisez un automate fini déterministe le reconnaissant. Considérez l'ensemble des symboles terminaux $I = \{0,1\}$. Donnez ensuite l'expression régulière représentant ce langage. Justifiez vos réponses.

a) Le langage des mots qui contiennent un nombre de 1 divisible par 3.

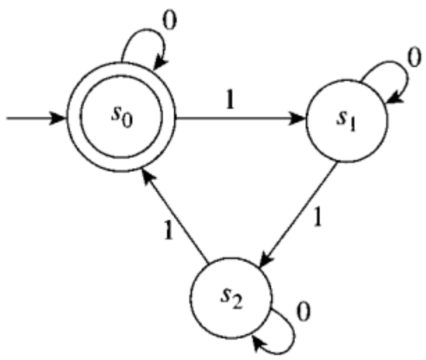
Solution

Considérons 3 états possibles. L'état correspondant à un nombre de 1 divisible par 3 (nombre de 1 congru à 0 modulo 3) : s_0. L'état correspondant à un nombre de 1 congru à 1 modulo 3 : s_1. Et finalement l'état correspondant à un nombre de 1 congru à 2 modulo3 : s_2. Le seul état terminal est l'état s_0 car le seul état ayant un nombre de 1 divisible par 3.

Initialement, nous avons zéro 1, ce qui est un nombre de 1 divisible par 3. L'état de départ est donc s_0 qui est un état terminal (à l'état s_0, on a une chaine contenant un nombre de 1 divisible par 3). On reste à s_0 tant qu'on n'ajoute pas un 1. Ensuite, quand on ajoute un 1, on passe à l'état s_1 et on reste à cet état tant que l'on n'ajoute pas un autre 1.

Si on ajoute un autre 1, on passe à l'état s_2 et comme précédemment, on reste à s_2 tant que l'on n'ajoute pas un nouveau 1. Dans ce cas, on passe à l'état s_0.

De retour à l'état s_0, on a rajouté trois 1, on a donc encore un nombre de 1 divisible par 3. L'automate fini est donc simplement :



Le langage reconnu est donc :

$$L = 0^* + (0^*10^*10^*10^*)^*$$

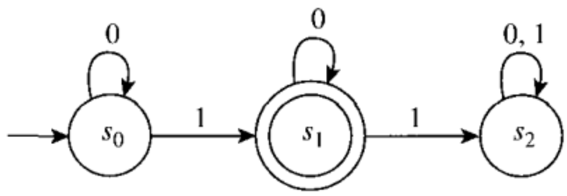
b) Le langage des mots qui contient exactement un seul 1.

Solution

Considérons l'état de départ s_0 et l'état s_1 qui est l'état où le mot d'entrée contient un seul 1.

Initialement, on se trouve à l'état s_0. On reste à s_0 tant qu'on n'ajoute pas un 1. Ensuite, quand on ajoute un 1, on passe à l'état s_1. L'état s_1 est terminal car il reconnaît une chaine contenant exactement un seul 1.

De l'état s_1, tant que l'on ajoute des 0, on demeure dans cet état. Cependant, si on ajoute un 1, alors on doit quitter l'état s_1 (le mot ne contient plus un seul 1). On remarque cependant qu'il n'est pas possible de retourner à l'état s_0 car de l'état s_0 il est possible de retourner à l'état s_1 or notre mot contient plus qu'un seul 1. On ajoute ainsi un état s_2 à partir duquel il n'est pas possible de retourner à l'état s_1. L'automate fini est donc :



Le langage reconnu est donc :

$$L = 0^*10^*$$

Exercice 5 :

Pour les langages suivants, proposez une grammaire $G = (V, T, S, P)$ qui engendre le langage. Précisez *V*, *T*, *S* et *P*.

a) Soit le langage construit sur l'alphabet $I = \{a, b\}$ correspondant à un a suivi d'un nombre impair de b.

Solution *

$G = (V, T, S, P)$
$V = \{a, b, A, S\}$
$T = \{a, b\}$
S est l'axiome
P est constitué des productions suivantes :

$S \rightarrow abA$
$A \rightarrow bbA \mid \epsilon$

*plusieurs solutions sont possibles.

b) Soit le langage $L = \{<^a \#^b >^a \mid a, b \in \mathbb{N}\}$ construit sur l'alphabet $I = \{<, >, \#\}$

Solution *

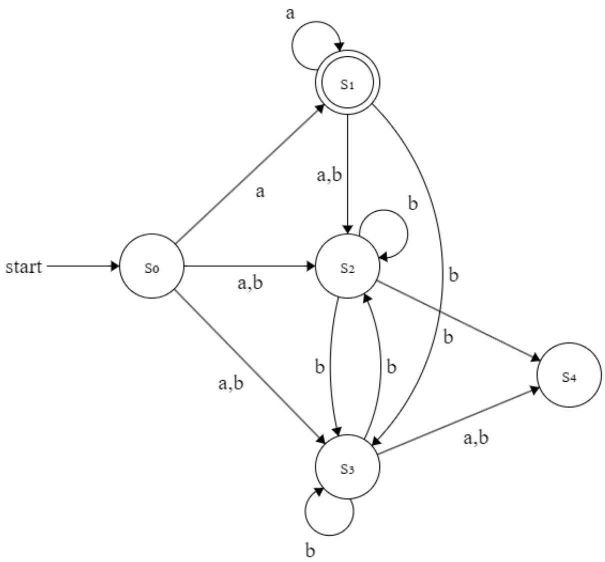
$G = (V, T, S, P)$
$V = \{<, >, \#, A, S\}$
$T = \{<, >, \#\}$
S est l'axiome
P est constitué des productions suivantes :

$S \rightarrow < S > \mid A \mid \epsilon$
$A \rightarrow \#A \mid \#$

*plusieurs solutions sont possibles.

Exercice 6 :

Transformez en automate déterministe l'automate suivant.



Solution

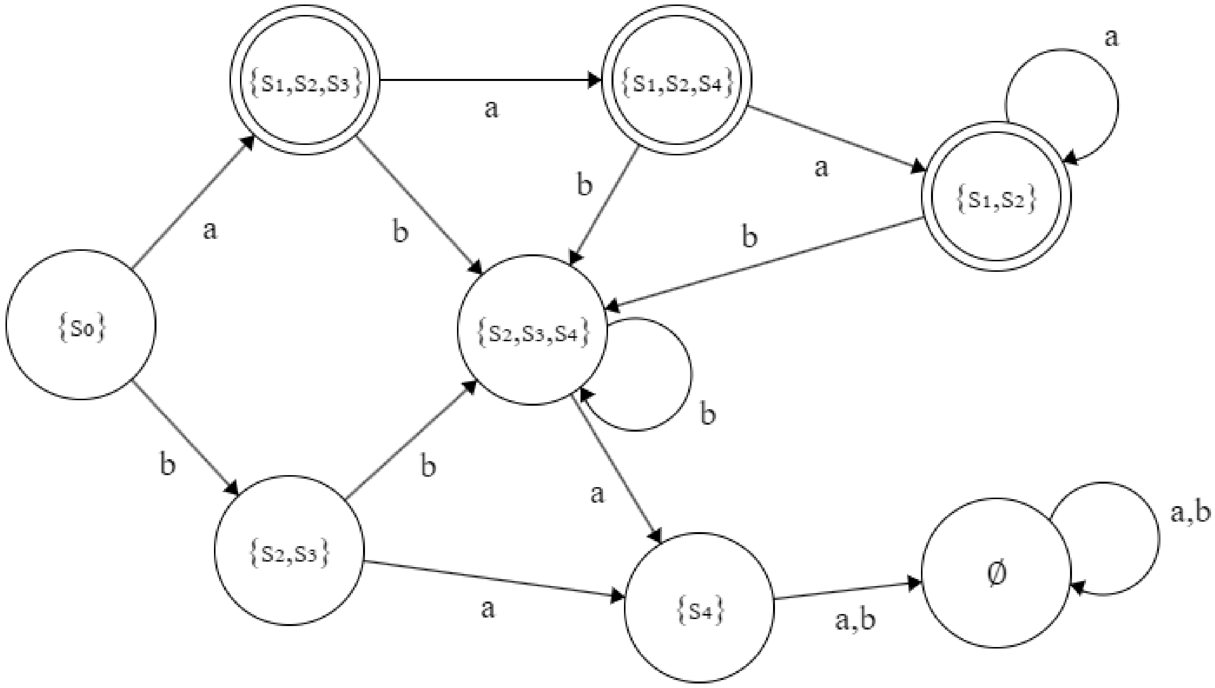
Table d'états-transition de l'automate initial :

États	Entrées	
	a	b
$\rightarrow S_0$	$\{S_1, S_2, S_3\}$	$\{S_2, S_3\}$
$\leftarrow S_1$	$\{S_1, S_2\}$	$\{S_2, S_3\}$
S_2	$\emptyset$	$\{S_2, S_3, S_4\}$
S_3	$\{S_4\}$	$\{S_2, S_3, S_4\}$
S_4	$\emptyset$	$\emptyset$

Table d'états-transition de l'automate déterministe :

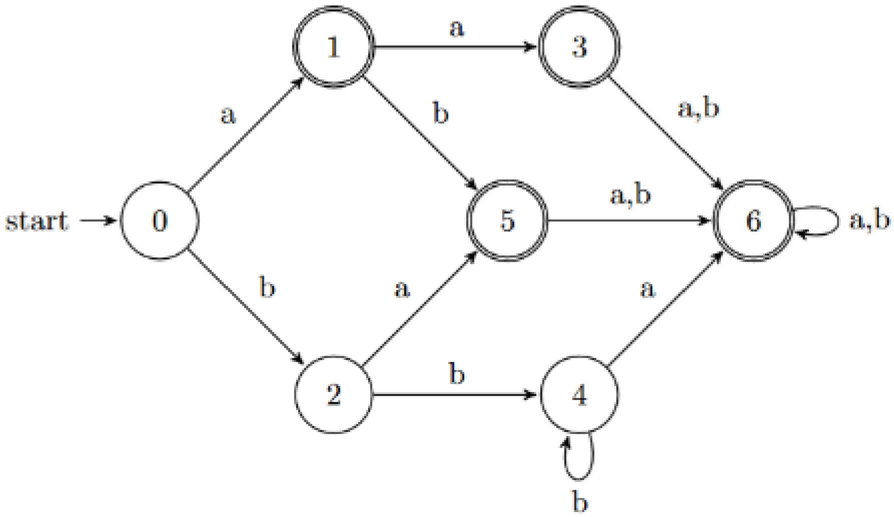
États	Entrées	
	a	b
$\rightarrow \{S_0\}$	$\{S_1, S_2, S_3\}$	$\{S_2, S_3\}$
$\leftarrow \{S_1, S_2, S_3\}$	$\{S_1, S_2, S_4\}$	$\{S_2, S_3, S_4\}$
$\{S_2, S_3\}$	$\{S_4\}$	$\{S_2, S_3, S_4\}$
$\leftarrow \{S_1, S_2, S_4\}$	$\{S_1, S_2\}$	$\{S_2, S_3, S_4\}$
$\{S_2, S_3, S_4\}$	$\{S_4\}$	$\{S_2, S_3, S_4\}$
$\{S_4\}$	$\emptyset$	$\emptyset$
$\leftarrow \{S_1, S_2\}$	$\{S_1, S_2\}$	$\{S_2, S_3, S_4\}$

L’automate est :



Exercice 7 (facultatif) :

Donnez la grammaire G qui génère le langage reconnu par l'automate suivant. Vous devez préciser l'alphabet V, l'ensemble des symboles terminaux T, l'axiome S, et l'ensemble des règles de production P.



Solution

Soit les symboles non terminaux associés aux états comme suit :

- État 0 : Symbole non terminal S, axiome de la grammaire
- État 1 : Symbole non terminal A
- État 2 : Symbole non terminal B
- État 3 : Symbole non terminal C
- État 4 : Symbole non terminal D
- État 5 : Symbole non terminal E
- État 6 : Symbole non terminal F

Nous avons les ensembles suivants :

N = {S, A, B, C, D, E, F},

T = {a, b},

V = {a, b, S, A, B, C, D, E, F}.

Les productions de **P** sont :

S → a | aA | bB

A → a | aC | b | bE

B → a | aE | bD

C → a | b | aF | bF

D → a | aF | bD

E → a | b | aF | bF

F → a | b | aF | bF