Théorie des Langages - Feuille n° 5
AUTOMATES À PILE
CORRECTION

Exercice 1 - Trouvez une grammaire algébrique pour chacun des langages suivants :

1. $L = \{a^k b^j c^l | k \leq j \text{ ou } j \leq l\}$

On doit décomposer le problème en sous-problèmes plus simples puis les combiner. Pour cela, on commence par remarquer que L peut être décomposé en l'union deux langages $L = L_1 + L_2$ où

$$L_1 = \{a^k b^j c^l | k \leq j, l \geq 0\}$$
$$L_2 = \{a^k b^j c^l | j \leq l, k \geq 0\}$$

On doit écrire les deux grammaires correspondantes à chacun de ces langages. Si S est l'axiome de la grammaire engendrant L, S_1 l'axiome de la grammaire engendrant L_1 et S_2 l'axiome de la grammaire engendrant L_2 alors on a $S \to S_1|S_2$ (car $L = L_1 + L_2$).

- $L_1 = \{a^k b^j c^l | k \leq j, l \geq 0\}$

  On peut constater que L_1 est le produit de deux langages : $L_1 = L_{1_1}.L_{1_2}$, avec $S_1 \to AB$
  - $L_{1_1} = \{a^k b^j | k \leq j\}$ dont on peut écrire la grammaire : $A \to \varepsilon|aAb|Ab$
  - $L_{1_2} = \{c^l | l \geq 0\}$ dont on peut écrire la grammaire : $B \to \varepsilon|cB$
- $L_2 = \{a^k b^j c^l | j \leq l, k \geq 0\}$

  L_2 est aussi le produit de deux langages : $L_2 = L_{2_1}.L_{2_2}$, avec $S_2 \to CD$
  - $L_{2_1} = \{a^k | k \geq 0\}$ dont on peut écrire la grammaire : $C \to \varepsilon|aC$
  - $L_{2_2} = \{b^j c^l | j \leq l\}$ dont on peut écrire la grammaire : $D \to \varepsilon|bDc|Dc$

Il faut maintenant rassembler tous les éléments : on obtient $G = \langle V, \Sigma, P, S\rangle$, $\Sigma = \{a, b, c\}$, $V \setminus \Sigma = \{S, S_1, S_2, A, B, C, D\}$, $P =$

$$\begin{array}{lcl} S & \to & S_1|S_2 \\ S_1 & \to & AB \\ S_2 & \to & CD \\ B & \to & \varepsilon|cB \\ A & \to & \varepsilon|aAb|Ab \\ C & \to & \varepsilon|aC \\ D & \to & \varepsilon|bDc|Dc \end{array}$$

2. $L = \{a^n b^m | n > 0, m = n \text{ ou } m = 2n\}$

Le raisonnement est le même qu'à la question précédente. L peut être décomposé en union de deux langages : $L = L_1 + L_2$, avec $S \to A|B$

- $L_1 = \{a^n b^m | n > 0, m = n\}$
  - $A \to ab|aAb$
- $L_2 = \{a^n b^m | n > 0, m = 2n\}$
  - $B \to abb|aBbb$

On obtient $G = \langle V, \Sigma, P, S \rangle$, $\Sigma = \{a, b\}$, $V \setminus \Sigma = \{S, A, B\}$, $P =$

$$\begin{array}{lcl} S & \to & A|B \\ A & \to & ab|aAb \\ B & \to & abb|aBbb \end{array}$$

3. $L = \{a^n b^r a^s b^t | n = s \text{ ou } r = t\}$

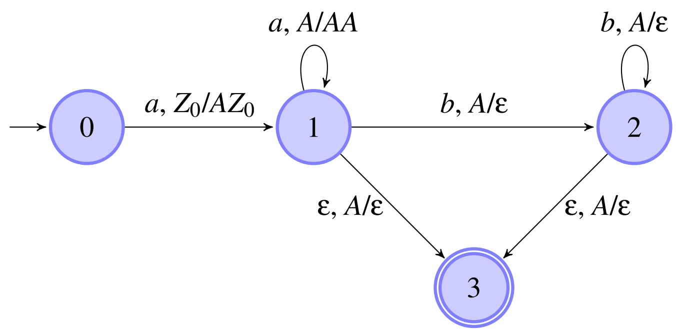
$L = L_1 + L_2$, avec $S \to S_1|S_2$

- $L_1 = \{a^n b^r a^s b^t | n = s, r \geq 0, t \geq 0\}$
  - $L_1 = L_{1_1}.L_{1_2}$, avec $S_1 \to AC$
    - $L_{1_1} = \{a^n b^r a^s | n = s, r \geq 0\}$
      - $A \to \varepsilon|aAa|bB$
      - $B \to \varepsilon|bB$
    - $L_{1_2} = \{b^t | t \geq 0\}$ – $C \to \varepsilon|bC$
- $L_2 = \{a^n b^r a^s b^t | r = t, n \geq 0, s \geq 0\}$
  - $L_2 = L_{2_1}.L_{2_2}$, avec $S_2 \to DE$
    - $L_{2_1} = \{a^n | n \geq 0\}$ – $D \to \varepsilon|aD$
    - $L_{2_2} = \{b^r a^s b^t | r = t, s \geq 0\}$
      - $E \to \varepsilon|bEb|aF$
      - $F \to \varepsilon|aF$

On obtient $G = \langle V, \Sigma, P, S \rangle$, $\Sigma = \{a, b, c\}$, $V \setminus \Sigma = \{S, S_1, S_2, A, B, C, D, E, F\}$, $P =$

$$\begin{array}{lcl} S & \to & S_1|S_2 \\ S_1 & \to & AC \\ S_2 & \to & DE \\ A & \to & \varepsilon|aAa|bB \\ B & \to & \varepsilon|bB \\ C & \to & \varepsilon|bC \\ D & \to & \varepsilon|aD \\ E & \to & \varepsilon|bEb|aF \\ F & \to & \varepsilon|aF \end{array}$$

Exercice 2 - Soit l'automate à pile suivant reconnaissant le langage L par état final :



<u>*Indications :*</u> *Pour rappel :*

- *Acceptation par pile vide : un mot est accepté s'il est entièrement lu et que la pile est totalement vide, quel que soit l'état dans lequel on s'arrête*
- *Acceptation par état final : un mot est accepté s'il est entièrement lu et qu'on est dans un état final, quel que soit l'état de la pile*

1. Cet automate est-il déterministe ?

A un état donné, il faut prendre en compte le symbole en sommet de pile pour savoir quel symbole lire. Ici, l'automate **n'est pas déterministe** car à partir de l'état 1, on peut lire a ou ε avec A en sommet de pile. C'est à dire qu'à partir d'un même état (l'état 1) et d'un même sommet de pile (symbole A), on peut lire deux symboles (a ou ε).

2. Donner les différentes étapes de reconnaissance du mot $aaab$

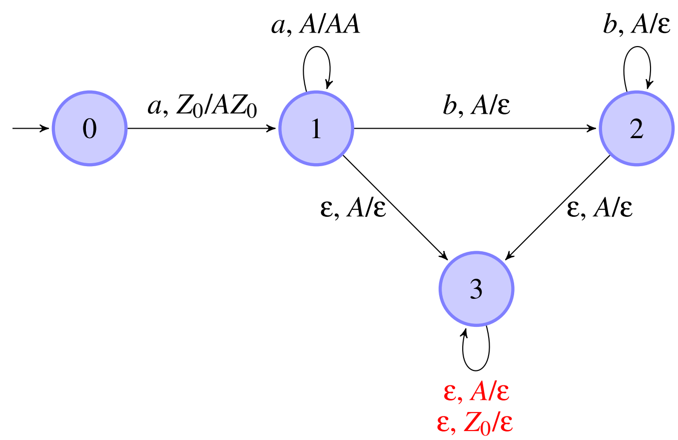
$(aaab,0,Z_0) \mapsto (aab,1,AZ_0) \mapsto (ab,1,AAZ_0) \mapsto (b,1,AAAZ_0) \mapsto (\varepsilon,2,AAZ_0) \mapsto (\varepsilon,3,AZ_0)$ Le mot est reconnu car on arrive avec ε sur 3 qui est un état final.

3. Quel langage est généré par cet automate ?

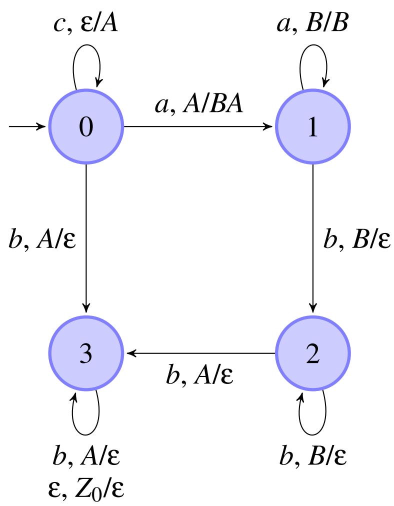
Il faut tout d'abord remarquer que l'on empile des symboles A tant qu'on lit le symbole a puis on dépile en lisant le symbole b. De plus, il ne peut pas y avoir plus de b que de a. $L = \{a^n b^m \mid n > 0, m \geq 0, m < n\}$

4. Modifier l'automate de façon à avoir une reconnaissance par pile vide

Au lieu de terminer parce qu'on arriverait sur l'état 3 en état final, il faut que l'on modifie l'automate de sorte que sur 3 la pile puisse être vide.
On peut remarquer que si le mot contient plus de a que de b, en arrivant sur l'état 3 dans l'automate ci-dessus, la pile ne sera pas vide. On rajoute donc deux transitions sur l'état 3 qui vont permettre de vider la pile : on dépile soit A soit Z_0 mais on ne rempile rien.



Exercice 3 - Soit l'automate à pile suivant reconnaissant le langage L par pile vide :



1. Cet automate est-il déterministe ?

> Oui, une transition par lettre lue et sommet de pile.
> Même si l'état 2 contient deux transitions b, la tête de pile de ces deux transitions n'est pas la même : d'un côté on a A de l'autre B.

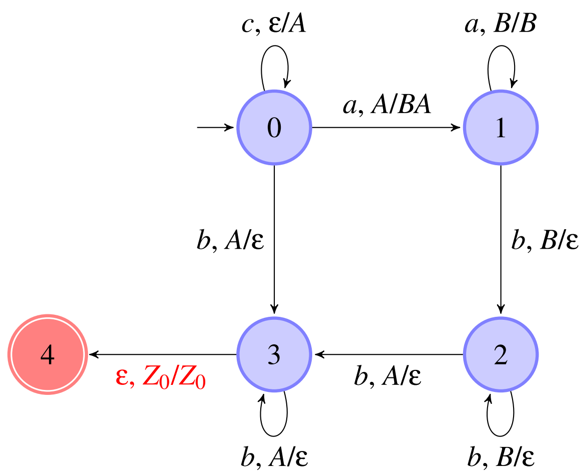
2. Donner les différentes étapes de reconnaissance du mot *caabbb*

> $(caabbb, 0, Z_0) \mapsto (aabbb, 0, AZ_0) \mapsto (abbb, 1, BAZ_0) \mapsto (bbb, 1, BBAZ_0) \mapsto (bb, 2, BAZ_0) \mapsto (b, 2, AZ_0) \mapsto (\varepsilon, 3, Z_0) \mapsto (\varepsilon, 3, \varepsilon)$
> Le mot est reconnu car à la fin de la lecture, on est avec le mot vide et la pile est vide.

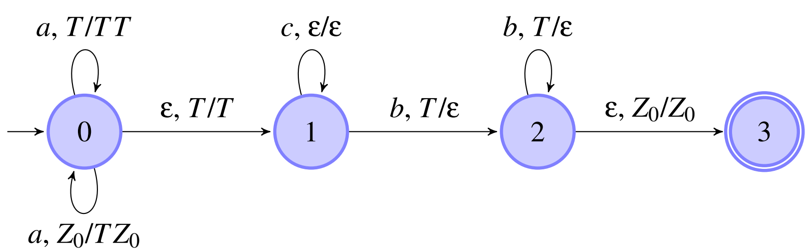
3. Quel langage est généré par cet automate ?

> Cette fois, on empile le symbole A en lisant des c puis on continue d'empiler des A en lisant des a et on dépile A en lisant b. On ne dépile Z_0 que s'il reste ε à lire. C'est à dire s'il y a eu autant de b lu qu'on avait lu de c et a.
>
> $$L = \{c^n a^m b^{(n+m)} | n > 0, m \geq 0\}$$

4. Modifier l'automate de façon à avoir une reconnaissance par état final

> A la question précédente, on a vu que l'on dépilait Z_0 qu'en lisant ε. Pour obtenir un automate qui accepte par état final au lieu de pile vide, on va rajouter un état 4 final avec la transition ε, Z_0/Z_0. C'est à dire qu'on ne vide plus complètement la pile mais on transite vers un état final si Z_0 est en tête de pile.
>
> 
>

Exercice 4 - Soit l'automate à pile suivant reconnaissant le langage L par état final :



1. Cet automate est-il déterministe ?

> Non : à partir de l'état 1, on peut lire a ou ε avec T en sommet de pile.

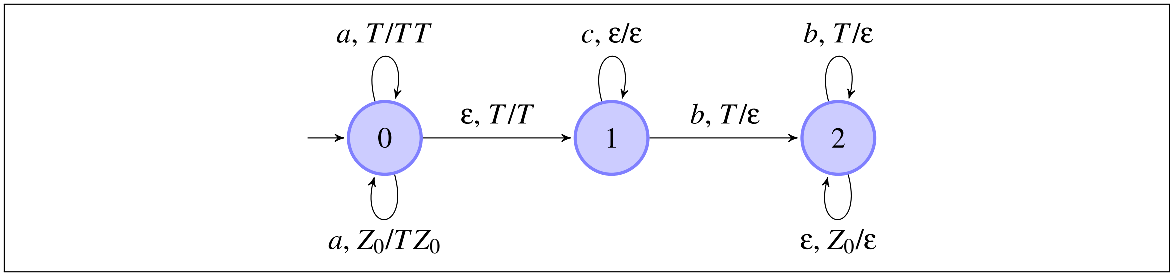
2. Donner les différentes étapes de reconnaissance des mot *aacbb* et *ab*

> $(aacbb, 0, Z_0) \mapsto (acbb, 0, TZ_0) \mapsto (cbb, 0, TTZ_0) \mapsto (cbb, 1, TTZ_0) \mapsto (bb, 1, TTZ_0) \mapsto (b, 2, TZ_0) \mapsto (\varepsilon, 2, Z_0) \mapsto (\varepsilon, 3, Z_0)$
> $(ab, 0, Z_0) \mapsto (b, 0, TZ_0)) \mapsto (b, 1, TZ_0) \mapsto (\varepsilon, 2, Z_0) \mapsto (\varepsilon, 3, Z_0)$

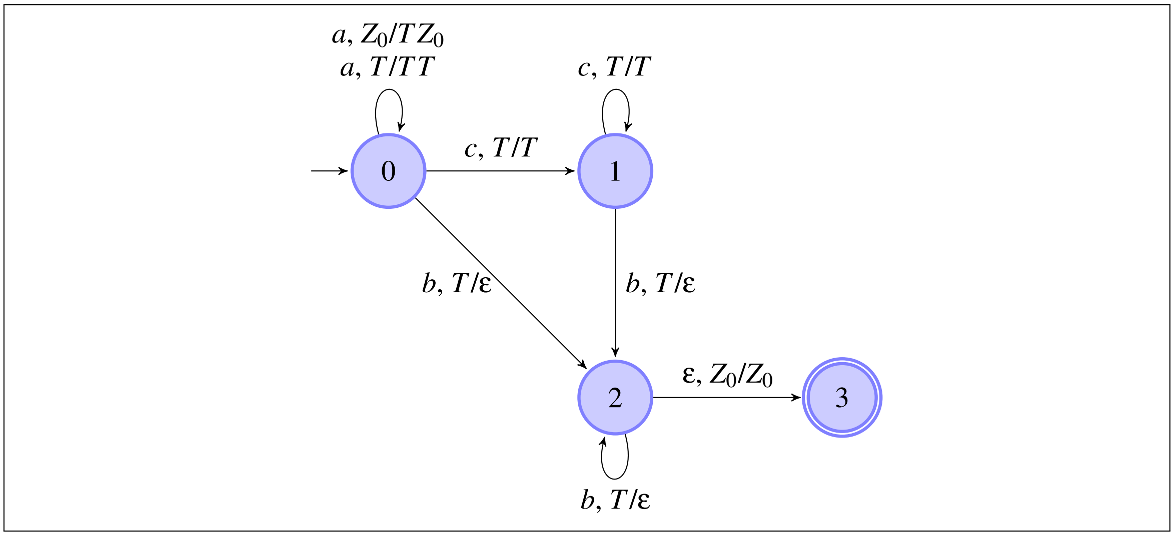
3. Quel langage est généré par cet automate ?

> $L = \{a^n c^m b^n | n > 0, m \geq 0\}$

4. Modifier l'automate de façon à avoir une reconnaissance par pile vide



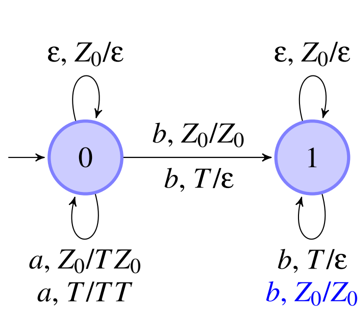
5. Modifier l'automate de façon à avoir un automate déterministe

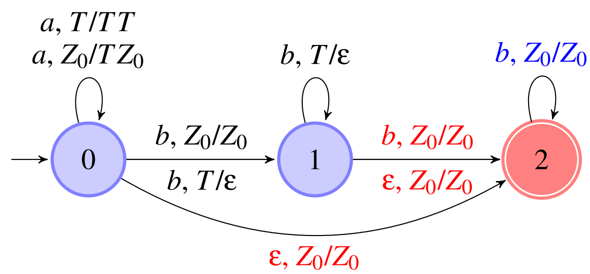


Exercice 5 - Donnez deux automates à pile (acceptation par pile vide ; par état final) qui reconnaissent chacun des langages suivants :

1. $L_1 = \{a^n b^m | n, m \geq 0 \text{ et } n \leq m\}$

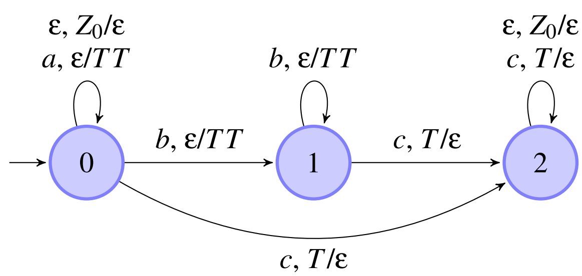
Une aide peut être de construire d'abord l'automate qui reconnait le langage $L_1 = \{a^n b^m | n, m \geq 0$ et $n = m\}$. Pour cela, on peut rajouter un symbole (par exemple T) dans la pile à chaque a lu puis dépiler ce même symbole lorsqu'un b est lu.
Ensuite il faut rajouter le traitement de $n \leq m$ (la transition qui permet de continuer à lire des b est en bleu).

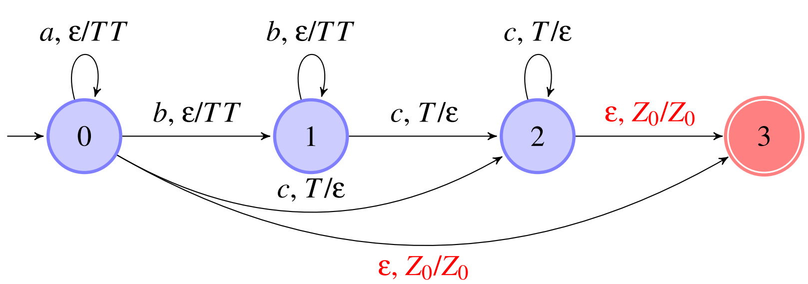




2. $L_2 = \{a^n b^m c^{2(n+m)} | n, m \geq 0\}$

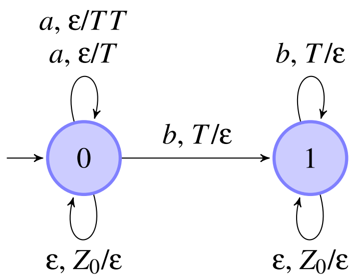
Ici on souhaite lire des mots où le nombre de c est le même que la somme du nombre de a et du nombre de b. L'idée est donc d'empiler un symbole (ici T) a chaque fois qu'on lit un a puis un b et enfin de dépiler un symbole lorsqu'on lit un c.

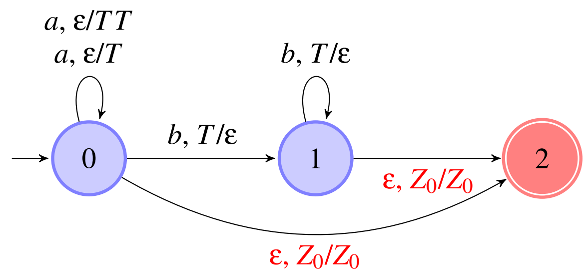




3. $L_3 = \{a^n b^m | n,m \geq 0 \text{ et } n \leq m \leq 2n\}$

Cette fois le nombre de b est conditionné par le nombre de a. On a déjà vu (question 1) comment avoir $n \leq m$, il faut voir comment on peut avoir $m \leq 2n$.
Une solution est d'empiler deux symboles (ici T) lorsqu'on lit un a de sorte qu'en lisant un b si on ne dépile qu'un seul symbole on a exactement $m = 2n$.
Il faut ensuite combiner le tout.

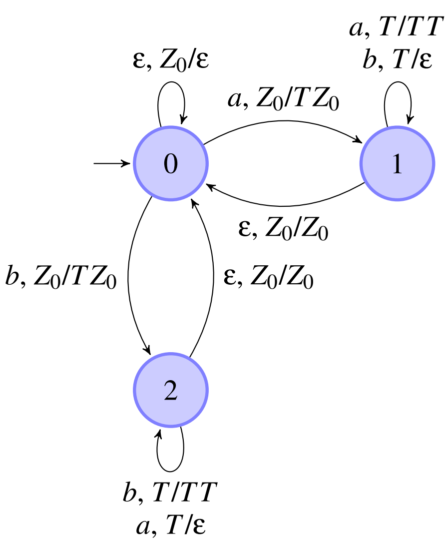


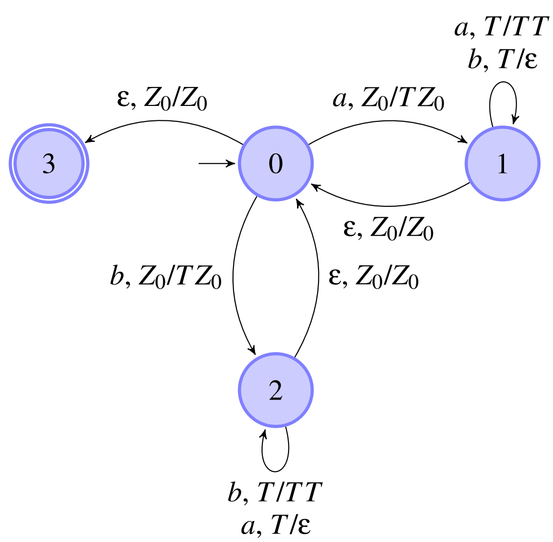


4. $L_4 = \{w \in (a+b)^* || w|_a = |w|_b\}$

Dans cet exemple, on on souhaite lire des mots qui contiennent autant de a que de b, quel que soit l'ordre de lecture. L'idée est donc d'écrire un automate à 3 états :

- Dans l'état 0, on sait qu'on a lu le même nombre de a que de b (état initial, cas du mot vide). Dans ce cas, les mots peuvent être acceptés, on a Z_0 en sommet de pile
- Dans l'état 1, on a lu plus de a que de b.
- Dans l'état 2, on a lu plus de b que de a



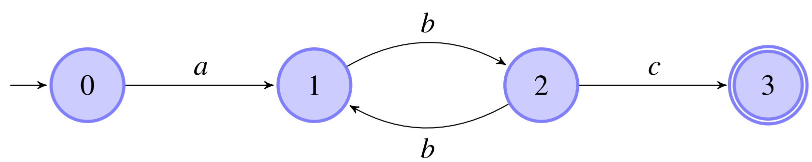


Exercice 6 - Soient $L_1 = \{ab^{2p+1}c | p \geq 0\}$ et $L_2 = \{a^n b^m c^m d^n | m, n > 0\}$

<u>Indications :</u> Rappel : Les langages réguliers sont des sous-ensemble strict des langages algébriques.
Vous devez d'abord avoir une intuition sur le langage.
Si vous pensez que le langage est régulier, vous devez trouver soit un automate fini (déterministe ou non), soit une expression régulière, soit une grammaire régulière pour génerer ce langage.
Si vous pensez que ce langage n'est pas régulier, mais est algébrique, vous devez trouver soit une grammaire hors-contexte, soit un automate à pile pour générer ce langage.
Pour répondre à cet exercice, vous devez ensuite trouver les automates correspondant si vous ne les avez pas trouvé en première étape.

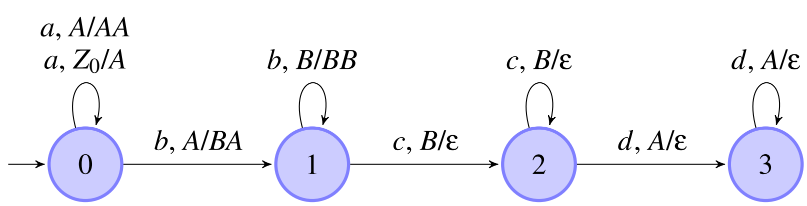
1. L_1 est-il régulier ? Construire l'automate qui reconnaît L_1

L_1 est l'ensemble des mots qui commencent par a, sont suivis d'un nombre impair de b et finissent par la lettre c. Dans un précédent TD nous avions déjà écrit l'automate qui reconnait un nombre impair d'un symbole. Il faut simplement rajouter la lecture d'un a au début et d'un c à la fin. L_1 est donc régulier.



2. L_2 est-il hors-contexte ? Construire l'automate qui reconnaît L_2

Le nombre de a doit être retenu pour avoir le même nombre de d, pareil pour le nombre de b avec le nombre de c. Ce n'est pas possible de le faire sans pile, L_2 est donc hors-contexte. Acceptation par pile vide :



Acceptation par état final :