Automates, langages et compilation

Automates à pile

Isabelle Ryl

2024 – 2025

Cours de L3 - Université Paris Dauphine-PSL

1. Automates à pile

Automates à pile

Idée des automates à pile

- Une extension des automates finis
- Permettant de pallier une limitation importante des automates finis : le nombre fini de configurations encodables
- En utilisant une mémoire annexe sous forme de pile

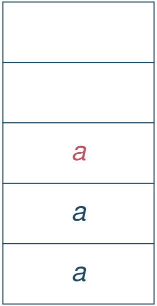
Rappel

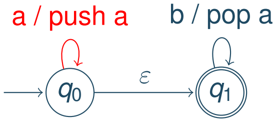
Une pile est une structure de données manipulable par 2 opérations : ajouter un élément en sommet de pile (push), retirer l'élément sommet de pile (pop)

Idée générale - $a^n b^n$

- Pour reconnaitre $a^n b^n$ il nous manque dans un automate fini la possibilité de « compter » le nombre de a puis de s'assurer que celui-ci est équivalent au nombre de b

➥ On va utiliser une pile comme mémoire annexe





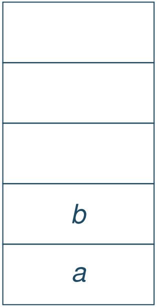
Exemple de mot : *aaabbb*

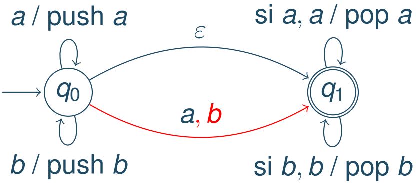
Exemple - Palindromes

Le langage des palindromes n'est pas régulier, mais on peut le reconnaitre avec un automate qui :

- lit une à une les lettres du mot de gauche à droite (comme un automate fini)
- place les lettres une à une dans la pile
- lorsqu'il décide qu'il a atteint le milieu du mot, « change » d'état et à partir de là
  - pour chaque lettre lue, dépile une lettre de la pile et vérifie que les deux sont égales
- s'arrête quand le mot d'entrée et la pile sont vides et que l'automate est dans un état final

Idée générale - Palindromes





Exemple de mot : *abbba*

Automates à pile

Un automate à pile est un septuplet $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ dans lequel

- Σ est l'alphabet fini des symboles d'entrée
- Z est un alphabet fini des symboles utilisés sur la pile
- $z_0 \in Z$ est le symbole initial de la pile
- Q un ensemble fini d'états, $q_0 \in Q$ l'état initial, $F \subseteq Q$ un ensemble d'états finaux
- δ la relation de transition avec $\delta \subseteq (Q \times (\Sigma \cup \{\varepsilon\}) \times Z) \times (Q \times Z^*)$

Exemple. Construction pour $a^n b^n$

$\mathcal{A} = (\{a, b\}, \{a, z_0\}, z_0, \{q_0, q_1\}, q_0, \{q_1\}, \delta)$

Automates à pile - Transitions

Les transitions sont des éléments de $(Q \times (\Sigma \cup \{\varepsilon\}) \times Z) \times (Q \times Z^*)$, soit :

- la transition effectuée dépend de
  - la lettre lue dans le mot d'entrée, un symbole de Σ ou ε, qui est appelée étiquette de la transition
  - l'état q de l'automate
  - le symbole z lu en sommet de pile (seul le sommet de pile est visible)
- lorsque la transition est effectuée
  - l'automate est dans un nouvel état de Q
  - le sommet de pile z a été remplacé par un mot de l'alphabet de pile Z^*

Exemple. Construction de δ pour $a^n b^n$:

$$
\begin{array}{lcl}
(q_0, z_0) & \xrightarrow{a} & (q_0, a) \text{ notation pour } (q_0, a, z_0) \to (q_0, a) \\
(q_0, a) & \xrightarrow{a} & (q_0, aa) \\
(q_0, z_0) & \xrightarrow{\varepsilon} & (q_1, z_0) \\
(q_0, a) & \xrightarrow{\varepsilon} & (q_1, a) \\
(q_1, a) & \xrightarrow{b} & (q_1, \varepsilon)
\end{array}
$$

Automates à pile - Configuration

Comme dans le cas des automates finis, une configuration de l'automate à pile correspond à l'état de l'automate à un instant donné, *i.e.* son état interne, le mot restant à lire et l'état de la pile :

Une **configuration** d'un automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ est un élément de $Q \times \Sigma^* \times Z^*$

Convention. Si $u \in Z^*$ représente l'état de la pile, la première lettre de u est le sommet de pile, la dernière lettre de u est le fond de la pile

La **configuration initiale** d'un automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ est (q_0, u, z_0) où u est le mot à lire

Automates à pile - Calcul

Un **pas de calcul** ou une **étape de calcul** d'un automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ est une paire de configurations (q, au, zv) et $(q', u, z'v)$ telles que $(q, a, z, q', z') \in \delta$ (ou $(q, z) \xrightarrow{a} (q'z') \in \delta$) et on note

$$(q, au, zv) \xrightarrow{a} (q', u, z'v)$$

Un **calcul** d'un automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ est une suite de pas de calcul

$$(q_1, x_1 \dots x_{n-1}u, z_1) \xrightarrow{x_1} (q_2, x_2 \dots x_{n-1}u, z_2) \xrightarrow{x_2} \dots \xrightarrow{x_{n-1}} (q_n, u, z_n)$$

La concaténation des étiquettes des transitions $x_1 x_2 \dots x_{n-1}$ est l'étiquette du calcul, et on note :

$$(q_1, x_1 \dots x_{n-1}u, z_1) \xrightarrow[*]{x_1 \dots x_{n-1}} (q_n, u, z_n)$$

$$\textit{ou} \qquad (q_1, x_1 \dots x_{n-1}u, z_1) \xrightarrow[*]{} (q_n, u, z_n)$$

Calculs - Remarques

Dans la notation d'un pas de calcul $(q, au, zv) \xrightarrow{a} (q', u, z'v)$,

- le sommet de la pile est à gauche, il s'agit d'une convention, l'inverse est possible
- si $|z'| = 0$, z a été dépilé
- si $|z'| = 1$, z a été remplacé en sommet de pile (éventuellement par lui-même)
- si $|z'| > 1$, la transition a empilé un ou plusieurs symboles (en dépilant z ou pas)

Critères d'acceptation

Dans les exemples précédents, le mot était accepté lorsque l'état atteint était un état final et que la pile était vide, il s'agit d'un cas particulier, plusieurs critères sont possibles :

- acceptation par **état final**, dans ce cas l'état de la pile peut être quelconque
- acceptation par **pile vide**, dans ce cas l'état de l'automate peut être quelconque
- la combinaison des deux

Langage reconnu par état final

Un mot u est **accepté par état final** par l'automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$, s'il existe un calcul $(q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, z)$ avec $q \in F$ et $z \in Z^*$

Le langage **accepté par état final** par l'automate à pile $\mathcal{A}$ est l'ensemble des mots acceptés par état final par $\mathcal{A}$, on le note $\mathcal{L}^{\mathcal{F}}(\mathcal{A})$

$$\mathcal{L}^{\mathcal{F}}(\mathcal{A}) = \{u \in \Sigma^* \mid \exists q \in F \text{ avec } (q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, z)\}$$

Langage reconnu par pile vide

Un mot u est **accepté par pile vide** par l'automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$, s'il existe un calcul de $(q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, \varepsilon)$

Le langage **accepté par pile vide** par l'automate à pile $\mathcal{A}$ est l'ensemble des mots acceptés par pile vide par $\mathcal{A}$, on le note $\mathcal{L}^{\mathcal{V}}(\mathcal{A})$

$$\mathcal{L}^{\mathcal{V}}(\mathcal{A}) = \{u \in \Sigma^* \mid \exists q \in Q \text{ avec } (q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, \varepsilon)\}$$

Le langage **accepté par pile vide et état final** par l'automate à pile $\mathcal{A}$ est

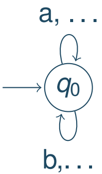
$$\mathcal{L}^{\mathcal{V}\mathcal{F}}(\mathcal{A}) = \{u \in \Sigma^* \mid \exists q \in F \text{ avec } (q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, \varepsilon)\}$$

Langage reconnu par pile vide - Exemple

Question. Trouver un automate à pile qui reconnait le langage des mots sur $\Sigma = \{a, b\}$ contenant autant de *a* que de *b*

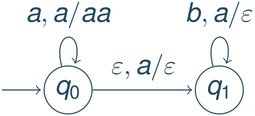
Définition de δ

$$
\begin{array}{lcl}
(q_0, z_0) & \xrightarrow{a} & (q_0, az_0) \\
(q_0, z_0) & \xrightarrow{b} & (q_0, bz_0) \\
(q_0, a) & \xrightarrow{a} & (q_0, aa) \\
(q_0, b) & \xrightarrow{b} & (q_0, bb) \\
(q_0, a) & \xrightarrow{b} & (q_0, \varepsilon) \\
(q_0, b) & \xrightarrow{a} & (q_0, \varepsilon) \\
(q_0, z_0) & \xrightarrow{\varepsilon} & (q_0, \varepsilon)
\end{array}
$$



Exemple de différence des critères

Soit $\mathcal{A} = (\{a, b\}, \{a\}, a, \{q_0, q_1\}, q_0, \{q_1\}, \delta)$



Principes d'un calcul :

- démarre avec un a en fond de pile et empile un a par a lu
- il y aura donc sur la pile un symbole a de plus que le nombre de a lu
- puis une ε-transition, permet de dépiler un a en changeant d'état
- la pile contient donc autant de a que le nombre de a qui ont été lus
- puis chaque lecture de b permet de dépiler un a

Acceptation

- par état final : $\{a^n b^m \mid 0 \leq m \leq n\}$
- par pile vide : $\{a^n b^n \mid 0 \leq n\}$

Équivalence des critères d'acceptation (1/4)

Proposition Les différents critères d'acceptation pour les automates à pile non déterministes sont équivalents : ils permettent de reconnaitre la classe des langages agébriques

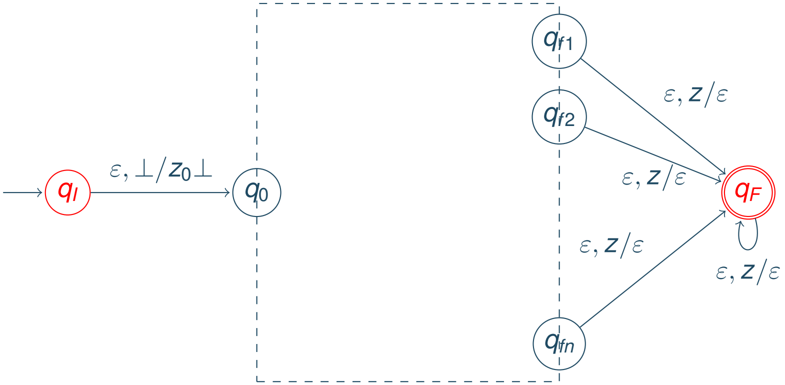
Idée de preuve - Transformation acceptation par état final $\rightarrow$ par pile vide

Soit $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ un automate à pile et $\mathcal{L}^{\mathcal{F}}(\mathcal{A})$ le langage reconnu par état final. Alors pour tout mot u de $\mathcal{L}^{\mathcal{F}}(\mathcal{A})$, il existe un calcul $(q_0, u, z_0) \underset{*}{\rightarrow} (q, \varepsilon, z)$ avec $q \in F$. Pour que le langage reconnu par pile vide soit le même, il « suffit » :

- de vider la pile z à partir de l'état final q
- de s'assurer que la pile n'est jamais vide dans état non final

Construisons $\mathcal{A}' = (\Sigma, Z', z_0', Q', q_I, F', \delta')$ un automate à pile tel que $\mathcal{L}^{\mathcal{F}}(\mathcal{A}) = \mathcal{L}^{\mathcal{V}}(\mathcal{A}')$

Équivalence des critères d'acceptation (2/4)



Équivalence des critères d'acceptation (3/4)

Formellement l'automate construit est
$\mathcal{A}' = (\Sigma, Z \cup \{\perp\}, \perp, Q \cup \{q_I, q_F\}, q_I, \{q_F\}, \delta')$ tel que :

- q_I et q_F sont de nouveaux états
- $\perp$ est un nouveau symbole de pile
- $\delta' = \delta \cup \{(q, z) \xrightarrow{\varepsilon} (q_F, \varepsilon) \mid q \in F \cup \{q_F\} \text{ et } z \in Z \cup \{\perp\}\}$
  $\cup \{(q_I, \perp) \xrightarrow{\varepsilon} (q_0, z_0 \perp)\}$

On montre par équivalence des calculs que les langages reconnus sont équivalents

Équivalence des critères d'acceptation (4/4)

Idée de preuve - Transformation acceptation par pile vide $\rightarrow$ par état final

Soit $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ un automate à pile et $\mathcal{L}^{\mathcal{V}}(\mathcal{A})$ le langage reconnu par pile vide. Pour que le langage reconnu par état final soit le même, il « suffit » :

- de passer dans un état final lorsque la pile est vide
- pour cela il faut savoir tester la pile vide

Construisons $\mathcal{A}' = (\Sigma, Z \cup \{\perp\}, \perp, Q \cup \{q_I, q_F\}, q_I, \{q_F\}, \delta')$ un automate à pile tel que

- q_I et q_F sont de nouveaux états
- $\perp$ est un nouveau symbole de pile
- $\delta' = \delta \cup \{(q, \perp) \xrightarrow{\varepsilon} (q_F, \varepsilon)\} \cup \{(q_I, \perp) \xrightarrow{\varepsilon} (q_0, z_0\perp)\}$

On montre par équivalence des calculs que les langages reconnus sont équivalents que $\mathcal{L}^{\mathcal{V}}(\mathcal{A}) = \mathcal{L}^{\mathcal{F}}(\mathcal{A}')$

Choix d’un critère d’acceptation

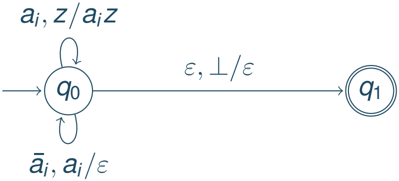
Pour la suite, lorsque ce n’est pas précisé, les automates à pile utilisés sont à acceptation par état final et pile vide

Exemple - Automate du langage de Dyck

Question. Construire un automate pour le langage de Dyck sur $\Sigma = \{a_1, \bar{a}_1, \ldots, a_n, \bar{a}_n\}$

Solution. L'automate à pile $\mathcal{A} = (\Sigma, Z, \perp, \{q_0, q_1\}, q_0, \{q_1\}, \delta)$ avec $Z = \{\perp\} \cup \{a_i \mid 1 \leq i \leq n\}$ et

$$\begin{array}{rcl} \delta & = & \{(q_0, z) \xrightarrow{a_i} (q_0, a_i z) \mid 1 \leq i \leq n \text{ et } z \in Z\} \\ & \cup & \{(q_0, a_i) \xrightarrow{\bar{a}_i} (q_0, \varepsilon) \mid 1 \leq i \leq n \text{ et } z \in Z\} \\ & \cup & \{(q_0, \perp) \xrightarrow{\varepsilon} (q_1, \varepsilon)\} \end{array}$$



Grammaires algébriques et automates à pile (1/2)

Proposition Un langage $\mathcal{L}$ est context-free (ou hors contexte ou algébrique) si et seulement si il existe un automate à pile qui le reconnait

Idée de preuve - Grammaire $\rightarrow$ Automate

Soit $\mathcal{G} = (\Sigma, V, S, R)$ une grammaire algébrique

L'automate à pile $\mathcal{A} = (\Sigma, V, S, \{q_0\}, q_0, F, \delta)$ à acceptation par pile vide avec δ définie par :

$$\begin{array}{rcl} \delta & = & \{(q_0, T) \xrightarrow{a} (q_0, \varepsilon) \mid a \in \Sigma \text{ et } T \rightarrow a \in R\} \\ & \cup & \{(q_0, T) \xrightarrow{\varepsilon} (q_0, \alpha) \mid \alpha \in V^* \text{ et } T \rightarrow \alpha \in R\} \end{array}$$

reconnait $\mathcal{L}(\mathcal{G})$

Exemple de construction d'AP pour une GA - Palindromes

$$S \longrightarrow aSa \mid bSb \mid cSc \mid a \mid b \mid c \mid \varepsilon$$

$$\begin{aligned} S &\longrightarrow ASA \mid BSB \mid CSC \mid A \mid B \mid C \mid AA \mid BB \mid CC \\ A &\longrightarrow a \\ B &\longrightarrow b \\ C &\longrightarrow c \end{aligned}$$

$$\begin{aligned} S &\longrightarrow aSa \\ &\longrightarrow abSba \\ &\longrightarrow abbSbba \\ &\longrightarrow abbcScbba \\ &\longrightarrow abbccbba \end{aligned}$$

$$\begin{aligned} (q_0, abbccbba, S) &\xrightarrow{\varepsilon} (q_0, abbccbba, ASA) \xrightarrow{a} (q_0, bbccbba, SA) \\ &\xrightarrow{\varepsilon} (q_0, bbccbba, BSBA) \xrightarrow{b} (q_0, bccbba, SBA) \\ &\xrightarrow{\varepsilon} (q_0, bccbba, BSBBA) \xrightarrow{b} (q_0, ccbba, SBBA) \\ &\xrightarrow{\varepsilon} (q_0, ccbba, CCBBA) \xrightarrow{c} (q_0, cbba, CBBA) \\ &\xrightarrow{c} (q_0, bba, BBA) \xrightarrow{b} (q_0, ba, BA) \\ &\xrightarrow{b} (q_0, a, A) \xrightarrow{a} (q_0, \varepsilon, \varepsilon) \end{aligned}$$

Grammaires algébriques et automates à pile (2/2)

Idée de preuve - Automate $\rightarrow$ Grammaire

Soit $\mathcal{L} = \mathcal{L}^{\mathcal{V}}(\mathcal{A})$ avec $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$.

Notons $L_{(q,q',z)}$ le langage des mots u tels que $(q, u, z) \underset{*}{\rightarrow} (q', \varepsilon, \varepsilon)$

➥alors pour tout $w \in Z^*$, $(q, u, zw) \underset{*}{\rightarrow} (q', \varepsilon, w)$ est un calcul de $\mathcal{A}$

Alors $\bigcup_{q \in Q} L_{(q_0,q,z_0)}$ est l'ensemble des mots qui permettent

- partant de l'état initial q_0
- avec une pile contenant le symbole initial z_0
- d'atteindre une configuration dans laquelle la pile est vide

➥$\mathcal{L} = \bigcup_{q \in Q} L_{(q_0,q,z_0)}$

Grammaires algébriques et automates à pile (3/2)

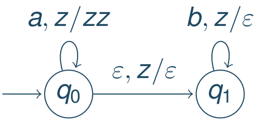
Analysons les premières transitions possibles pour les calculs étiquetés par les mots de $L_{(q,q',z)}$, supposons pour simplifier l'écriture que chaque transition de δ empile au plus 2 symboles

$$
\begin{array}{lclcl}
(q,x,z) & \xrightarrow{x} & (q',\varepsilon,\varepsilon) & & \\
(q,xu,z) & \xrightarrow{x} & (s,u,z_1) & \underset{*}{\rightarrow} & (q',\varepsilon,\varepsilon) \\
(q,xu_1u_2,z) & \xrightarrow{x} & (s,u_1u_2,z_1z_2) & \underset{*}{\rightarrow} & (t,u_2,z_2) \underset{*}{\rightarrow} (q',\varepsilon,\varepsilon)
\end{array}
$$

Ce qui nous permet de décomposer le langage

$$
\begin{array}{rcl}
L_{(q,q',z)} & = & \{x \mid (q,z) \xrightarrow{x} (q',\varepsilon)\} \\
 & \cup & \bigcup_{(q,z)\xrightarrow{x}(s,z_1)\in\delta} xL_{(s,q',z_1)} \\
 & \cup & \bigcup_{(q,z)\xrightarrow{x}(s,z_1z_2)\in\delta} xL_{(s,t,z_1)}L_{(t,q',z_2)}
\end{array}
$$

Exemple de construction d'une GA pour un AP - $a^n b^n$ (1/2)



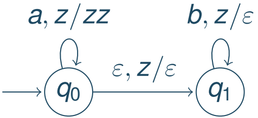
$$\begin{array}{ccc} (q_0, z) & \xrightarrow{a} & (q_0, zz) \\ (q_0, z) & \xrightarrow{\varepsilon} & (q_1, \varepsilon) \\ (q_1, z) & \xrightarrow{b} & (q_0, \varepsilon) \end{array}$$

$\mathcal{L} = L_{(q_0,q_0,z)} \cup L_{(q_0,q_1,z)}$

$$\begin{array}{rcl} L_{(q_0,q_1,z)} & = & \{x \mid (q_0, z) \xrightarrow{x} (q_1, \varepsilon)\} = \{\varepsilon\} \\ & \cup & \bigcup_{(q_0,z)\xrightarrow{x}(s,z)\in\delta} xL_{(s,q_1,z)} = \emptyset \\ & \cup & \bigcup_{(q,z)\xrightarrow{x}(s,z_1z_2)\in\delta} xL_{(s,t,z_1)} L_{(t,q',z_2)} \\ & & = \bigcup_{(q_0,z)\xrightarrow{x}(q_0,zz)\in\delta} aL_{(q_0,q_1,z)} L_{(q_1,q_1,z)} \end{array}$$

Grammaire :

$$\begin{array}{llll} (q_0, q_1, z) & S & \rightarrow & \varepsilon + aST \\ (q_1, q_1, z) & T & & \end{array}$$



$$
\begin{array}{ccc}
(q_0, z) & \xrightarrow{a} & (q_0, zz) \\
(q_0, z) & \xrightarrow{\varepsilon} & (q_1, \varepsilon) \\
(q_1, z) & \xrightarrow{b} & (q_0, \varepsilon)
\end{array}
$$

$\mathcal{L} = L_{(q_0,q_0,z)} \cup L_{(q_0,q_1,z)}$

$$
\begin{array}{rcl}
L_{(q_1,q_1,z)} & = & \{x \mid (q_1, z) \xrightarrow{x} (q_1, \varepsilon)\} = \{b\} \\
& \cup & \bigcup_{(q_1,z)\xrightarrow{x}(s,z)\in\delta} xL_{(s,q_1,z)} = \emptyset \\
& \cup & \bigcup_{(q_1,z)\xrightarrow{x}(s,zz)\in\delta} xL_{(s,t,z)}L_{(t,q_1,z)} = \emptyset
\end{array}
$$

Grammaire :

$$
\begin{array}{llcl}
(q_0, q_1, z) & S & \to & \varepsilon + aST \\
(q_1, q_1, z) & T & \to & b
\end{array}
$$

Automates à pile déterministes - Idée

L'idée de base est la même que pour les automates finis :

- à chaque pas de calcul une seule transition est possible
- il n'y a pas de « choix »

⚠ les ε-transitions sont possibles, mais si une ε-transition est possible, aucune autre transition n'est possible

Automates à pile déterministes - Définition

Un automate à pile $\mathcal{A} = (\Sigma, Z, z_0, Q, q_0, F, \delta)$ est **déterministe** si δ est telle que pour tout $q \in Q$ et tout $z \in Z$:

- si $(q, z) \xrightarrow{\varepsilon} (q', u') \in \delta$ alors
  - $\forall (q, z) \xrightarrow{\varepsilon} (q'', u'') \in \delta$, $q' = q''$ et $u' = u''$
  - $\nexists a \in \Sigma \mid (q, z) \xrightarrow{a} (q'', u'') \in \delta$
- si $\nexists (q, z) \xrightarrow{\varepsilon} (q', u') \in \delta$ alors
  - $\forall a \in \Sigma$, $|\{(q', u') \mid (q, z) \xrightarrow{a} (q', u') \in \delta\}| \leq 1$

⚠ Les différents modes d'acceptation ne sont pas équivalents dans le cas des automates à pile déterministes, l'acceptation par pile vide permet d'accepter moins de langages que l'acceptation par état final

Langages algébriques déterministes et propriétés

Un langage algébrique est dit **déterministe** s'il est accepté par un automate à pile déterministe à acceptation par état final

Proposition

Les langages algébriques déterministes sont clos par complémentaire

Rappel : les langages algébriques ne sont pas clos par complémentaire

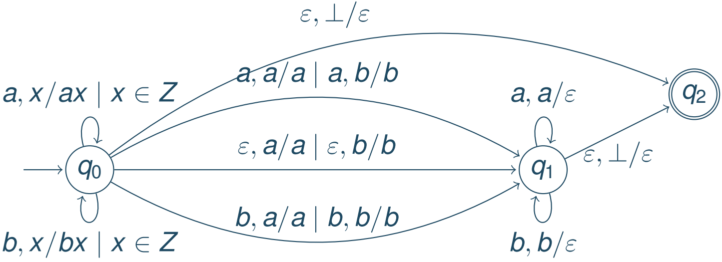
Proposition

Tout langage algébrique déterministe est non ambigu

Remarque : la réciproque est fausse

Exemple - déterminisme

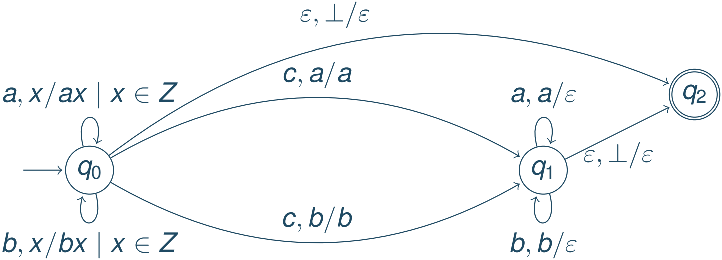
Automate à pile $\mathcal{A} = (\Sigma, Z, \perp, Q, q_0, \{q_2\}, \delta)$ avec $\Sigma = \{a, b\}$, $Z = \{a, b, \perp\}$



➥ Le langage est non déterministe intuitivement, il faut « deviner » où est le milieu du mot

Exemple - déterminisme

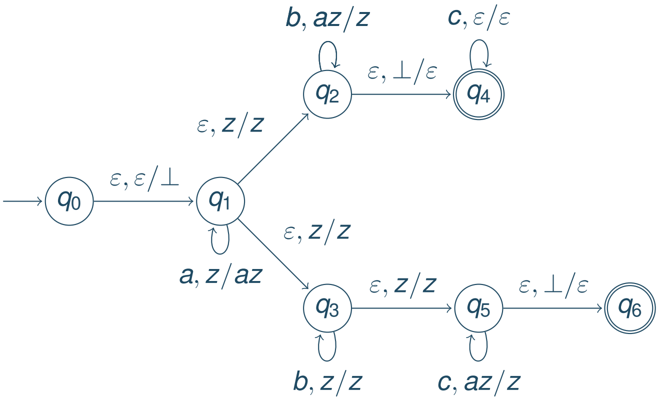
Automate à pile $\mathcal{A} = (\Sigma, Z, \perp, Q, q_0, \{q_2\}, \delta)$ avec $\Sigma = \{a, b, c\}$, $Z = \{a, b, \perp\}$



➥ Le langage est déterministe intuitivement, on a « marqué » le milieu du mot

Exemple $a^i b^j c^k$

Soit $\mathcal{L} = \{a^n b^n c^m \mid n, m \in \mathbb{N}\} \cup \{a^n b^m c^n \mid n, m \in \mathbb{N}\}$



➥ Ce langage ne peut être reconnu par un automate à pile déterministe