Langages, Compilation, Automates. Partie 7: Automates à pile

Florian Bridoux

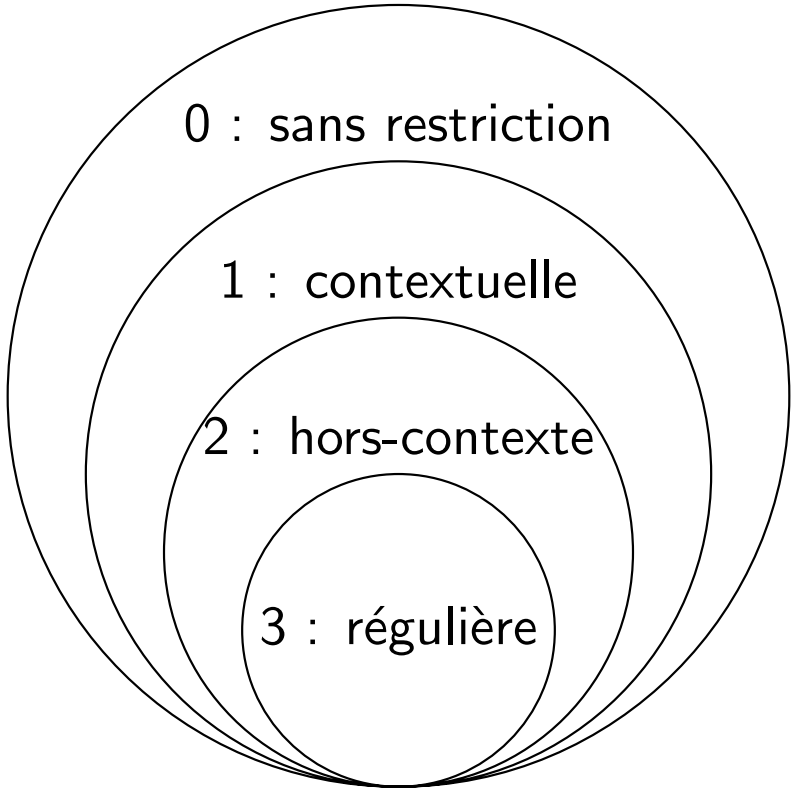
Polytech Nice Sophia

2022-2023

Table des matières

Table des matières

Hiérarchie de Chomsky-Schützenberger



Langage hors-contexte = langage algébrique.

Règles de la grammaire et type de langage

- Une règle est **régulière à gauche** si et seulement si elle est de la forme $A \rightarrow xB$, $A \rightarrow x$ ou $A \rightarrow \epsilon$ avec $A, B \in N$ et $x \in \Sigma^*$.
- Une règle est **régulière à droite** si et seulement si elle est de la forme $A \rightarrow Bx$, $A \rightarrow x$ ou $A \rightarrow \epsilon$ avec $A, B \in N$ et $x \in \Sigma^*$.

Si toutes les règles d'une grammaire sont régulières gauches ou toutes les règles d'une grammaire sont régulières droites alors **le langage engendré est régulier**.

- Une règle est **hors-contexte** si et seulement si elle est de la forme : $A \rightarrow \alpha$ avec $A \in N$ et $\alpha \in (N \cup \Sigma)^*$.

Si toutes les règles d'une grammaire sont hors-contexte alors **le langage engendré est hors-contexte**.

Automates finis déterministe

Définition (Automate fini déterministe (AFD))

Un **automate fini déterministe (AFD)** est un quintuplet $(\Sigma, Q, \delta, q_0, F)$ où:

- Σ est un alphabet des symboles d'entrée,
- Q est un ensemble **fini** d'états,
- δ est la fonction de transition: $Q \times \Sigma \rightarrow Q$,
- $q_0 \in Q$ est l'état initial,
- $F \subseteq Q$ est l'ensemble des états d'acceptation.

Un langage L est régulier si et seulement s'il existe un AFD A tel que $L(A) = L$.

Table des matières

Automates à pile (AP)

Définition (Automate à pile (AP))

Un **automate à pile (AP)** est un septuplet $(\Sigma, \Gamma, Q, \delta, Z, q_0, F)$ où:

- Σ est un alphabet des symboles d'entrée,
- Γ est un alphabet des symboles de pile,
- Q est un ensemble **fini** d'états,
- δ est la fonction de transition: $Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\}) \rightarrow P(Q \times \Gamma^*)$,
- $Z \in \Gamma$ symbole initial de la pile (aussi appelé *symbole de fond de pile*)
- $q_0 \in Q$ est l'état initial,
- $F \subseteq Q$ est l'ensemble des états d'acceptation.

Rappel: $P(E)$ désigne l'ensemble des $2^{|E|}$ sous-ensembles de E.

Automates à pile (AP)

Un AP commence dans l'état initial avec une pile vide.

$$\delta : \ Q \ \times \ (\Sigma \cup \{\epsilon\}) \ \times \ (\Gamma \cup \{\epsilon\}) \ \rightarrow P(\ \ Q \ \times \ \Gamma^* \ \ )$$

L'AP commence dans l'état initial q_0 avec une pile contenant uniquement Z. À chaque étape, l'AP:

- lit l'état courant $q \in Q$.
- Choisit (indéterministe) de consommer le prochain symbole ($a \in \Sigma$) ou pas de symbole ($a = \epsilon$).
- Choisit (indéterministe) d'ignorer la pile $\gamma = \epsilon$ ou de dépiler le sommet de la pile $\gamma \in \Gamma$.
- Passe dans un état $q' \in Q$ et empile un mot $w \in \Gamma^*$ avec

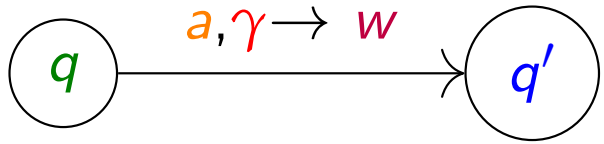
$$(\ q', \ w) \in \delta(q,a,\gamma)$$

L'AP accepte le mot si après avoir lu tout le mot il peut finir dans un état acceptant avec une pile vide.
Remarque: on verra plus tard des variations de ce mode d'acceptation.

Représentation des transitions

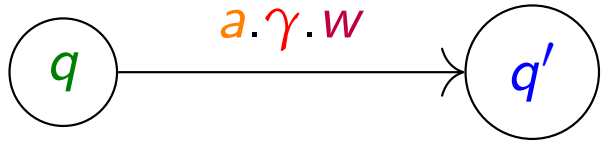
Classiquement:

$$(\ q',\ w) \in \delta(q,a,\gamma) \Rightarrow$$



Dans ce cours:

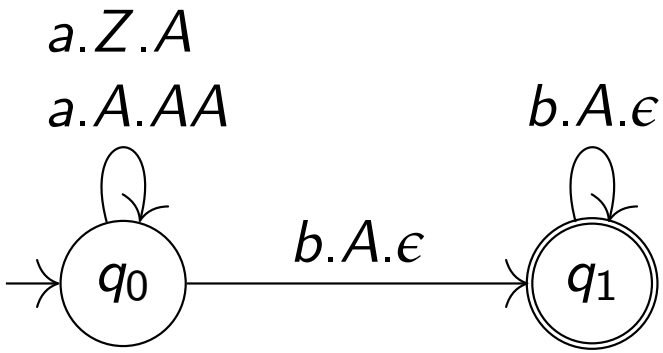
$$(\ q',\ w) \in \delta(q,a,\gamma) \Rightarrow$$



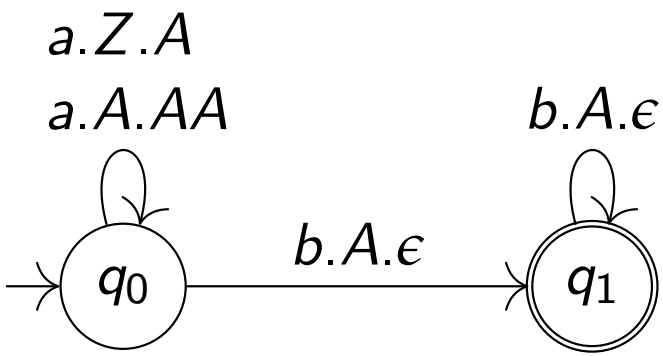
Passage d'une **configuration** à une autre:

$$(q, CBA) \xrightarrow{a.C.ED} (q', EDBA)$$

Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



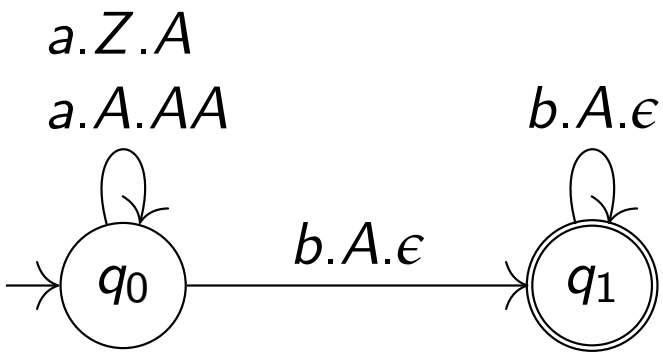
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z

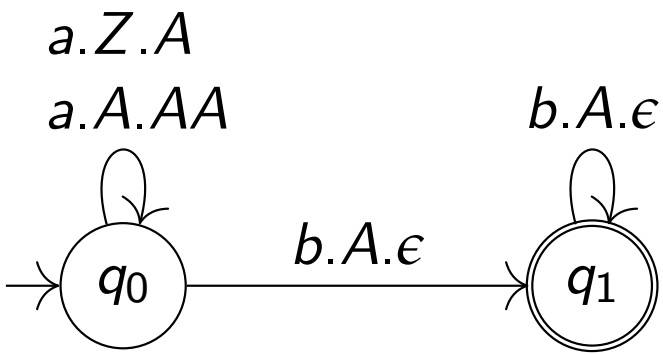
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A

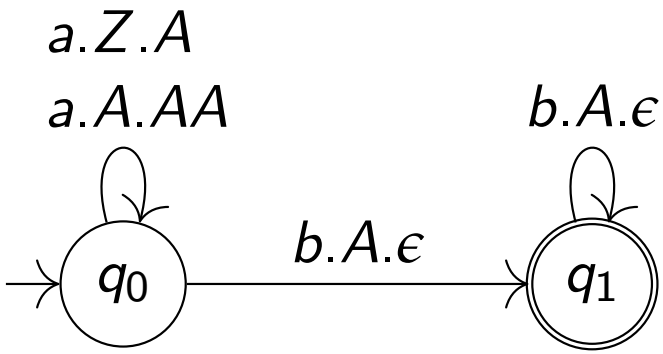
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A
q_0	$abbb$	AA

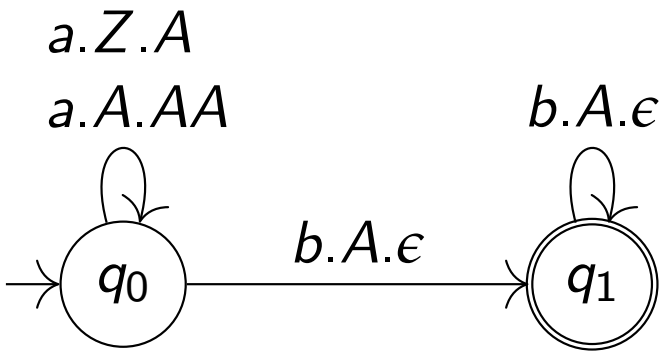
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A
q_0	$abbb$	AA
q_0	bbb	AAA

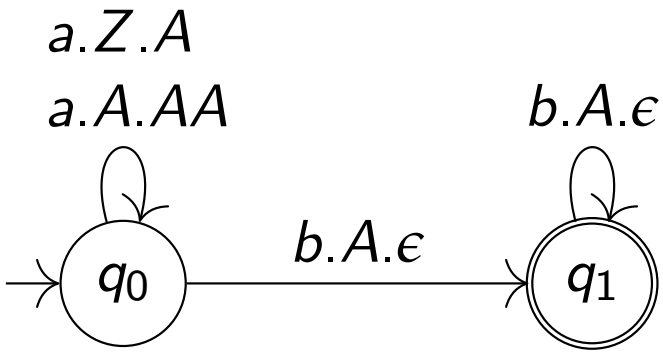
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A
q_0	$abbb$	AA
q_0	bbb	AAA
q_1	bb	AA

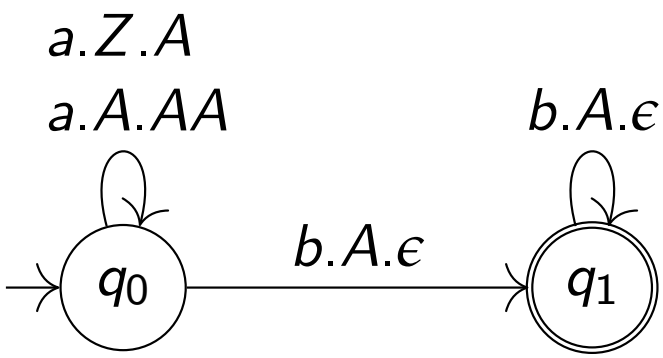
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A
q_0	$abbb$	AA
q_0	bbb	AAA
q_1	bb	AA
q_1	b	A

Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$

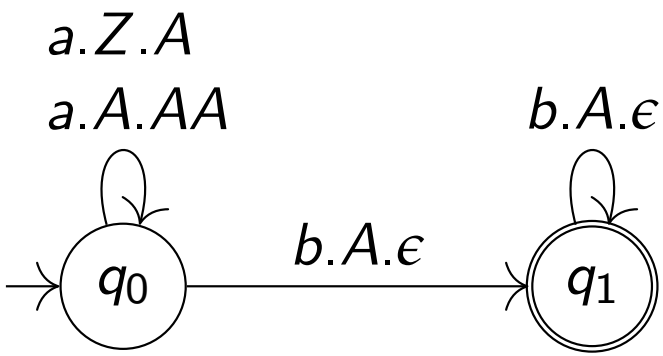


Exemple avec le mot $w = aaabbb$:

état	mot restant	pile
q_0	$aaabbb$	Z
q_0	$aabbb$	A
q_0	$abbb$	AA
q_0	bbb	AAA
q_1	bb	AA
q_1	b	A
q_1	ϵ	ϵ

On finit dans l'état q_1 (acceptant) avec une pile vide: mot accepté.

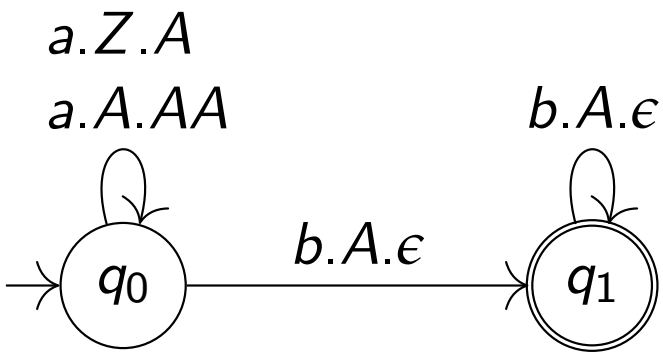
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z

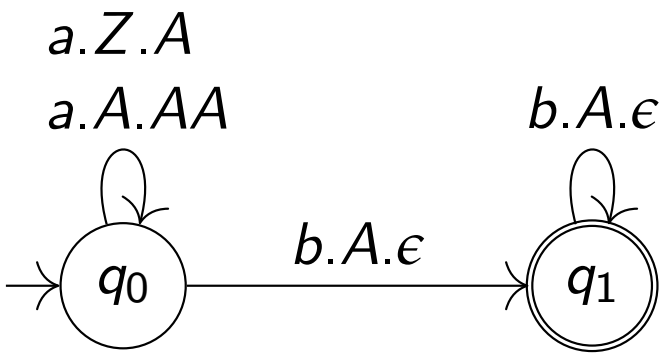
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z
q_0	$aabb$	A

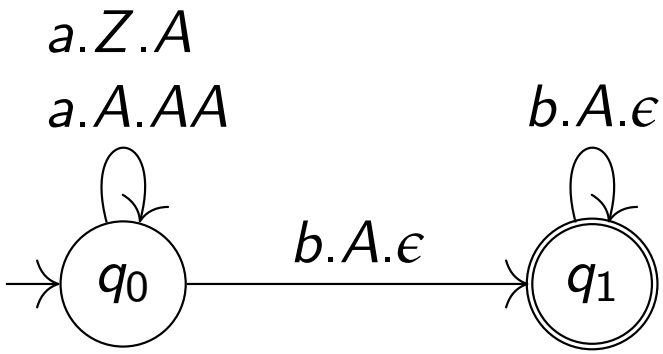
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z
q_0	$aabb$	A
q_0	abb	AA

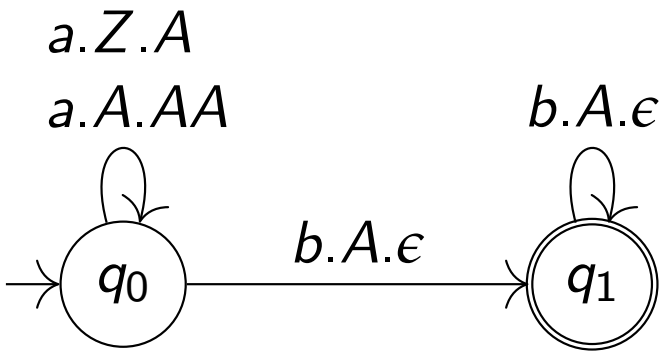
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z
q_0	$aabb$	A
q_0	abb	AA
q_0	bb	AAA

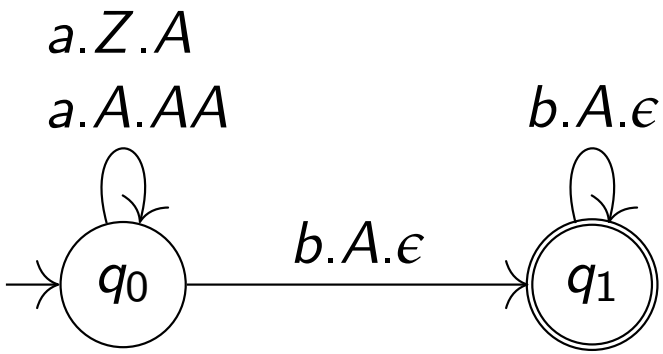
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z
q_0	$aabb$	A
q_0	abb	AA
q_0	bb	AAA
q_1	b	AA

Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$

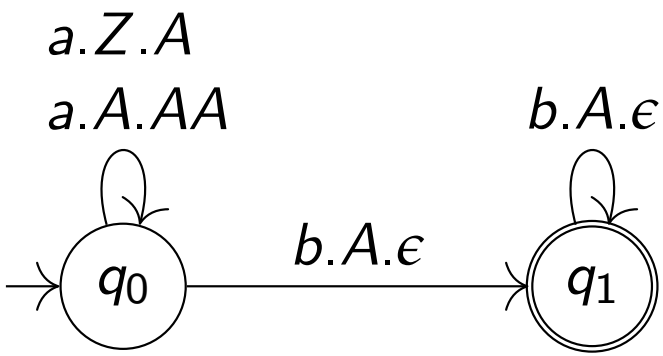


Exemple avec le mot $w = aaabb$:

état	mot restant	pile
q_0	$aaabb$	Z
q_0	$aabb$	A
q_0	abb	AA
q_0	bb	AAA
q_1	b	AA
q_1	ϵ	A

On finit (nécessairement) dans l'état q_1 (acceptant) mais avec une pile non vide: mot rejeté.

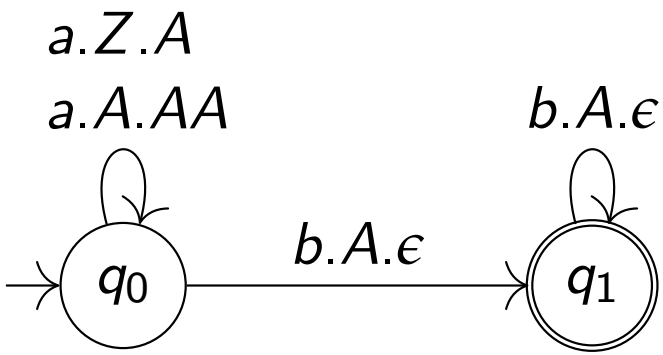
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aabbb$:

état	mot restant	pile
q_0	$aabbb$	Z

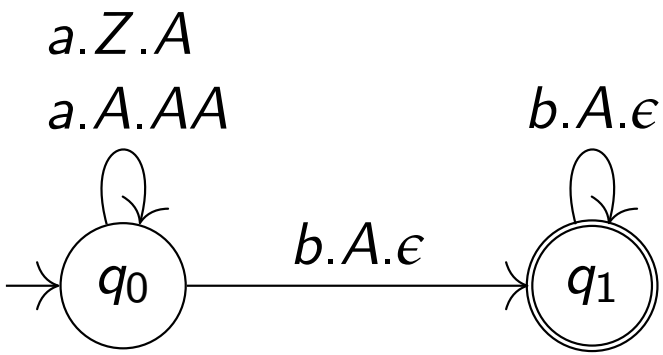
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aabbb$:

état	mot restant	pile
q_0	$aabbb$	Z
q_0	$abbb$	A

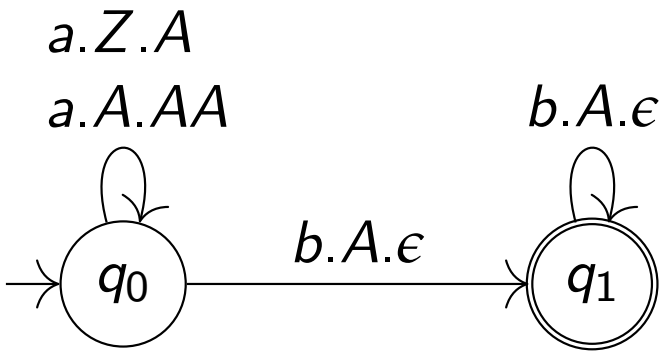
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aabbb$:

état	mot restant	pile
q_0	$aabbb$	Z
q_0	$abbb$	A
q_0	bbb	AA

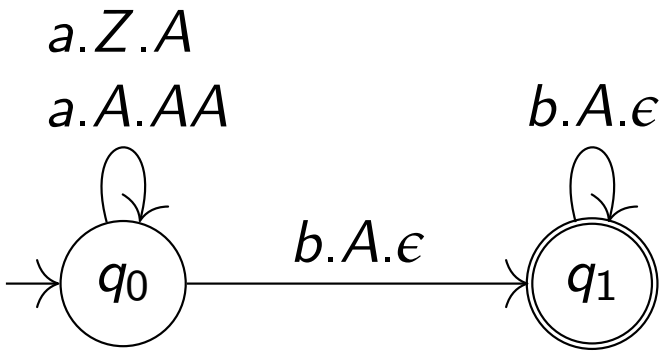
Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aabbb$:

état	mot restant	pile
q_0	$aabbb$	Z
q_0	$abbb$	A
q_0	bbb	AA
q_1	bb	A

Exemple: reconnaître le langage $\{a^n b^n \mid n \geq 1\}$



Exemple avec le mot $w = aabbb$:

état	mot restant	pile
q_0	$aabbb$	Z
q_0	$abbb$	A
q_0	bbb	AA
q_1	bb	A
q_1	b	ϵ

Pas de transition possible: mot rejeté.

Table des matières

Équivalence langage hors-contexte et automates à pile

Theorem

Un langage L est hors-contexte si et seulement s'il existe un automate à pile A qui le reconnaît ($L(A) = L$).

- Si un langage est engendré par une grammaire hors-contexte alors il existe un automate à pile qui le reconnaît.
- Si un langage est reconnu par un automate à pile alors il est engendré par une grammaire hors-contexte.

grammaire hors-contexte $\Rightarrow$ automates à pile

Principe:

1. Empiler l'axiome S de la grammaire.
2. Si le sommet de la pile est un non-terminal N_i alors on le remplace par un la partie droite d'une règle de la forme $N_i \rightarrow \alpha$ de telle sorte que le premier symbole x de α se trouve en sommet de pile.
3. Si le sommet de la pile est un terminal x alors on le compare avec le prochain caractère de l'entrée. S'ils sont égaux alors on dépile sinon on "rejette".
4. Si la pile est vide et que l'entrée a été totalement lue alors on accepte, sinon on revient à l'étape 2.

grammaire hors-contexte $\Rightarrow$ automate à pile

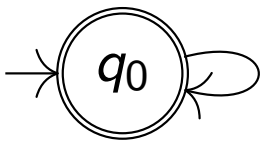
Construction de l'automate à pile A correspondant à la grammaire hors-contexte $G = (N, \Sigma, R, S)$:

- $A = (\Sigma, \Gamma, Q, \delta, Z, q_0, F)$,
- $\Gamma = \Sigma \cup N \cup \{Z\}$, $Q = \{q_0\}$,
- $Z = S$: à l'état initial, dans la pile on a juste l'axiome.
- La fonction de transition δ est définie de la façon suivante :
  - $\delta(q_0, \epsilon, N_i) = \{(q_0, \alpha) \mid N_i \rightarrow \alpha \in R\}$ pour tout $N_i \in N$: Si un symbole non-terminal N_i occupe le sommet de la pile, on le remplace par la partie droite α d'une règle $N_i \rightarrow \alpha$. (ϵ-transition)
  - $\delta(q_0, a, a) = \{(q_0, \epsilon)\}$ pour tout $a \in \Sigma$: Si le même symbole terminal occupe le sommet de la pile et la lettre suivante de l'entrée, on dépile.
- $F = \{q_0\}$: quand l'entrée est entièrement consommée et que la pile est vide on accepte le mot.

Exemple

Grammaire G:

$$\begin{array}{lcl} E & \rightarrow & E + T \\ T & \rightarrow & T * F \\ F & \rightarrow & (E) \mid a \end{array}$$



$a.a.\epsilon$

$+.+.\epsilon$

$*.*.\epsilon$

$(.(.\epsilon$

$).).\epsilon$

$\epsilon.E.E+T$

$\epsilon.T.T*F$

$\epsilon.F.(E)$

$\epsilon.F.a$

grammaire hors-contexte $\Rightarrow$ automates à pile

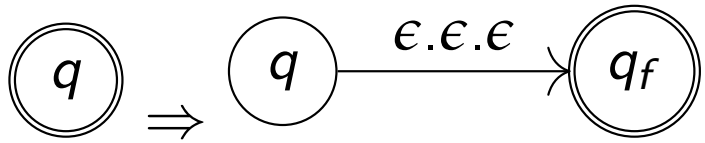
Remarque:

Lorsqu'un non-terminal N_i doit être remplacé au sommet de la pile, il peut l'être par la partie droite d'une règle de la forme $N_i \longrightarrow \alpha$. Plusieurs règles de cette forme peuvent exister dans la grammaire. L'automate correspondant est généralement non déterministe.

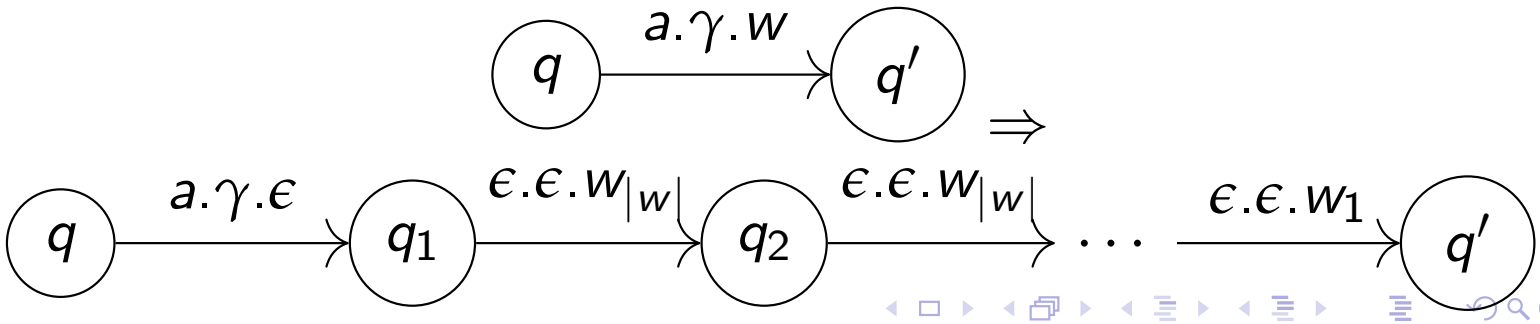
automates à pile $\Rightarrow$ grammaire hors-contexte

Première étape: On simplifie notre automate à pile pour lui donner 3 propriétés:

- Un seul état acceptant.



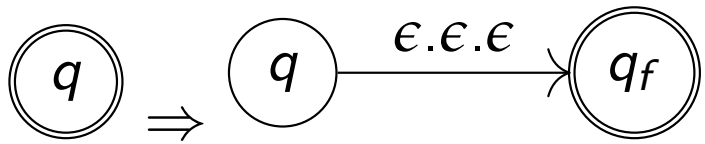
- Chaque transition soit:
  - ignore la pile et empile un unique symbole: $\delta(q, a, \epsilon) = (q', \gamma)$ $(a \in \Sigma \cup \{\epsilon\})$;
  - Ou dépile un symbole et n'empile rien: $\delta(q, a, \gamma) = (q', \epsilon)$ $(a \in \Sigma \cup \{\epsilon\})$.
  - On n'empile et ne dépile rien. $\delta(q, a, \epsilon) = (q', \epsilon)$ $(a \in \Sigma \cup \{\epsilon\})$.



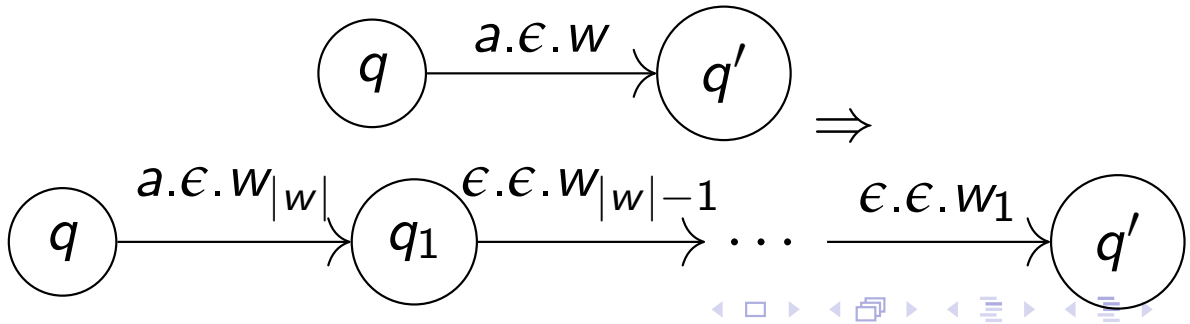
automates à pile $\Rightarrow$ grammaire hors-contexte

Première étape: On simplifie notre automate à pile pour lui donner 3 propriétés:

- Un seul état acceptant.



- Chaque transition soit:
  - ignore la pile et empile un unique symbole: $\delta(q, a, \epsilon) = (q', \gamma)$ $(a \in \Sigma \cup \{\epsilon\})$;
  - Ou dépile un symbole et n'empile rien: $\delta(q, a, \gamma) = (q', \epsilon)$ $(a \in \Sigma \cup \{\epsilon\})$.
  - On n'empile et ne dépile rien. $\delta(q, a, \epsilon) = (q', \epsilon)$ $(a \in \Sigma \cup \{\epsilon\})$.



automates à pile $\Rightarrow$ grammaire hors-contexte

Construction de la grammaire hors-contexte $G = (N, \Sigma, R, S)$ à partir de l'automate à pile $A = (\Sigma, \Gamma, \{q_0, \ldots, q_r\}, \delta, Z, q_0, \{q_r\})$ (simplifié comme indiqué à la diapo précédente).

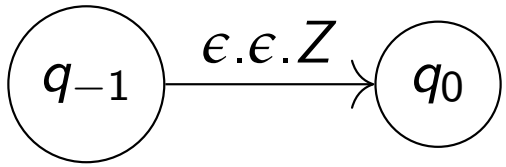
$$\text{On prend } N = \{N_{i,j} \mid i \in [-1, r], j \in [0, r]\}.$$

Idée: $N_{i,j}$ représente l'ensemble des mots w tel que:

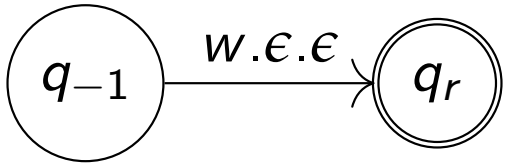
(on peut passer de q_i à q_j sans "toucher à la pile" et en consommant le mot w)

automates à pile $\Rightarrow$ grammaire hors-contexte

q_{-1} est un "faux état" qu'on ajoute pour mettre l'état initial Z sur la pile.



On prend $S = N_{-1,r}$.



grammaire hors-contexte $\Rightarrow$ automates à pile

Cas de base:

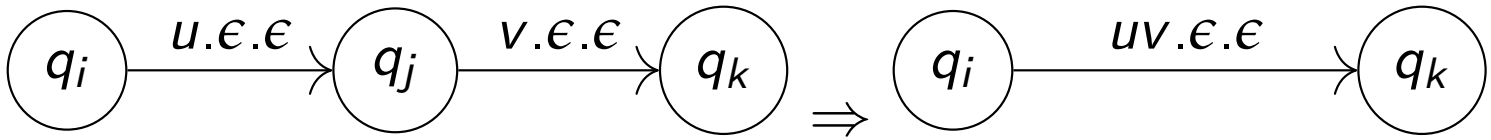
- Pour tout $i \in [0, r]$, on ajoute une règle $N_{i,i} \rightarrow \epsilon$.

- Pour toute transition $(q_j, \epsilon) \in \delta(q_i, a, \epsilon)$ on ajoute une règle $N_{i,j} \rightarrow a$.

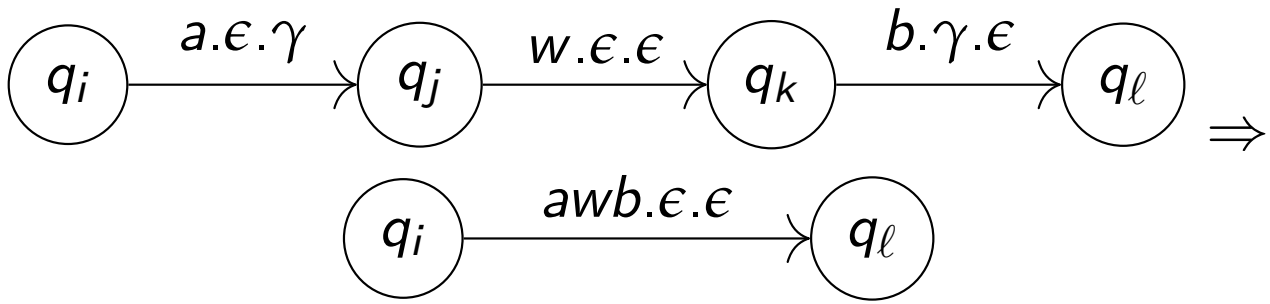
grammaire hors-contexte $\Rightarrow$ automates à pile

Récursion:

- Pour tout $i \in [-1, r]$, $j, k \in [0, r]$, on ajoute une règle $N_{i,k} \to N_{i,j} N_{j,k}$.

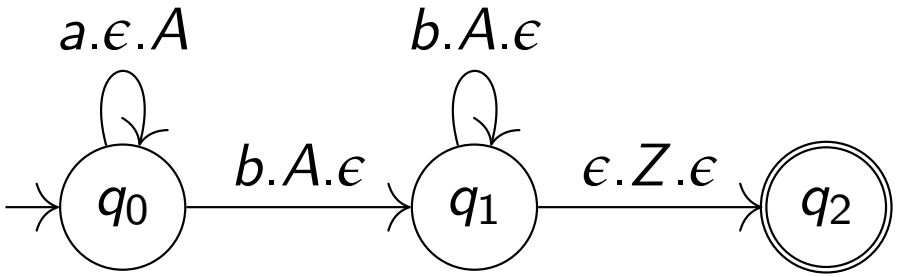


- Pour tout $(q_j, \gamma) \in \delta(q_i, a, \epsilon)$ et $(q_\ell, \epsilon) \in (q_k, b, \gamma)$ (une règle qui empile γ et l'autre qui dépile γ), on ajoute: $N_{i,\ell} \to a N_{j,k} b$.



(On suppose $q_{-1} \xrightarrow{\epsilon.\epsilon.z} q_0$.)

Exemple: grammaire pour le langage $\{a^n b^n \mid n \geq 1\}$

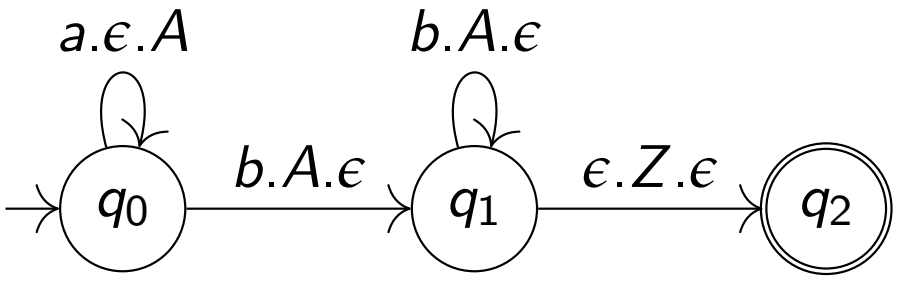


$S = N_{-1,2}$,

$N = \{N_{-1,0}, N_{-1,1}, N_{-1,2}, N_{-1,1}N_{0,0}, N_{0,1}, N_{0,2}, N_{1,0}, N_{1,1}, N_{1,2}, N_{2,0}, N_{2,1}, N_{2,2}\}$

$$
\begin{array}{lcl}
N_{0,0} & \rightarrow & \epsilon \\
N_{1,1} & \rightarrow & \epsilon \\
N_{2,2} & \rightarrow & \epsilon \\
N_{0,2} & \rightarrow & \epsilon \\
N_{1,2} & \rightarrow & \epsilon
\end{array}
\qquad
\begin{array}{lcl}
N_{-1,0} & \rightarrow & N_{-1,0}N_{0,0} \mid \dots \\
 & \dots & \\
N_{2,2} & \rightarrow & N_{2,0}N_{0,2} \mid N_{2,1}N_{1,2} \mid N_{2,2}N_{2,2} \\
N_{0,1} & \rightarrow & aN_{0,0}b \\
N_{0,1} & \rightarrow & aN_{0,1}b \\
N_{-1,2} & \rightarrow & \epsilon N_{0,1}\epsilon
\end{array}
$$

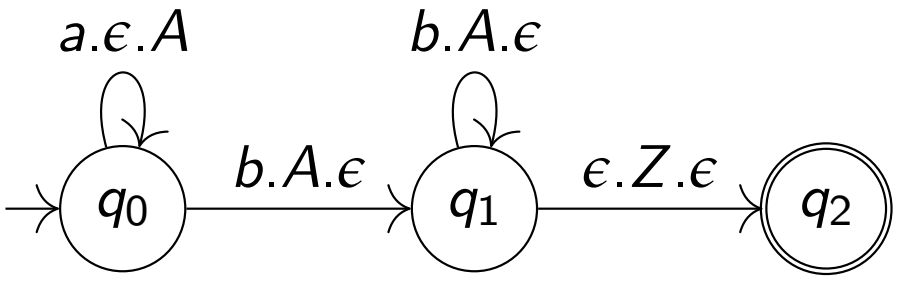
Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



Reconnaissance de *aaabbb*:

q_{-1}

Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$

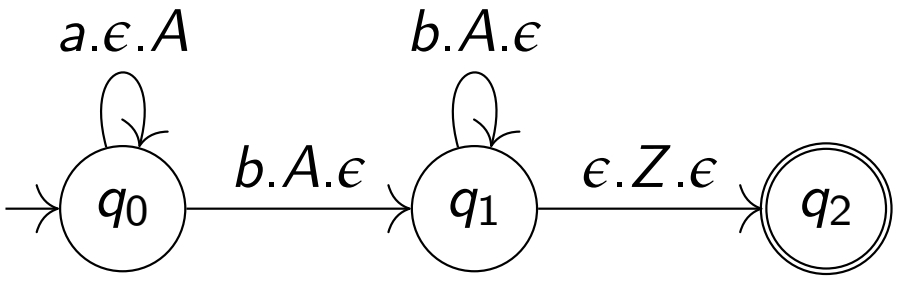


Reconnaissance de *aaabbb*:

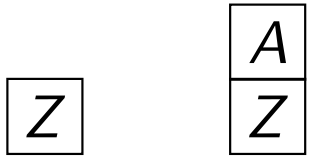
$\boxed{Z}$

$q_{-1} \xrightarrow{\epsilon.\epsilon.Z} q_0$

Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$

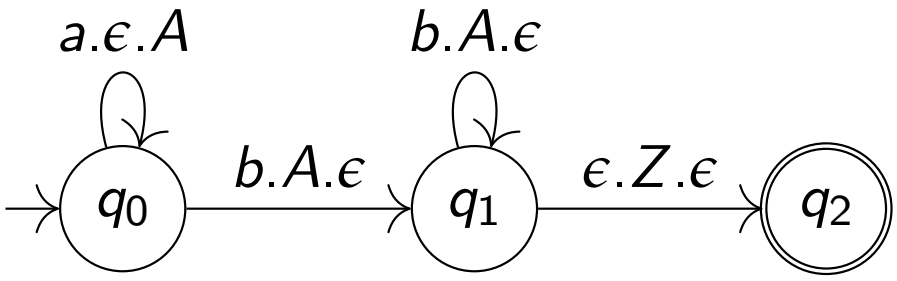


Reconnaissance de *aaabbb*:

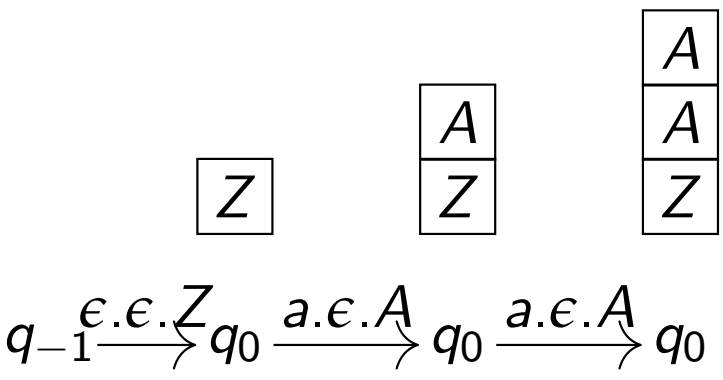


$$q_{-1} \xrightarrow{\epsilon.\epsilon.Z} q_0 \xrightarrow{a.\epsilon.A} q_0$$

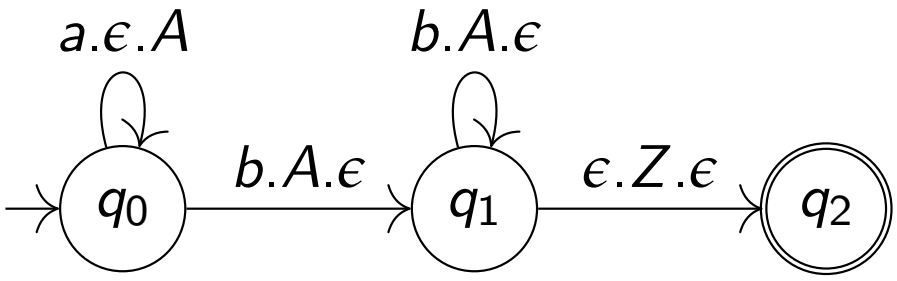
Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



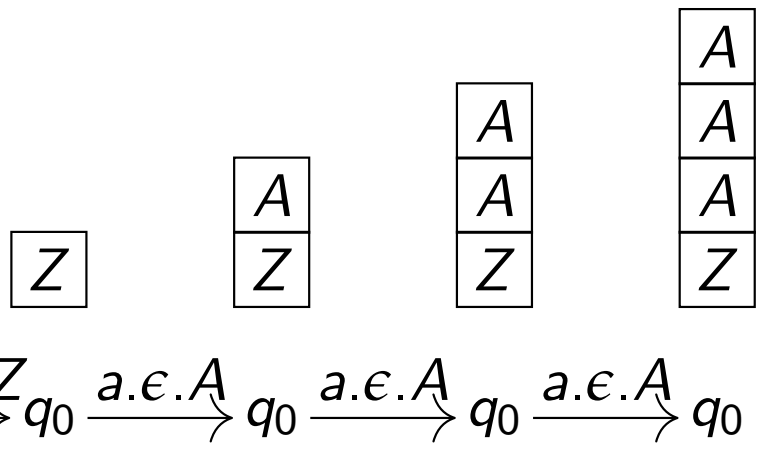
Reconnaissance de *aaabbb*:



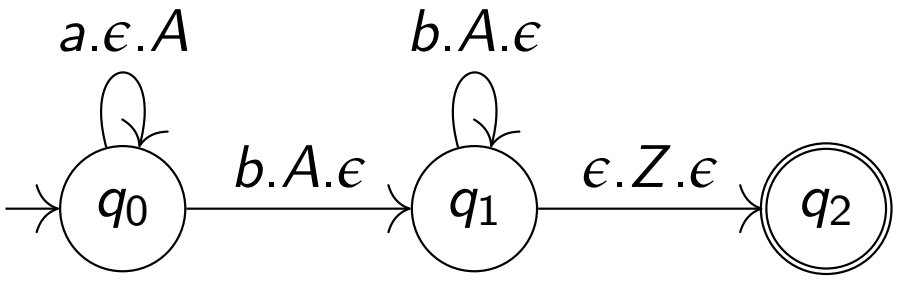
Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



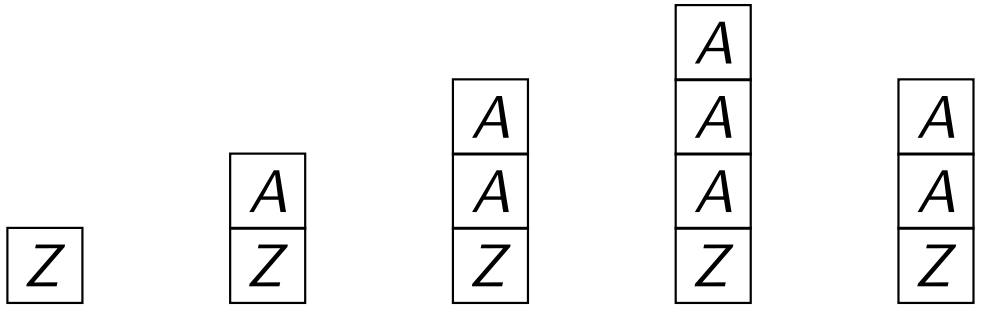
Reconnaissance de *aaabbb*:



Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$

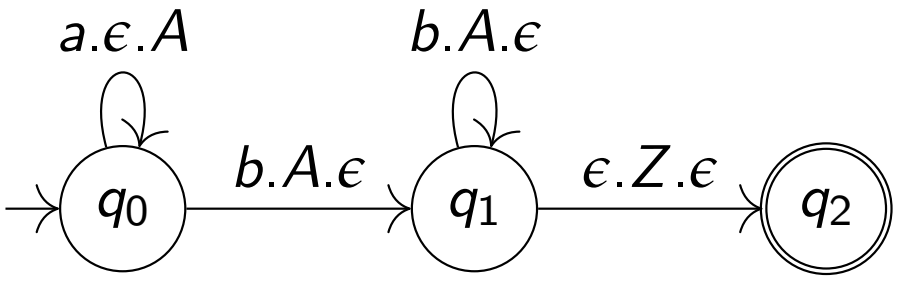


Reconnaissance de *aaabbb*:

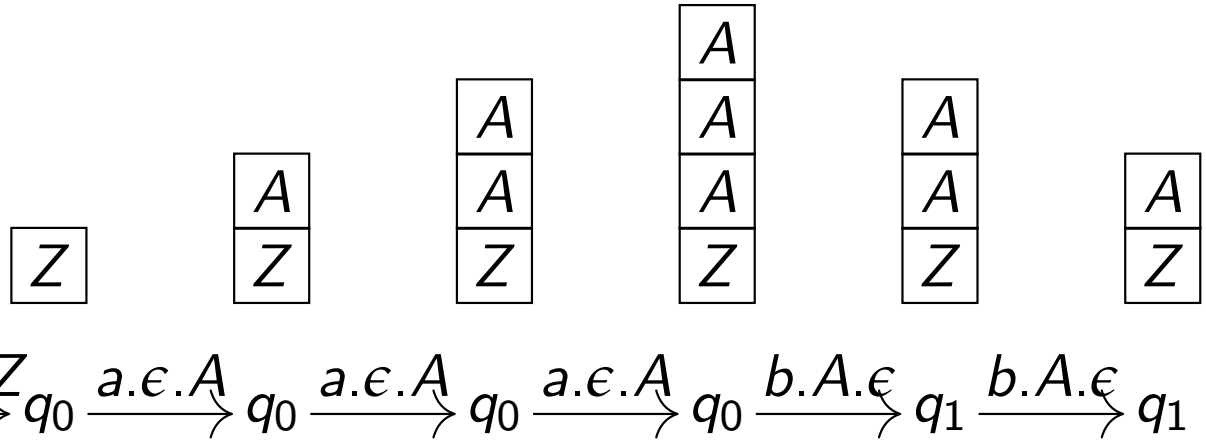


$$q_{-1} \xrightarrow{\epsilon.\epsilon.Z} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{b.A.\epsilon} q_1$$

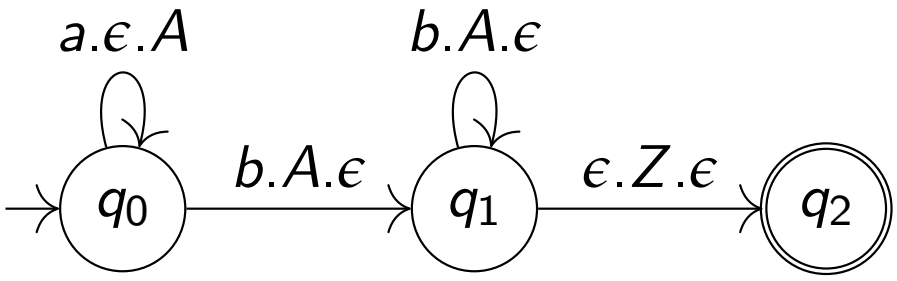
Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



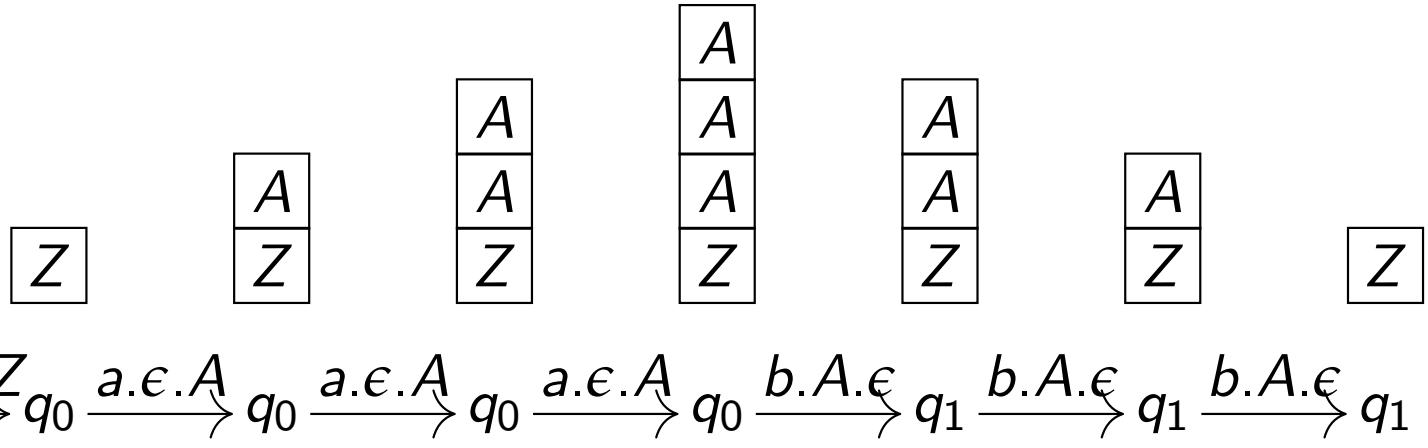
Reconnaissance de *aaabbb*:



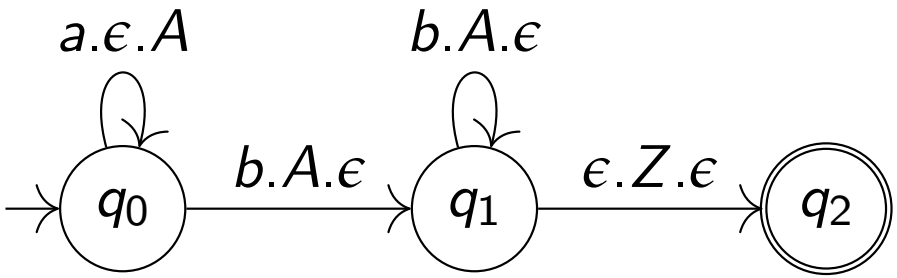
Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



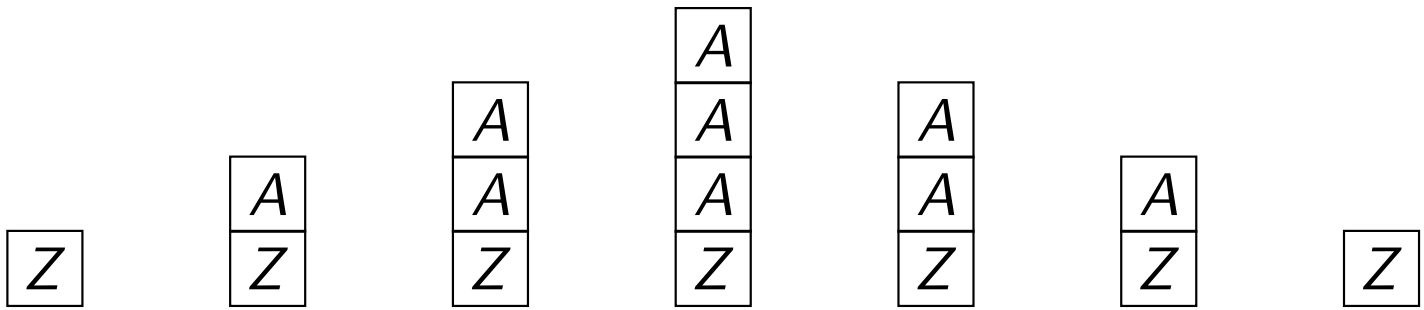
Reconnaissance de *aaabbb*:



Exemple: grammaire pour le langage $\{a^n b^n \mid n \in \mathbb{N}\}$



Reconnaissance de *aaabbb*:



$$q_{-1} \xrightarrow{\epsilon.\epsilon.Z} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{a.\epsilon.A} q_0 \xrightarrow{b.A.\epsilon} q_1 \xrightarrow{b.A.\epsilon} q_1 \xrightarrow{b.A.\epsilon} q_1 \xrightarrow{\epsilon.Z.\epsilon} q_2$$

$$N_{-1,2} \Rightarrow N_{0,1} \Rightarrow aN_{0,1}b \Rightarrow aaN_{0,1}bb \Rightarrow aaaN_{0,0}bbb \Rightarrow aaabbb.$$

Quelques résultats

- On peut déduire des constructions précédentes que chaque automate à pile est équivalent à un automate à pile à **un seul état**.
- En revanche, il n'y a pas de notion d'automate à pile minimal comme pour les AFD...
- Et en général, savoir si deux automates à pile sont équivalent est un problème indécidable!
- (C'est en revanche décidable pour les automates à pile déterministes...)