Théorie des langages

Automates à pile

Jérôme Delobelle

jerome.delobelle@u-paris.fr

LIPADE - Université de Paris

Automates à pile

1. Introduction
2. Rappels sur les piles
3. Automates à pile : définition
4. Automates à pile : configurations
5. Les critères d'acceptation
6. Automates à pile déterministes

Introduction

Introduction

- Grammaires hors contexte : génèrent des langages algébriques

Introduction

- Grammaires hors contexte : génèrent des langages algébriques
- Les automates finis acceptent (exactement) les langages réguliers

Introduction

- Grammaires hors contexte : génèrent des langages algébriques
- Les automates finis acceptent (exactement) les langages réguliers
- Langages réguliers : sous-ensemble strict des langages algébriques

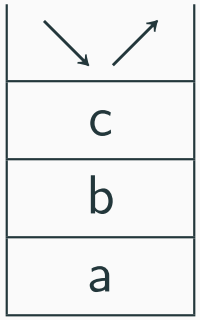
Introduction

- Grammaires hors contexte : génèrent des langages algébriques
- Les automates finis acceptent (exactement) les langages réguliers
- Langages réguliers : sous-ensemble strict des langages algébriques
- Comment obtenir des automates qui acceptent les langages algébriques non réguliers ?
    - Un automate fini dispose par définition d'une mémoire finie

Introduction

- Grammaires hors contexte : génèrent des langages algébriques
- Les automates finis acceptent (exactement) les langages réguliers
- Langages réguliers : sous-ensemble strict des langages algébriques
- Comment obtenir des automates qui acceptent les langages algébriques non réguliers ?
  - Un automate fini dispose par définition d'une mémoire finie

  $\Rightarrow$ Ajouter une pile permet d'étendre les possibilités de mémorisation
    - $\rightarrow$ Garder en mémoire les étapes de calculs passées
    - $\rightarrow$ Conditionner les étapes de calculs à venir



Introduction

- **Automate fini** : défini principalement à partir de sa fonction de transition
- **Automate à pile** : enrichit la fonction de transition par :

Introduction

- **Automate fini** : défini principalement à partir de sa fonction de transition
- **Automate à pile** : enrichit la fonction de transition par :
  1. un nouvel alphabet fini qui contient les symboles qui peuvent être empilés et dépilés

Introduction

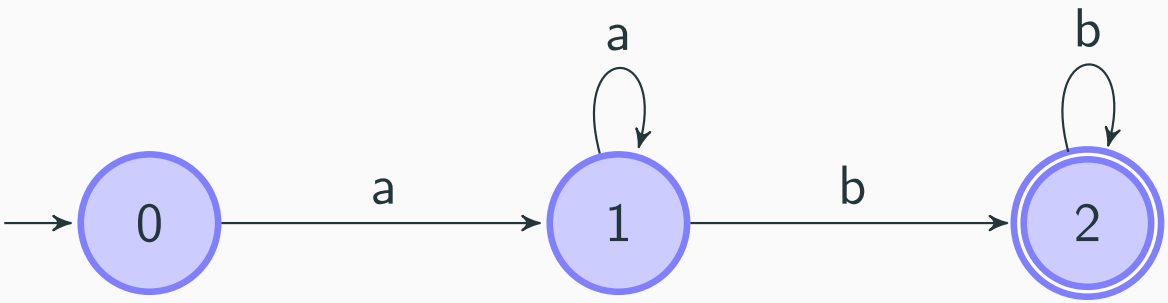
- **Automate fini** : défini principalement à partir de sa fonction de transition
- **Automate à pile** : enrichit la fonction de transition par :
  1. un nouvel alphabet fini qui contient les symboles qui peuvent être empilés et dépilés
  2. des transitions conditionnées par le symbole en haut de la pile

Introduction

- **Automate fini** : défini principalement à partir de sa fonction de transition
- **Automate à pile** : enrichit la fonction de transition par :
  1. un nouvel alphabet fini qui contient les symboles qui peuvent être empilés et dépilés
  2. des transitions conditionnées par le symbole en haut de la pile
  3. lors d'une transition dans l'automate, il est possible d'empiler ou de dépiler un symbole dans la pile

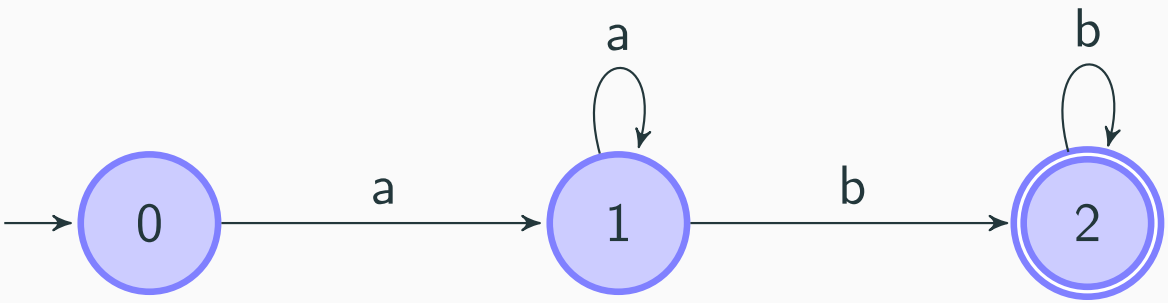
Automate à pile : un exemple introductif

- Soit l'automate suivant qui reconnaît le langage $\{a^n b^m | n, m > 0\}$



Automate à pile : un exemple introductif

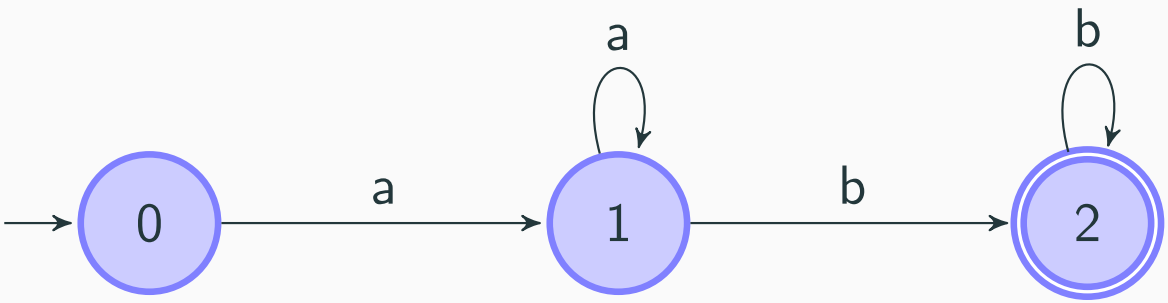
- Soit l'automate suivant qui reconnaît le langage $\{a^n b^m | n, m > 0\}$



- Cet automate ne peut pas reconnaitre le langage $\{a^n b^m | n = m > 0\}$: il est impossible de compter le nombre de a vus, et donc s'assurer de lire le même nombre de b.

Automate à pile : un exemple introductif

- Soit l'automate suivant qui reconnaît le langage $\{a^n b^m | n, m > 0\}$

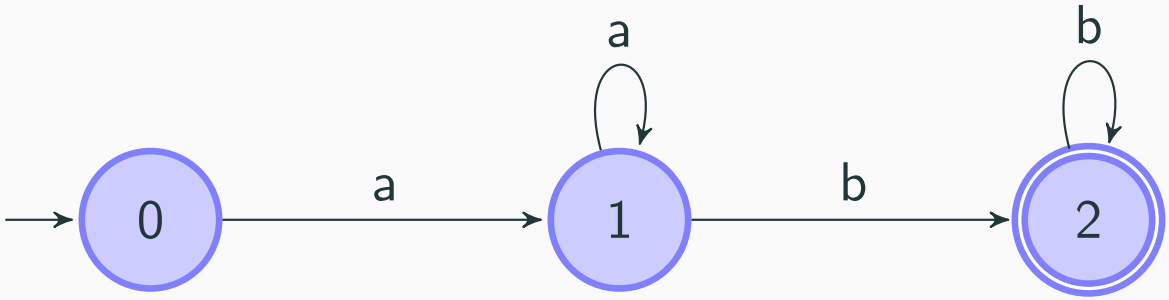


- Cet automate ne peut pas reconnaitre le langage $\{a^n b^m | n = m > 0\}$: il est impossible de compter le nombre de a vus, et donc s'assurer de lire le même nombre de b.

$\Rightarrow$ Automate à pile :

  - Empiler un symbole (T) à chaque lecture d'un symbole a
  - Dépiler un symbole à chaque lecture d'un symbole b
  - Calcul réussi : la pile est vide – on a lu alors autant de a que de b

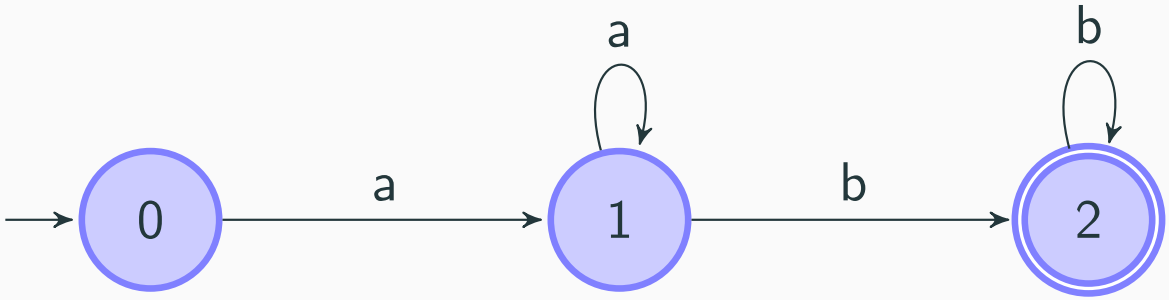
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile

Automate à pile : un exemple introductif

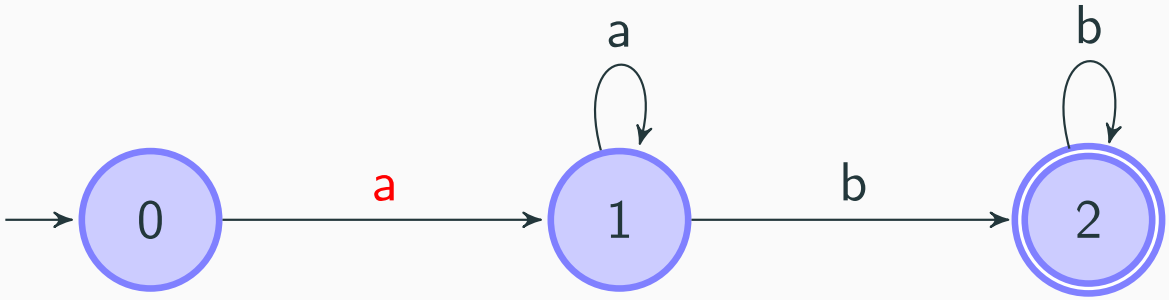


Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile
aaabbb	0	Pile vide

on commence à l'état initial avec la pile vide

Automate à pile : un exemple introductif



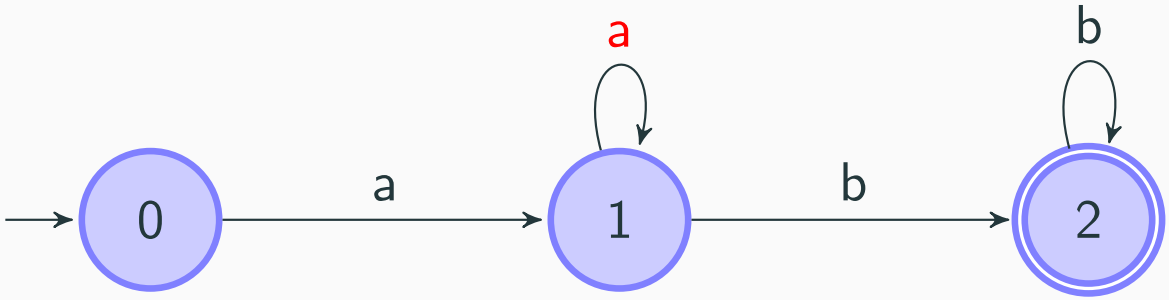
Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile
aaabbb	0	Pile vide
aabbb	1	T

on commence à l'état initial avec la pile vide

on lit un a, on empile T

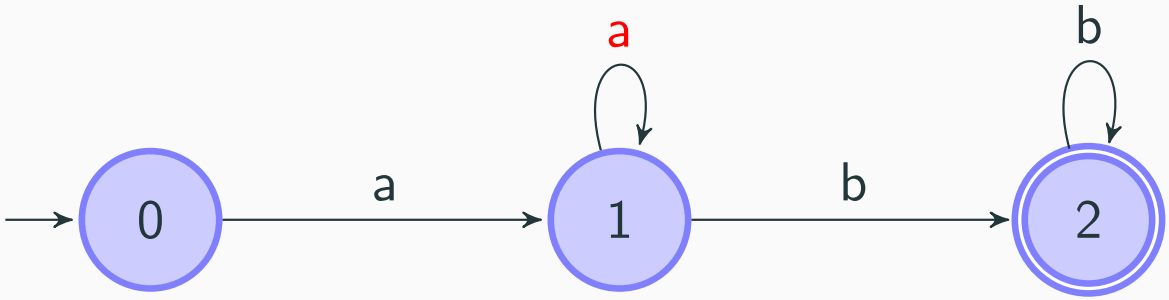
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	T	*on lit un a, on empile T*
abbb	1	TT	*on lit un a, on empile T*

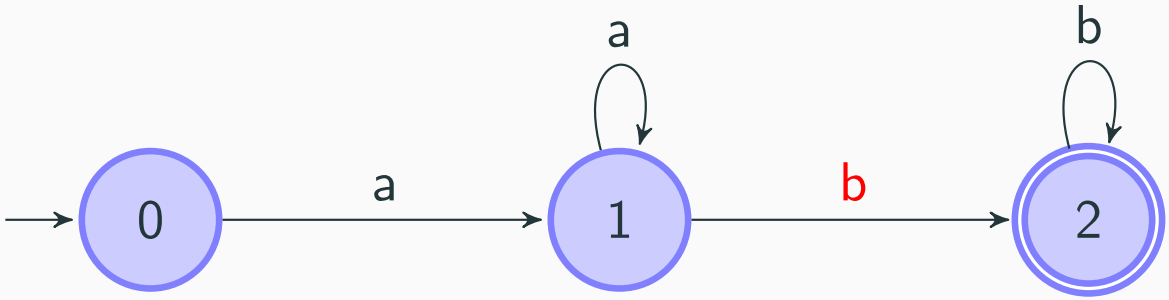
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	*T*	*on lit un a, on empile T*
abbb	1	*TT*	*on lit un a, on empile T*
bbb	1	*TTT*	*on lit un a, on empile T*

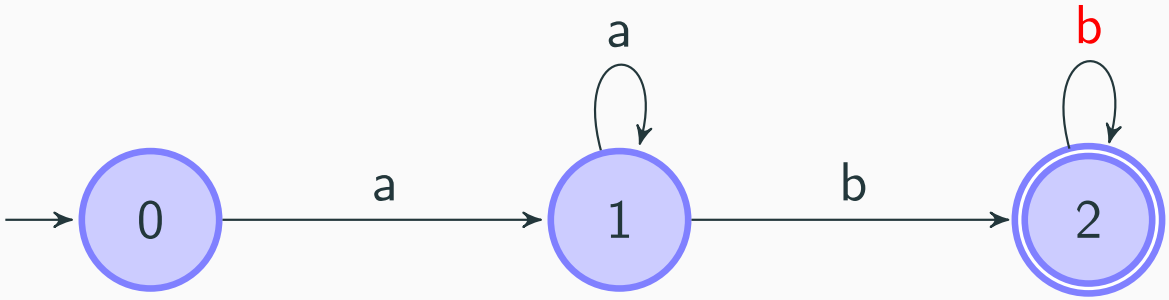
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	*T*	*on lit un a, on empile T*
abbb	1	*TT*	*on lit un a, on empile T*
bbb	1	*TTT*	*on lit un a, on empile T*
bb	2	*TT*	*on lit un b, on dépile T*

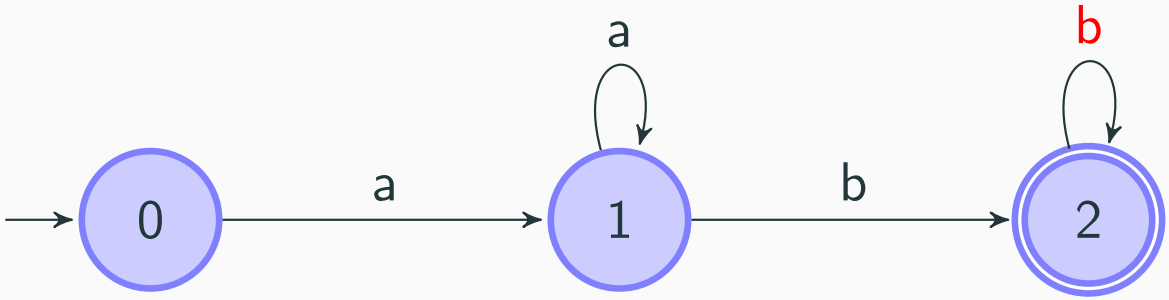
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	*T*	*on lit un a, on empile T*
abbb	1	*TT*	*on lit un a, on empile T*
bbb	1	*TTT*	*on lit un a, on empile T*
bb	2	*TT*	*on lit un b, on dépile T*
b	2	*T*	*on lit un b, on dépile T*

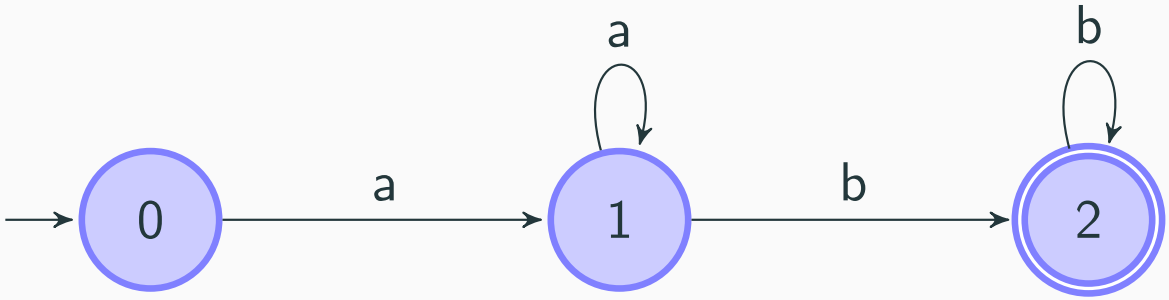
Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	*T*	*on lit un a, on empile T*
abbb	1	*TT*	*on lit un a, on empile T*
bbb	1	*TTT*	*on lit un a, on empile T*
bb	2	*TT*	*on lit un b, on dépile T*
b	2	*T*	*on lit un b, on dépile T*
ϵ	2	Pile vide	*on lit un b, on dépile T*

Automate à pile : un exemple introductif



Trace du calcul pour $w = aaabbb$

Entrée	Etat	Pile	
aaabbb	0	Pile vide	*on commence à l'état initial avec la pile vide*
aabbb	1	*T*	*on lit un a, on empile T*
abbb	1	*TT*	*on lit un a, on empile T*
bbb	1	*TTT*	*on lit un a, on empile T*
bb	2	*TT*	*on lit un b, on dépile T*
b	2	*T*	*on lit un b, on dépile T*
ϵ	2	Pile vide	*on lit un b, on dépile T*

Le mot est lu entièrement, on est dans un état final, la pile est vide, le mot est accepté par l'automate.

Automate à pile

Définition informelle

Un automate à pile est un automate fini asynchrone (i.e. un AFD avec ϵ-transition) auquel on a ajouté une pile de capacité illimitée initialement vide.

Vérifier si un mot est accepté par un automate à pile est globalement semblable à celle d'un automate fini.

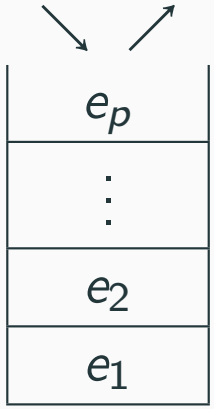
⚠ Deux conditions supplémentaires ⚠

- à chaque étape, l'automate à pile consulte le sommet de sa pile et le remplace éventuellement par une suite de symboles
- la condition d'acceptation (pile vide ? état final ? les deux ?)

Rappels sur les piles

Rappels sur les piles

- **Pile** : Type P
- **LIFO (Last In First Out)**
  - constante $\text{pilevide} \in P$
  - $\text{empiler} : E \times P \to P$
  - $\text{depiler} : P \setminus \{\text{pilevide}\} \to P$
  - $\text{sommet} : P \setminus \{\text{pilevide}\} \to E$
  - $\text{est\_vide} : P \to \mathbb{B}$



Applications en mathématique/informatique :

- résursivité
- backtracking utilisé par exemple dans l'algorithme de recherche en profondeur
- notation post-fixée
- ...

Piles et automates à pile

- On introduit un alphabet de pile Γ
- Une pile p est un mot $p \in \Gamma^*$
- **Opérations sur les piles** :
  - Tester si la pile est vide : déterminer si $p = \epsilon$
  - Empiler un élément $x \in \Gamma$ dans une pile $p \in \Gamma^*$: $p \to xp$
  - Si la pile est non vide, elle est de la forme xp, où $x \in \Gamma$ et $p \in \Gamma^*$. Dépiler l'élément x : $xp \to p$
  - On peut étendre ces notions à des mots. Ainsi, empiler un mot $u = u_1 u_2 \dots u_l$ revient à empiler successivement les lettres $u_1, \dots, u_l \in \Gamma$. Partant de la pile $p \in \Gamma^*$, on obtient

$$p \to u_1 p \to u_2 u_1 p \to \dots \to u_l \dots u_2 u_1 p = u^R p$$

- **Attention** : On obtient le miroir du mot u dans la pile

Automates à pile : définition

Automate à pile

Automate à pile

Un automate à pile (AP) non déterministe (en anglais pushdown automaton) est un septuplet $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$, où

- Σ est l'alphabet d'entrée
- Γ est l'alphabet de pile
- $Z_0 \in \Gamma$ est le symbole initial de la pile
- Q est un ensemble fini d'états
- $q_0 \in Q$ est l'état initial de l'automate
- $F \subseteq Q$ est l'ensemble des état finaux (on peut avoir $F = \emptyset$)
- δ est une fonction de $Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\})$ vers l'ensemble des parties de $Q \times (\Gamma \cup \{\epsilon\})$

Automate à pile

Automate à pile

Un automate à pile (AP) non déterministe (en anglais pushdown automaton) est un septuplet $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$, où

- Σ est l'alphabet d'entrée
- Γ est l'alphabet de pile
- $Z_0 \in \Gamma$ est le symbole initial de la pile
- Q est un ensemble fini d'états
- $q_0 \in Q$ est l'état initial de l'automate
- $F \subseteq Q$ est l'ensemble des état finaux (on peut avoir $F = \emptyset$)
- δ est une fonction de $Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\})$ vers l'ensemble des parties de $Q \times (\Gamma \cup \{\epsilon\})$

Généralisation possible de δ en $Q \times (\Sigma \cup \{\epsilon\}) \times (\Gamma \cup \{\epsilon\})^* \rightarrow Q \times (\Gamma \cup \{\epsilon\})^*$

Automate à pile

Automate à pile : automate fini non-déterministe, à la différence près que la fonction de transition δ comporte trois arguments en entrée :

- l'état courant $q \in Q$
- le symbole d'entrée courant $a \in \Sigma \cup \{\epsilon\}$
- le symbole courant en haut de la pile $Y \in \Gamma \cup \{\epsilon\}$

Et un tuple en sortie composé de :

- l'état de sortie $r \in Q$
- le symbole à remplacer en haut de la pile $T \in \Gamma \cup \{\epsilon\}$

Automate à pile

L'utilisation de la transition $(q, a, Y) \rightarrow (r, T)$ conduira à :

1. lire le symbole a
2. dépiler Y
3. empiler T
4. transiter de l'état q vers l'état r

Automate à pile

L'utilisation de la transition $(q, a, Y) \rightarrow (r, T)$ conduira à :

1. lire le symbole a
2. dépiler Y
3. empiler T
4. transiter de l'état q vers l'état r

- On part de l'état q, on lit a, on dépile Y, on empile T, on arrive à l'état r

Automate à pile

L'utilisation de la transition $(q, a, Y) \to (r, T)$ conduira à :

1. lire le symbole a
2. dépiler Y
3. empiler T
4. transiter de l'état q vers l'état r

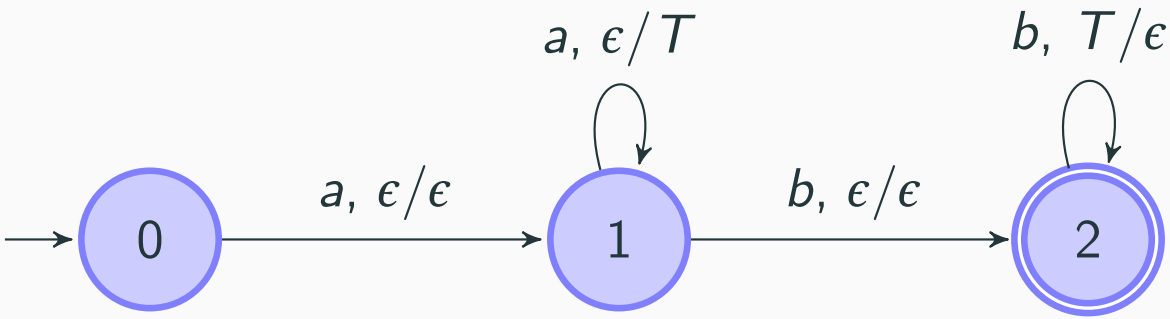
- On part de l'état q, on lit a, on dépile Y, on empile T, on arrive à l'état r
- Pour pouvoir passer par cette transition, il faut donc être dans l'état q, que a soit la prochaine lettre à lire, et que Y soit en sommet de la pile

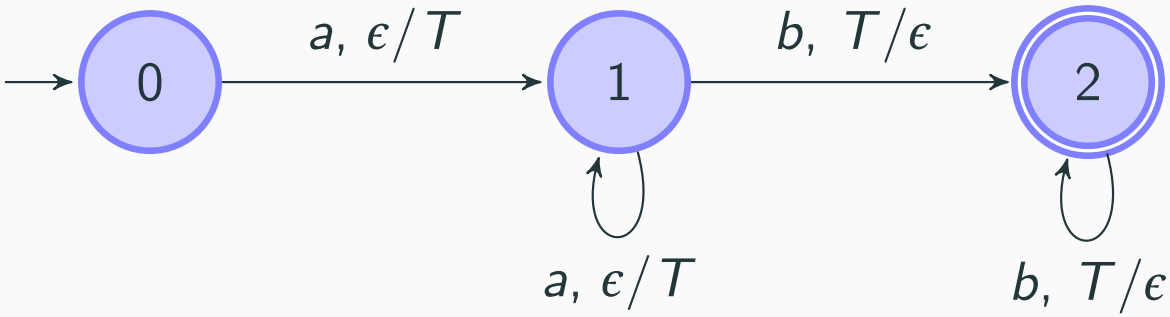
Règles de transition

- Chaque transition $a, Y/T$ précise la lettre a qui doit être lu, le symbole de pile Y que l'on doit trouver en haut de la pile et le symbole T par lequel on doit le remplacer.
  - si en haut de la pile ce n'est pas le bon symbole, alors on ne peut pas emprunter cette transition ;
  - si c'est le bon symbole de pile, on effectue l'opération sur la pile.
- Y et T peuvent être égaux à ϵ
  - Si c'est Y alors on ne se soucis pas du symbole en haut de la pile et qu'on ne dépile rien
  - Si c'est T alors on n'empile rien
- La lettre a peut aussi être ϵ : dans ce cas cela se passe comme pour les ϵ-transitions dans les automates finis (mais on effectue quand même les opérations de pile).

Automate à pile : exemple

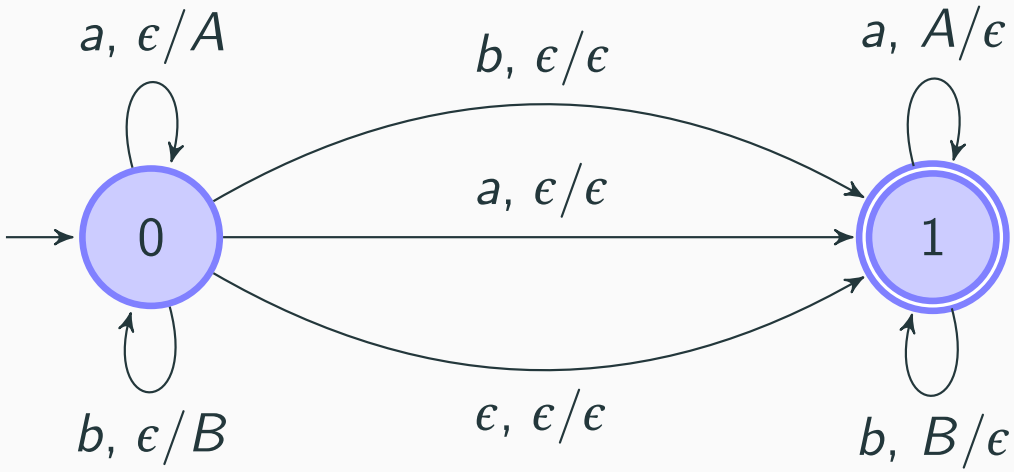
Deux automates à pile reconnaîssant le langage $\{a^n b^n | n > 0\}$





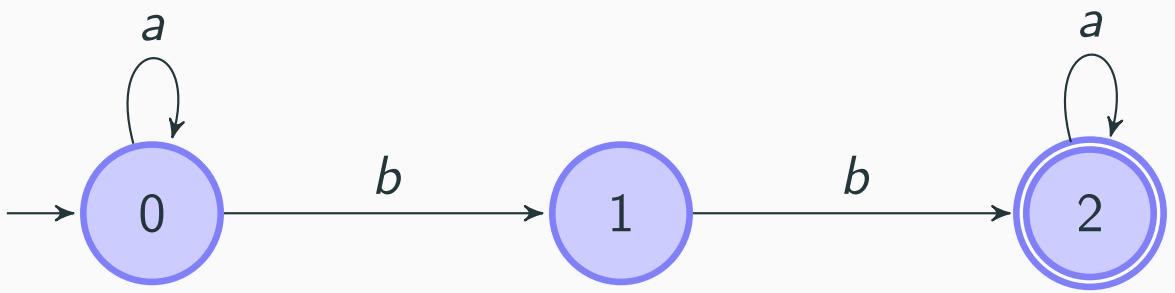
Automate à pile : exemple 2

Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$

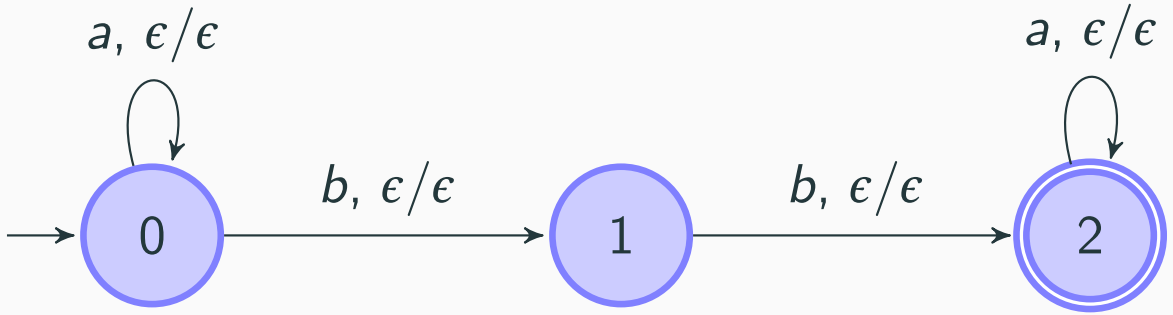


Lien entre AFD et AP

Un automate fini "traditionnel" est un automate à pile particulier, défini sur un alphabet de pile vide ($\Gamma = \emptyset$) et dont toutes les transitions laissent la pile inchangée.



AFD



AP

Automates à pile : configurations

Exécution et configurations

- Une exécution est une suite de configurations.

Exécution et configurations

- Une exécution est une suite de configurations.
- Pour un **automate fini**, une configuration est :
  - mot restant à lire $m \in \Sigma^*$
  - état courant $q \in Q$
  - Exemple : $(q, abbb)$

Exécution et configurations

- Une exécution est une suite de configurations.
- Pour un **automate fini**, une configuration est :
  - mot restant à lire $m \in \Sigma^*$
  - état courant $q \in Q$
  - Exemple : $(q, abbb)$
- Pour un **automate à pile**, une configuration est définie par :
  - le mot restant à lire $m \in \Sigma^*$
  - l'état courant $q \in Q$
  - le contenu de la pile, l'élément le plus à gauche étant le sommet de pile
  - Exemple : $(q, abbb, ABBZ_0)$

Configurations

Configuration

La pile contient, à tout moment, un mot h sur Γ^*. L'automate se trouve dans un état q, et doit lire encore le mot $m \in \Sigma^*$

- Le couple (q, m, h) est appelé une configuration de l'automate.
- L'ensemble des configurations est $Q \times \Sigma^* \times \Gamma^*$.
- La configuration initiale $(q_0, m, Z_0) \in Q \times \Sigma^* \times \Gamma$ est formée de l'état initial et du symbole initial de la pile

Configurations

Configuration

La pile contient, à tout moment, un mot h sur Γ^*. L'automate se trouve dans un état q, et doit lire encore le mot $m \in \Sigma^*$

- Le couple (q, m, h) est appelé une configuration de l'automate.
- L'ensemble des configurations est $Q \times \Sigma^* \times \Gamma^*$.
- La configuration initiale $(q_0, m, Z_0) \in Q \times \Sigma^* \times \Gamma$ est formée de l'état initial et du symbole initial de la pile

Un **mouvement** de l'automate représente le passage d'une configuration à une autre.

Configurations

Configuration

La pile contient, à tout moment, un mot h sur Γ^*. L'automate se trouve dans un état q, et doit lire encore le mot $m \in \Sigma^*$

- Le couple (q, m, h) est appelé une configuration de l'automate.
- L'ensemble des configurations est $Q \times \Sigma^* \times \Gamma^*$.
- La configuration initiale $(q_0, m, Z_0) \in Q \times \Sigma^* \times \Gamma$ est formée de l'état initial et du symbole initial de la pile

Un **mouvement** de l'automate représente le passage d'une configuration à une autre.
On note indifféremment une configuration $(q_0, m, Z_0) \in Q \times \Sigma^* \times \Gamma$ ou $(m, q_0, Z_0) \in \Sigma^* \times Q \times \Gamma$

Passage d'une configuration à une autre

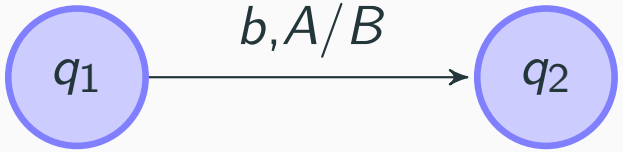
- Le passage d'une configuration c_1 à une configuration c_2 dans un automate M s'écrit :

$$c_1 \vdash_M c_2$$

- On note $\vdash_M^*$ la clôture réflexive et transitive de $\vdash_M$
- Il y a deux modes de transition pour changer de configuration :
  - Sur une Σ-transition
  - Sur une ϵ-transition

Σ-transition : exemple

- Transition $(q_1, b, A) \rightarrow (q_2, B)$



- Configuration (q_1, bba, AZ_0)
- On aura alors :

$$(q_1, bba, AZ_0) \vdash_M (q_2, ba, BZ_0)$$

On part de l'état q_1, on lit b, on dépile A, on empile B, on arrive à l'état q_2

ϵ-transition : exemple

- Transition $(q_1, \epsilon, A) \rightarrow (q_2, B)$

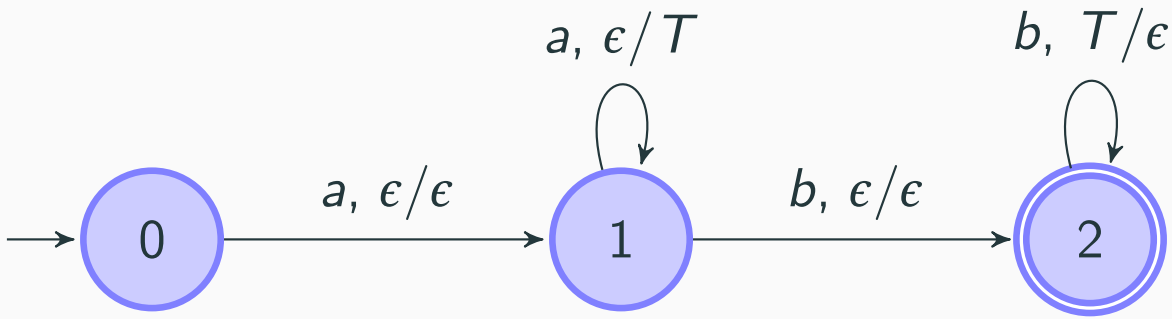
- Configuration (q_1, bba, AZ_0)
- On aura alors :

$$(q_1, bba, AZ_0) \vdash_M (q_2, bba, BZ_0)$$

- On ne touche pas à la tête de lecture
  On part de l'état q_1, **on ne lit rien**, *on dépile A, on empile B, on arrive à l'état q_2*

Transitions : exemple

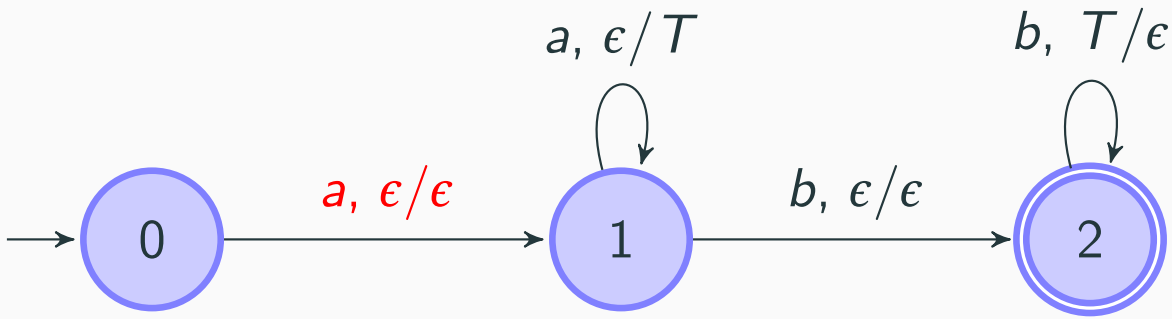
Soit l'automate à pile suivant qui reconnaît le langage $\{a^n b^n | n > 0\}$



$(aabb, 0, Z_0)$

Transitions : exemple

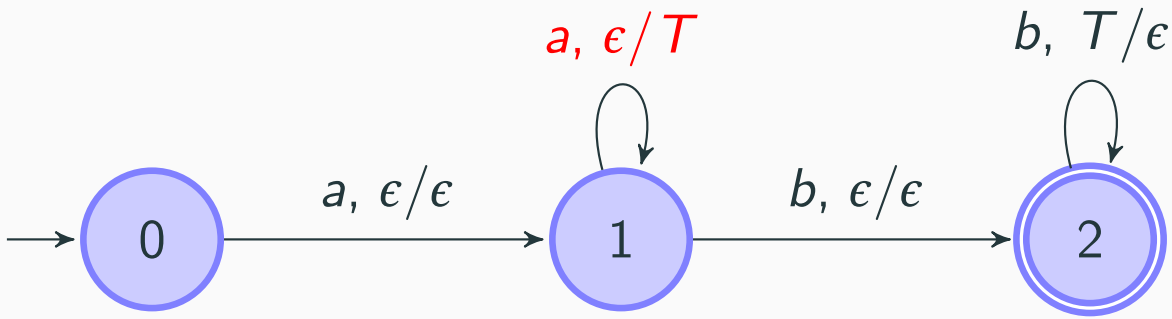
Soit l'automate à pile suivant qui reconnaît le langage $\{a^n b^n | n > 0\}$



$(aabb, 0, Z_0) \vdash_M (abb, 1, Z_0)$ on lit a, on ne dépile rien, on empile rien

Transitions : exemple

Soit l'automate à pile suivant qui reconnaît le langage $\{a^n b^n | n > 0\}$

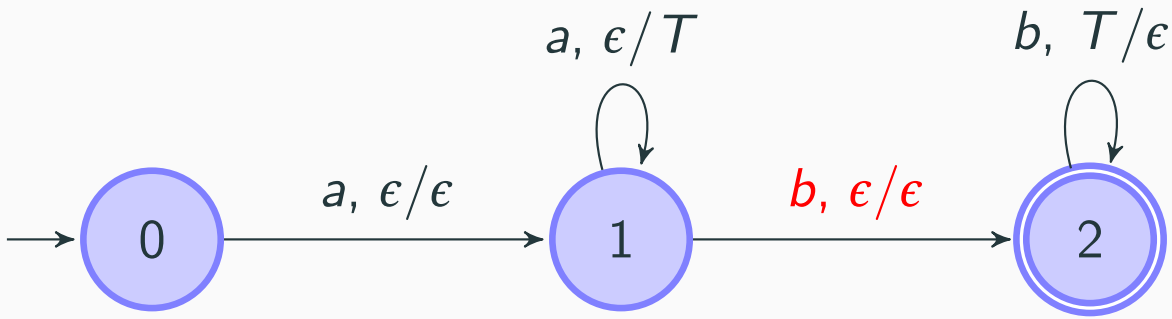


$(aabb, 0, Z_0)$ $\vdash_M$ $(abb, 1, Z_0)$ on lit a, on ne dépile rien, on empile rien

$\vdash_M$ $(bb, 1, TZ_0)$ on lit a, on ne dépile rien, on empile T

Transitions : exemple

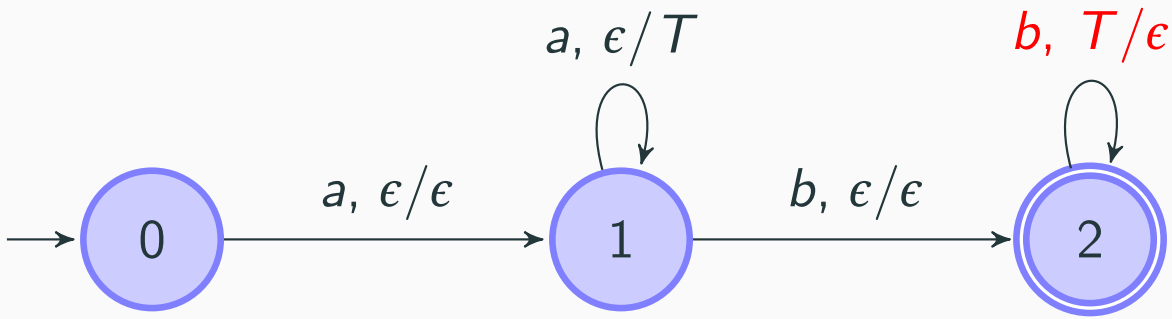
Soit l'automate à pile suivant qui reconnaît le langage $\{a^n b^n | n > 0\}$



$$
\begin{array}{lll}
(aabb, 0, Z_0) & \vdash_M & (abb, 1, Z_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile rien} \\
& \vdash_M & (bb, 1, TZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } T \\
& \vdash_M & (b, 2, TZ_0) \quad \text{on lit } b\text{, on ne dépile rien, on empile rien}
\end{array}
$$

Transitions : exemple

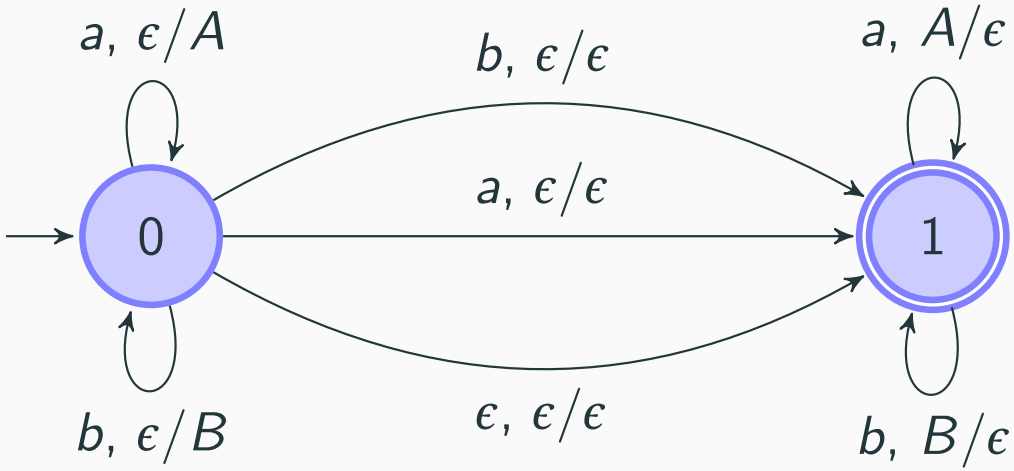
Soit l'automate à pile suivant qui reconnaît le langage $\{a^n b^n | n > 0\}$



$$
\begin{array}{lll}
(aabb, 0, Z_0) & \vdash_M & (abb, 1, Z_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile rien} \\
& \vdash_M & (bb, 1, TZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } T \\
& \vdash_M & (b, 2, TZ_0) \quad \text{on lit } b\text{, on ne dépile rien, on empile rien} \\
& \vdash_M & (\epsilon, 2, Z_0) \quad \text{on lit } b\text{, on dépile } T\text{, on empile rien}
\end{array}
$$

Transitions : exemple

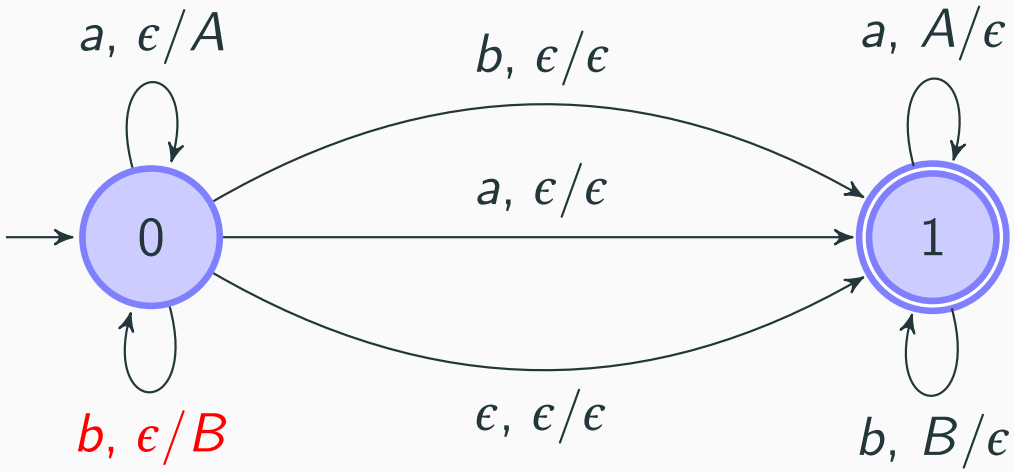
Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$(baaab, 0, Z_0)$

Transitions : exemple

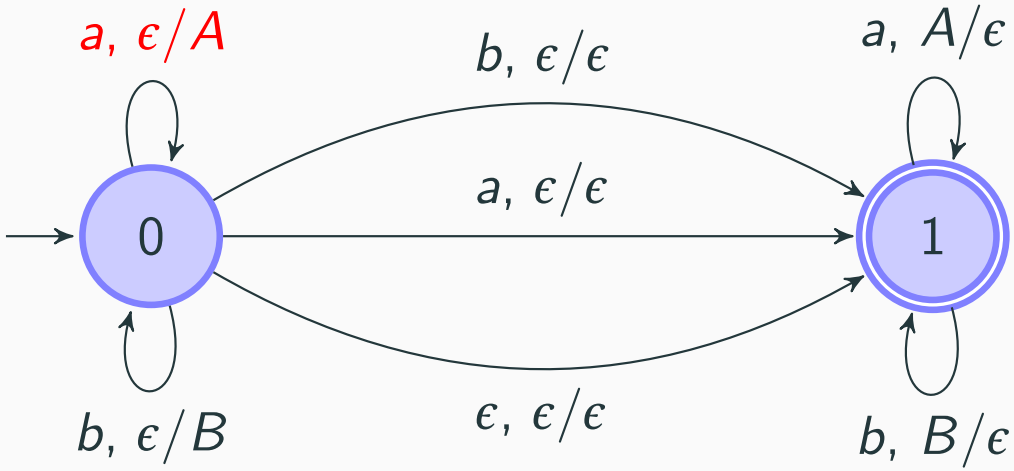
Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$(baaab, 0, Z_0) \vdash_M (aaab, 0, BZ_0)$ on lit b, on ne dépile rien, on empile B

Transitions : exemple

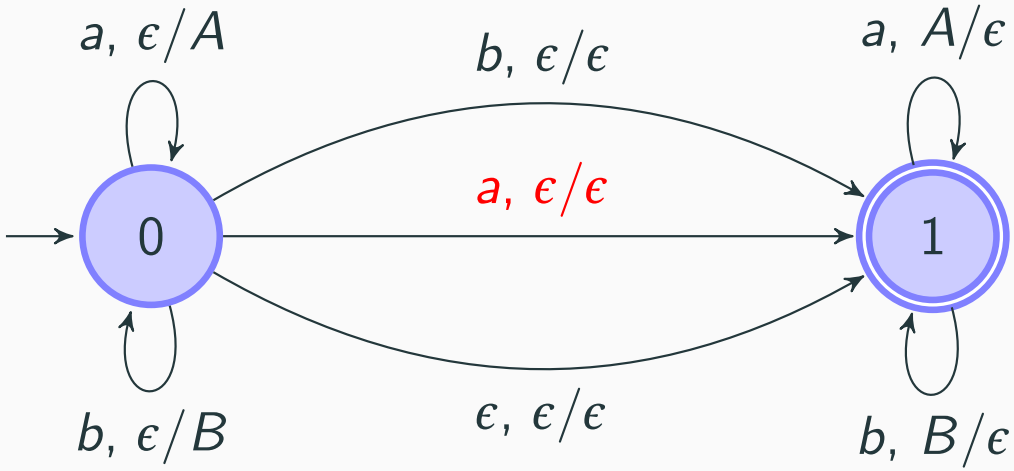
Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$$
\begin{aligned}
(baaab, 0, Z_0) &\vdash_M (aaab, 0, BZ_0) && \text{on lit } b\text{, on ne dépile rien, on empile } B \\
&\vdash_M (aab, 0, ABZ_0) && \text{on lit } a\text{, on ne dépile rien, on empile } A
\end{aligned}
$$

Transitions : exemple

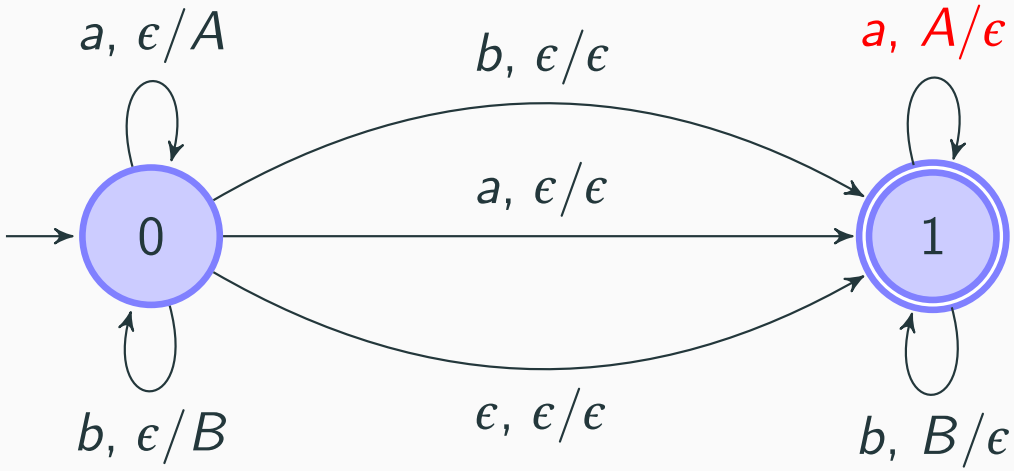
Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$$
\begin{array}{lll}
(baaab, 0, Z_0) & \vdash_M & (aaab, 0, BZ_0) \quad \text{on lit } b\text{, on ne dépile rien, on empile } B \\
& \vdash_M & (aab, 0, ABZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } A \\
& \vdash_M & (ab, 1, ABZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile rien}
\end{array}
$$

Transitions : exemple

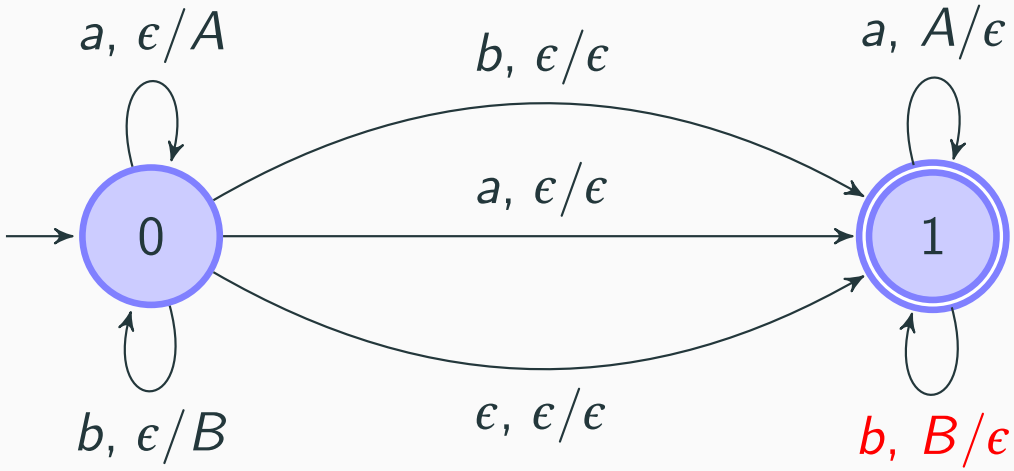
Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$$
\begin{array}{lll}
(baaab, 0, Z_0) & \vdash_M & (aaab, 0, BZ_0) \quad \text{on lit } b\text{, on ne dépile rien, on empile } B \\
& \vdash_M & (aab, 0, ABZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } A \\
& \vdash_M & (ab, 1, ABZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile rien} \\
& \vdash_M & (b, 1, BZ_0) \quad \text{on lit } a\text{, on dépile } A\text{, on empile rien}
\end{array}
$$

Transitions : exemple

Soit l'automate à pile suivant qui reconnaît le langage $\{w \in \Sigma^* | w$ est un palindrome$\}$



$$
\begin{array}{lll}
(baaab, 0, Z_0) & \vdash_M (aaab, 0, BZ_0) & \text{on lit } b\text{, on ne dépile rien, on empile } B \\
& \vdash_M (aab, 0, ABZ_0) & \text{on lit } a\text{, on ne dépile rien, on empile } A \\
& \vdash_M (ab, 1, ABZ_0) & \text{on lit } a\text{, on ne dépile rien, on empile rien} \\
& \vdash_M (b, 1, BZ_0) & \text{on lit } a\text{, on dépile } A\text{, on empile rien} \\
& \vdash_M (\epsilon, 1, Z_0) & \text{on lit } b\text{, on dépile } B\text{, on empile rien}
\end{array}
$$

Les critères d'acceptation

Les critères d'acceptation

- Dans nos exemples, on accepte un mot si le ruban est vide, on est sur l'état final et la pile est vide
- Ce sont des cas particuliers
- Il y a deux critères d'acceptation possibles :
  - Acceptation par état final (quelle que soit la pile quand on s'arrête)
  - Acceptation par pile vide (quel que soit l'état dans lequel on s'arrête)
- Mais le ruban doit toujours être vide !
- Ces deux critères sont équivalents

Acceptation par état final

Acceptation par état final

Un mot $m \in \Sigma^*$ est accepté par état final par un automate à pile $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$ si pour la configuration initiale (m, q_0, Z_0), il existe un état $q_f \in F$ et un mot $z \in \Gamma^*$ tel que

$$(m, q_0, Z_0) \vdash_M^* (\epsilon, q_f, z)$$

Acceptation par état final

Acceptation par état final

Un mot $m \in \Sigma^*$ est accepté par état final par un automate à pile $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$ si pour la configuration initiale (m, q_0, Z_0), il existe un état $q_f \in F$ et un mot $z \in \Gamma^*$ tel que

$$(m, q_0, Z_0) \vdash_M^* (\epsilon, q_f, z)$$

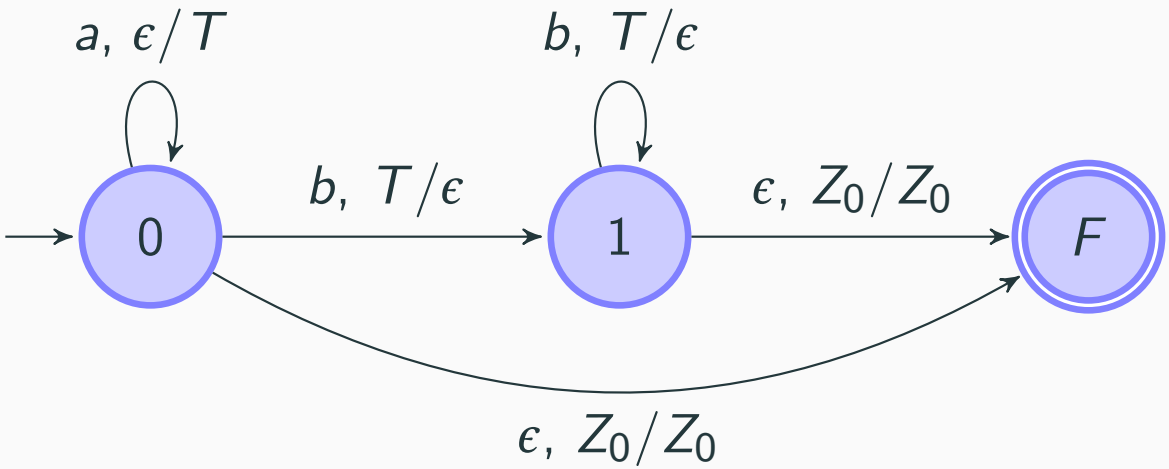
Langage accepté par état final

Le langage accepté par état final par un automate à pile est l'ensemble des mots acceptés par cet automate.

$$L^F(M) = \{m \in \Sigma^* | (m, q_0, Z_0) \vdash_M^* (\epsilon, q_f, z)\}$$

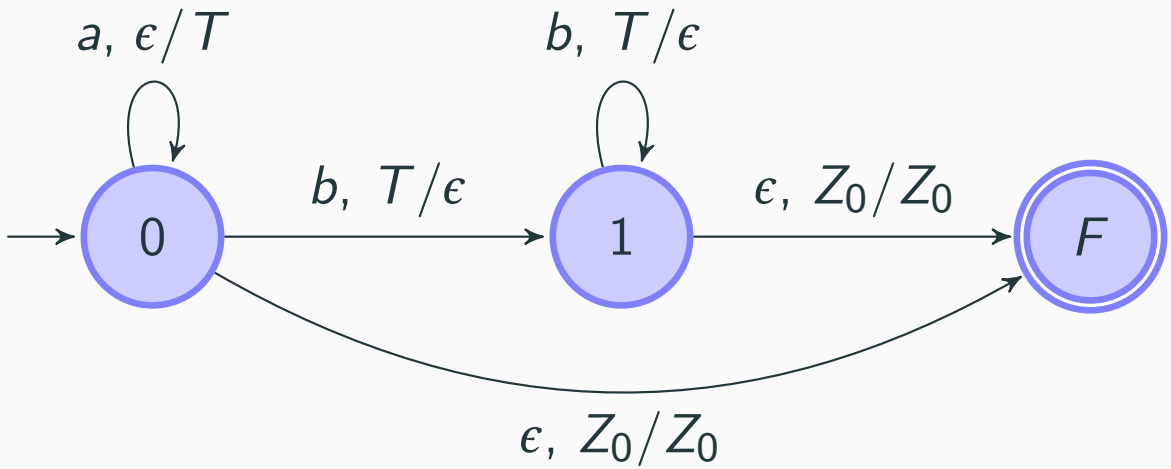
Acceptation par état final : exemple

Soit l'automate à pile avec **acceptation par état final** qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



Acceptation par état final : exemple

Soit l'automate à pile avec **acceptation par état final** qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

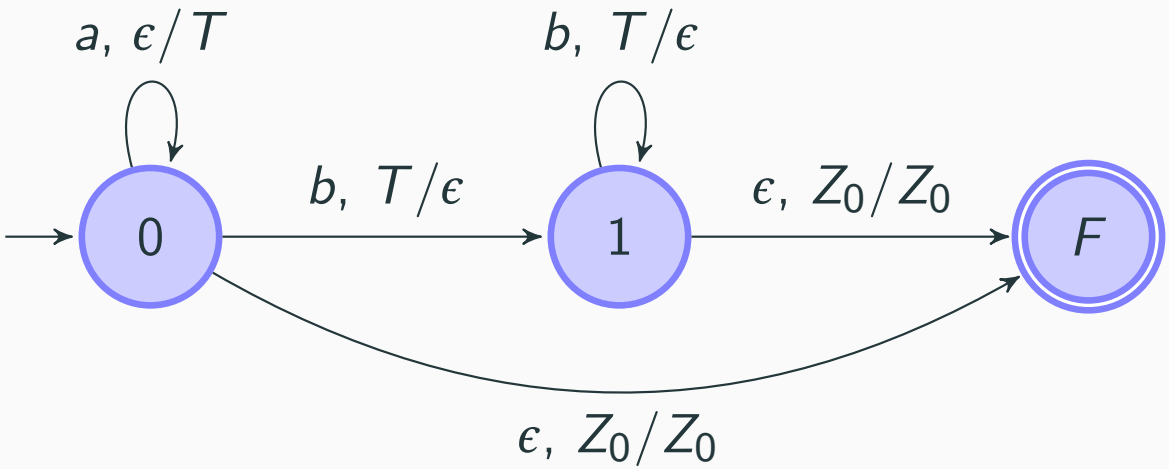


$(aabb, 0, Z_0)$	$\vdash_M$	$(abb, 0, TZ_0)$	on lit a, on ne dépile rien, on empile T
	$\vdash_M$	$(bb, 0, TTZ_0)$	on lit a, on ne dépile rien, on empile T
	$\vdash_M$	$(b, 1, TZ_0)$	on lit b, on dépile T, on empile rien
	$\vdash_M$	$(\epsilon, 1, Z_0)$	on lit b, on dépile T, on empile rien
	$\vdash_M$	(ϵ, F, Z_0)	on ne lit rien, on dépile Z_0, on empile Z_0

On obtient le mot vide dans un état final, *quel que soit l'état de la pile*.
aabb est accepté par l'automate.

Acceptation par état final : exemple

Soit l'automate à pile avec **acceptation par état final** qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

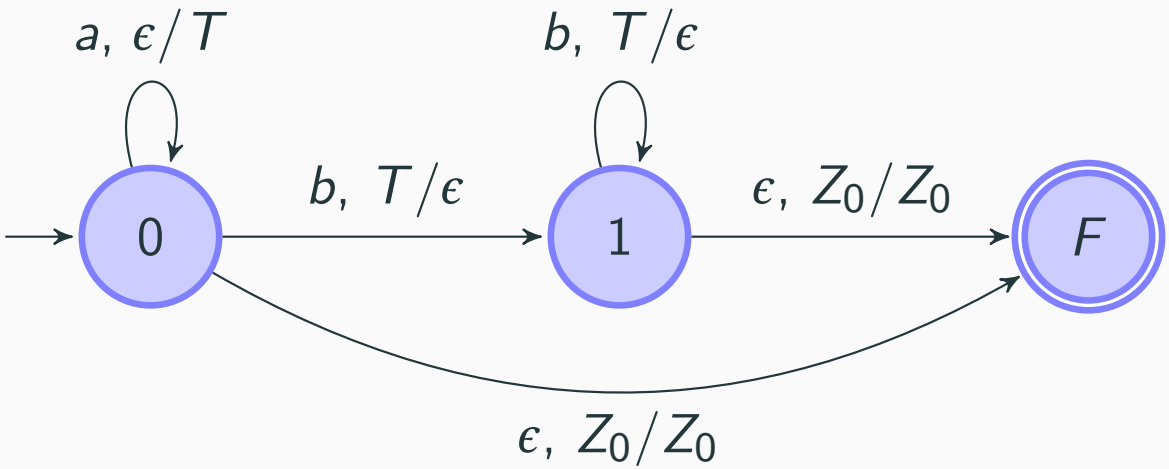


$$
\begin{array}{lll}
(aab, 0, Z_0) & \vdash_M & (ab, 0, TZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } T \\
& \vdash_M & (b, 0, TTZ_0) \quad \text{on lit } a\text{, on ne dépile rien, on empile } T \\
& \vdash_M & (\epsilon, 1, TZ_0) \quad \text{on lit } b\text{, on dépile } T\text{, on empile rien}
\end{array}
$$

Echec : On est bloqué : on ne pas dépiler T (pas de b à lire) ; on ne peut pas aller dans l'état final (on ne peut pas dépiler Z_0 qui n'est pas en sommet de la pile). aab n'est pas accepté par l'automate.

Acceptation par état final : exemple

Soit l'automate à pile avec **acceptation par état final** qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

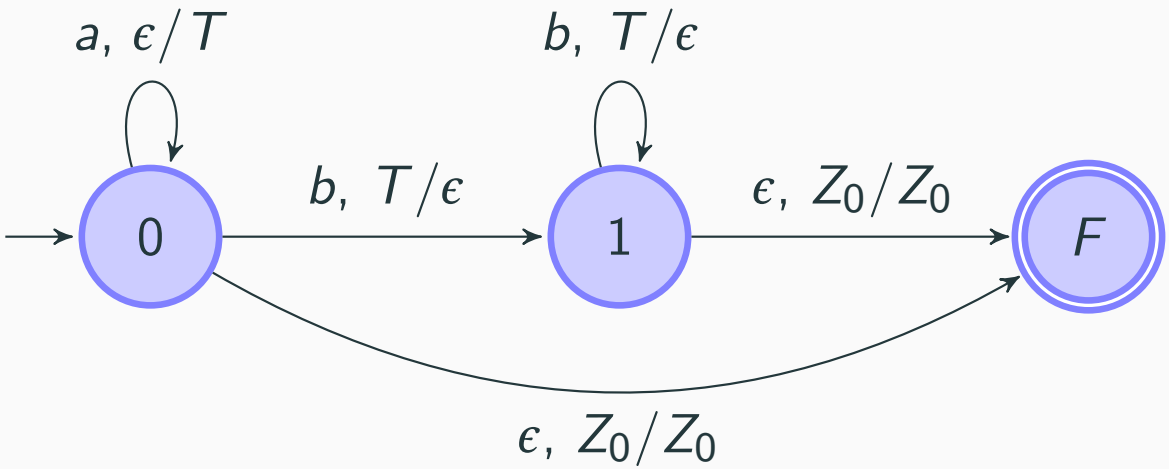


$(abb, 0, Z_0)$	$\vdash_M$	$(bb, 0, TZ_0)$	on lit a, on ne dépile rien, on empile T
	$\vdash_M$	$(b, 1, Z_0)$	on lit b, on dépile T, on empile rien (*)
	$\vdash_M$	(b, F, Z_0)	on ne lit rien, on dépile Z_0, on empileZ_0

Echec : à l'étape (*), on ne peut pas lire b car on ne peut pas dépiler T. On finit bien dans l'état final, mais le mot n'est pas vide. abb n'est pas accepté par l'automate.

Acceptation par état final : exemple

Soit l'automate à pile avec **acceptation par état final** qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



$(\epsilon, 0, Z_0) \vdash_M (\epsilon, F, Z_0)$ on ne lit rien, on dépile Z_0, on empileZ_0

On obtient le mot vide dans un état final, *quel que soit l'état de la pile*. ϵ est accepté par l'automate.

Acceptation par pile vide

Acceptation par pile vide

Un mot $m \in \Sigma^*$ est accepté par pile vide par un automate à pile $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$ si pour la configuration (m, q_0, Z_0), il existe un état $q \in Q$ tel que

$$(m, q_0, Z_0) \vdash_M^* (\epsilon, q, \epsilon)$$

Attention, la pile vide ne contient plus Z_0 !

Acceptation par pile vide

Acceptation par pile vide

Un mot $m \in \Sigma^*$ est accepté par pile vide par un automate à pile $M = (\Sigma, \Gamma, Z_0, Q, q_0, F, \delta)$ si pour la configuration (m, q_0, Z_0), il existe un état $q \in Q$ tel que

$$(m, q_0, Z_0) \vdash_M^* (\epsilon, q, \epsilon)$$

Attention, la pile vide ne contient plus Z_0 !

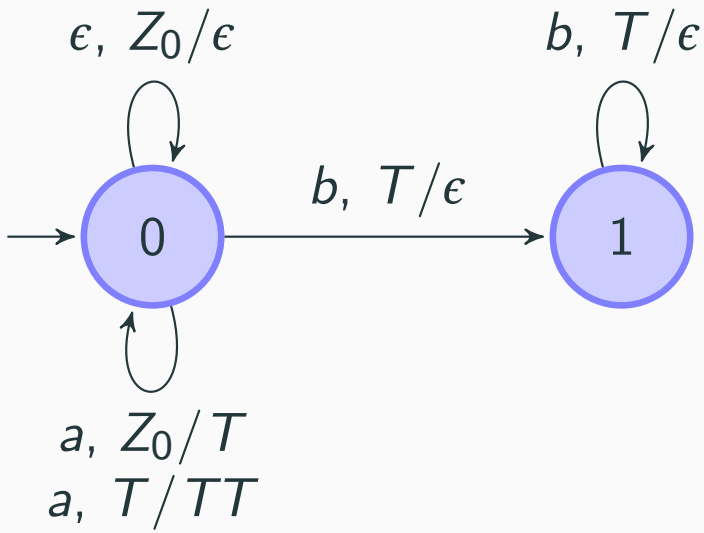
Langage accepté par pile vide

Le langage accepté par pile vide par un automate à pile est l'ensemble des mots acceptés par cet automate.

$$L^V(M) = \{m \in \Sigma^* | (m, q_0, Z_0) \vdash_M^* (\epsilon, q, \epsilon)\}$$

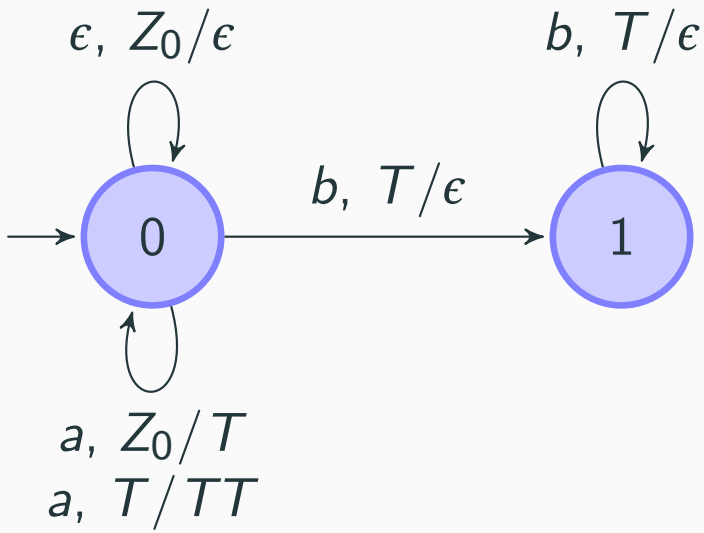
Acceptation par pile vide : exemple

Soit l'automate à pile avec **acceptation par pile vide** suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



Acceptation par pile vide : exemple

Soit l'automate à pile avec **acceptation par pile vide** suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

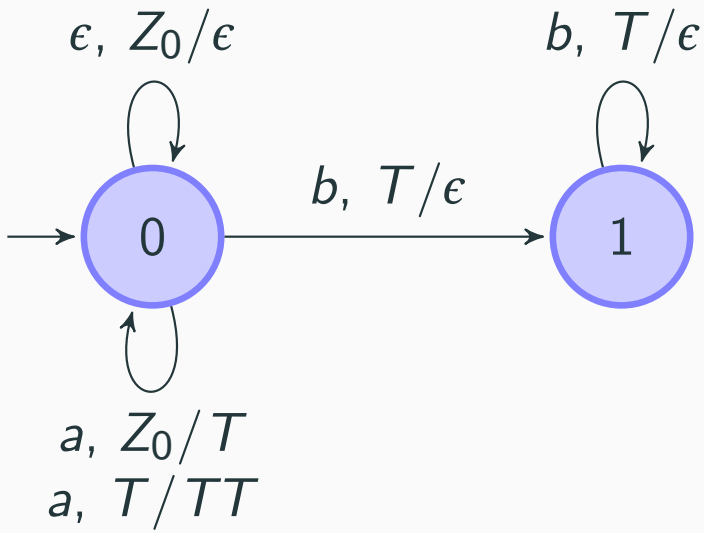


$$
\begin{array}{lll}
(aabb, 0, Z_0) & \vdash_M (abb, 0, T) & \text{on lit } a\text{, on dépile } Z_0\text{, on empile } T \\
& \vdash_M (bb, 0, TT) & \text{on lit } a\text{, on dépile } T\text{, on empile } TT \\
& \vdash_M (b, 1, T) & \text{on lit } b\text{, on dépile } T\text{, on empile rien} \\
& \vdash_M (\epsilon, 1, \epsilon) & \text{on lit } b\text{, on dépile } T\text{, on empile rien}
\end{array}
$$

On obtient le mot vide et une pile vide, *quel que soit l'état dans lequel on se trouve*. *aabb* est accepté par l'automate.

Acceptation par pile vide : exemple

Soit l'automate à pile avec **acceptation par pile vide** suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

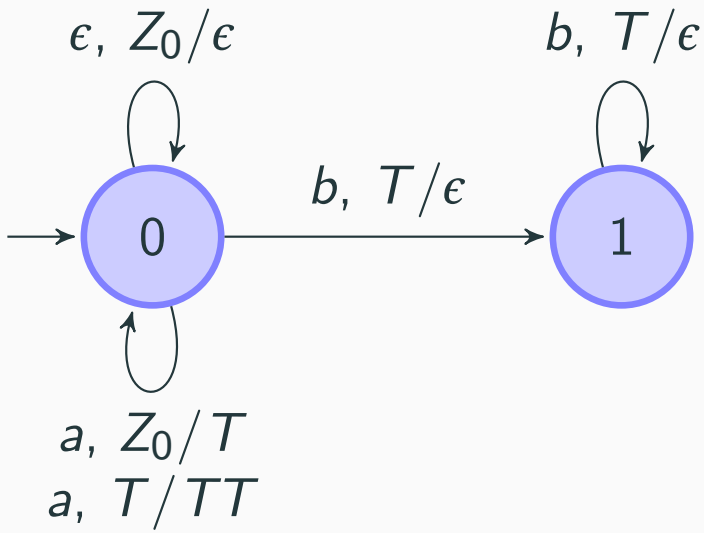


$$
\begin{array}{lll}
(aab, 0, Z_0) & \vdash_M & (ab, 0, T) \quad \text{on lit } a\text{, on dépile } Z_0\text{, on empile } T \\
& \vdash_M & (b, 0, TT) \quad \text{on lit } a\text{, on dépile } T\text{, on empile } TT \\
& \vdash_M & (\epsilon, 1, T) \quad \text{on lit } b\text{, on dépile } T\text{, on empile rien}
\end{array}
$$

Echec : le mot est vide, mais la pile ne l'est pas. On ne peut suivre aucune autre transition. *aab* n'est pas accepté par l'automate.

Acceptation par pile vide : exemple

Soit l'automate à pile avec **acceptation par pile vide** suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$

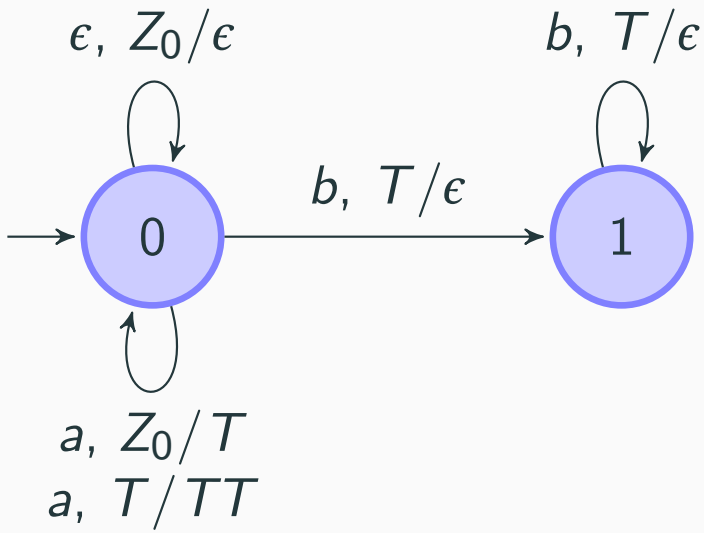


$$
\begin{array}{lll}
(abb, 0, Z_0) & \vdash_M & (bb, 0, T) \quad \text{on lit } a\text{, on dépile } Z_0\text{, on empile } T \\
 & \vdash_M & (b, 1, \epsilon) \quad \text{on lit } b\text{, on dépile } T\text{, on empile rien}
\end{array}
$$

Echec : on ne peut pas lire b car on ne peut pas dépiler T. On a bien une pile vide, mais le mot n'est lui pas vide. abb n'est pas accepté par l'automate.

Acceptation par pile vide : exemple

Soit l'automate à pile avec **acceptation par pile vide** suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



$(\epsilon, 0, Z_0) \quad \vdash_M \quad (\epsilon, 0, \epsilon)$ on ne lit rien, on dépile Z_0, on empile rien

On obtient le mot vide et la pile vide, *quel que soit l'état dans lequel on se trouve*. ϵ est accepté par l'automate.

Critères d'acceptation

- Les deux critères d'acceptation (par état final et par pile vide) sont équivalents

Théorème

Un langage est accepté par un automate à pile avec le critère d'acceptation sur pile vide si et seulement si il est accepté par un automate à pile avec acceptation par état final.

Critères d'acceptation

- Les deux critères d'acceptation (par état final et par pile vide) sont équivalents

Théorème

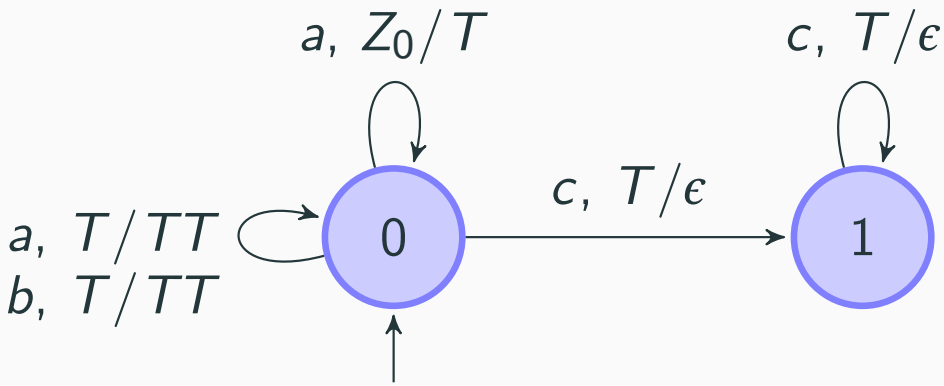
Un langage est accepté par un automate à pile avec le critère d'acceptation sur pile vide si et seulement si il est accepté par un automate à pile avec acceptation par état final.

$\Rightarrow$ Chaque transition dans laquelle Z_0 est dépilé est remplacée par une transition vers un **nouvel** état final

$\Leftarrow$ Après avoir atteint un état final, on vide entièrement la pile

Automate avec acceptation par pile vide

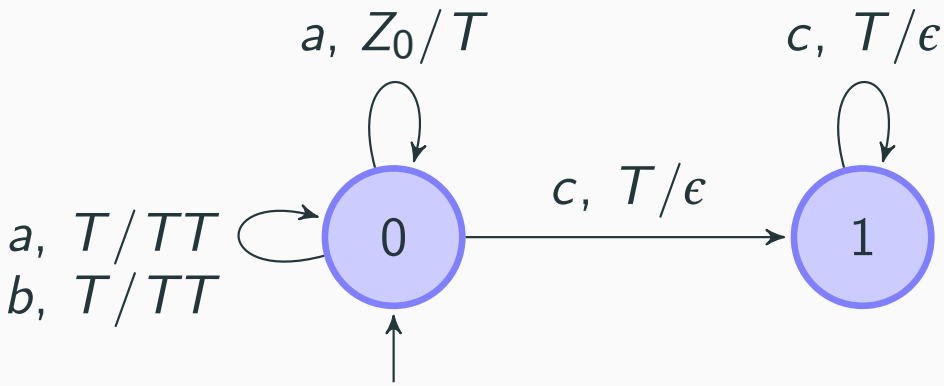
Soit l'automate à pile avec **acceptation par pile vide** suivant :



:

Automate avec acceptation par pile vide

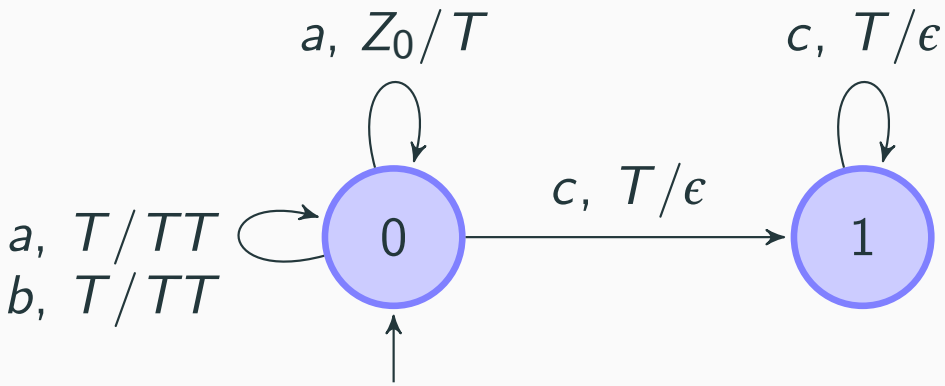
Soit l'automate à pile avec **acceptation par pile vide** suivant :



Cet automate accepte le langage $\{a(a+b)^{n-1}c^n | n > 0\}$:

Automate avec acceptation par pile vide

Soit l'automate à pile avec **acceptation par pile vide** suivant :

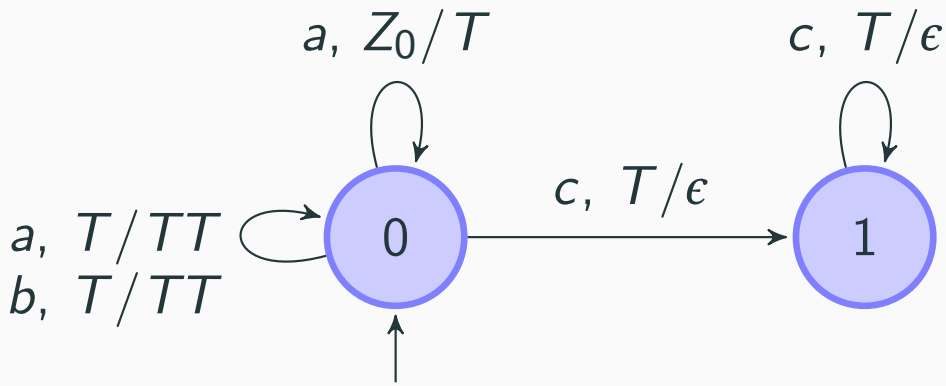


Cet automate accepte le langage $\{a(a+b)^{n-1}c^n | n > 0\}$:

- On doit lire a en premier (Z_0 en sommet de pile)

Automate avec acceptation par pile vide

Soit l'automate à pile avec **acceptation par pile vide** suivant :

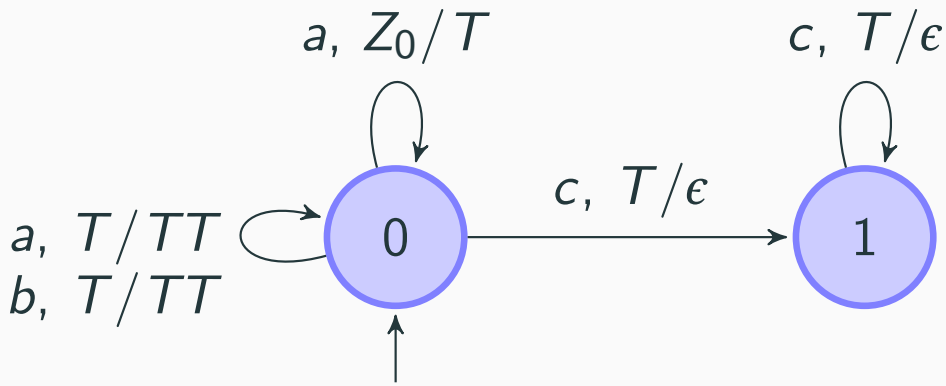


Cet automate accepte le langage $\{a(a+b)^{n-1}c^n | n > 0\}$:

- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
  → il y a autant de T dans la pile que de a et b lus

Automate avec acceptation par pile vide

Soit l'automate à pile avec **acceptation par pile vide** suivant :

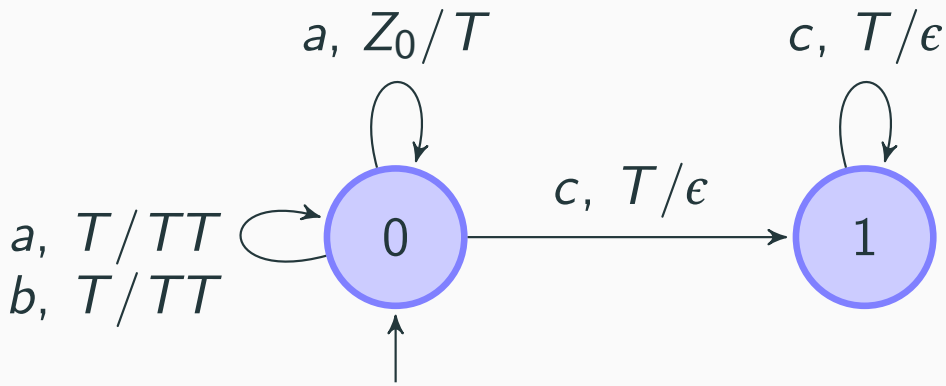


Cet automate accepte le langage $\{a(a+b)^{n-1}c^n | n > 0\}$:

- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
  → il y a autant de T dans la pile que de a et b lus
- On doit ensuite lire autant de c que de a et b lu (pour dépiler tous les T et avoir une pile vide)

Automate avec acceptation par pile vide

Soit l'automate à pile avec **acceptation par pile vide** suivant :

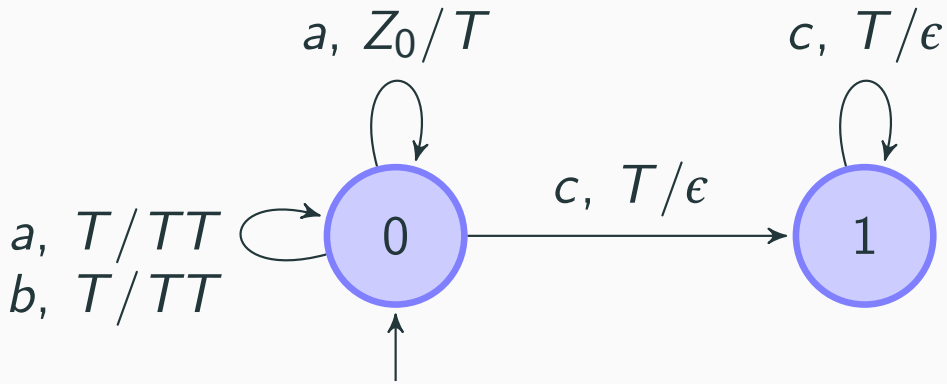


Cet automate accepte le langage $\{a(a+b)^{n-1}c^n | n > 0\}$:

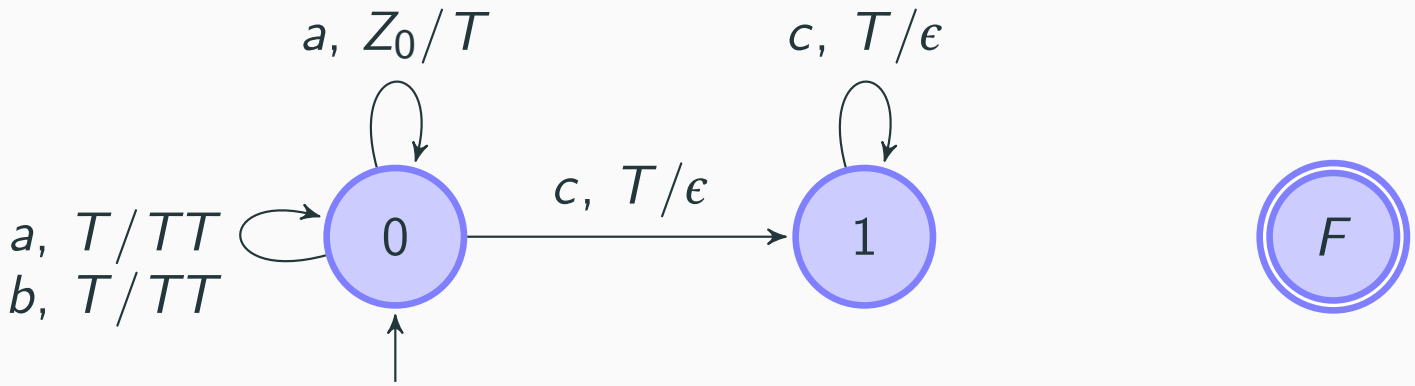
- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
  → il y a autant de T dans la pile que de a et b lus
- On doit ensuite lire autant de c que de a et b lu (pour dépiler tous les T et avoir une pile vide)
- Le mot vide n'est pas accepté : il faut lire a pour dépiler Z_0

Acceptation par pile vide → acceptation par état final

Soit l'automate à pile avec **acceptation par pile vide** suivant :

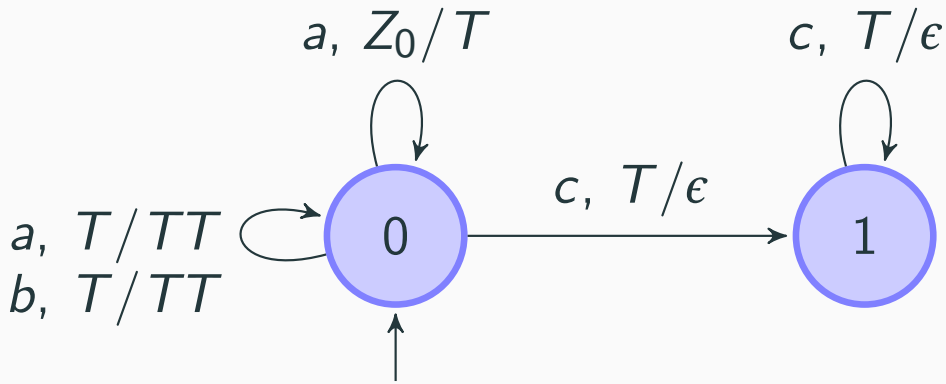


Construction de l'aut. à pile avec **acceptation par état final** équivalent :

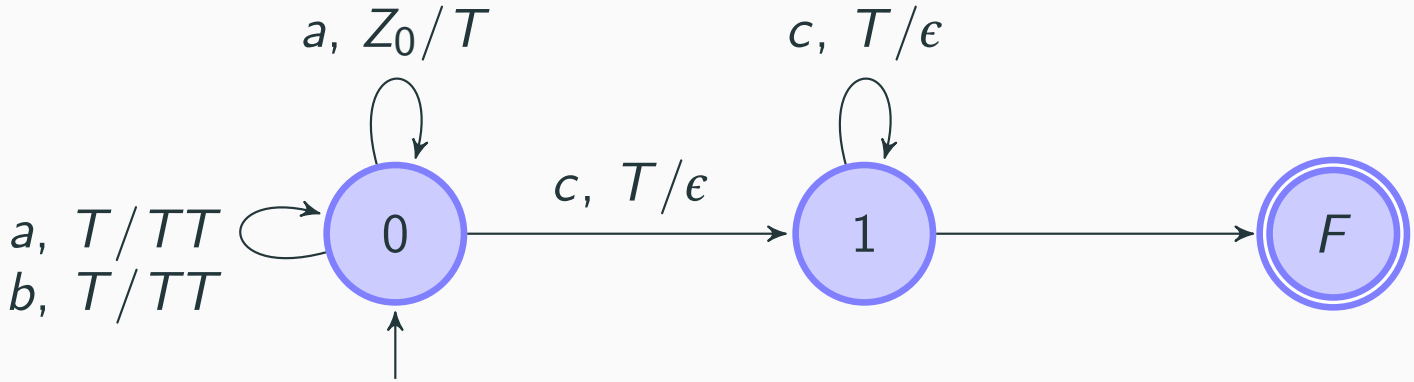


Acceptation par pile vide → acceptation par état final

Soit l'automate à pile avec **acceptation par pile vide** suivant :



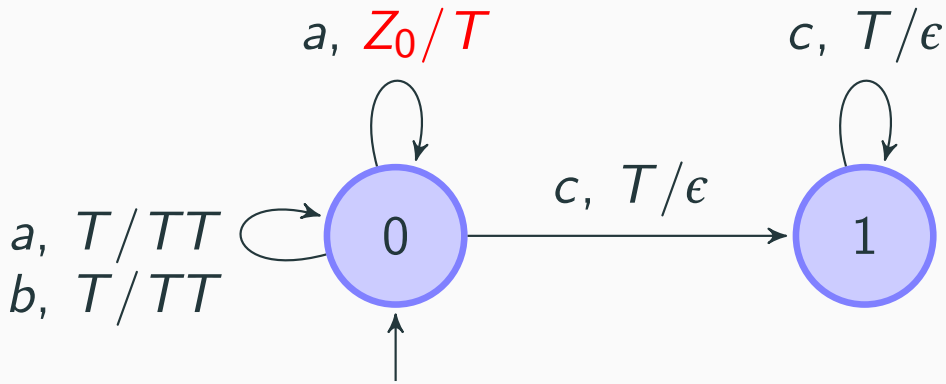
Construction de l'aut. à pile avec **acceptation par état final** équivalent :



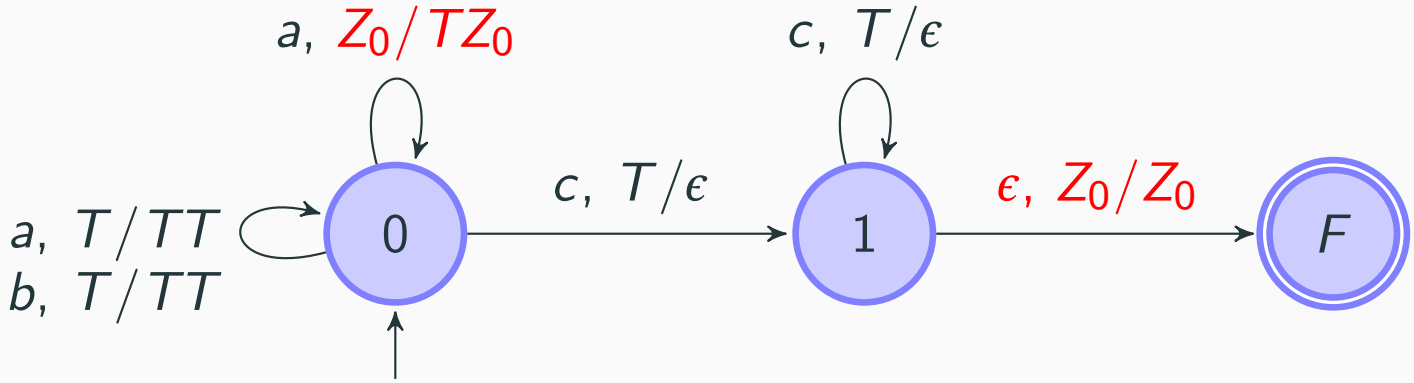
- Mot le plus court : ac. On peut donc aller à l'état final uniquement à partir de l'état 1

Acceptation par pile vide → acceptation par état final

Soit l'automate à pile avec **acceptation par pile vide** suivant :



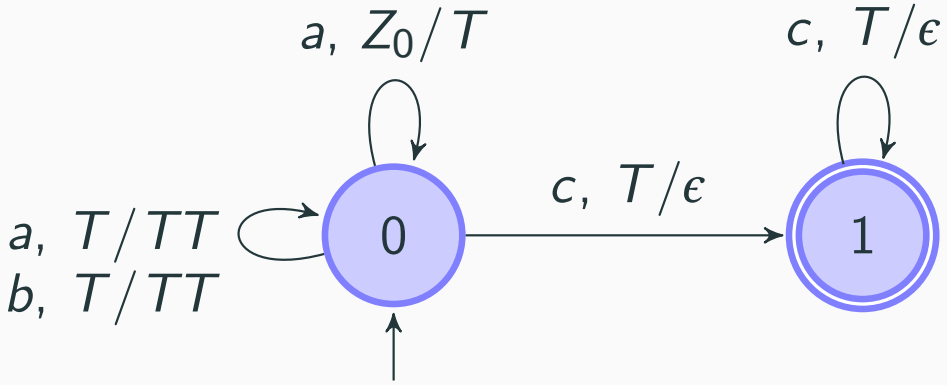
Construction de l'aut. à pile avec **acceptation par état final** équivalent :



- Il faut avoir dépilé tous les T : Z_0 doit être en sommet de pile
- Il faut donc que Z_0 ne soit pas dépilé à l'état 0

Automate avec acceptation par état final

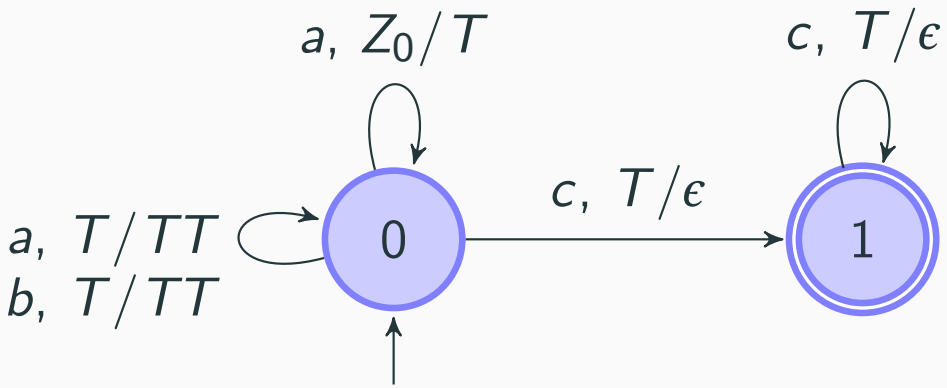
Soit l'automate à pile avec **acceptation par état final** suivant :



:

Automate avec acceptation par état final

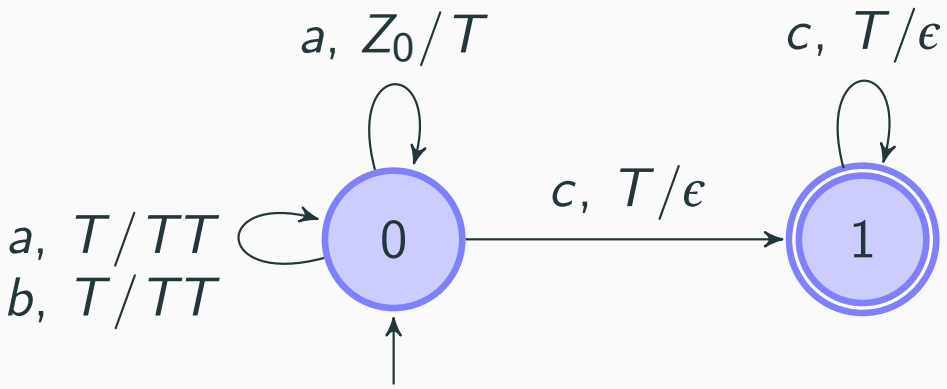
Soit l'automate à pile avec **acceptation par état final** suivant :



Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

Automate avec acceptation par état final

Soit l'automate à pile avec **acceptation par état final** suivant :

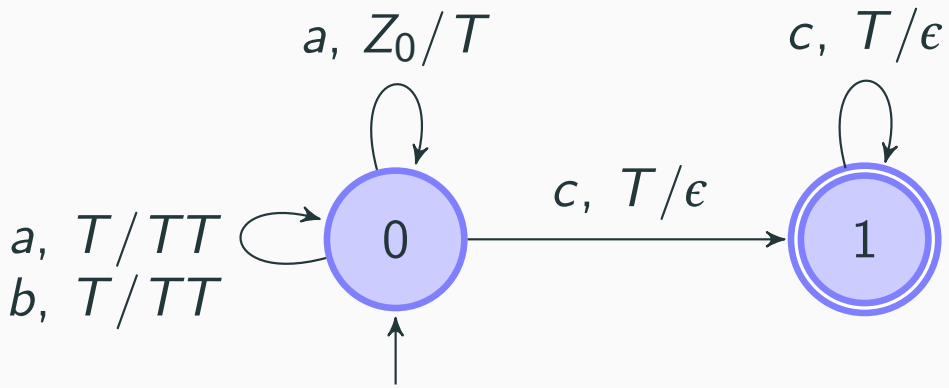


Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

- On doit lire a en premier (Z_0 en sommet de pile)

Automate avec acceptation par état final

Soit l'automate à pile avec **acceptation par état final** suivant :

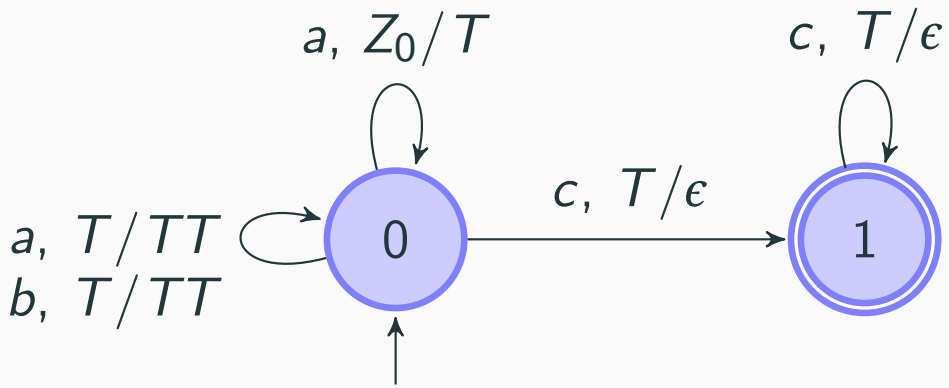


Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)

Automate avec acceptation par état final

Soit l'automate à pile avec **acceptation par état final** suivant :

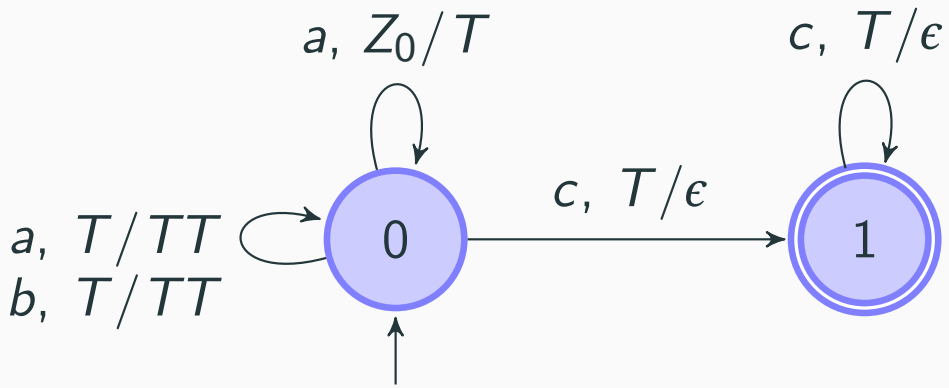


Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
- On doit ensuite lire au moins un c pour arriver dans l'état final

Automate avec acceptation par état final

Soit l'automate à pile avec **acceptation par état final** suivant :

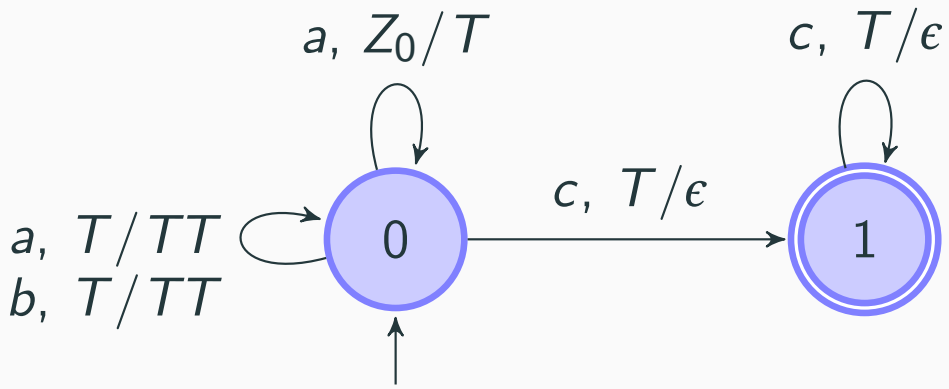


Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
- On doit ensuite lire au moins un c pour arriver dans l'état final
- On peut s'arreter quand on veut, et on ne peux pas lire plus de c que de a et de b

Automate avec acceptation par état final

Soit l'automate à pile avec **acceptation par état final** suivant :

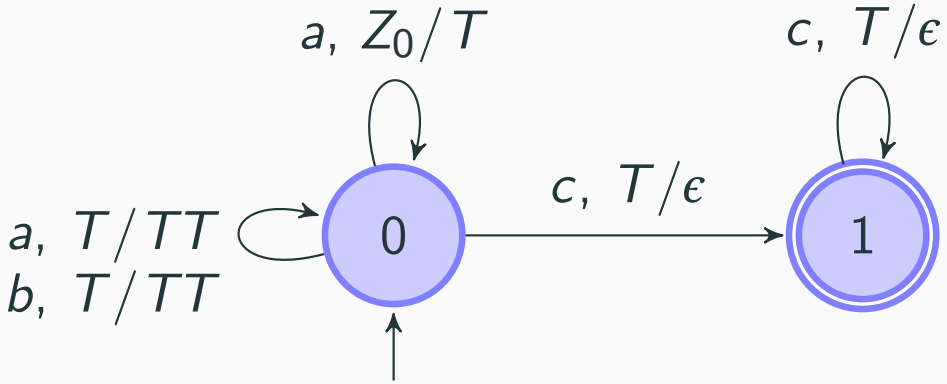


Cet automate accepte le langage $\{a(a+b)^n c^m | n \geq 0, m > 0, m \leq n\}$:

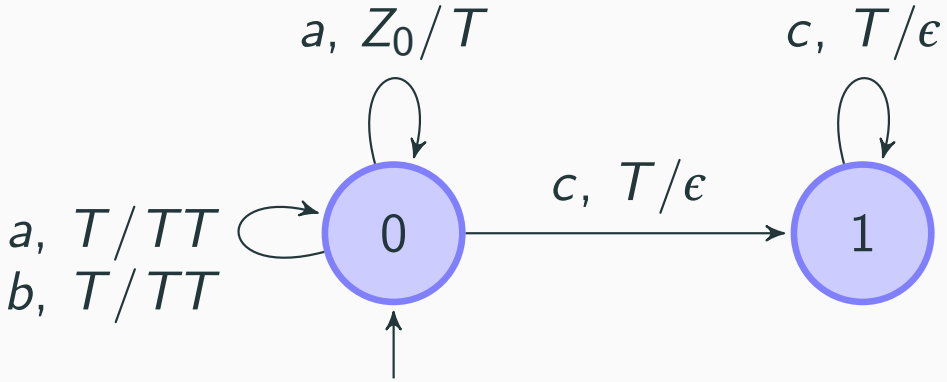
- On doit lire a en premier (Z_0 en sommet de pile)
- On lit ensuite autant de a et de b que l'on veut (on dépile un T et empile deux T à chaque fois)
- On doit ensuite lire au moins un c pour arriver dans l'état final
- On peut s'arreter quand on veut, et on ne peux pas lire plus de c que de a et de b
- Le mot vide n'est pas accepté : il faut avoir un T dans la pile pour arriver dans l'état final

Acceptation par état final → acceptation par pile vide

Soit l'automate à pile avec **acceptation par état final** suivant :

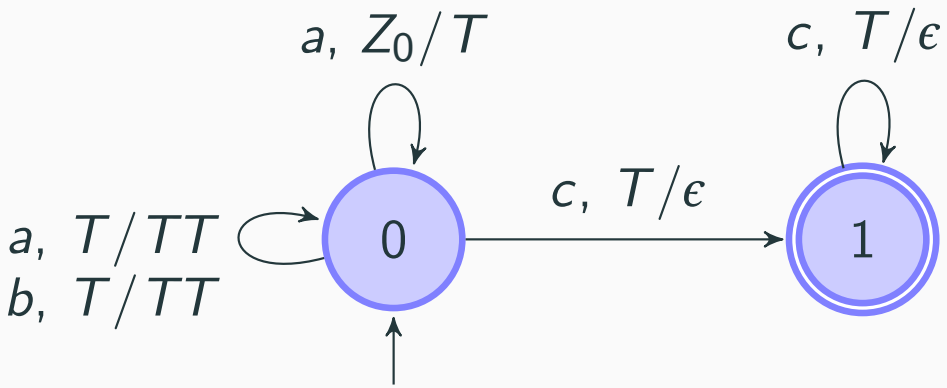


Construction de l'aut. à pile avec **acceptation par pile vide** équivalent :

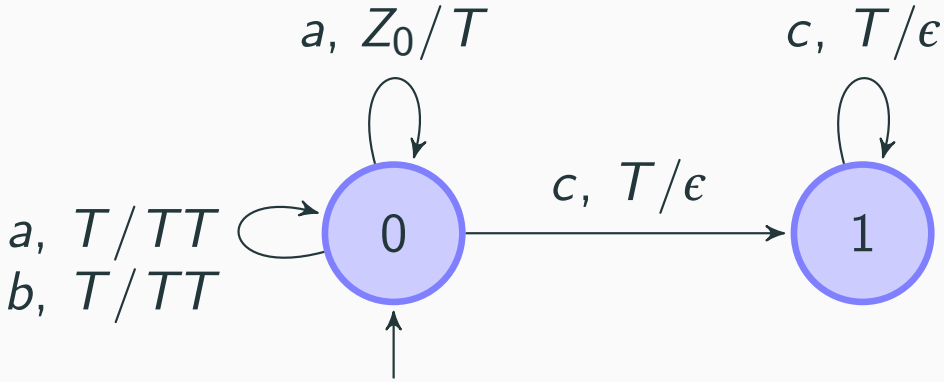


Acceptation par état final → acceptation par pile vide

Soit l'automate à pile avec **acceptation par état final** suivant :



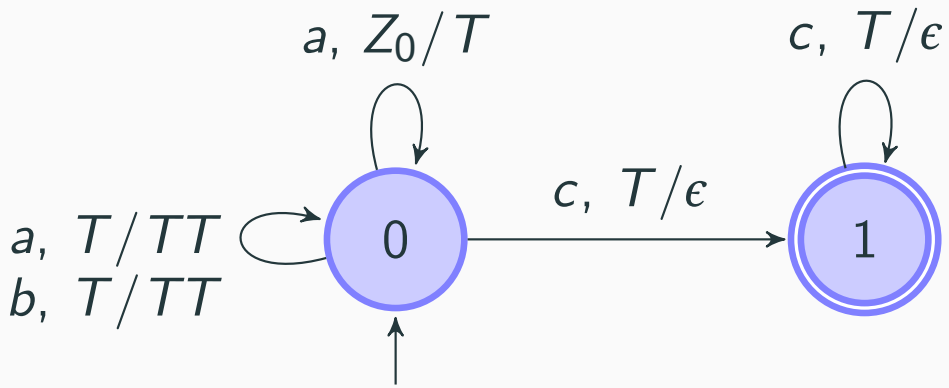
Construction de l'aut. à pile avec **acceptation par pile vide** équivalent :



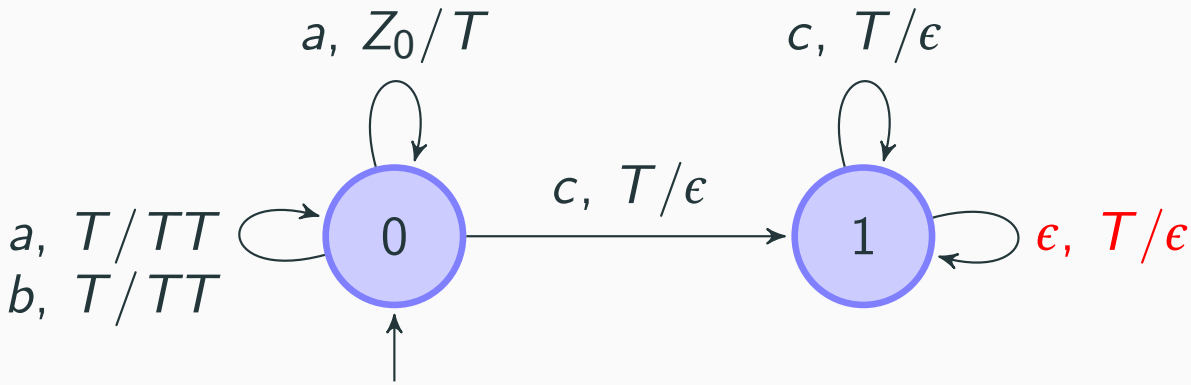
- Il suffit de vider la pile dans l'ancien état final

Acceptation par état final → acceptation par pile vide

Soit l'automate à pile avec **acceptation par état final** suivant :



Construction de l'aut. à pile avec **acceptation par pile vide** équivalent :



- Il suffit de vider la pile dans l'ancien état final
- Seuls des T peuvent être présents encore dans la pile

Automates à pile déterministes

Automates déterministes

- Les automates à pile que nous avons défini jusqu'à maintenant sont indéterministes
  - Un mot est accepté s'il existe **au moins** une suite de configurations qui conduit à l'acceptation
  - Mais il peut y en avoir plusieurs
  - Et il peut il y avoir plusieurs suites de configuration qui mènent à l'échec

$\Rightarrow$ Automate à pile déterministe ?

Automates déterministes

- Un automate à pile M est déterministe à 2 conditions :

Automates déterministes

- Un automate à pile M est déterministe à 2 conditions :
- Première condition
  - pour un état q donné
  - pour un symbole d'entrée x donné
  - pour un sommet de pile Z donné

  il existe au plus une transition partant de (q, x, Z)

Automates déterministes

- Un automate à pile M est déterministe à 2 conditions :
- Première condition
  - pour un état q donné
  - pour un symbole d'entrée x donné
  - pour un sommet de pile Z donné

  il existe au plus une transition partant de (q, x, Z)
- Seconde condition
  - pour un état q donné
  - pour un sommet de pile Z donné

  s'il existe une transition partant de (q, ϵ, Z), elle est unique et pour toute lettre x, il n'en existe pas partant de (q, x, Z).

Automates déterministes

- Un automate à pile M est déterministe à 2 conditions :
- Première condition
  - pour un état q donné
  - pour un symbole d'entrée x donné
  - pour un sommet de pile Z donné

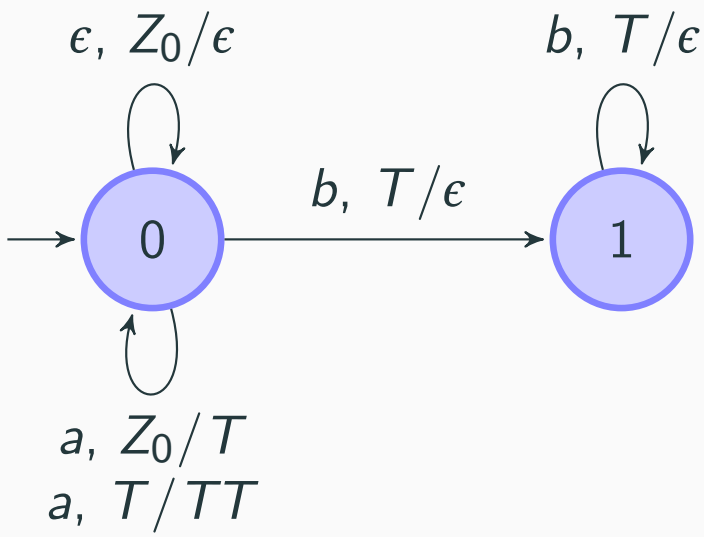
  il existe au plus une transition partant de (q, x, Z)
- Seconde condition
  - pour un état q donné
  - pour un sommet de pile Z donné

  s'il existe une transition partant de (q, ϵ, Z), elle est unique et pour toute lettre x, il n'en existe pas partant de (q, x, Z).

$\Rightarrow$ **Dans une configuration donnée, on ne peut pas avoir le choix sur la transition à appliquer**

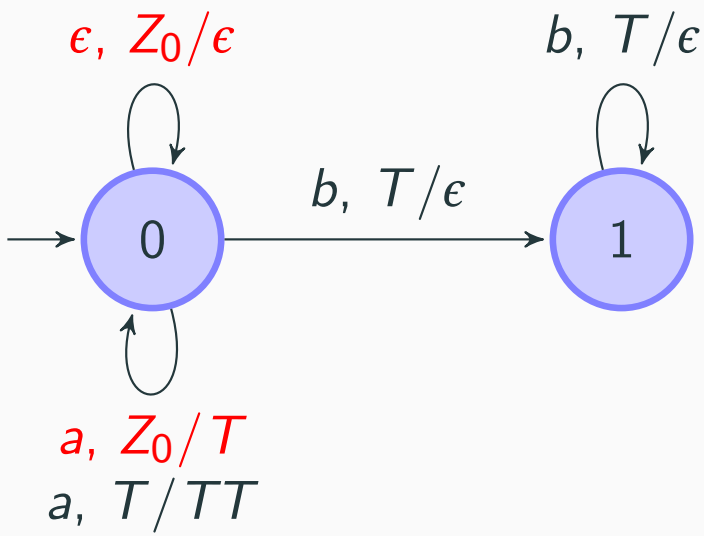
Automate déterministe ?

Soit l'automate à pile avec acceptation par pile vide suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



Automate déterministe ?

Soit l'automate à pile avec acceptation par pile vide suivant qui reconnaît le langage $\{a^n b^n | n \geq 0\}$



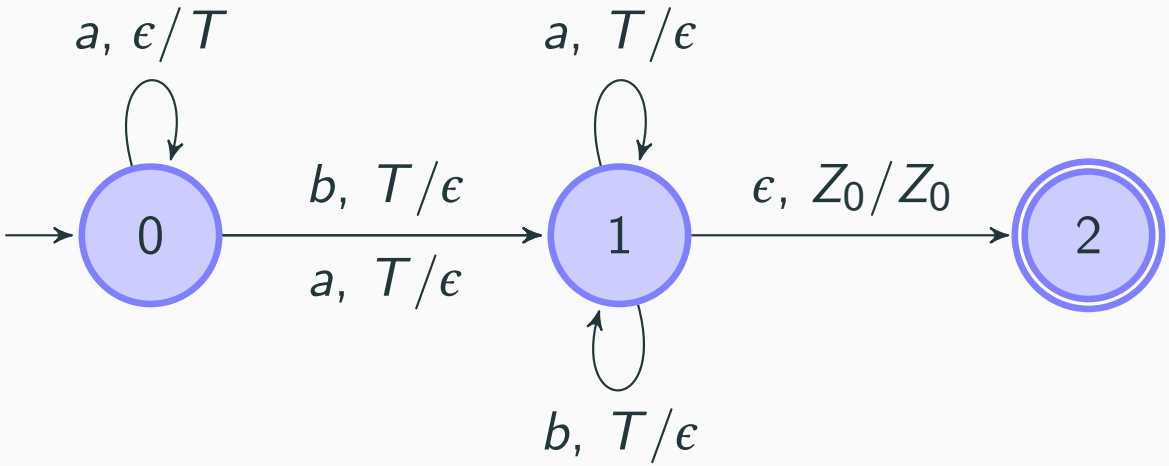
Cet automate n'est **pas déterministe** *(condition 2)* :

$(aabb, 0, Z_0)$	$\vdash_M$	$(abb, 0, T)$	on lit a, on dépile Z_0, on empile T
$(aabb, 0, Z_0)$	$\vdash_M$	$(aabb, 0, \epsilon)$	on ne lit rien, on dépile Z_0, on empile rien

Il y a le choix entre 2 transitions.

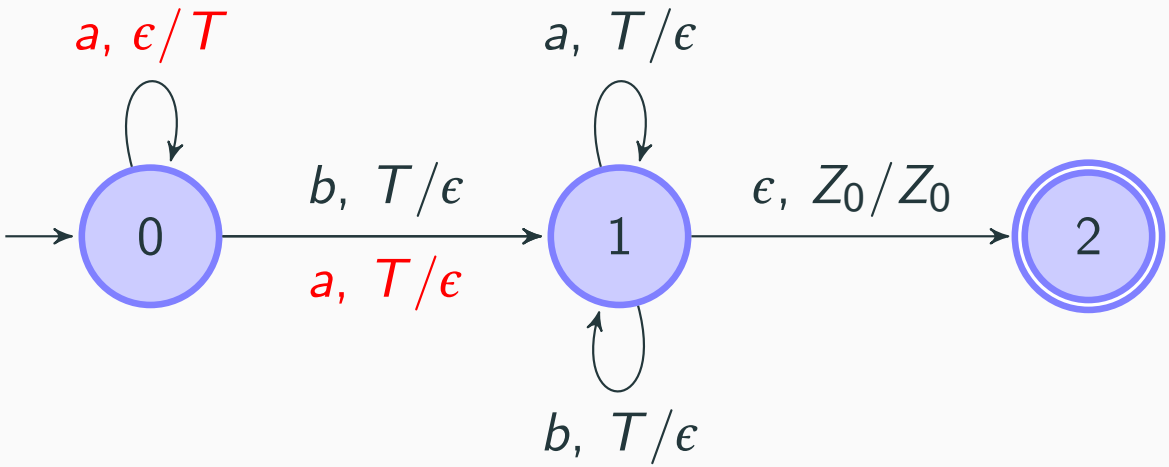
Automate déterministe ?

Soit l'automate à pile avec acceptation par état final suivant qui reconnaît le langage $\{a^n(a+b)^n | n \geq 0\}$



Automate déterministe ?

Soit l'automate à pile avec acceptation par état final suivant qui reconnaît le langage $\{a^n(a+b)^n | n \geq 0\}$



Cet automate n'est **pas déterministe** *(condition 1)* :

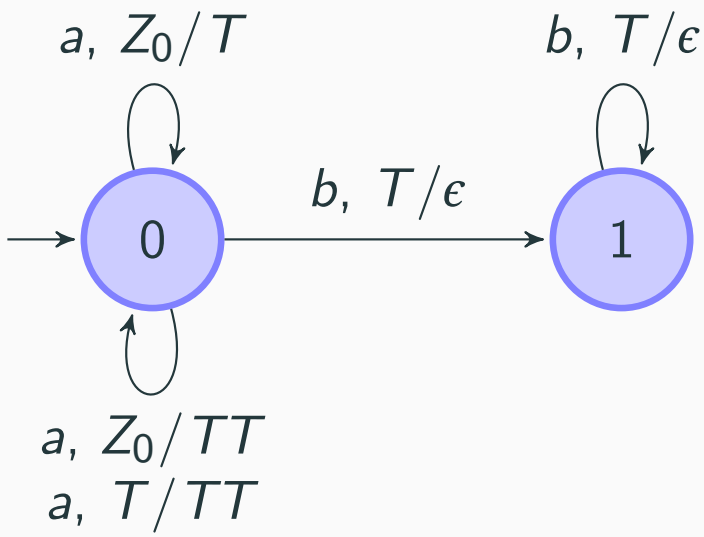
$(aba, 0, TZ_0) \vdash_M (ba, 0, TTZ_0)$ on lit a, on ne dépile rien, on empile T

$(aba, 0, TZ_0) \vdash_M (ba, 1, Z_0)$ on lit a, on dépile T, on empile rien

Il y a le choix entre 2 transitions.

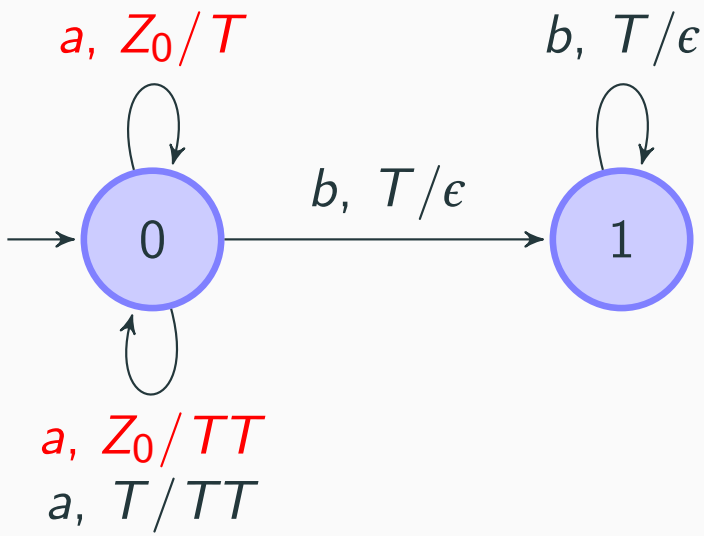
Automate déterministe ?

Soit l'automate à pile avec acceptation par pile vide suivant qui reconnaît le langage $\{a^n b^m | n > 0, m = n \text{ ou } m = n+1\}$



Automate déterministe ?

Soit l'automate à pile avec acceptation par pile vide suivant qui reconnaît le langage $\{a^n b^m | n > 0, m = n \text{ ou } m = n+1\}$



Cet automate n'est **pas déterministe** *(condition 1)* :

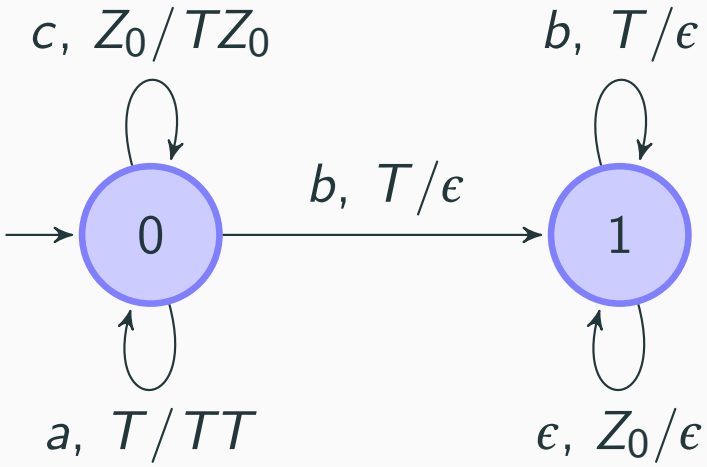
$(abb, 0, Z_0) \vdash_M (bb, 0, T)$ on lit a, on dépile Z_0, on empile T

$(abb, 0, Z_0) \vdash_M (bb, 0, TT)$ on lit a, on dépile Z_0, on empile TT

Il y a le choix entre 2 transitions.

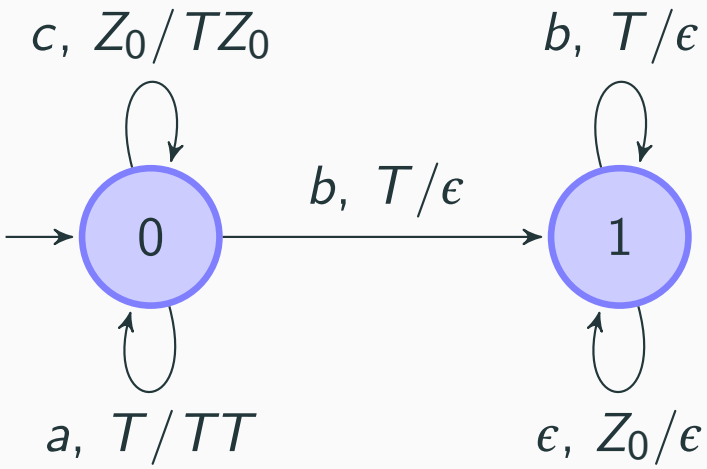
Automate déterministe ?

Soit l'automate à pile par acceptation par pile vide suivant qui reconnaît le langage $\{ca^{n-1}b^n | n > 0\}$



Automate déterministe ?

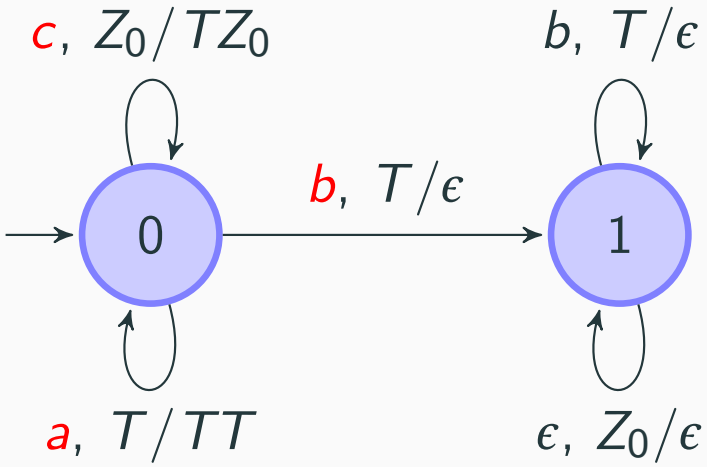
Soit l'automate à pile par acceptation par pile vide suivant qui reconnaît le langage $\{ca^{n-1}b^n | n > 0\}$



Cet automate **est déterministe** :

Automate déterministe ?

Soit l'automate à pile par acceptation par pile vide suivant qui reconnaît le langage $\{ca^{n-1}b^n | n > 0\}$

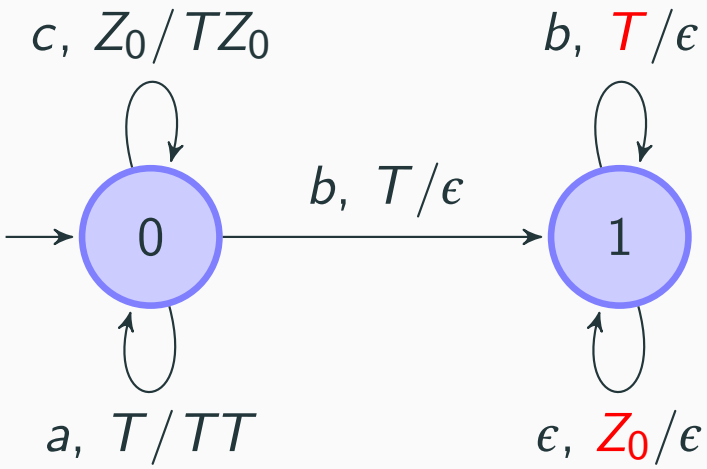


Cet automate **est déterministe** :

- Transitions à partir de l'état 0 : pas le choix sur la lettre à lire

Automate déterministe ?

Soit l'automate à pile par acceptation par pile vide suivant qui reconnaît le langage $\{ca^{n-1}b^n | n > 0\}$



Cet automate **est déterministe** :

- Transitions à partir de l'état 0 : pas le choix sur la lettre à lire
- Transitions à partir de l'état 1 : pas le choix sur le symbole à dépiler