Chapitre 2 : Automates à Piles (Semaines 5 & 6)

Etienne Kouokam

Département d'Informatique
Université de Yaoundé I, Cameroun

Année académique 2019-2020
Univ de Ydé I : Mars-Juin 2020

Plan

1. Automates à Piles
   - Généralités sur les automates à pile
   - Équivalence entre PDA et CFG

Automates finis Vs Automates à Pile

Limites des automates finis

Le modèle des automates finis souffre d'une limitation manifeste liée à sa mémoire qui est :

- limitée
- intégralement résumée par l'état courant de l'automate

En se servant d'un automate fini :

- Possible de représenter le langage $L_1 = a^n b^m, \ n, m \geq 0$ dénoté par l'expression régulière $a^* b^*$
- Impossible de représenter le langage $L_2 = a^n b^n$ pourtant contenu dans $L_1 = a^n b^m$. Il n'y a véritablement aucun mécanisme mis en œuvre pour compter.

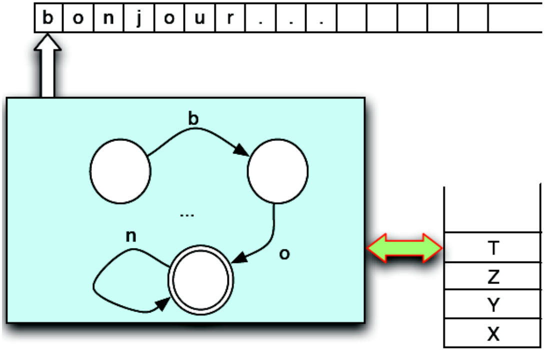
Pour palier cette limitation, on dote l'automate fini d'une mémoire infinie, que nous allons modéliser par une pile qui :

- aide à garder une mémoire (non-bornée en taille) des étapes de calculs passées
- peut conditionner les étapes de calcul à venir

Automate à Pile ou PDA (PDA pour Push Down Automata)

Un PDA est essentiellement un ϵ-AFN avec une pile (stack). Lors d'une transition, le PDA :

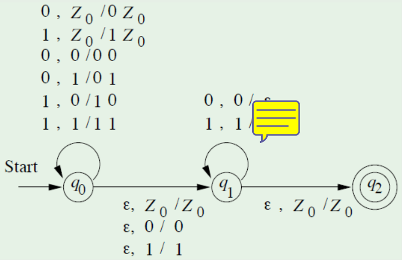
1. Consomme un symbole d'entrée (ou non si ϵ-transition)
2. Change d'état de contrôle
3. Remplace le symbole T en sommet de la pile par un string (ϵ (pop), T (pas de changement), AT (push un A))



Exemple : PDA pour $L_{ww^r} = \{ww^R \mid w \in \{0,1\}^*\})$

La grammaire correspondante est définie par $P \rightarrow 0P0 \mid 1P1 \mid \epsilon$
On peut construire un PDA équivalent à 3 états fonctionnels comme suit :

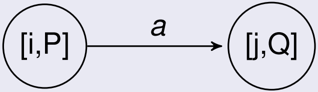
- En 0 : Il peut supposer lire w : push le symbole sur la pile
- En 0 : Il peut supposer être au milieu de l'input : va dans l'état 1
- En 1 : Il compare ce qui est lu et ce qui est sur la pile : s'ils sont identiques, la comparaison est correcte, il pop le sommet de la pile et continue (sinon bloque)
- En 1 : S'il retombe sur le symbole de pile initial, va dans l'état final 2.



Définition formelle d'un PDA

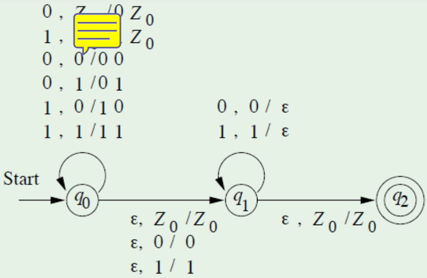
Un PDA est un heptuplet $P = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ tel que :

- Q est l'ensemble fini des états dont q_0 est l'initial (unique)
- Σ est un ensemble fini de symboles (alphabet d'entrée)
- Γ est un ensemble fini de symboles de pile dont Z_0 est l'initial.
- $\delta : Q \times (\Sigma \cup \{\epsilon\} \times \Gamma) \rightarrow Q \times \Gamma^*$ est la **fonction** de transition dans le cas où l'automate est déterministe.
  Parfois, plus précisément dans le cas où l'automate à pile n'est pas déterministe, on utilise plutôt des **relations** de transition, auquel cas on a : $\delta : Q \times (\Sigma \cup \{\epsilon\} \times \Gamma) \rightarrow 2^{Q \times \Gamma^*}$
- La transition ([i,P], a, [j,Q]) fait passer de l'état i à l'état j en dépilant P puis en empilant Q.



- $F \subseteq Q$ est un ensemble d'états accepteurs

Exemple(1/2)

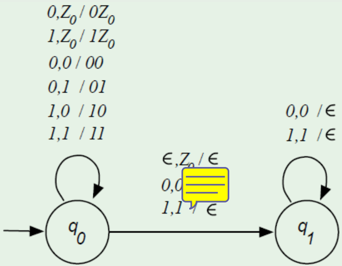


$$P = \langle \{q_0, q_1, q_2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_0, Z_0, \{q_2\} \rangle$$

δ	$(0, Z_0)$	$(1, Z_0)$	$(0, 0)$	$(0, 1)$	$(1, 0)$	$(1, 1)$
$\rightarrow q_0$	$\{(q_0, 0Z_0)\}$	$\{(q_0, 1Z_0)\}$	$\{(q_0, 00)\}$	$\{(q_0, 01)\}$	$\{(q_0, 10)\}$	$\{(q_0, 11)\}$
q_1	$\emptyset$	$\emptyset$	$\{(q_1, \epsilon)\}$	$\emptyset$	$\emptyset$	$\{(q_1, \epsilon)\}$
$*q_2$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$

δ	(ϵ, Z_0)	$(\epsilon, 0)$	$(\epsilon, 1)$
$\rightarrow q_0$	$\{(q_1, Z_0)\}$	$\{(q_1, 0)\}$	$\{(q_1, 1)\}$
q_1	$\{(q_2, Z_0)\}$	$\emptyset$	$\emptyset$
$*q_2$	$\emptyset$	$\emptyset$	$\emptyset$

Exemple(2/2)



$$P = \langle \{q_0, q_1\}, \{0,1\}, \{0,1,Z_0\}, \delta, q_0, Z_0, \emptyset \rangle$$

δ	$(0, Z_0)$	$(1, Z_0)$	$(0, 0)$	$(0, 1)$
$\rightarrow q_0$	$\{(q_0, 0Z_0)\}$	$\{(q_0, 1Z_0)\}$	$\{(q_0, 00), (q_1, \epsilon)\}$	$\{(q_0, 01)\}$
q_1	$\emptyset$	$\emptyset$	$\{(q_1, \epsilon)\}$	$\emptyset$

δ	$(1, 0)$	$(1, 1)$	(ϵ, Z_0)
$\rightarrow q_0$	$\{(q_0, 10)\}$	$\{(q_0, 11), (q_1, \epsilon)\}$	$\{(q_1, \epsilon)\}$
q_1	$\emptyset$	$\{(q_1, \epsilon)\}$	$\{(q_1, \epsilon)\}$

Autres variantes

Table de transitions

Un automate à pile peut se décrire aussi à l'aide d'une table de transitions dans laquelle les transitions sont spécifiées.

- Les lignes correspondent aux états,
- Les colonnes correspondent aux étiquettes possibles.
- Dans les cases, figurent les triplets (état d'arrivée ; symbole dépilé ; symbole empilé) ; il peut y en avoir plusieurs par case, ou aucun. Les états particuliers sont signalés dans les dernières colonnes.

Plus généralement, on a dans cette représentation
$\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow Q \times \Gamma \times \Gamma$ pour la **fonction** de transition ou encore
$\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow 2^{Q \times \Gamma \times \Gamma}$ pour la **relation** de transition

Applications

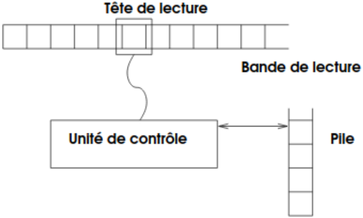
Exemple 1 : Cas de $\mathbf{a^n b^n}$

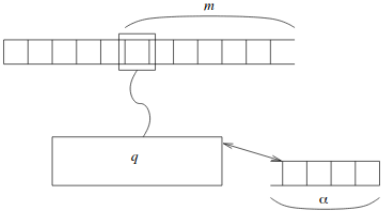
	a	**b**	ϵ	**accept**
S_0	$(1, \epsilon, u)$			X
1	$(1, \epsilon, u)$	$(2, u, \epsilon)$		
2		$(2, u, \epsilon)$		X

Exemple 2 : Cas du palindrome

	a	**b**	ϵ	**accept**
S_0	(S_0, ϵ, P) $(1, \epsilon, \epsilon)$	(S_0, ϵ, Q) $(1, \epsilon, \epsilon)$	$(1, \epsilon, \epsilon)$ $(1, \epsilon, \epsilon)$	
1	$(1, \epsilon, u)$	$(2, u, \epsilon)$		
2		$(2, u, \epsilon)$		

Autres variantes





Définition

Configuration = triplet $(q, m, \alpha) \in Q \times \Sigma^* \times \Gamma^*$, où :

- q est l'état courant de l'unité de contrôle
- m représente la partie du mot à reconnaître non encore lue. Le symbole le plus à gauche de m est le caractère sous la tête de lecture.
- α représente le contenu de la pile. Le symbole le plus à gauche de α est le sommet de la pile.

Changement de configuration

Configuration = triplet $(q, m, \alpha) \in Q \times \Sigma^* \times \Gamma^*$, où :

- Comportement : Lors du mouvement de transition on note

$$(q, aw, Z\alpha) \vdash (q', w, \gamma\alpha) \text{si } (q', \gamma) \in \delta(q, a, Z)$$

  - l'unité de contrôle passe de q à q'
  - le symbole *a* a été lu
  - la tête de lecture s'est déplacée d'une case vers la droite
  - le symbole Z a été dépilé et le mot γ empilé.
- Pour un automate à pile A, une chaîne $w \in \Sigma^*$ est reconnue par état acceptant et pile vide s'il y a une suite de transitions partant de l'état initial avec la pile vide, dont la suite des étiquettes est w , et aboutissant à un état acceptant où la pile est vide.

Configuration & Mode de reconnaissance

Mode de reconnaissance d'un mot dans un PDA

Un mot reconnu dans un PDA peut l'être de 2 façons :

- Par état final : Une chaîne testée est acceptée si, à partir de l'état initial, elle peut être entièrement lue en arrivant à un état de F, ceci quel que soit le contenu de la pile à ce moment-là. La pile est supposée vide au départ, mais on peut aussi convenir d'un contenu initial qui sera imposé.
- Par pile vide : Mode de reconnaissance autorisant un automate à pile à accepter une chaîne si, à partir de l'état initial, elle peut être entièrement lue en vidant la pile. Il n'y a donc pas lieu de préciser F, il est sous-entendu que $F = Q$ (tous les états sont acceptants).

Configuration & Mode de reconnaissance

Il existe des variantes qui ne portent pas sur la forme des transitions, mais sur l'état dans lequel doit être la pile pour qu'une chaîne soit acceptée.

Remarque

- Pour un PDA P, 2 langages (à priori différents) sont définis :
  1. N(P) (acceptation par pile vide)

     $$N(P) = \{w \mid w \in \Sigma^* \ \wedge \ \exists q \in Q, (q_0, w, Z_0) \vdash_P^* (q, \epsilon, \epsilon)\}$$

  2. L(P) (acceptation par état final)

     $$L(P) = \{w \mid w \in \Sigma^* \ \wedge \ \exists q \in F, \gamma \in \Gamma^* \ (q_0, w, Z_0) \vdash_P^* (q, \epsilon, \gamma)\}$$

- N(P) n'utilise pas F et n'est donc pas modifié si l'on définit $F = \emptyset$

Configuration

Configuration et langage accepté

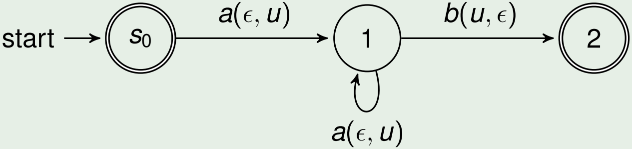
Configuration = triplet $(q, m, \alpha) \in Q \times \Sigma^* \times \Gamma^*$

- Configuration initiale : (q_0, w, Z_0) où w est la chaîne à accepter
- Configuration finale avec acceptation par pile vide : (q, ϵ, ϵ)
- Configuration finale avec acceptation par état final : (q, ϵ, γ) avec $q \in F$ ($\gamma \in \Gamma^*$ quelconque)

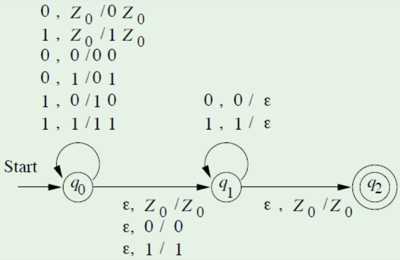
- Si $\gamma = \epsilon$, la pile a été dépilée (sauf si $Z = \epsilon$)
- $a = \epsilon$, le changement d'état et la modification de la pile se font sans mouvement de la tête.
- $Z = \epsilon$, il s'agit d'une transition permise quel que soit le symbole sur la pile.
- $\gamma = \epsilon$ et $Z = \epsilon$, alors la pile est inchangée.
- Donc la transition $(q, \epsilon, \epsilon) \vdash (q, \epsilon)$ est un changement d'état sans autre modification (, $\Leftrightarrow$ ϵ-transition dans AFD)

Configuration

Configuration et langage accepté



PDA pour $L_{ww^r} = \{ww^R \mid w \in \{0,1\}^*\})$ et exécution sur 1111



$$
\begin{tikzcd}
(q_0, 1111, Z_0) \arrow[d] \arrow[dr] & & \\
(q_0, 111, 1Z_0) \arrow[d] \arrow[dr] & (q_1, 1111, Z_0) \arrow[r] & (q_2, 1111, Z_0) \\
(q_0, 11, 11Z_0) \arrow[d] \arrow[dr] & (q_1, 111, 1Z_0) \arrow[r] & (q_1, 11, Z_0) \arrow[d] \\
(q_0, 1, 111Z_0) \arrow[d] \arrow[dr] & (q_1, 11, 11Z_0) \arrow[dr] & (q_2, 11, Z_0) \\
(q_0, \varepsilon, 1111Z_0) \arrow[d] & (q_1, 1, 111Z_0) \arrow[d] & (q_1, 1, 1Z_0) \arrow[d] \\
(q_1, \varepsilon, 1111Z_0) & (q_1, \varepsilon, 11Z_0) & (q_1, \varepsilon, Z_0) \arrow[d] \\
 & & (q_2, \varepsilon, Z_0)
\end{tikzcd}
$$

Résultats

Exemple pour L(P) et N(P)

- Dans la slide 7, on a :

$$L(P) = \{ww^R \mid w \in \{0,1\}^*\}) \text{ et } N(P) = \emptyset$$

- En revanche, l'exemple de la slide 8 aboutit à

$$L(P) = \emptyset \text{ et } N(P) = \{ww^R \mid w \in \{0,1\}^*\})$$

Théorème

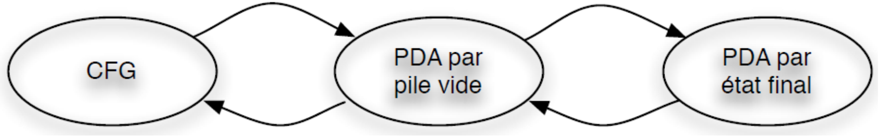
La classe des langages définis par un PDA déterministe est strictement incluse dans la classe des langages définie par un PDA (général)

Preuve ? ? ?

Pour la démonstration, on pourra alors montrer que le langage L_{ww^r} ayant servi comme exemple à la slide 7 ne peut être défini par un PDA déterministe

Équivalence entre PDA et CFG

On ambitionne de montrer les inclusions suivantes (chaque flèche montrant une inclusion).



Ce qui prouvera que :

Théorème

Les trois classes de langages suivants sont équivalentes :

- Les langages définis par une CFG (càd les CFL)
- Les langages définis par un PDA avec acceptation par pile vide
- Les langages définis par un PDA avec acceptation par état final

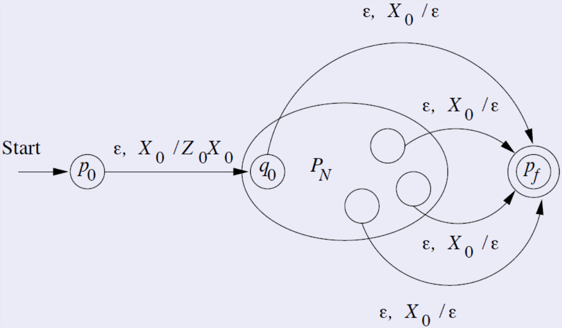
Équivalence entre état final et pile vide (1/2)

Théorème

Pour tout PDA $P_N = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, \emptyset)$, il existe un PDA P_F tel que $L(P_F) = N(P_N)$

Démonstration

La preuve est faite par construction directe d'un tel automate, comme suit :



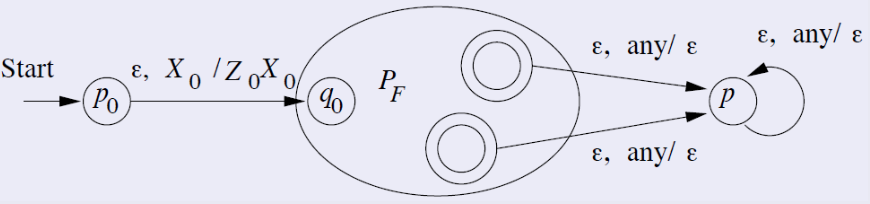
Équivalence entre état final et pile vide (2/2)

Théorème

Pour tout PDA $P_F = (Q, \Sigma, \Gamma, \delta_N, q_0, Z_0, F)$, il existe un PDA P_N tel que $N(P_N) = L(P_F)$

Démonstration

La preuve est aussi faite par construction directe d'un tel automate, comme suit :



Équivalence entre PDA et CFG

Théorème

Pour toute CFG G on peut définir un PDA M tel que $L(G) = N(M)$

Démonstration

Ayant $G = (V, T, P, S)$ une CFG, on construit un PDA M à un seul état qui simule les dérivations gauches de G comme suit : $P = (\{q\}, T, V \cup T, \delta, q, S, \emptyset)$ sachant que $\forall\, A \to X_1X_2 \ldots X_k \in P, (q, X_1X_2 \ldots X_k) \in \delta(q, \epsilon, A)$ et $\forall a \in T, \delta(q, \epsilon, a) = \{(q, \epsilon)\}$

- Initialement, le symbole de départ S est sur la pile
- Toute variable A au sommet de la pile avec $A \to X_1X_2 \ldots X_k \in P$ peut être remplacée par sa partie droite $X_1X_2 \ldots X_k$ avec X_1 au sommet de la pile
- Tout terminal au sommet de la pile qui est égal au prochain symbole d'input est matché avec l'input (on lit l'input et on pop le symbole)
- À la fin, si la pile est vide, le string est accepté

Théorème

Pour tout PDA P on peut définir une CFG G avec tel que $L(G) = N(P)$