Automates à pile 1

Mirabelle Nebut

Bureau 223 - extension M3
mirabelle.nebut at lifl.fr

2009-2010

But de ce cours

On sait maintenant écrire une grammaire algébrique... mais pas l'analyseur syntaxique qui va avec !

langage	spécification	modèle exécutable
régulier	expression régulière	AFD
algébrique	grammaire algébrique	automate à pile

Dans un premier temps : découverte des automates à pile.

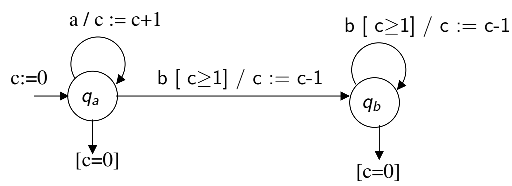
Plus tard : comment on s'en sert pour l'analyse syntaxique.

Plan du cours

Reconnaître un langage algébrique, intuition - 1

Pour reconnaître $\{a^n b^n \mid n \geq 0\}$:

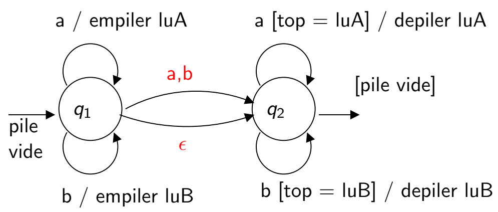
- un automate à nombre fini d'états pour lire des a puis des b ;
- un compteur c pour compter les a et décompter les b ;
- arrêt quand le ruban est vide et état final **et** c vaut 0 .



Reconnaître un langage algébrique, intuition - 2

Pour reconnaître $\{m \in \Sigma^* \mid m \text{ est un palindrome }\}$:

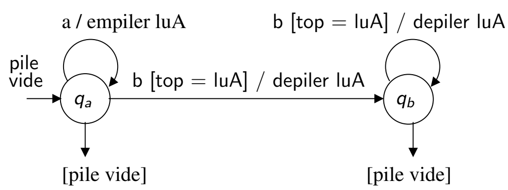
- un compteur ne suffit pas !
- il faut mémoriser les symboles lus puis les consulter ;
- mémorisation par empilement, vérification par dépilement.



Reconnaître un langage algébrique, intuition - 3

Ça marche aussi pour reconnaître $\{a^n b^n | n \geq 0\}$:

- on empile *luA* quand on lit un a ;
- on dépile *luA* quand on lit un b ;
- arrêt quand le ruban est vide et état final **et** la pile est vide.



Et ça marche pour tous les langages algébriques !

Automate à pile, intuition

un automate à nombre fini d'états classique
+
une pile non bornée

Et voilà notre mémoire non bornée !

Automate à pile, intuition

Automate à nombre fini d'états — ex des palindromes :

- ensemble d'état Q ; — ex : $\{q_1, q_2\}$
- état initial q_0 ; — ex : q_1
- ensemble d'états finaux $F \subseteq Q$; — ex : $\{q_2\}$
- alphabet d'entrée Σ. — ex : {a,b}

Pile

- contient des éléments de l'alphabet de pile Z — ex : {luA, luB}

Relation de transition ?

Transitions, intuition - 1

Pour un AF, une transition c'est :

- quand je suis dans l'état $q \in Q$
- et que j'ai $a \in \Sigma$ sous la tête de lecture
- ou que je transite sur ϵ
- alors je passe dans l'état $q' \in Q$

$$q, a \rightarrow q'$$
$$q, \epsilon \rightarrow q'$$

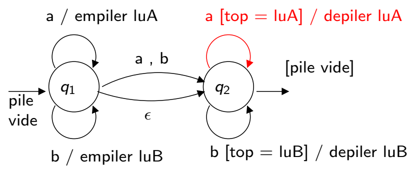
Transitions, intuition - 2

Pour un automate à pile, une transition c'est :

- quand je suis dans l'état $q \in Q$
- et que j'ai $a \in \Sigma$ sous la tête de lecture
- ou que je transite sur ϵ
- et que le sommet de pile est $z \in Z$
- je passe dans l'état $q' \in Q$
- et je modifie le sommet de pile en le remplaçant par des éléments de Z ou ϵ.

$$\begin{array}{rcl} q, a, z & \rightarrow & q', z_1 z_2 \\ q, \epsilon, z & \rightarrow & q', z \\ q, a, z & \rightarrow & q', \epsilon \end{array}$$

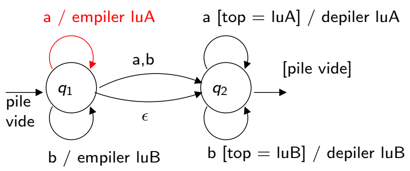
Exemple des palindromes - 1



- dans l'état $q_2 \in Q$;
- avec $a \in \Sigma$ sous la tête de lecture (Σ-transition) ;
- et avec $luA \in Z$ en sommet de pile ;
- alors on reste dans l'état $q_2 \in Q$;
- et on dépile : on remplace luA par ϵ.

$$q_2, a, luA \rightarrow q_2, \epsilon$$

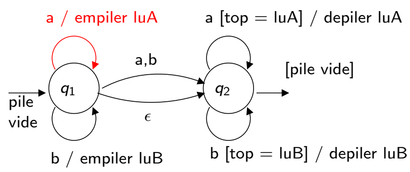
Exemple des palindromes - 2



- dans l'état $q_1 \in Q$ et avec a sous la tête de lecture ;
- et quel que soit le sommet de pile... quel peut-il être ?
  - si je viens de lire un a (resp. b) : luA (resp. luB) ;
  - si je n'ai encore rien lu : pile initiale (vide)

Pas de transition sur pile vide : symbole initial de pile $z_\perp \in Z$

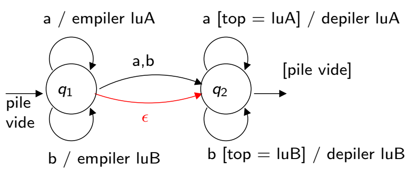
Exemple des palindromes - 3



- dans l'état $q_1 \in Q$, avec a sous la tête de lecture ;
- et avec luA, luB ou $z_\perp$ en sommet de pile ;
- alors on reste dans $q_1 \in Q$;
- et on empile luA : on remplace le sommet x par x luA

$$q_1, a, luA \rightarrow q_1, luA\ luA \qquad q_1, a, z_\perp \rightarrow q_1, z_\perp\ luA$$
$$q_1, a, luB \rightarrow q_1, luB\ luA \qquad \text{(lecture bas vers haut de pile)}$$

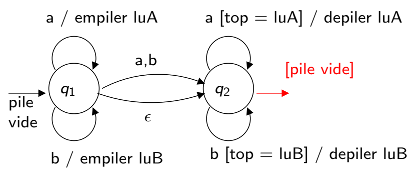
Exemple des palindromes - 4



- dans l'état $q_1 \in Q$;
- sans toucher la tête de lecture (ϵ-transition) ;
- et avec luA, luB ou $z_\perp$ en sommet de pile ;
- alors on passe dans $q_2 \in Q$ et on ne touche pas à la pile.

$$q_1, \epsilon, luA \rightarrow q_2, luA \qquad q_1, \epsilon, z_\perp \rightarrow q_2, z_\perp$$
$$q_1, \epsilon, luB \rightarrow q_2, luB$$

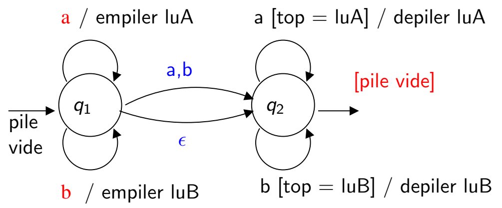
Exemple des palindromes - 5



Pour terminer on vide la pile (ϵ-transition) :

$$q_2, \epsilon, z_\perp \rightarrow q_2, \epsilon$$

Exemple des palindromes, récapitulatif



$q_1, a, luA \to q_1, luA\ luA$	$q_1, a, z_\perp \to q_1, z_\perp\ luA$	$q_1, a, luB \to q_1, luB\ luA$
$q_1, b, luA \to q_1, luA\ luB$	$q_1, b, z_\perp \to q_1, z_\perp\ luB$	$q_1, b, luB \to q_1, luB\ luB$
$q_1, a, luA \to q_2, luA$	$q_1, a, z_\perp \to q_2, z_\perp$	$q_1, a, luB \to q_2, luB$
$q_1, b, luA \to q_2, luA$	$q_1, b, z_\perp \to q_2, z_\perp$	$q_1, b, luB \to q_2, luB$
$q_1, \epsilon, luA \to q_2, luA$	$q_1, \epsilon, z_\perp \to q_2, z_\perp$	$q_1, \epsilon, luB \to q_2, luB$
$q_2, a, luA \to q_2, \epsilon$	$q_2, b, luB \to q_2, \epsilon$	$q_2, \epsilon, z_\perp \to q_2, \epsilon$

Définition formelle

Definition (Automate à pile (AP))

Un automate à pile A est un tuple $(\Sigma, Z, z_\perp, Q, q_0, F, \Delta)$ où :

- Σ est un alphabet d'entrée fini (les terminaux) ;
- Z est un alphabet de pile fini ;
- $z_\perp \in Z$ est le symbole initial de pile,
- Q est un ensemble fini d'états,
- $q_0 \in Q$ est l'état initial ;
- $F \subseteq Q$ est l'ensemble des états finaux ;
- $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Z \times Q \times Z^*$ est la relation de transition.

NB : on pourrait choisir $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Z^* \times Q \times Z^*$.

Exécution et configurations

Une exécution est une suite de configurations.

Pour un AF, une configuration est :

- mot restant à lire $m \in \Sigma^*$;
- état courant $q \in Q$;

$$(abbb, q)$$

Pour un AP, configuration définie par :

- le mot restant à lire $m \in \Sigma^*$;
- l'état courant $q \in Q$;
- le contenu de la pile de Z^*, lu du bas vers le haut de la pile.

Exécution et configurations

Ex : $(abbb, q_1, z_\perp luA\ luA)$ pour la pile $\begin{array}{|c|}\hline luA \\ luA \\ z_\perp \\ \hline\end{array}$

Definition (configuration)

Une configuration c d'un AP $(\Sigma, Z, z_\perp, Q, q_0, F, \Delta)$ est un élément de $\Sigma^* \times Q \times Z^*$.

Transiter d'une configuration à une autre

Le passage dans A d'une configuration c_1 à une configuration c_2 s'écrit :

$$c_1 \vdash_A c_2$$

On note $\vdash_A^*$ la clôture réflexive et transitive de $\vdash_A$.

Deux modes de transition pour changer de configuration :

- sur une Σ-transition ;
- sur une ϵ-transition.

Changement de configuration sur Σ-transition : exemple

Transition $q_1, b, luA \to q_1, luA\ luB$

Configuration $(bba,\ q_1,\ z_\perp luA)$

On aura alors :

$$(bba,\ q_1,\ z_\perp luA) \vdash_A (ba,\ q_1,\ z_\perp luA\ luB)$$

Changement de configuration sur Σ-transition

Definition ($c_1 \vdash_A c_2$ sur Σ-transition)

A passe d'une config $c_1 = (m_1, q_1, \alpha_1)$ à $c_2 = (m_2, q_2, \alpha_2)$ si :

- il existe une transition $(q_1, x, z) \to (q_2, \beta_2) \in \Delta$;
- m_1 est de la forme xm_2 ;
- α_1 est de la forme $\beta_1 z$;
- α_2 est de la forme $\beta_1\beta_2$.

$$\left(xm_2 \ ,\ q_1 \ ,\ \overset{\alpha_1}{\begin{array}{|c|}\hline z \\ \hline \beta_1 \\ \hline\end{array}} \right) \vdash_A \left(m_2 \ ,\ q_2 \ ,\ \overset{\alpha_2}{\begin{array}{|c|}\hline \beta_2 \\ \hline \beta_1 \\ \hline\end{array}} \right)$$

Changement de configuration sur Σ-transition - exemple

Transition $q_1, b, luA \to q_1, luA\ luB$

Configuration $(bba,\ q_1,\ z_\perp luA)$

$$(\overbrace{b\ \underbrace{ba}_{m_2}}^{m_1},\ q_1,\ \underbrace{\overbrace{z_\perp}^{\beta_1} luA}_{\alpha_1}) \vdash (\underbrace{ba}_{m_2},\ q_1,\ \underbrace{\overbrace{z_\perp}^{\beta_1} \overbrace{luA\ luB}^{\beta_2}}_{\alpha_2})$$

$$\left(xm_2 \ ,\ q_1 \ ,\ \overset{\alpha_1}{\begin{array}{|c|}\hline z \\ \hline \beta_1 \\ \hline\end{array}} \right) \vdash_A \left(m_2 \ ,\ q_2 \ ,\ \overset{\alpha_2}{\begin{array}{|c|}\hline \beta_2 \\ \hline \beta_1 \\ \hline\end{array}} \right)$$

Changement de configuration sur ϵ-transition - exemple

Transition $q_1,\ \epsilon,\ luB \to q_2,\ luB$

Configuration $(ba,\ q_1,\ z_\perp luA\ luB)$

On aura alors :

$$(ba,\ q_1,\ z_\perp luA\ luB) \vdash (ba,\ q_2,\ z_\perp luA\ luB)$$

On ne touche pas à la tête de lecture.

Changement de configuration sur ϵ-transition

Definition ($c_1 \vdash_A c_2$ sur ϵ-transition)

A passe d'une config $c_1 = (m, q_1, \alpha_1)$ à $c_2 = (m, q_2, \alpha_2)$ si :

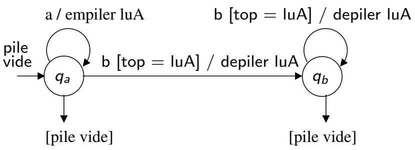
- il existe une transition $(q_1, \epsilon, z) \to (q_2, \beta_2) \in \Delta$;
- α_1 est de la forme $\beta_1 z$ (z sommet de pile) ;
- α_2 est de la forme $\beta_1\beta_2$.

$$\left(\ m\ ,\ q_1\ ,\ \overset{\alpha_1}{\begin{array}{|c|}\hline z \\ \hline \beta_1 \\ \hline \end{array}}\ \right) \vdash_A \left(\ m\ ,\ q_2\ ,\ \overset{\alpha_2}{\begin{array}{|c|}\hline \beta_2 \\ \hline \beta_1 \\ \hline \end{array}}\ \right)$$

Exécution - exemple

Pour le langage $\{a^n b^n \mid n \geq 0\}$:

$$\Delta = \left\{ \begin{array}{ll} q_a, a, z_\perp \to q_a, z_\perp luA & q_a, a, luA \to q_a, luA\ luA \\ q_a, b, luA \to q_b, \epsilon & q_a, \epsilon, z_\perp \to q_a, \epsilon \\ q_b, b, luA \to q_b, \epsilon & q_b, \epsilon, z_\perp \to q_b, \epsilon \end{array} \right\}$$



$(q_a, aabb, z_\perp) \vdash_A^* (q_b, \epsilon,\quad)$

$(q_a, \epsilon, z_\perp) \vdash_A^* (q_a, \epsilon,\quad)$

Critère d'acceptation

Dans nos exemples, on accepte un mot si ruban vide et état final et pile vide.

Ce sont des cas particuliers.

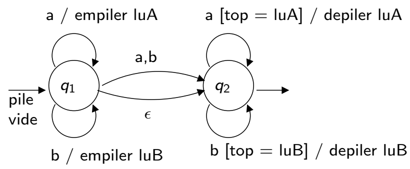
Il y a deux critères d'acceptation possibles :

- acceptation par état final (pour toute pile quand on s'arrête) ;
- acceptation par pile vide (pour tout état quand on s'arrête).

Ces deux critères sont équivalents.

Acceptation par état final - exemple

L'exemple des palindromes sans vider la pile en q_2 :



On supprime $q_2, \epsilon, z_\perp \to q_2, \epsilon$.

$(q_1, abba, z_\perp) \vdash^* (q_2, \epsilon, z_\perp)$: acceptation.

Acceptation par état final - définition

Definition (Acceptation par état final)

Un mot $m \in \Sigma^*$ est accepté par état final par un AP $A = (\Sigma, Z, z_\perp, Q, q_0, F, \Delta)$ si pour la configuration $(m, q_0, z_\perp)$, il existe un état $q_f \in F$ et un mot $\mathbf{z} \in Z^*$ tel que $(m, q_0, z_\perp) \vdash_A^* (\epsilon, q_f, \mathbf{z})$

Definition (Langage accepté)

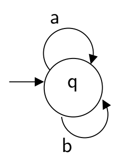
Le langage accepté par état final par un AP est l'ensemble des mots acceptés par cet automate.

$$L^F(A) = \{m \in \Sigma^* \mid (m, q_0, z_\perp) \vdash_A^* (\epsilon, q_f, \mathbf{z})\}$$

Acceptation par pile vide - exemple

Reconnaître $\{m \in \{a, b\}^* \mid \quad |m|_a = |m|_b\ \}$?

$(q, a, z_\perp) \to (q, z_\perp\ luA)$	`empiler luA`
$(q, b, z_\perp) \to (q, z_\perp\ luB)$	`empiler luB`
$(q, a, luA) \to (q, z_\perp\ luA\ luA)$	`empiler luA`
$(q, b, luB) \to (q, z_\perp\ luB\ luB)$	`empiler luB`
$(q, a, luB) \to (q, \epsilon)$	`a consomme luB`
$(q, b, luA) \to (q, \epsilon)$	`b consomme luA`
$(q, \epsilon, z_\perp) \to (q, \epsilon)$	`vider la pile`



Acceptation par pile vide - définition

Definition (Acceptation par pile vide)

Un mot $m \in \Sigma^*$ est accepté par pile vide par un AP $A = (\Sigma, Z, z_\perp, Q, q_0, F, \Delta)$ si pour la configuration $(m, q_0, z_\perp)$, il existe un état $q \in Q$ tel que $(m, q_0, z_\perp) \vdash_A^* (\epsilon, q, \epsilon)$

Definition (Langage accepté)

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

$$L^V(A) = \{m \in \Sigma^* \mid (m, q_0, z_\perp) \vdash_A^* (\epsilon, q, \epsilon)\}$$

Automates à pile et langages algébriques

Theorem

L est un langage algébrique si et seulement s'il existe un AP A (acceptant par pile vide) tel que $L = L^V(A)$.

Automates déterministes

Les AP définis précédemment sont indéterministes (APND) :

- un mot est accepté s'il existe au moins une suite de configurations conduisant à l'acceptation ;
- mais il peut en avoir plusieurs ;
- et il peut y avoir des suites conduisant à l'échec ;

$\Rightarrow$ automate à pile déterministe ?

Definition (intuitive, APD)

Un AP (acceptant par état final) est déterministe (APD) si, dans chaque configuration, il n'y a qu'une seule transition possible.

Automates déterministes vs non déterministes

Les automates à pile non déterministes :

- sont strictement plus puissants que les APD ;
- ne sont pas tous déterminisables.

$$\text{APD} \not\Leftrightarrow \text{APND}$$

Différent des AF : AFD $\Leftrightarrow$ AFND.

Langage algébrique déterministe

Definition

Un langage algébrique L est déterministe s'il existe un AP A (acceptant par état final) déterministe tel que $L^F(A) = L$.

Exemple1 : $\{m \in (a+b)^* \mid m \text{ est un palindrome}\}$ est algébrique, non ambigu, mais. . . n'est pas déterministe.

Intuitivement, on ne sait pas deviner où est le milieu du mot.

Exemple1 : $\{m_1 c m_2 \mid m_1 m_2 \in \{a, b\}^* \text{ est un palindrome }\}$ est un langage algébrique déterministe.

Associer un AP à une grammaire algébrique

- Les AP sont nécessaires pour reconnaître les langages algébriques. . .
- . . . mais les AP ne sont pas si faciles à concevoir ;
- et on ne voit pas bien le lien entre dérivations d'une grammaire et exécution d'un AP.

Les langages algébriques sont spécifiés par des grammaires algébriques :

- dériver automatiquement un AP à partir d'une grammaire algébrique ?
- $\Rightarrow$ automate des items (malheureusement pas déterministe).

Automate des items, présentation

Automate à pile particulier :

- dérivé à partir d'une grammaire algébrique ;
- non déterministe donc inutilisable en pratique ;
- permet de définir les automates réellement utilisés par les analyseurs syntaxiques ;
  - analyse descendante par simplification ;
  - surtout analyse ascendante.
- repose sur la notion d'item.

Item, définition

Un item d'une grammaire G est de la forme :

$$[X \to \alpha \bullet \beta] \text{ avec } X \to \alpha\beta \in P \text{ et } \alpha, \beta \in (V_N \cup V_T)^*$$

Interprété comme :
"en cherchant à dériver de X un mot $m = uv \in V_T{}^*$, on a déjà dérivé un mot u de α, et il reste à dériver v de β"

L'ensemble des items de G est noté It_G.

Association des items aux productions de G

À $X \to \alpha$ on associe :

- $[X \to \bullet\alpha]$: « on cherche à reconnaître un mot pour X » ;
- $[X \to \alpha\bullet]$ (item terminal) : « on a reconnu un mot pour X »

À $X \to \alpha\beta$ on associe $[X \to \alpha \bullet \beta]$.

À $X \to \epsilon$ est associé par convention l'unique item $[X \to \bullet]$.

Ex : $S \to \epsilon \,|\, aSb$
$It_G = \{[S \to \bullet],\ [S \to \bullet aSb],\ [S \to a \bullet Sb],\ [S \to aS \bullet b],\ [S \to aSb\bullet]\}$

Extension de la grammaire

Ex : $S \to \epsilon \,|\, aSb$

Récursivité sur l'axiome S.

$\Rightarrow$ reconnaître un mot pour S ne signifie pas forcément avoir fini !

$\Rightarrow$ extension de la grammaire avec un nouvel axiome S' :

$$S' \to S,\ S \to aSb \,|\, \epsilon$$

$It_G = \{[S' \to \bullet S],\ [S' \to S\bullet],\ [S \to \bullet],\ [S \to \bullet aSb],\ [S \to a \bullet Sb],\ [S \to aS \bullet b],\ [S \to aSb\bullet]\}$

Automate des items, intuition

Tout est item !

- les états de l'automate sont les items de It_G ;
- l'alphabet de pile est aussi l'ensemble It_G
  $\Rightarrow$ la pile sert à mémoriser des séquences d'états ;
- l'état courant de l'automate est l'item de sommet de pile.

Pour le moment on n'explicitera pas l'automate à nombre fini d'états sous-jacent.

États particuliers :

- $[S' \to \bullet S]$: item initial = état initial, sert de $z_\perp$;
- $[S' \to S\bullet]$: item final = unique état final (arrêt par état final).

Exemple

$G = (V_T, V_N, S, P)$ avec $V_T = \{a, b, d\}$, $V_N = \{S, D\}$ et :

$S' \to S$
$S \to aSbD \ | \ b$
$D \to d \ | \ \epsilon$

Reconnaître $abbd$?
Reconnaître abb ?
Lien avec dérivation / arbre syntaxique ?

Configurations et transitions de l'automate - 1

État courant indiqué par sommet de pile :

$\Rightarrow$ Configurations simplifiées du type $(m \in \Sigma^*, z_1 \dots z_n \in It^*)$

Trois types de transitions :

- Lecture (Σ-transition) : lire a et avancer la tête de lecture ;
- Expansion (ϵ-transition) par une production $X \to \alpha$: tenter de reconnaître X en reconnaissant α ;
- Réduction (ϵ-transition) par une production $X \to \alpha$: indiquer qu'on a reconnu X en reconnaissant α.

Configurations et transitions de l'automate - 2

On simplifie / modifie un peu la forme de la relation de transition :

- AP classique : $\Delta \subseteq Q \times (\Sigma \cup \{\epsilon\}) \times Z \times Q \times Z^*$;
- aut des items : $\Delta \subseteq (\Sigma \cup \{\epsilon\}) \times (Z \times Z) \times Z^*$

Trois types de transitions :

Lecture Σ-transition :

$$(a, [\cdots \bullet \dots]) \to [\cdots \bullet \dots] \dots [\cdots \bullet \dots]$$

Expansion ϵ-transition :

$$(\epsilon, [\cdots \bullet \dots]) \to [\cdots \bullet \dots] \dots [\cdots \bullet \dots]$$

Réduction ϵ-transition :

$$(\epsilon, [\cdots \bullet \dots][\cdots \bullet \dots]) \to [\cdots \bullet \dots] \dots [\cdots \bullet \dots]$$

qui fait intervenir le sous-sommet de pile.

Transition de lecture, définition

- si le sommet de pile est $[X \to \alpha \bullet a\beta]$;
- et que a est sous la tête de lecture ;
- alors remplacer le sommet de pile par $[X \to \alpha a \bullet \beta]$;
- et avancer la tête de lecture.

$$(a , [X \to \alpha \bullet a\beta]) \to [X \to \alpha a \bullet \beta]$$

$$(am , z_1 \ldots z_n [X \to \alpha \bullet a\beta]) \vdash (m , z_1 \ldots z_n [X \to \alpha a \bullet \beta])$$

Transition d'expansion par $Y \to \gamma$, définition

- si le sommet de pile est $[X \to \alpha \bullet Y\beta]$;
- et que la production $Y \to \gamma$ appartient à la grammaire ;
- alors remplacer le sommet de pile par $[X \to \alpha \bullet Y\beta]$ $[Y \to \bullet\gamma]$

$$(\epsilon , [X \to \alpha \bullet Y\beta]) \to [X \to \alpha \bullet Y\beta] [Y \to \bullet\gamma]$$

$$(m , z_1 \ldots z_n [X \to \alpha \bullet Y\beta]) \vdash (m , z_1 \ldots z_n [X \to \alpha \bullet Y\beta] [Y \to \bullet\gamma])$$

Transition d'expansion par $Y \to \gamma$, intuition

Remplacer le sommet de pile $[X \to \alpha \bullet Y\beta]$ par
$[X \to \alpha \bullet Y\beta]$ $[Y \to \bullet\gamma]$ pour une production $Y \to \beta$

Intuitivement :

- pour reconnaître un mot pour X il faut reconnaître un mot pour Y ;
- on a le choix (non-déterministe) entre toutes les productions de la forme $Y \to \gamma$;
- on garde en mémoire dans la pile qu'on est en train de reconnaître un mot pour X.

Transition de réduction par $Y \to \gamma$, définition

- si la partie supérieure de pile est $[X \to \alpha \bullet Y\beta]$ $[Y \to \gamma\bullet]$;
- alors la remplacer par $[X \to \alpha Y \bullet \beta]$.

$$(\epsilon , [X \to \alpha \bullet Y\beta] [Y \to \gamma\bullet]) \to [X \to \alpha Y \bullet \beta]$$

$$(m , z_1 \ldots z_n [X \to \alpha \bullet Y\beta] [Y \to \gamma\bullet]) \vdash (m , z_1 \ldots z_n [X \to \alpha Y \bullet \beta])$$

Transition de réduction par $Y \to \gamma$, intuition

Remplacer la partie supérieure de pile $[X \to \alpha \bullet Y\beta]$ $[Y \to \gamma\bullet]$
par $[X \to \alpha Y \bullet \beta]$.

Intuitivement :

- si pour reconnaître un mot pour X il fallait reconnaître un mot pour Y ;
- et qu'on a effectivement reconnu un mot pour Y ;
- alors on passe Y ;
- et on continue la reconnaissance de X.

Lien avec $L(G)$

Theorem
Le langage reconnu par l'automate des items de G est $L(G)$.

Qu'est-ce que l'analyse syntaxique ?

Étant donnés une grammaire G (ou par ex. son automate des items) et un mot $m \in {V_T}^*$:

- a-t-on $m \in L(G)$?
- si oui : arbre syntaxique pour m / dérivation pour m ?

Comment faire ?

Approche (très) naïve - 1

On essaie toutes les dérivations possibles partant de l'axiome, en essayant de tomber sur m.

Problème :

- quand s'arrête-t-on ?
- le langage engendré peut être infini ;
- et si ça boucle ? Allongement inutile des dérivations.

Ex (td) : $A \longrightarrow aSb \,|\, bSa \,|\, AA \,|\, \epsilon$ et $A \Rightarrow AA \Rightarrow A \Rightarrow AA \Rightarrow \ldots$

Ex : $S \rightarrow X \,|\, a$, $X \rightarrow S$ et $S \Rightarrow X \Rightarrow S \Rightarrow \ldots$

Approche (très) naïve - 2

On peut éliminer les productions qui « causent les boucles » :

- $X \rightarrow \epsilon$
- $X \rightarrow Y$ $(Y \in V_N)$

On obtient alors une grammaire G' propre telle que :

$$L(G') = \begin{cases} L(G) \setminus \{\epsilon\} \text{ si } \epsilon \in L(G) \\ L(G) \text{ sinon} \end{cases}$$

Propriétés des grammaires propres = corrélation entre :

- longueur de m
- longueur de la dérivation pour m

$$\text{si } S \Rightarrow^k m \text{ alors } k \leq |m|$$

Approche (très) naïve - 3

Pour le mot m et la grammaire G : $m \in G$?

Si $m = \epsilon$, on regarde si $S \Rightarrow^* \epsilon$ (facile).
Si $m \neq \epsilon$:

- on transforme G en G' réduite et propre ;
- on essaye toutes les dérivations possibles de taille $\leq |m|$.

Complexité : $O(n^{|m|})$ avec n le nombre de règles de production.

Bref, c'est naïf et pas efficace du tout ! Il faut chercher plus futé.

En pratique

On supprime tout indéterminisme.

Il faut savoir à tout instant quelle expansion effectuer.

Plus de tentatives inutiles :

- construction d'une unique dérivation ;
- si la dérivation échoue, le mot est rejeté, il est accepté sinon.

2 approches :

- analyses universelles ;
- analyses dédiées à certaines classes de grammaires.

Analyses « universelles »

Fonctionnent pour toute grammaire algébrique :

- algorithme de Cocke-Younger-Kasami pour des grammaires en forme normale de Chomsky ;
- algorithme de Earley pour des grammaires algébriques quelconques.

Complexité en $o(n^3)$, où n est la longueur du mot à analyser.

On préférerait $o(n)$... quitte à se restreindre à certains types de grammaires.

Analyses dédiées à certaines grammaires

	Analyse descendante	Analyse ascendante
nom	LL(k)	LR(k)
dérivation construite	gauche	droite
ordre constr arbre	préfixe	postfixe
opérations	lecture et expansion	lecture et réduction
on part	de l'axiome	du mot à reconnaître
outil	javaCC	Cup

- plus efficaces ;
- cas particulier de l'automate des items ;
- ne fonctionnent que pour certaines classes de grammaires.