CYK et Earley

Méthodes générales de reconnaissance des grammaires non contextuelles

– Deux approches possibles

. Soit on utilise un automate déterministe, mais on l'éclate en plusieurs automates tournant en parallèle, chacun sur sa pile, en cas de conflit (méthode GLR par exemple)

. Soit une approche *interprétée* de l'analyse (on travaille directement sur la grammaire), par opposition à la production d'un automate (sorte de *compilation* de la grammaire)

. Dans les deux cas, on peut avoir plusieurs arbres de dérivation en cas d'ambiguïté

∗ partage des sous-arbres communs → *forêt partagée*

∗ plutôt que de construire la forêt, on peut produit une grammaire qui la décrit

Approches interprétées

– Deux méthodes largement employées

  . CYK, découverte indépendamment par Cocke, Kasami, Younger (voir D. H. Younger, Recognition and parsing of context-free languages in time n^3, Information and Control **10**(2), 1967)

  . Earley (J. Earley, An efficient context-free parsing algorithm. CACM **13**(2), 1970)

  . Bien qu'elles semblent assez différentes, en fait elles sont au fond assez proches (S. L. Graham, M. A. Harrison & W. L. Ruzzo, An Improved Context-Free Recognizer, ACM Transactions on Programming Languages and Systems –TOPLAS– **2**(3), 1980)

– Permettent toutes deux une analyse en temps $O(n^3)$ pour une phrase de longueur n,

  . optimisations permettant de descendre à $O(n^{3-\cdots})$ pour CYK

  . au pire $O(n^3)$ pour Earley

Détour par la méthode de Unger

- La grammaire ne doit pas avoir de production vide, ni produire de cycle (c'est à dire avoir des règles permettant $A \Rightarrow^+ A$)
- Pour une entrée w, chercher un découpage de w en $w_1 w_2 \cdots w_k$ tel qu'il existe (au moins) une règle $S \rightarrow A_1 A_2 \cdots A_k$ et $A_i \Rightarrow^+ w_i$
- Donc, chercher, pour $1 \leq i \leq k$, un découpage de $w_{i_1} \cdots w_{i_l}$ de w_i par une (ou plus) production de A_i, etc.
- Pour $\mathcal{G} = \{S \rightarrow ABC,\ A \rightarrow a|ab,\ B \rightarrow bc|c,\ C \rightarrow d\}$ et le mot $abcd$

S		
A	B	C
a	b	cd

KO

S		
A	B	C
a	bc	d

OK

S		
A	B	C
ab	c	d

OK

- Comme il faut, à la base, essayer (presque) tous les découpages possibles, méthode exponentielle

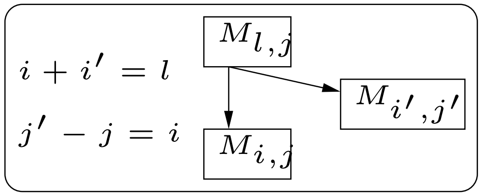
Principes de CYK

- La grammaire doit être en forme normale de Chomsky (CNF), c'est à dire n'avoir que des productions de la forme $A \rightarrow a$ ou $A \rightarrow BC$ (elle est donc sans productions vides, et elle ne produit pas de cycles)
- On construit les découpages possibles "en remontant", c'est à dire en calculant les découpages de longueur 1, 2...n si n est la longueur de l'entrée $w = a_1 \cdots a_n$,
- Ces découpages sont calculés dans une matrice triangulaire $n \times n$ dans laquelle l'élement M_{ij} est l'ensemble des non-terminaux produisant $a_j \cdots a_{j+i-1}$
- L'entrée est reconnue si $M_{n,1}$ contient l'axiome S
- Cette matrice est appelée *table de reconnaissance*

Exemple de calcul avec CYK

– La grammaire BNF $S{\rightarrow}ABC,\ A{\rightarrow}a|ab,\ B{\rightarrow}c|bc\ C{\rightarrow}d$ en CNF $S{\rightarrow}AD,\ D{\rightarrow}BC,\ A{\rightarrow}$

– la matrice calculée pour *abcd* :



4				
3				
2				
1	{AG}	{EH}	{BF}	{C}
	a	b	c	d

⇒

4				
3				
2	{A}	{B}	{D}	
1	{AG}	{EH}	{BF}	{C}
	a	b	c	d

⇒

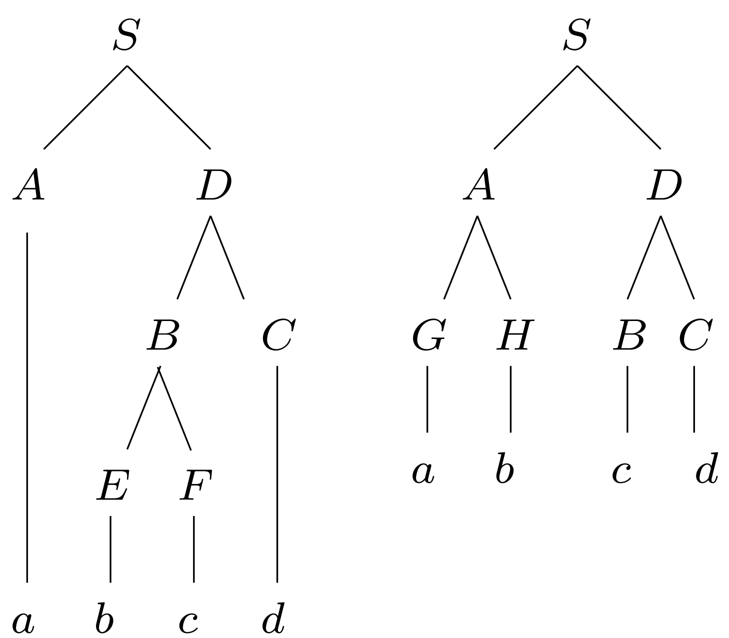
4				
3		{D}		
2	{A}	{B}	{D}	
1	{AG}	{EH}	{BF}	{C}
	a	b	c	d

⇒

4	{S}			
3		{D}		
2	{A}	{B}	{D}	
1	{AG}	{EH}	{BF}	{C}
	a	b	c	d

Arbres pour *abcd* et grammaire de la forêt

Les arbres



La grammaire de la forêt

$S_{1,4} \rightarrow A_{1,1} D_{2,4}$ $\quad$ $S_{1,4} \rightarrow A_{1,2} D_{3,4}$

$D_{2,4} \rightarrow B_{2,3} C_{4,4}$ $\quad$ $D_{3,4} \rightarrow B_{3,3} C_{4,4}$

$B_{2,3} \rightarrow E_{2,2} F_{3,3}$ $\quad$ $A_{1,2} \rightarrow G_{1,1} H_{2,2}$

$A_{1,1} \rightarrow a$ $\quad$ $G_{1,1} \rightarrow a$

$E_{2,2} \rightarrow b$ $\quad$ $H_{2,2} \rightarrow b$

$F_{3,3} \rightarrow c$ $\quad$ $B_{3,3} \rightarrow c$

$C_{4,4} \rightarrow d$

Un non-terminal $A_{i,j}$ peut donc avoir plusieurs productions, et peut apparaître dans plusieurs parties droites

Algorithme CYK sur une grammaire en CNF

données : grammaire $G = (V, N, P, S)$ et phrase $a_1 \cdots a_n$
M est une matrice triangulaire $n \times n$ d'ensemble de non-terminaux

pour $j := 1$ **à** n **faire** $M_{1,j} := \{A \mid A \rightarrow a_j \in P\}$ **fin pour**
pour $l := 2$ **à** n **faire**
  pour $j := 1$ **à** $n - l + 1$ **faire**
    $M_{l,j} := \emptyset;$
    pour $i := 1$ **à** $l - 1$ **faire**
      $i' := l - 1;\ j' := j + i$
      $M_{l,j} := M_{l,j} \cup \{A \mid A \rightarrow B\,C \in P, B \in M_{i,j}, C \in M_{i',j'}\}$
    fin pour
  fin pour
fin pour
si $S \notin M_{1,n}$ **alors** erreur

Algorithme CYK sur toute grammaire

– L'algorithme précédent peut être adapté pour fonctionner sur une grammaire qui n'est pas en CNF :
  On met A dans $M_{l,j}$ si $A \rightarrow X_1 \cdots X_k$ et s'il existe $M_{i_1,j_1} \cdots M_{i_k,j_k}$ tels que

  $$i_1 + i_2 + \cdots + i_k = l$$

  $$j_1 = j \text{ et } j_p = j_{p-1} + i_{p-1} \text{ pour } 2 \leq p \leq k$$

  $$X_p \in M_{i_p,j_p} \text{ pour } 1 \leq p \leq k$$

– Ce qui donne un algorithme plus compliqué, et pose problème
  . avec les ε productions : il faut considérer que tout X_p tel que $X_p \Rightarrow^+ \varepsilon$ est virtuellement présent dans la matrice
  . avec les cycles, qui demandent quelques précautions de traitement

– Pour ces raisons, peu utilisé

Calcul d'un arbre de dérivation avec CYK - I

– La matrice contient implicitement les partitions valides de la méthode de Unger

– Si l'entrée est a, alors il existe nécessairement $S \rightarrow a$: c'est l'arbre

– Si l'entrée est $a_1 \cdots a_i \, a_{i+1} \cdots a_n, 1 \leq i < n$, alors il existe nécessairement $S \rightarrow AB, A \Rightarrow^+ a_1 \cdots a_i$ et $B \Rightarrow^+ a_{i+1} \cdots a_n$

– Les mêmes remarques s'appliquent récursivement à A et à B

– La fonction $\Pi(A, j, l), A \in M_{l,j}$, calcule *toutes* les dérivations possibles de $A \Rightarrow^* a_j ... a_{j+l-1}$

– $\Pi(S, 1, n)$ donne tous les arbres compatibles avec la grammaire et la phrase analysée

Calcul d'un arbre de dérivation avec CYK - II

– La fonction Π

$\Pi(A, n, j, l)$
  si $l = 1$ **alors**
    ajouter $A_n \rightarrow a_j$ aux règles si elle n'y est pas
  sinon
    pour tout B, C, i $(1 \leq i < l)$
      tels que $A \rightarrow B\,C$, $B \in M_{i,j}$, $C \in M_{l-i,j+i}$,
    $\Pi(B, j, i)$
    $\Pi(C, j+i, l-i)$
    ajouter $A_{j,j+l-1} \rightarrow B_{j,j+i-1} C_{j+i,j+l-1}$ aux règles si elle n'y est pas

– On pourrait aussi mettre directement les règles de la grammaire dans la matrice

Exemple d'utilisation de CYK - I

– Soit une grammaire de nombres comme 1, 21.5, $1e+10$, $1.72e-2$

$S \rightarrow NDX$	$N \rightarrow C$	$N \rightarrow NC$	$D \rightarrow .N$	$D \rightarrow \varepsilon$
$X \rightarrow e+N$	$X \rightarrow e-N$	$X \rightarrow \varepsilon$		
$C \rightarrow 0$	$C \rightarrow 1$	$\cdots$	$\cdots$	$C \rightarrow 9$

– En forme CNF, on a la grammaire équivalente (dur dur !)

$C \rightarrow 0$	$\cdots$	$C \rightarrow 9$	$S \rightarrow 0$	$\cdots$	$S \rightarrow 9$
$V \rightarrow .$	$P \rightarrow +$	$M \rightarrow -$	$E \rightarrow e$		
$S \rightarrow CC$	$S \rightarrow NC$	$N \rightarrow CC$	$N \rightarrow NC$		
$S \rightarrow CD$	$S \rightarrow CX$	$S \rightarrow CY$	$S \rightarrow ND$	$S \rightarrow NX$	$S \rightarrow NY$
$Y \rightarrow DX$	$D \rightarrow VC$	$D \rightarrow VN$			
$X \rightarrow FC$	$X \rightarrow FN$	$F \rightarrow EP$	$F \rightarrow EM$		

Exemple d'utilisation de CYK - II

– l'entrée 1 donne $S_{1,1}{\rightarrow}1$

– 12 donne $S_{1,2}{\rightarrow}C_{1,1}C_{2,2}$, $C_{1,1}{\rightarrow}1$, $C_{2,2}{\rightarrow}2$

– 123 donne $S_{1,3}{\rightarrow}N_{1,2}C_{3,3}$, $N_{1,2}{\rightarrow}C_{1,1}C_{2,2}$, $C_{1,1}{\rightarrow}1$, $C_{2,2}{\rightarrow}2$, $C_{3,3}{\rightarrow}3$

– 12.34 donne $S_{1,5}{\rightarrow}N_{1,2}D_{3,5}$, $N_{1,2}{\rightarrow}C_{1,1}C_{2,2}$, $C_{1,1}{\rightarrow}1$, $C_{2,2}{\rightarrow}2$,
$D_{3,5}{\rightarrow}V_{3,3}N_{4,5}$, $V_{3,3}{\rightarrow}.$, $N_{4,5}{\rightarrow}C_{4,4}C_{5,5}$, $C_{4,4}{\rightarrow}3$, $C_{5,5}{\rightarrow}4$

– $12e+2$ donne $S_{1,5}{\rightarrow}N_{1,2}X_{3,5}$, $N_{1,2}{\rightarrow}C_{1,1}C_{2,2}$, $C_{1,1}{\rightarrow}1$, $C_{2,2}{\rightarrow}2$,
$X_{3,5}{\rightarrow}F_{3,4}C_{5,5}$, $F_{3,4}{\rightarrow}E_{3,3}P_{4,4}$, $E_{3,3}{\rightarrow}e$, $P_{4,4}{\rightarrow}+$, $C_{5,5}{\rightarrow}2$

– ...

Méthode d'Earley

– Pour une entrée $a_1 \cdots a_n$, l'algorithme construit une suite d'états $q_0 \cdots q_n$ tels que
  . $q_{i-1} \xrightarrow{a_i} q_i$, donc dans l'état q_i, on a lu $a_1 \cdots a_i$
  . à chaque état q_i est associé un ensemble d'items de la forme $[A \rightarrow \alpha \cdot \beta, k]$, qui signifie que
    * on est dans la reconnaissance de A, qui commence à partir de a_k
    * on a reconnu α (qui couvre donc $a_k \cdots a_i$)
    * on s'apprête à reconnaître β (si $\beta \neq \varepsilon$)
  . dans un ensemble d'items, on distingue
    * le sous-ensemble, appelé *complets*, des items de la forme $[A \rightarrow \alpha \cdot, k]$
    * le sous-ensemble, appelé *actifs*, des items de la forme $[A \rightarrow \alpha \cdot X\beta, k]$

Calcul des états dans la méthode d'Earley - I

– Une transition $q_{i-1} \xrightarrow{a_i} q_i$ a lieu si q_{i-1} contient un (ou plusieurs) item de la forme $[A \rightarrow \alpha \cdot a_i \beta, k]$

– Le noyau de q_i est formé des items $[A \rightarrow \alpha a_i \cdot \beta, k]$, et

  . si $\beta = B\gamma$, il faut faire une fermeture, c'est à dire ajouter à q_i les items $[B \rightarrow \cdot \varphi, i+1]$ (on s'apprête à reconnaître φ à partir du prochain terminal a_{i+1})

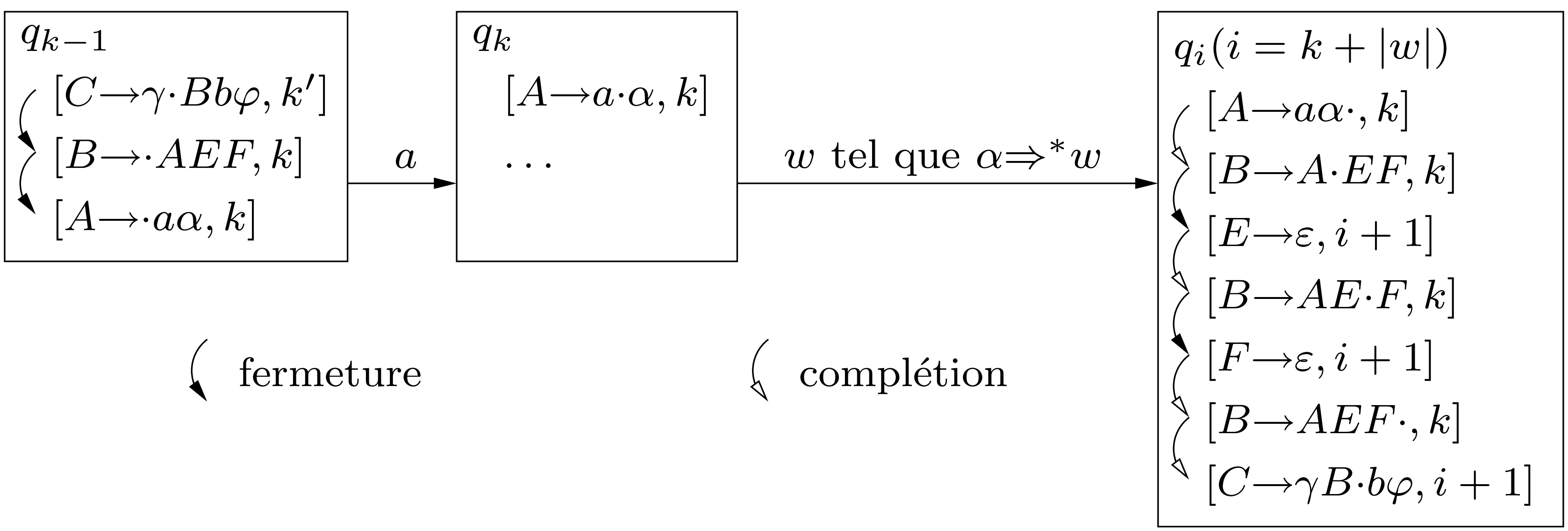
  . si $\beta = \varepsilon$, il faut *compléter*, c'est à dire ajouter à q_i les items $[C \rightarrow \gamma A \cdot \varphi, k']$, tels qu'il existe $[A \rightarrow \cdot \alpha a_i \beta, k]$ et $[C \rightarrow \gamma \cdot A \varphi, k']$ dans q_{k-1}

  . et ceci récursivement selon la forme de φ

– L'entrée $a_1 \cdots a_n$ appartient au langage si, et seulement si, q_n contient $[S \rightarrow \alpha \cdot, 1]$

Illustration de la fermeture et de la complétion

– Le traitement des productions vides est pris en charge automatiquement



– On peut éviter la recherche d'items dans un état si chaque item se "souvient" de celui dont il est issu : si $[C\to\gamma\cdot Bb\varphi, \nu_0] = \nu_1$, alors on aura $[B\to AEF, \nu_1] = \nu 2$ et $[A\to\cdot a\alpha, \nu_2] = \nu_3$

Calcul des états dans la méthode d'Earley - II

– Fermeture et complétion : c'est l'ensemble minimum défini par

$$
\begin{aligned}
C(q_i) = q_i & \\
& \cup \{[B\to\cdot\gamma, i+1] \,|\, [A\to\alpha\cdot B\beta, k] \in C(q_i), B\to\gamma \in P\} \\
& \cup \{[B\to\gamma A\cdot\varphi, k'] \,|\, [A\to\alpha\cdot, k] \in C(q_i), [A\to\cdot\alpha, k] \in q_{k-1}, \\
& \qquad [B\to\gamma\cdot A\varphi, k'] \in q_{k-1}\}
\end{aligned}
$$

– Etat initial :

$$q_0 = C(\, \{[S\to\cdot\alpha, 1] \,|\, S\to\alpha \in P\}\,)$$

– Transitions :

$$\Theta(q_i, a) = C(\, \{[A\to\alpha a\cdot\beta, k] \,|\, [A\to\alpha\cdot a\beta, k] \in q_i\}\,)$$

Exemple de fonctionnement de la méthode d'Earley - I

– On prend la grammaire des nombres :

$S \rightarrow NDX$	$N \rightarrow C$	$N \rightarrow NC$	$D \rightarrow .N$	$D \rightarrow \varepsilon$
$X \rightarrow e + N$	$X \rightarrow e - N$	$X \rightarrow \varepsilon$		
$C \rightarrow 0$	$C \rightarrow 1$	$\cdots$	$\cdots$	$C \rightarrow 9$

– L'entrée 1 donne (en haut de l'état, items complets)

$q_0 =$

$[S \rightarrow \cdot NDX, 1]$	$[N \rightarrow \cdot C, 1]$	$[N \rightarrow \cdot NC, 1]$
$[C \rightarrow \cdot 0, 1]$	$\cdots$	$[C \rightarrow \cdot 9, 1]$

$\xrightarrow{1}$

$q_1 =$

$[C \rightarrow 1 \cdot, 1]$	$[N \rightarrow C \cdot, 1]$	$[D \rightarrow \cdot, 2]$	$[X \rightarrow \cdot, 2]$
$[S \rightarrow NDX \cdot, 1]$			
$[S \rightarrow N \cdot DX, 1]$	$[N \rightarrow N \cdot C, 1]$	$[D \rightarrow \cdot .N, 2]$	$[C \rightarrow \cdot 0, 2]$
$[C \rightarrow \cdot 1, 2]$	$[C \rightarrow \cdot 2, 2]$	$[C \rightarrow \cdot 3, 2]$	$[C \rightarrow \cdot 4, 2]$
$[C \rightarrow \cdot 5, 2]$	$[C \rightarrow \cdot 6, 2]$	$[C \rightarrow \cdot 7, 2]$	$[C \rightarrow \cdot 8, 2]$
$[C \rightarrow \cdot 9, 2]$	$[S \rightarrow ND \cdot X, 1]$	$[X \rightarrow \cdot e + N, 2]$	$[X \rightarrow \cdot e - N, 2]$

Exemple de fonctionnement de la méthode d'Earley - II

– L'entrée $12.3e + 4$ donne

$$q_0 = \begin{array}{|lll|}
\hline
 & & \\
\hline
[S\rightarrow \cdot NDX, 1] & [N\rightarrow \cdot C, 1] & [N\rightarrow \cdot NC, 1] \\
[C\rightarrow \cdot 0, 1] & \cdots & [C\rightarrow \cdot 9, 1] \\
\hline
\end{array} \quad \xrightarrow{1}$$

$$q_1 = \begin{array}{|lll|}
\hline
[C\rightarrow 1\cdot, 1] & [N\rightarrow C\cdot, 1] & [D\rightarrow \cdot, 2] \\
[X\rightarrow \cdot, 2] & [S\rightarrow NDX\cdot, 1] & \\
\hline
[S\rightarrow N\cdot DX, 1] & [N\rightarrow N\cdot C, 1] & [D\rightarrow \cdot .N, 2] \\
[C\rightarrow \cdot 0, 2] & \cdots & [C\rightarrow \cdot 9, 2] \\
[S\rightarrow ND\cdot X, 1] & [X\rightarrow \cdot e + N, 2] & [X\rightarrow \cdot e - N, 2] \\
\hline
\end{array} \quad \xrightarrow{2}$$

$$q_2 = \begin{array}{|lll|}
\hline
[C\rightarrow 2\cdot, 2] & [N\rightarrow NC\cdot, 1] & [D\rightarrow \cdot, 3] \\
[X\rightarrow \cdot, 3] & [S\rightarrow NDX\cdot, 1] & \\
\hline
[S\rightarrow N\cdot DX, 1] & [N\rightarrow N\cdot C, 1] & [D\rightarrow \cdot .N, 3] \\
[C\rightarrow \cdot 0, 3] & \cdots & [C\rightarrow \cdot 9, 3] \\
[S\rightarrow ND\cdot X, 1] & [X\rightarrow \cdot e + N, 3] & [X\rightarrow \cdot e - N, 3] \\
\hline
\end{array} \quad \xrightarrow{\cdot}$$

Exemple de fonctionnement de la méthode d'Earley - IIa

$$q_3 = \begin{array}{|lll|}\hline \\ \hline [D\rightarrow.\cdot N,3] & [N\rightarrow\cdot C,4] & [N\rightarrow\cdot NC,4] \\ [C\rightarrow\cdot 0,4] & \cdots & [C\rightarrow\cdot 9,4] \\ \hline \end{array} \quad \xrightarrow{3}$$

$$q_4 = \begin{array}{|lll|}\hline [C\rightarrow 3\cdot,4] & [N\rightarrow C\cdot,4] & [D\rightarrow.N\cdot,3] \\ [X\rightarrow\cdot,5] & [S\rightarrow NDX\cdot,1] & \\ \hline [N\rightarrow N\cdot C,4] & [S\rightarrow ND\cdot X,1] & \\ [C\rightarrow\cdot 0,5] & \cdots & [C\rightarrow\cdot 9,5] \\ [X\rightarrow\cdot e+N,5] & [X\rightarrow\cdot e-N,5] & \\ \hline \end{array} \quad \xrightarrow{e}$$

$$q_5 = \begin{array}{|ll|}\hline \\ \hline [X\rightarrow e\cdot+N,5] & [X\rightarrow e\cdot-N,5] \\ \hline \end{array} \quad \xrightarrow{+}$$

$$q_6 = \begin{array}{|lll|}\hline \\ \hline [X\rightarrow e+\cdot N,5] & [N\rightarrow\cdot C,7] & [N\rightarrow\cdot NC,7] \\ [C\rightarrow\cdot 0,7] & \cdots & [C\rightarrow\cdot 9,7] \\ \hline \end{array} \quad \xrightarrow{4}$$

$$q_7 = \begin{array}{|llll|}\hline [C\rightarrow 4\cdot,7] & [N\rightarrow C\cdot,7] & [X\rightarrow e+N\cdot,5] & [S\rightarrow NDX\cdot,1] \\ \hline [N\rightarrow N\cdot C,7] & [C\rightarrow\cdot 0,8] & \cdots & [C\rightarrow\cdot 9,8] \\ \hline \end{array}$$

Exemple de fonctionnement de la méthode d'Earley - III

– Avec la grammaire $\mathcal{G}_{ASa}$ suivante, qui a des récursivités gauches cachées, $S \rightarrow ASa$, $S \rightarrow b$, $A \rightarrow \varepsilon$ et le mot $ba \ldots a$, on obtient

$$q_0 = \begin{array}{|lll|}\hline [A \rightarrow \cdot, 1] & & \\ \hline [S \rightarrow \cdot ASa, 1] & [S \rightarrow \cdot b, 1] & [S \rightarrow A \cdot Sa, 1] \\ \hline \end{array} \quad \xrightarrow{b}$$

$$q_1 = \begin{array}{|l|}\hline [S \rightarrow b \cdot, 1] \\ \hline [S \rightarrow AS \cdot a, 1] \\ \hline \end{array} \quad \xrightarrow{a} \quad q_2 = \begin{array}{|l|}\hline [S \rightarrow ASa \cdot, 1] \\ \hline [S \rightarrow AS \cdot a, 1] \\ \hline \end{array} \quad \xrightarrow{a} \cdots \xrightarrow{a}$$

$$q_n = \begin{array}{|l|}\hline [S \rightarrow ASa \cdot, 1] \\ \hline [S \rightarrow AS \cdot a, 1] \\ \hline \end{array}$$

Sait-on compter le nombre de réductions du vide ?

Réponse plus loin.

Calcul des arbres avec la méthode d'Earley

– Si la phrase d'entrée est correcte, l'état final q_n contient un (ou plusieurs) item de la forme $[S \rightarrow \alpha\cdot, 1]$ si $\alpha \Rightarrow^* a_1 \cdots a_n$

– Pour chacun de ces items, on produira la règle $\Pi(\alpha)$ qui décrit l'arbre de α, puis on produira $S_{1,n} \rightarrow \Pi(\alpha)$

– Pour produire $\Pi(\alpha)$ dans un état q_i

  . on a $[A \rightarrow \alpha\cdot, j]$ dans q_i, et $\alpha \Rightarrow^* a_j \cdots a_i$

  . si $\alpha = \alpha_1 B$, alors q_i contient $[B \rightarrow \beta\cdot, k]$ et $\beta \Rightarrow^* a_k \cdots a_i$

    ∗ on aura à produire $\Pi(\beta)$, puis $B_{k,i} \rightarrow \Pi(\beta)$, et ainsi de suite pour tous les items complets de q_i (si $\beta = \beta_1 C, \ldots$)

    ∗ l'état q_{k-1} contient $[A \rightarrow \alpha_1 \cdot B, j]$: si $\alpha_1 \neq \varepsilon$, alors $\alpha_1 \Rightarrow^* a_j \ldots a_{k-1}$, et il faut produire $\Pi(\alpha_1)$ dans q_{k-1}, puis $A_{j,i} \rightarrow \Pi(\alpha_1) B_{k,i}$

  . si $\alpha = \alpha_1 a$, on produit $\Pi(\alpha_1)$ dans q_{i-1}, puis $A_{j,i} \rightarrow \Pi(\alpha_1) a$

Calcul des arbres avec la méthode d'Earley - II

- On produira des règles par exemple de la forme $A_{i,j} \rightarrow B_{i,k-1} b C_{k+1,j}$ si $a_k = b$
- Les symboles de la règle à produire seront rangés dans un vecteur $\pi_0 \cdots \pi_r$, ou r est la longueur de la partie droite qu'on traite
- L'algorithme Π est donc

  $\Pi(q_i, A \rightarrow X_1 \cdots X_r \cdot)$
  pour $k := r$ **à** 1 **faire**
    si $X_k = a$ **alors**
      $\pi_k := a$; $i := i - 1$
    sinon avec $X_k = B$ % donc il existe $[B \rightarrow \beta \cdot, j]$ dans q_i
      $\pi_k := B_{j,i}$; $\Pi(q_{j-1}, [B \rightarrow \beta \cdot, j])$; $i := j - 1$
  fin pour
  produire $\pi_0 \rightarrow \pi_1 \cdots \pi_r$

- On l'appelle par $\Pi(q_n, [S \rightarrow \alpha \cdot, 1])$

Exemple de calcul d'arbre avec la méthode d'Earley - I

– Avec l'entrée 1 et la grammaire des nombres :

$$\Pi(q_1, [S \rightarrow NDX\cdot, 1])$$
$$\Pi(q_1, [X \rightarrow \cdot, 2])$$
$$\Pi(q_1, [D \rightarrow \cdot, 2])$$
$$\Pi(q_1, [N \rightarrow C\cdot, 1])$$
$$\Pi(q_1, [C \rightarrow 1\cdot, 1])$$

$$X_{2,1} \rightarrow \varepsilon$$
$$D_{2,1} \rightarrow \varepsilon$$
$$C_{1,1} \rightarrow 1$$
$$N_{1,1} \rightarrow C_{1,1}$$
$$S_{1,1} \rightarrow N_{1,1} D_{2,1} X_{2,1}$$

Exemple de calcul d'arbre avec la méthode d'Earley - II

– avec l'entrée $12.3e + 4$:

$\Pi(q_7, [S \rightarrow NDX\cdot, 1])$	$C_{7,7} \rightarrow 4$
$\Pi(q_7, [X \rightarrow e + N\cdot, 5])$	$N_{7,7} \rightarrow C_{7,7}$
$\Pi(q_7, [N \rightarrow C\cdot, 7])$	$X_{5,7} \rightarrow e + N_{7,7}$
$\Pi(q_7, [C \rightarrow 4\cdot, 7])$	$C_{4,4} \rightarrow 3$
$\Pi(q_4, [D \rightarrow .N\cdot, 3])$	$N_{4,4} \rightarrow C_{4,4}$
$\Pi(q_4, [N \rightarrow C\cdot, 4])$	$D_{3,4} \rightarrow .N_{4,4}$
$\Pi(q_4, [C \rightarrow 3\cdot, 4])$	$C_{2,2} \rightarrow 2$
$\Pi(q_2, [N \rightarrow NC\cdot, 1])$	$C_{1,1} \rightarrow 1$
$\Pi(q_2, [C \rightarrow 2\cdot, 2])$	$N_{1,1} \rightarrow C_{1,1}$
$\Pi(q_1, [N \rightarrow C\cdot, 1])$	$N_{1,2} \rightarrow N_{1,1} C_{2,2}$
$\Pi(q_1, [C \rightarrow 1\cdot, 1])$	$S_{1,7} \rightarrow N_{1,2} D_{3,4} X_{5,7}$

Exemple de calcul d'arbre avec la méthode d'Earley - III

– avec l'entrée b et la grammaire $\mathcal{G}_{ASa}$:

$$q_0 = \begin{array}{|lll|} \hline [A\rightarrow\cdot, 1] & & \\ \hline [S\rightarrow\cdot ASa, 1] & [S\rightarrow\cdot b, 1] & [S\rightarrow A\cdot Sa, 1] \\ \hline \end{array} \quad \xrightarrow{b} \quad q_1 = \begin{array}{|l|} \hline [S\rightarrow b\cdot, 1] \\ \hline [S\rightarrow AS\cdot a, 1] \\ \hline \end{array}$$

$\Pi(q_1, [S\rightarrow b\cdot, 1])$ $\qquad$ $S_{1,1}\rightarrow b$

– avec l'entrée baa :

$\Pi(q_3, [S\rightarrow ASa\cdot, 1])$	$S_{1,1}\rightarrow b$
$\Pi(q_2, [S\rightarrow ASa\cdot, 1])$	$A_{1,0}\rightarrow\varepsilon$
$\Pi(q_1, [S\rightarrow b\cdot, 1])$	$S_{1,2}\rightarrow A_{1,0}S_{1,1}a$
$\Pi(q_0, [A\rightarrow\cdot, 1])$	$S_{1,3}\rightarrow A_{1,0}S_{1,2}a$

on a bien deux réductions du vide