Langages formels et automates

Reconnaissance de mots dans les grammaires hors-contexte

Catalin Dima

Algorithme d'Earley

- Algorithme permettant de vérifier si un mot $w = w_1 w_2 \dots w_n$ est accepté par une grammaire.
- *Configurations* de l'algorithme : ensemble d'items $[i]X \longrightarrow \alpha \bullet \beta[j]$, où
  - $X \longrightarrow \alpha\beta$ est une production, dont le membre droit est découpé en deux mots, $\alpha \cdot \beta$.
  - $1 \leq i \leq j \leq n$ sont des indices représentant le sous-mot $w_i \dots w_j$ de w que la production courante est censée produire.
  - Le $\bullet$ donne la *position courante* dans le mot w jusqu'où la production courante est "capable de générer" w.
- L'algorithme fonctionne en construisant itérativement des items , à partir de la configuration contenant toutes les productions $S \longrightarrow \bullet\, \alpha$, jusqu'à ce qu'une configuration contenant $[0]S \longrightarrow \alpha \bullet [n]$ soit construite.

Algorithme d'Earley

1. **Initialisation** : Configuration
   $C_0 = \{[0]S \longrightarrow \bullet\, \alpha[0] \mid S \longrightarrow \alpha \in P\}$.
2. **Lecture** : Si, dans C_i on a un item $[j]X \longrightarrow \alpha \bullet a\beta[i]$ avec $a = w_{i+1}$ ($(i+1)$-ième lettre du mot d'entrée), alors on rajoute dans C_{i+1} l'item $[j]X \longrightarrow \alpha a \bullet a\beta[i+1]$.
3. **Prédiction** : Si, dans C_i on a un item $[j]X \longrightarrow \alpha \bullet Y\beta[i]$ alors on rajoute dans C_i tous les items $[i]Y \longrightarrow \bullet\, \gamma[i]$, pour toute production $Y \longrightarrow \gamma \in P$.
4. **Complétion** : Si, dans C_i on a un item $[j]Y \longrightarrow \gamma \bullet [i]$ alors pour tout autre item de la forme $[k]X \longrightarrow \alpha \bullet Y\beta[j]$ dans C_j on rajoute aussi l'item $[k]X \longrightarrow \alpha Y \bullet \beta[i]$ dans C_i (attention aux indices !).

Algorithme d'Earley

Grammaire :

$$E \longrightarrow T \mid E + T$$
$$T \longrightarrow F \mid T * F$$
$$F \longrightarrow a \mid b \mid (E)$$

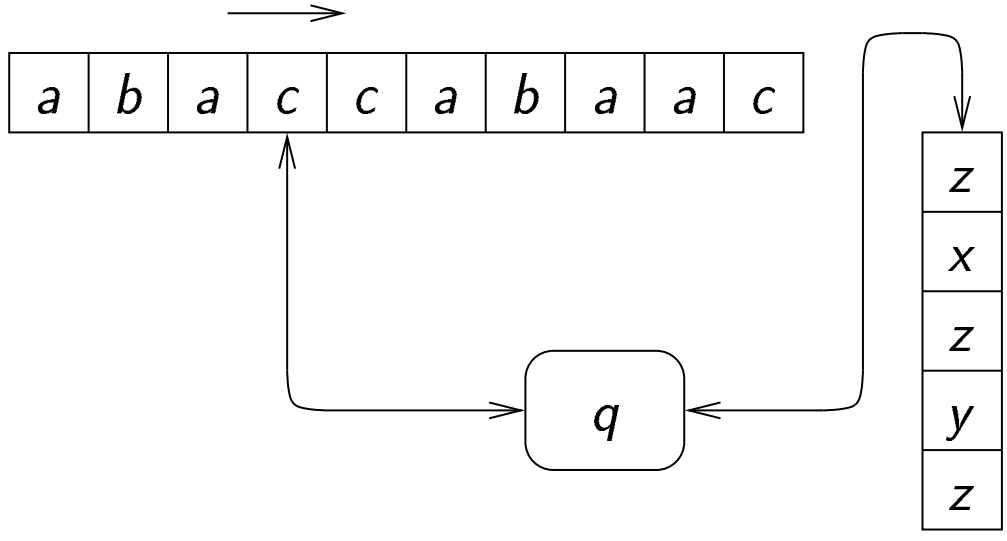
Génération de $a + b$:

C_0	C_1	C_2	C_3
$[0]E \rightarrow \bullet\, T[0]$	$[0]F \rightarrow a \bullet [1]$	$[0]E \rightarrow E + \bullet\, T[2]$	$[2]F \rightarrow b \bullet [3]$
$[0]E \rightarrow \bullet\, E + T[0]$	$[0]T \rightarrow F \bullet [1]$	$[2]T \rightarrow \bullet\, F[2]$	$[2]T \rightarrow F \bullet [3]$
$[0]T \rightarrow \bullet\, F[0]$	$[0]T \rightarrow T \bullet * F[1]$	$[2]T \rightarrow \bullet\, T * F[2]$	$[2]T \rightarrow T \bullet * F[3]$
$[0]T \rightarrow \bullet\, T * F[0]$	$[0]E \rightarrow T \bullet [1]$	$[2]F \rightarrow \bullet\, a[2]$	$[0]E \rightarrow E + T \bullet [3]$
$[0]F \rightarrow \bullet\, a[0]$	$[0]E \rightarrow E \bullet + T[1]$	$[2]F \rightarrow \bullet\, b[2]$	
$[0]F \rightarrow \bullet\, b[0]$		$[2]F \rightarrow \bullet\, (E)[2]$	
$[0]F \rightarrow \bullet\, (E)[0]$			

- Donc $a + b$ accepté (on trouve $[3]E \rightarrow E + T \bullet [3]$ dans la dernière colonne).
- Essayer aussi avec $a + b * a$, $a * (b * (b))$, $b+$!

Automates pour langages hors contexte

- Automate à pile : utilise une mémoire infinie, organisée à la manière d'une pile.



- Les lettres du mot sont lues de gauche à droite.
- À chaque pas, le top de la pile compte dans la modification de l'état de l'automate.
- À chaque pas, on peut empiler ou dépiler des symboles.

Automates à pile

- $\mathcal{A} = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, Q_f)$.
- Q = ensemble d'états.
- Σ = alphabet d'entrée (bande de lecture).
- Γ = alphabet de pile.
- q_0 = état initial, Q_f = ensemble d'états finaux.
- Z_0 = symbole initial dans la pile.
- δ = relation de transition :

$$\delta \subseteq Q \times \left(\Sigma \cup \{\varepsilon\}\right) \times \Gamma \times \Gamma^* \times Q$$

Configuration dans un automate à pile

- Une transition dans δ est un tuple $q \xrightarrow{a,z,\gamma} r$.
  - Si l'automate est dans l'état q, la lettre sous la tête de lecture est a et le top de la pile est z,
  - ... alors on remplace le top de la pile z par le mot $\gamma \in \Gamma^*$,
  - ... et on change d'état en r.
- Une configuration de l'automate : $(q, w, \gamma) \in Q \times \Sigma^* \times \Gamma^*$.
  - Tuple (état, partie du mot pas encore lue, contenu de la pile).
- De cette configuration on peut évoluer dans une autre en appliquant une transition $q \xrightarrow{a,z,\beta} r$:
  - Si la première lettre du mot d'entrée w est a,
  - Et la première lettre du mot de pile γ est z,
  - Alors on change d'état en r, on "bouffe" a et on remplace z par β.
  - On écrit alors $(q, w = aw', \gamma = z\gamma') \vdash (r, w', \beta\gamma')$.
- Configuration initiale : (q_0, w, Z_0).
- Configuration acceptante : (r, ε, γ) avec $r \in Q_f$.
- Si de (q, w, Z_0) on peut franchir un (r, ε, γ) pour $r \in Q_f$, alors w est accepté !

Trajectoires d'automates à pile

- Liste de transitions (ça ne sert plus à rien de représenter l'automate comme un graphe...).

$$q_0 \xrightarrow{a,A,AA} q \qquad q \xrightarrow{a,A,AA} q$$
$$q \xrightarrow{b,A,\varepsilon} r \qquad r \xrightarrow{b,A,\varepsilon} r$$

- État initial q_0, final r, symbole initial de pile A.
- Exemple de configuration initiale : (q_0, w, A).
- Exemple de trajectoire = suite de configurations reliées par des transitions :

$$(q_0, aaabb, A) \vdash (q, aabb, AA) \vdash (q, abb, AAA) \vdash (q, bb, AAAA)$$
$$\vdash (r, b, AAA) \vdash (r, \varepsilon, AA)$$

- Et donc on accepte *aaabb* !
- Essayer aussi avec *aaab* et *aabb*.

Trajectoires

- Exemple de trajectoire non-acceptante :

$$(q_0, abab, A) \vdash (q, bab, AA) \vdash (r, ab, A) \nvdash$$

- ... car il n'y a pas de transition qui puisse s'appliquer dans cette configuration !
- Même plus, aucune autre trajectoire ne peut être associée à $abab$!

Langage accepté

Langage accepté par un automate $\mathcal{A}$:

$$L(\mathcal{A}) = \left\{ w \in \Sigma^* \mid (q_0, w, \gamma_0) \vdash (r, \varepsilon, z) \text{ configuration acceptante } \right\}$$

Exemple :

$$q_0 \xrightarrow{a,A,AA} q \qquad q \xrightarrow{a,A,AA} q$$

$$q \xrightarrow{b,A,\varepsilon} r \qquad r \xrightarrow{b,A,\varepsilon} r$$

- Automate à pile pour $L = \{a^n b^m \mid n > m\}$

Exemples d'automates à pile

- Et si on voulait exactement $L_{anbn} = \{a^n b^n \mid n \in \mathbb{N}\}$?
- Il faut avoir bouffé, quand on entre dans l'état final, tous les A dans la pile !
- Automate avec ε-transitions :

$$q_0 \xrightarrow{a,Z_0,Z_0A} q \qquad q \xrightarrow{a,A,AA} q$$

$$q \xrightarrow{b,A,\varepsilon} r \qquad r \xrightarrow{b,A,\varepsilon} r$$

$$r \xrightarrow{\varepsilon,Z_0,\varepsilon} s \qquad q_0 \xrightarrow{\varepsilon,Z_0,\varepsilon} s$$

- Et maintenant c'est s qui est l'état final !
- Pas de transition en s !
  - Donc si on a appliqué la dernière transition dès le début, on est bloqué dans une configuration (s, w, ε) qui n'est pas finale !

Acceptation par pile vide

- On peut modifier la définition de l'acceptation par automate à pile :
  - On permet de s'arrêter dans n'importe que état.
  - Mais il faut avoir fini de parcourir le mot d'entrée.
  - Et aussi la pile devrait être vide !
- Modifions notre exemple pour L_{anbn} pour qu'il fonctionne par acceptation par pile vide :

$$q_0 \xrightarrow{a,A,AA} q \qquad q \xrightarrow{a,A,AA} q$$

$$q \xrightarrow{b,A,\varepsilon} r \qquad r \xrightarrow{b,A,\varepsilon} r$$

$$r \xrightarrow{\varepsilon,A,\varepsilon} s \qquad q_0 \xrightarrow{\varepsilon,A,\varepsilon} s$$

- On voit que le premier A dans la pile ne sert pas à compter les b, mais plutôt pour finaliser la trajectoire.
- **Théorème** : Si un langage est accepté par un automate à pile par acceptation sur états finaux, alors il sera accepté aussi par un automate à pile par acceptation par pile vide, et vice-versa !

Automates à pile et grammaires hors contexte

- **Théorème** : Tout langage hors contexte est accepté par un automate à pile. La réciproque est vraie aussi : tout langage d'un automate à pile est hors contexte.
- Association d'un automate à pile à une grammaire hors contexte :
  - Les nonterminaux et les terminaux forment l'alphabet de la pile.
  - Chaque production est simulée sur la pile.
  - Un seul état suffit !
  - Si le top de la pile est un terminal a, il faut le bouffer sur la bande d'entrée.
  - Si c'est un nonterminal X, il faut faire une ε-transition qui place sur la pile le résultat d'une production de X.
  - Acceptation par pile vide.

Automate à pile pour une grammaire hors contexte

- Exemple de grammaire :

$$S \longrightarrow ASB \mid \varepsilon$$
$$A \longrightarrow aAS \mid a$$
$$B \longrightarrow SbS \mid bb$$

- Automate associé :

$$q \xrightarrow{\varepsilon,S,ASB} q \qquad q \xrightarrow{\varepsilon,S,\varepsilon} q$$
$$q \xrightarrow{\varepsilon,A,aAS} q \qquad q \xrightarrow{\varepsilon,A,a} q$$
$$q \xrightarrow{\varepsilon,B,SbS} q \qquad q \xrightarrow{\varepsilon,B,bb} q$$
$$q \xrightarrow{a,a,\varepsilon} q \qquad q \xrightarrow{b,b,\varepsilon} q$$

- Symbole initial de pile : S !
- Acceptation de *aaabbb* : arbre syntaxique, et trajectoire dans l'automate.

Limites des langages hors contexte

- On peut définir des automates déterministes.
- Mais ils ne seront pas équivalents avec les non-déterministes.
  - Un automate déterministe génère un langage non-ambigu.
  - Et comme il y a des langages à ambiguité inhérente, ces langages ne peuvent pas être acceptés par un tel automate déterministe.
- Le problème du langage vide est décidable (c.à.d. il y a des algorithmes).
- Mais vérifier qu'une grammaire accepte tous les mots sur un alphabet n'est pas décidable !
- L'union et la concaténation de deux langages hors contexte est hors contexte ! (le prouver !)
- Mais il existe des langages hors contexte dont le complément n'est pas un langage hors contexte.
- Et aussi, l'intersection de deux langages hors contexte peut ne pas être hors contexte.
- Toutefois, si L est hors contexte et R est régulier alors $L \cap R$ sera hors contexte !

Lemme de "gonflement" ou de "pompage" pour les langages hors contexte

- Si L est hors contexte, alors il existe un entier $n_0 \in \mathbb{N}$ tel que tout mot de L ($z \in L$) contenant plus de n_0 lettres peut s'écrire $z = xyzuv$, avec les propriétés suivantes :
  1. $yu \neq \varepsilon$.
  2. Le "rayon hors contexte" $xy^n zu^n v$ est contenu dans L : pour tout $n \in \mathbb{N}$, $xy^n zu^n v \in L$.
- Généralisation du lemme de l'étoile des langages réguliers.
- S'applique de la même manière :
- Preuve par réduction à l'absurde :
  1. On suppose que L est hors contexte et on prouve que le lemme de gonflement nous amène à une contradiction.
- Comment trouver une contradiction :
  - On prend un mot $z \in L$ et on le décompose de toutes les manières possibles en $z = xyzuv$.
  - Pour chaque décomposition, on devrait prouver que le rayon $xy^n zu^n$ ne peut pas être inclus dans L.

Lemme de gonflement

- Idée de preuve du lemme :
  - On prend une grammaire pour le langage.
  - On l'amène à une forme qui ne contient plus d'ε-productions, ni de renommages, et tous les nonterminaux sont utiles.
  - On prend $n_0 = k^{card(N)+1}$, ou k est le nombre de lettres dans la plus grande production.
  - On prend un mot w ayant plus de n_0 lettres.
  - On prend aussi un arbre syntaxique pour w.
  - Du fait du choix de n_0, dans l'arbre syntaxique il doit y avoir un chemin racine $\longrightarrow$ feuille qui a plus de n_0 arêtes.
  - Alors un des noeuds de ce chemin se repète – et il correspond à un nonterminal X !
  - On a identifié une partie de l'arbre qui peut être répétée autant de foix qu'on veuille !
  - La frontière de cette partie de l'arbre, à gauche de sa feuille X, forme la partie y de notre mot, et celle de droite forme u.
  - Le reste de la frontiere forme x, z et v.
  - Répéter n fois la partie de l'arbre trouvée, revient à générer xy^nzu^nv.
  - Dessin sur le tableau...

Exemple d'application du lemme de gonflement

$L_{anbncn} = \{a^n b^n c^n \mid n \in \mathbb{N}\}$.

- Prenons toute décomposition d'un mot $a^n b^n c^n$ en $xyzuv$.
- Plusieurs cas possibles :
  1. y et u ne contiennent que des a.
  2. y contient des a et u contient des b.
  3. y contient des a et u contient des c.
  4. y et u ne contiennent que des b.
  5. y contient des b et u contient des c.
  6. ....
  7. y contient des a et des b et....
- Mais il y a deux types de cas :
  1. Quand les deux mots y, u prennent chacun un seul type de lettre.
  2. Quand un des deux mots y, u prend deux types de lettres.
- Dans chaque cas, il faut prouver qu'il existe des membres du rayon $xy^n zu^n v$ qui ne sont pas dans L.
- Dans tous les cas ici on peut prouver que $xyyzuuv = xy^2zu^2v$ n'est pas dans le langage.

Exemple de langage non-hors contexte

- $L = \{w \in \{a, b, c\}^* \mid \#_a(w) = \#_b(w) = \#_c(w)\}$.
- Au lieu d'appliquer le lemme de gonflement, on peut utiliser une idée vue pour les langages réguliers :
  - Si L était hors contexte, alors $L \cap R$ serait aussi hors contexte pour tout langage régulier R.
- Il nous faut R régulier tel que $L \cap R = \{a^n b^n c^n \mid n \in \mathbb{N}\}$.
- Qui pourrait être R ?