Algorithme d'Earley

Antoine Dequay

21 septembre 2022

Notes

— Prof : .

— Leçon : 907, 923.

— Références :

— Floyd & Biegel, *Le langage des machines.*

On cherche à résoudre le problème du mot pour une grammaire G quelconque. On suppose que $\varepsilon \notin L(G)$ et, sans perte de généralité, que la grammaire ne possède pas de règles de la forme $A \to B \in \mathcal{N} \cup \Gamma$ (règles unitaires).Pourquoi ? ? ?

Ecrire l'algo dans le plan

Algorithm 1: EARLEY$(w = w_1 \dots w_n, G)$

Entrée: Un mot w et une grammaire G.

Sortie : Un booléen indiquant si $w \in L(G)$.

1 $OBJET[0] = \{(0, 0, S, \varepsilon, \beta) : S \to \beta \text{ règle de } G\}$;
2 **for** $j \leftarrow 1$ **to** n **do**
3 Initialiser $OBJET[j] = \emptyset$
4 **for** $j \leftarrow 0$ **to** $n - 1$ **do**
5 **for** $(i, j, P, \alpha, Q\gamma)$ *dans* $OBJET[j]$ *(dont les objets rajoutés pendant cette boucle) et* $Q \to \delta$ *règle de* G **do**
6 Ajouter $(j, j, Q, \varepsilon, \delta)$ à $OBJET[j]$ s'il n'y est pas
7 **for** $(i, j, P, \alpha, c\gamma)$ *de* $OBJET[j]$ *tel que* $c = w[j + 1]$ **do**
8 Ajouter $(i, j + 1, P, \alpha c, \gamma)$ à $OBJET[j + 1]$
9 **for** $(i, j + 1, P, \alpha, \varepsilon) \in OBJET[j + 1]$ *et* $(h, i, R, \gamma, P\delta) \in OBJET[i]$ **do**
10 Ajouter $(h, j + 1, R, \gamma P, \delta)$ à $OBJET[j + 1]$ s'il n'y est pas (et le traiter dans cette boucle)
11 **if** *il existe un objet de la forme* $(0, n, S, \alpha, \varepsilon)$ *dans* $OBJET[n]$ **then**
12 **return** *True*
13 **else**
14 **return** *False*

Remarque

- On passe successivement par les étapes "initialisation", "fermeture", "progression" et "complétion",
- Pour écrire proprement les boucles 5 et 9, on fait une boucle while sur une queue/pile initialisée à $OBJET[j]$,
- un objet (i, j, P, α, β) est stocké dans $OBJET[j]$ si :
  - $P \to \alpha\beta$,
  - $\alpha \to^* w[i + 1 \dots j]$,
  - il reste à trouver $k \geq j + 1$, s'il existe, tel que $\beta \to^* w[j + 1 \dots k]$.

Ainsi :

- l'initialisation permet de traduire le fait qu'on cherche tous les préfixes de w qui sont dérivables à partir de G.

— La fermeture permet de chercher des préfixes de $w[j+1 \dots n]$ accessibles à partir de Q, en "oubliant" comment on est arrivé à ce Q,

— La progression s'intéresse à la lettre $w[j+1]$ et permet de se donner des (tous les) outils (possibles) pour accéder à cette lettre et continuer l'algorithme,

— La complétion permet de mettre à jour les objectifs en recombinant tout ce qu'il est possible de recombiner parmis ce que l'on a calculé.

Théorème 1 L'algorithme est correct.

Preuve. Par la structure de l'algorithme, tout objet construit représente une dérivation possible de la grammaire. On ne peut donc accepter que des mots dérivables dans la grammaire.

Montrons qu'on les accepte tous. Pour cela, soit $w \in L(G)$ et T un arbre de dérivation associé. On procède par récurrence structurelle.

On pose l'**hypothèse de récurrence** : On suppose que T possède un noeud Y qui produit $w[a+1 \dots d]$, et que pour tout β tel que $Y \to \beta$, $(a, a, Y, \varepsilon, \beta)$ ait été produit par l'algorithme. Alors l'algorithme produira un objet de la forme $(a, d, Y, \beta, \varepsilon)$ (c'est à dire que $Y \to^* w[a+1 \dots b]$).

L'hypothèse est vraie sur les feuilles de l'arbre (des terminaux) avec des objets de la forme $(a, a, c, c, \varepsilon)$ qui sont effectivement construits.

Soit Y la racine d'un sous-arbre de dérivation de T, supposons l'hypothèse vérifiée par tous les sous-arbres propres de ce sous-arbre. On note $(Z_i)_{i \in [\![1,n]\!]}$ les fils de Y, associés aux mots $w[z_i + 1 \dots z_{i+1}]$ avec $z_1 = a$ et $z_{n+1} = b$.

— Si Z_1 est terminal (i.e. $Z_1 = w[a+1]$), l'étape de progression permet de créer $(a, a+1, Y, Z_1, Z_2 \dots Z_n)$ à partir de $(a, a, Y, \varepsilon, Z_1 \dots Z_n)$, créé lors de la fermeture de Y.

— Si Z_1 est une variable, la fermeture (pour $j = a$) crée $(a, a, Z_1, \varepsilon, \beta)$ pour tout β tel que $Z_1 \to \beta$ dans G. L'hypothèse de récurrence permet d'obtenir l'existence de $(a, z_2, Z_1, \beta, \varepsilon)$. Pour $j = z_2$, la complétion permet de le combiner avec $(a, a, Y, \varepsilon, Z_1 \dots Z_n)$, ce qui permet d'obtenir $(a, z_2, Y, Z_1, Z_2 \dots Z_n)$.

On peut appliquer ce raisonnement successivement à $Z_2, \dots, Z_n$. On arrive bien, à la fin, à $(a, b, Y, Z_1 \dots Z_n, \varepsilon)$, d'où la preuve de l'hypothèse de récurrence !

Comme on commence avec tous les objets possibles de la forme $(0, 0, S, \varepsilon, \beta)$, l'hypothèse de récurrence sur la racine de T permet d'obtenir un objet de la forme $(0, n, S, \beta, \varepsilon)$, d'où le résultat !

□

Remarque Cf. complexité en début de la section.