Preuve d'un algorithme de calcul d'une table de parsing $LL(1)$

Thibaut Antoine (thibaut.antoine@ens-rennes.fr)

Phrase d'introduction

L'analyse syntaxique $LL(1)$ utilise une *table de parsing* pour calculer un AST d'une expression selon une grammaire passée en entrée. On illustre une manière de calculer la table et une preuve de sa correction.

1 Passage en « forme $LL(1)$ » [TIG]

Grammaire pour des sommes :
— $S \to E\$$
— $E \to E + E \mid (E) \mid \texttt{num}$
Exemple : $w = 1 + 2 + 3\$$

On présente une grammaire simple pour des sommes qui n'est pas $LL(1)$. En particulier, la règle $\boldsymbol{E} \to \boldsymbol{E} + E$ présente de la récursivité à gauche et par exemple, on ne peut pas parser le mot $w = 1 + 2 + 3\$$ car on ne peut pas savoir à l'avance combien de fois appliquer la règle.

$\leadsto$ Grammaire $LL(1)$ équivalente :
— $S \to E\$$
— $E \to FE'$; $E' \to +FE' \mid \varepsilon$
— $F \to \texttt{num} \mid (E)$
Attention : pas utilisé en pratique (parsing LR).

On peut éliminer la récursivité à gauche de cette grammaire avec l'ajout de variables et de règles. Cependant la transformation est assez lourde, et en pratique il existe d'autres techniques de parsing (comme l'analyse syntaxique $LR(k)$) qui permettent de l'éviter.

2 Calcul de nullable, first et follow [SCH]

On va ici énoncer la caractérisation des ensembles/valeurs nullable(X), first(X) et follow(X) pour une variable X par leurs équations de point fixe, qui permettent d'obtenir des algorithmes de calcul. On ne rappelle pas ici leurs définitions, il faut se référer au plan.

Théorème. *Pour* $X \in V$*,* $X \to Y_1 \cdots Y_p$*,*

1. $\text{nullable}(\varepsilon)$*, et* $[\forall i, \text{nullable}(Y_i)] \Rightarrow \text{nullable}(X)$*,*
2. $\text{first}(X)$ *est le pp. ens. tel que*

$$\begin{cases} \forall t \in T, \text{first}(t) = \{t\} \\ \forall i, \text{nullable}(Y_1 \cdots Y_{i-1}) \Rightarrow \text{first}(Y_i) \subseteq \text{first}(X) \end{cases}$$

3. $\forall i, j$ *:*

$$\begin{cases} \text{nullable}(Y_{i+1} \cdots Y_p) \Rightarrow \text{follow}(X) \subseteq \text{follow}(Y_i) \\ \text{nullable}(Y_{i+1} \cdots Y_{j-1}) \Rightarrow \text{first}(Y_j) \subseteq \text{follow}(Y_i) \end{cases}$$

On comprend assez bien ces équations intuitivement :

1. Si toutes les variables d'une règle se dérivent vers ε alors la concaténation de ces variables se dérive aussi vers ε par le lemme fondamental,
2. La seule lettre qui peut commencer un lexème est le lexème lui-même, et si dans la règle $X \to Y_1 \cdots Y_p$ les $Y_1, \ldots, Y_{i-1}$ se dérivent vers ε alors les premiers lexèmes de Y_i peuvent commencer des mots dérivés de X,
3. On comprend de manière similaire les équations pour follow.

Ces équations donnent un algorithme de calcul. On en donne un exemple d'utilisation sur la grammaire précédente avant de faire la preuve.

Exemple.

	nullable	**first**	**follow**
S	non	`num`, (	
E	non	`num`, (	), \$
E'	oui	+	), \$
F	non	`num`, (	), \$, +

On commence par le calcul de la colonne nullable en parcourant les règles de la grammaire une par une et en appliquant l'équation 1. On calcule ensuite la colonne first en ajoutant, pour chaque règle $X \to Y_1 \cdots Y_p$, les éléments de first(Y_i) à first(X) si $Y_1 \cdots Y_{i-1}$ est nullable (de sorte à ce que l'équation 2 soit toujours vérifiée). On fait enfin la même chose avec la colonne follow.

On remarque qu'on peut calculer aussi $\text{first}(\alpha)$ pour un mot en faisant l'union des first de chaque lettre succédant à un préfixe nullable de α.

On fait la preuve du théorème (seulement pour first, la preuve de nullable est similaire mais plus technique).

Preuve du théorème pour first. Soit, pour $X \in V \sqcup T$, $\mu(X)$ le pp. ens. décrit dans l'énoncé.

$\supseteq$ $\forall X, \mu(X) \subseteq \text{first}(X)$ car $\mu(X)$ le plus petit.

$\subseteq$ Montrons par récurrence forte sur $n \in \mathbb{N}$ la propriété :

$$\mathcal{P}(n) : \text{"}\forall X, \text{ si } \exists \beta, X \rightarrow^n t\beta \text{ alors } t \in \mu(X)\text{"}$$

- Si $n = 0$ et $X \rightarrow^n t\beta$ alors $X = t$ et $\mu(X) = \{t\} \ni t$ d'où $\mathcal{P}(0)$.
- Si $n \geq 0$ supposons $\mathcal{P}(k)$ pour $0 \leq k \leq n$.
  Si $X \rightarrow^{n+1} t\beta$ alors $X \rightarrow Y_1 \cdots Y_p \rightarrow^n t\beta$. Par le *lemme fondamental* on a $Y_i \rightarrow^{k_i} u_i$, $\sum_i k_i = n$ et $u_1 \cdots u_p = t\beta$.
  Soit j le plus petit indice tel que $u_j \neq \varepsilon$. Alors il existe γ tel que $u_j = t\gamma$ donc par $\mathcal{P}(k_j)$, $t \in \mu(Y_j)$.
  Or $u_1 \cdots u_{j-1} = \varepsilon$ donc $Y_1 \cdots Y_{j-1} \rightarrow^* \varepsilon$ i.e. $\text{nullable}(Y_1 \cdots Y_{j-1})$. Par hypothèse, $\mu(Y_j) \subseteq \mu(X)$ et donc $t \in \mu(X)$. D'où $\mathcal{P}(n+1)$.

$\square$

Clarifications sur la preuve :

$\supseteq$ On ne le montre pas, mais on voit que $\text{first}(X)$ vérifie bien les équations du théorème. Par minimalité de $\mu(X)$ parmi les ensembles vérifiant ces équations, on déduit l'inclusion.

$\subseteq$ Pour comprendre ce qui nous pousse à prouver cette propriété, il faut revenir à la définition de first :

$$\text{first}(X) = \{t \in T \mid \exists \beta \in (T \sqcup V)^*, X \rightarrow^* t\beta\}.$$

On voit alors que si $\mathcal{P}(n)$ est vraie pour tout n, alors on a bien $\text{first}(X) \subseteq \mu(X)$ pour tout X. En effet, si $t \in \text{first}(X)$ alors par définition il existe n et β tels que $X \rightarrow^n t\beta$. Alors par $\mathcal{P}(n)$ on conclut $t \in \mu(X)$.

Le reste de la preuve est plutôt clair, il faut faire attention cependant à bien quantifier universellement X (sur $T \sqcup V$) dans $\mathcal{P}$, car dans le cas contraire on ne peut pas utiliser la propriété de récurrence sur Y_j.

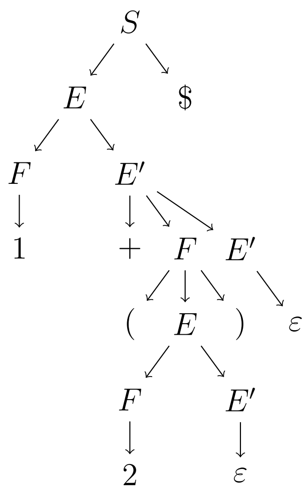
3 Table de prédiction

Grâce au calcul de nullable, first, et follow, on peut construire la table de prédiction $LL(1)$. On se réfère au plan pour savoir quand ajouter une règle $X \to \gamma$ dans la case (X, t).

	`num`	(	)	+	\$
S	$S \to E\$$	$S \to E\$$			
E	$E \to FE'$	$E \to FE'$			
E'			$E' \to \varepsilon$	$E' \to +FE'$	$E' \to \varepsilon$
F	$F \to$ `num`	$F \to (E)$			

Utilisation. On construit l'AST en parcourant simultanément les lettres du mot et les non-terminaux de la frontière, et en appliquant la règle associée dans la table.

Exemple. Pour le mot $w = 1 + (2)\$$:



On commence par S et la première lettre de w est 1, donc on applique la règle $S \to E\$$. On n'a toujours pas consommé la lettre 1, donc on recommence et on applique $E \to FE'$, puis $F \to 1$. On essaye ensuite de consommer + et on applique $E' \to +FE$, etc.

Organisation du tableau

Titre 1. Forme $LL(1)$ *1ère grammaire* $\rightsquigarrow$ *2ème gram.*	2. Calcul *Théorème* *Exemple*	*Preuve*	3. Table *Table* *Utilisation*

Remarques

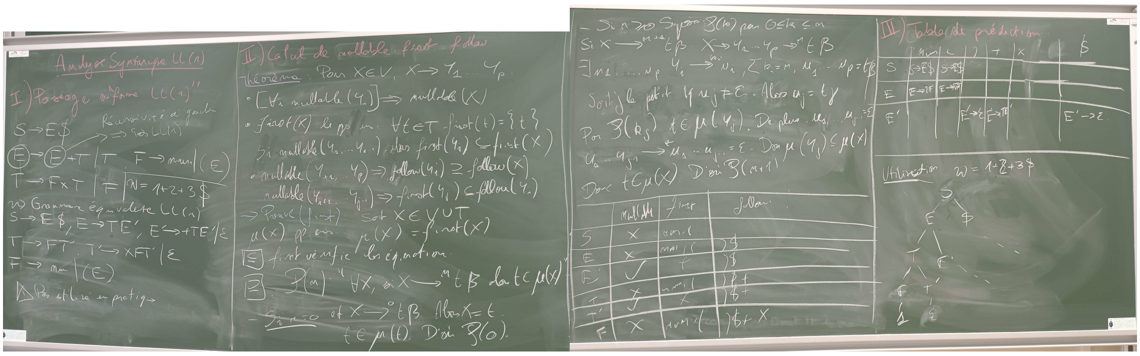
- Dans tout le développement je n'ai pas fait la différence entre les règles de la grammaire et la relation de dérivation, en écrivant toutes les flèches « $\to$ ». Il faut être prêt à savoir les distinguer en cas de question.
- La mention de $LR(k)$ peut donner lieu à des questions : il peut être intéressant de jeter un œil à la partie en question de [**SCH**].
- Quelques éléments de preuve de la première inclusion dans le théorème :
  - Si $t \in T$, $\text{first}(t) = \{u \mid \exists \beta, t \to^* u\beta\} = \{t\}$ car il est naturel de supposer qu'aucun terminal n'est préfixe d'un autre.
  - Si on a $X \to Y_1 \cdots Y_p$ une règle de la grammaire et $\text{nullable}(Y_1 \cdots Y_{i-1})$ pour un certain i, alors si $t \in \text{first}(Y_i)$ on a un β tel que $Y_i \to^* t\beta$. Ainsi on a :

$$\begin{aligned} X &\to Y_1 \cdots Y_{i-1} Y_i \cdots Y_p \\ &\to^* \varepsilon Y_i \cdots Y_p && \text{(car nullable}(Y_1 \cdots Y_{i-1})) \\ &\to^* t\, \underbrace{\beta Y_{i+1} \cdots Y_p}_{=\beta'} \to^* t\beta' && \text{(par hypothèse sur } t) \end{aligned}$$

    d'où l'inclusion.
- Ce développement est assez long, il faut être efficace tout au long, quitte à ne pas finir l'exemple de la fin ou la table, et éventuellement diminuer la grammaire en exemple (en supprimant les parenthèses). Il faut être prêt(e) à terminer la table si le jury le demande, cependant.
- La grammaire est en partie tirée de [**TIG**], il faut juste enlever les opérations inutiles (soustraction, division, multiplication).
- La preuve ne vient pas directement de [**SCH**], bien qu'elle en soit fortement inspirée. L'essentiel pour ce développement est de ne pas faire la preuve pour les mots entiers mais bien pour les variables uniquement. La propriété à prouver par récurrence est légèrement modifiée, mais elle guide en suite presque

complètement la preuve (il faut juste se souvenir de l'utilisation du lemme fondamental des grammaires).

Exemple de tableau



Références

[TIG] *Modern Compiler implementation in Java (Tiger Book)*, A.W. Appel

[SCH] *Compilation : Analyse lexicale et syntaxique*, Romain Legendre, François Schwarzentruber