Chapitre 4 Sémantiques des BNF et arbres d'analyse

Problèmatique

On considère le mini-langage **E** d'expressions arithmétiques sur ensemble de terminaux $V_T \overset{def}{=} \{\text{`I'}, \text{`-'}, \text{`('}, \text{`)'}\}$

$$\begin{array}{lcl} \mathbf{E} & ::= & \text{`I'} \\ & | & \text{`-'}\ \mathbf{E} \\ & | & \mathbf{E}\ \text{`-'}\ \mathbf{E} \\ & | & \text{`('}\ \mathbf{E}\ \text{`)'} \end{array}$$

Comment définir une sémantique aux mots dans ce langage ?
e.g. une fonction de $V_T^* \to \mathbb{Z}$ de domaine **E**.

Idée

1. Associer une structure *d'arbres d'analyse* à cette BNF avec une signature tq chaque alternative correspond à un constructeur distinct.
2. Définir la sémantique sur cette structure d'arbre.

Définition des arbres d'analyse

Déf. À la BNF de **E**, on associe la BNF d'arbres (sur $V_T \cup [1,4]$)

$$\begin{array}{lcl} \mathbf{E_P} & ::= & 1\ \text{`I'} \\ & | & 2\ \text{`-'}\ \mathbf{E_P} \\ & | & 3\ \mathbf{E_P}\ \text{`-'}\ \mathbf{E_P} \\ & | & 4\ \text{`('}\ \mathbf{E_P}\ \text{`)'} \end{array}$$

Déf. Pour $t \in (V_T \cup [1,4])^*$, on note $\chi(t)$ le mot de V_T^* obtenu en "*effaçant*" de t les symboles de $[1,4]$.

Exo 4.1 Calculer $\chi(t)$ avec t valant "`2 - 4 ( 3 1 I - 1 I )`".
Définir χ par induction sur les mots.

Déf. Si $t \in \mathbf{E_P}$, on dit que t est un **arbre d'analyse** de $\chi(t)$.
Exo 4.2† Donner l'ensemble des arbres d'analyse du mot `I--I-I`

Exo 4.3 Pour $L \subseteq (V_T \cup [1,4])^*$, on pose $\chi(L) = \{\chi(t) \mid t \in L\}$.
Montrer que $\mathbf{E} = \chi(\mathbf{E_P})$.
(C-à-d $w \in \mathbf{E}$ ssi il existe $t \in \mathbf{E_P}$ avec t arbre d'analyse de w).

Une sémantique sur la BNF de **E**

$$\begin{array}{llcll} 1 & \mathbf{E}{\uparrow}r & ::= & \text{`I'} & r := 1 \\ 2 & & | & \text{`-'}\ \mathbf{E}{\uparrow}r_1 & r := -r_1 \\ 3 & & | & \mathbf{E}{\uparrow}r_1\ \text{`-'}\ \mathbf{E}{\uparrow}r_2 & r := r_1 - r_2 \\ 4 & & | & \text{`('}\ \mathbf{E}{\uparrow}r\ \text{`)'} & \end{array}$$

Définition Un mot w admet la sémantique r ssi il existe un arbre d'analyse t de w tq $r = [\![t]\!]$, où $[\![t]\!]$ est calculée avec la BNF

$$\begin{array}{lcll} \mathbf{E_P}{\uparrow}r & ::= & 1\ \text{`I'} & r := 1 \\ & | & 2\ \text{`-'}\ \mathbf{E_P}{\uparrow}r_1 & r := -r_1 \\ & | & 3\ \mathbf{E_P}{\uparrow}r_1\ \text{`-'}\ \mathbf{E_P}{\uparrow}r_2 & r := r_1 - r_2 \\ & | & 4\ \text{`('}\ \mathbf{E_P}{\uparrow}r\ \text{`)'} & \end{array}$$

Ambiguïté et non-déterminisme

Exo 4.4† Pour chacun des mots suivants, dessiner l'ensemble de ses arbres d'analyse puis la propagation d'attributs sur ces arbres d'analyse.

1. `I-I-I`
2. `(I-I)-I`
3. `I-(I-I)`

Définition Une BNF engendrant deux arbres d'analyses distincts du même mot w est dite *ambiguë* : ce mot a *éventuellement* plusieurs sémantiques (donc une sémantique non-déterministe).

NB La BNF associée à un AFD (Automate Fini Déterministe) est non-ambiguë.

Idée de la suite rendre sémantiques déterministes avec *règles de priorités* qui éliminent les arbres d'analyses indésirables.

Élimination des ambiguïtés pour $L = \{a^n b^m \mid m \geq n\}$

Exo 4.5† Donner tous les arbres d'analyse du mot *abbb* pour la BNF du langage L suivante

$$\mathbf{L} ::= a\ \mathbf{L}\ b \mid \mathbf{L}\ b \mid \varepsilon$$

Exo 4.6† Donner tous les arbres d'analyse du mot *abbb* pour la BNF du langage L suivante

$$\mathbf{L} ::= \mathbf{L}\ b \mid \mathbf{A} \qquad \mathbf{A} ::= a\ \mathbf{A}\ b \mid \varepsilon$$

Montrer que cette BNF est non-ambiguë en dessinant l'unique arbre d'analyse de tout mot de L écrit sous la forme $a^n b^{n+p}$.

Exo 4.7† Même exercice que le précédent pour cette BNF alternative du langage L

$$\mathbf{L} ::= a\ \mathbf{L}\ b \mid \mathbf{B} \qquad \mathbf{B} ::= b\ \mathbf{B} \mid \varepsilon$$

Chapitre 4 Sémantiques des BNF et arbres d'analyse

Priorité des opérateurs

Sur expressions arithmétiques $20 - 2 \times 3$ pourrait *à priori* représenter "$(20 - 2) \times 3$" ou "$20 - (2 \times 3)$".

Par convention

$$x - y \times z \stackrel{def}{=} x - (y \times z) \quad \text{et} \quad x \times y - z \stackrel{def}{=} (x \times y) - z$$

Formellement "$\times$" de *priorité plus élévée* que "$-$".
En pratique, utilise *niveaux de priorité inversement ordonnés*.
(i.e. un petit nombre correspond à une priorité élévée !)

Exemple pour le langage C multiplication `*` et division `/` de niveau 5 versus soustraction `-` et addition `+` de niveau 6.

ATTENTION, priorité aussi pour opérateurs unaires !
Exo 4.8† parenthésage explicite de "$-\ 1 \mid 2\ \&\ 3$" ?

Associativité des opérateurs

Pb du parenthésage implicite pour opérateurs non associatifs :
$(5-3)-2 \neq 5-(3-2)$ et $(2^3)^2 \neq 2^{(3^2)}$

associativité à gauche $x-y-z \stackrel{def}{=} (x-y)-z$.
le cas de tous opérateurs arithmétiques sauf puissance ci-dessus.

associativité à droite $x^{y^z} \stackrel{def}{=} x^{(y^z)}$.

ATTENTION, associativité en fait définie par niveau de priorité
$x+y-z=(x+y)-z$ et $x-y+z=(x-y)+z$

Retour sur l'exemple

$$
\begin{array}{llll}
1 & \mathbf{E}\uparrow r ::= & \text{`I'} & r := 1 \\
2 & \quad | & \text{`-'}\ \mathbf{E}\uparrow r_1 & r := -r_1 \\
3 & \quad | & \mathbf{E}\uparrow r_1\ \text{`-'}\ \mathbf{E}\uparrow r_2 & r := r_1 - r_2 \\
4 & \quad | & \text{`('}\ \mathbf{E}\uparrow r\ \text{`)'} &
\end{array}
$$

Exo 4.9† Avec ce système d'attributs, quel arbre d'analyse associer à "`I--I-I`" pour que le résultat corresponde à la convention usuelle.

Encore un exemple

On considère la BNF de profils $\mathbf{S}\uparrow\mathbb{Z}$ et $\mathbf{E}\downarrow\mathbb{N}\uparrow\mathbb{Z}$:

$$
\begin{array}{llll}
(1) & \mathbf{S}\uparrow n ::= & \mathbf{E}\downarrow 1\uparrow n & \\
(2) & \mathbf{E}\downarrow p\uparrow n ::= & \text{`x'} & n := 3 \\
(3) & \quad | & \text{`('}\ \mathbf{E}\downarrow p\uparrow n\ \text{`)'} & \\
(4) & \quad | & \mathbf{E}\downarrow(p{+}1)\uparrow n_0\ \text{`\#'} & n := n_0 \times 2^p \\
(5) & \quad | & \mathbf{E}\downarrow p\uparrow n_1\ \text{`-'}\ \mathbf{E}\downarrow p\uparrow n_2 & n := n_1 - n_2
\end{array}
$$

Exo 4.10† La BNF étant ambiguë, dessiner tous les arbres possibles du mot '`x # - x #`', avec la propagation d'attributs.

Exo 4.11† On considère que '`#`' est prioritaire sur '`-`' (qui est associatif à gauche). Quel est le résultat du calcul ?

Chapitre 4 Sémantiques des BNF et arbres d'analyse

Problématique

Transformer "*spécification*" d'un analyseur
via BNF attribuée + priorités
en une BNF attribuée non-ambiguë "**équivalente**".

"**équivalente**" =
m̂ syntaxe (langage reconnu)
et m̂ sémantique associée (priorités + attributs)

Motivation ramener la théorie à l'étude des BNF non-ambiguës.

Difficultés

- Il n'existe pas forcément une BNF non-ambiguë équivalente.
- Le problème est indécidable : on applique des "patrons" à partir d'exemples types $\rightsquigarrow$ "heuristiques".

Encodage des priorités d'une BNF d'expressions

Introduire un non-terminal $\mathbf{E_n}$ *par* niveau de priorité n via
$\mathbf{E_n} \stackrel{def}{=}$ ensemble des expr tq tt opérateur de priorité $> n$
apparaît uniquement dans une sous-expr de forme "(e)"

Pour n maximal, $\mathbf{E_n} \equiv$ ensemble des expressions.

Construction des équations

- pour tout $n > 0$, $\mathbf{E_n} \supseteq \mathbf{E_{n-1}}$ (ce qui induit $\mathbf{E_n} ::= \mathbf{E_{n-1}} \mid \ldots$)
- pour n maximal, on a alternative $\mathbf{E_0} \supseteq (\ \mathbf{E_n}\ )$.
- Tout op binaire $\spadesuit$ de niveau n induit une des 3 alternatives
  si n associatif à gauche, $\mathbf{E_n} \supseteq \mathbf{E_n} \spadesuit \mathbf{E_{n-1}}$
  si n associatif à droite, $\mathbf{E_n} \supseteq \mathbf{E_{n-1}} \spadesuit \mathbf{E_n}$
  si n non-associatif, $\mathbf{E_n} \supseteq \mathbf{E_{n-1}} \spadesuit \mathbf{E_{n-1}}$

NB Associativité fixée par niveau de priorité !

Exemples

Exo 4.12† Appliquer cette méthode sur les BNF suivantes.
On se limitera à se convaincre "à la main" de la non-ambiguïté sur quelques exemples.

1. l'exemple de l'introduction.
2. la BNF de la section 6.1 du sujet de TP.

Langage algébrique intrinsèquement ambigu

Soit $A \stackrel{def}{=} \{a^n b^n c^k | n, k \text{ de } \mathbb{N}\}$ et $B \stackrel{def}{=} \{a^k b^n c^n | n, k \text{ de } \mathbb{N}\}$.

Exo 4.13† Trouver une BNF pour le langage $A \cup B$. Montrer que cette BNF est ambiguë.

Thm Toute BNF qui engendre le langage $A \cup B$ est ambiguë.

Raison "intuitive"
Le langage $A \cap B = \{a^n b^n c^n | n \in \mathbb{N}\}$ n'est pas algébrique (par lemme de l'étoile des langages algébriques).
Un mot de $A \cap B$ ne peut donc pas avoir un unique arbre d'analyse.

Indécidabilité de la détection des ambiguïtés

Exemple sur $V_T = \{a, b, 1, 2\}$ et $V_N = \{S, U, V\}$

$$\begin{array}{lcl} S & ::= & U \mid V \\ U & ::= & 1\ U\ a \mid 2\ U\ a\ b\ a \quad \mid 1\ a \mid 2\ a\ b\ a \\ V & ::= & 1\ V\ a\ a \mid 2\ V\ b \quad \mid 1\ a\ a \mid 2\ b \end{array}$$

Exo 4.14 Cette BNF est-elle ambiguë ?

...

Généralisable pour param $(n, (u_i, v_i)_{i \in 1..n})$ tq $u_i, v_i \in \{a, b\}^* \backslash \{\epsilon\}$ (sur $V_T = \{a, b, 1, \ldots, n\}$)

$$\begin{array}{lcl} U & ::= & 1\ U\ u_1 \mid \ldots \mid n\ U\ u_n \quad \mid 1\ u_1 \mid \ldots \mid n\ u_n \\ V & ::= & 1\ V\ v_1 \mid \ldots \mid n\ V\ v_n \quad \mid 1\ v_1 \mid \ldots \mid n\ v_n \end{array}$$

Problème "$U \cap V = \emptyset$?" indécidable !
(i.e : on sait montrer qu'il n'existe pas d'algorithme).
Cf *"Problème de correspondance de Post"* sur wikipédia.

Problèmes qu'il reste à examiner...

Étant donnée une BNF G qcq,

1. comment gérer le fait qu'on ne sait pas détecter les ambiguïtés éventuelles de G ?
2. comment avoir une analyse syntaxique "efficace" ?

Solutions connues depuis les années 1970 et outillées (yacc/bison, ANTLR, etc) via théorie des *grammaires hors-contextes* :

1. définition de familles de BNF non-ambigües avec parsing efficace (linéaire).
2. "méthodes" pour tenter de ramener G à une telle famille.