Analyse syntaxique (descendante)

Plan et objectifs

- Initiation à l'analyse syntaxique
- Problèmes : étant donnée une grammaire algébrique
  - déterminer si un mot appartient au langage défini par la grammaire
  - construire l'arbre syntaxique d'un mot appartenant au langage défini par la grammaire
- Plan
  - Rappels sur les grammaires algébriques
  - Descente récursive
  - Définition des premiers et suivants

Rappels : grammaire algébrique

Definition (Grammaire algébrique)

Une grammaire algébrique est un quadruplet $G = (N, T, \rightarrow, S)$ tel que

- N est l'ensemble des symboles non terminaux.
- T est l'ensemble des symboles terminaux.
- $\rightarrow$ est une relation de N vers $(N \cup T)^*$ telle que pour chaque élément A de N il n'y ait qu'un nombre fini de mots α appartenant à $(N \cup T)^*$ tels que $A \rightarrow \alpha$.
- S est un élément de N, appelé l'axiome de la grammaire.

Definition (Réécriture)

La relation de réécriture $\rightarrowtail$ est définie sur $(N \cup T)^*$. α se réécrit en β, qu'on note $\alpha \rightarrowtail \beta$, si $\alpha = \alpha_1 A \alpha_2$ et $\beta = \alpha_1 \gamma \alpha_2$ où

$\alpha_1,\ \alpha_2,\ \gamma \in (N \cup T)^*$, $A \in N$ et $A \rightarrow \gamma$ est une règle de G.

Definition (Derivation non stricte et stricte)

La relation de dérivation est la fermeture réflexive transitive de la relation $\rightarrowtail$, on la note $\overset{*}{\rightarrowtail}$.

α se dérive en β que l'on note $\alpha \overset{*}{\rightarrowtail} \beta$, si on passe de α à β par une suite finie de réécriture.

La relation de dérivation stricte est la fermeture transitive de la relation $\rightarrowtail$, on la note $\overset{+}{\rightarrowtail}$.
α se dérive strictement en β que l'on note $\alpha \overset{+}{\rightarrowtail} \beta$, si on passe de α à β par une suite finie de réécriture, de longueur non nulle.

Réécriture gauche, dérivation gauche

Definition (Réécriture à gauche)

La relation de réécriture à gauche $\overset{g}{\rightarrowtail}$ est définie sur $(N \cup T)^*$. α se réécrit à gauche en β, qu'on note $\alpha \overset{g}{\rightarrowtail} \beta$, si $\alpha = wA\alpha_1$ et $\beta = w\gamma\alpha_1$ où

$\alpha_1,\ \gamma \in (N \cup T)^*$, $w \in T^*$, $A \in N$ et $A \rightarrow \gamma$ est une règle de G.

Remarque : la réécriture à gauche consiste à réécrire le non terminal le plus à gauche.

Definition (Dérivation à gauche)

La relation de dérivation à gauche est la fermeture réflexive transitive de la relation de réécriture à gauche.

Arbre syntaxique

Definition

Soit $G = (N,\ T,\ \rightarrow,\ S)$, un arbre syntaxique pour la grammaire G est un arbre étiqueté par les éléments de $N \cup T \cup \{\varepsilon\}$ qui satisfait les conditions suivantes :

- la racine de l'arbre est étiquetée par S, l'axiome de G
- chaque nœud interne est étiqueté par un élément de N. Chaque feuille est étiquetée par un élément de T ou par ε.
- pour tout nœud interne, si son étiquette est un non-terminal A et si ses descendants immédiats sont les nœuds $n_1, \ldots, n_k$ ayant respectivement pour étiquettes $X_1, \ldots, X_k$ alors $A \rightarrow X_1 \ldots X_k$ doit être une règle de production de G.
- si un nœud est étiqueté par ε, alors ce nœud est le seul descendant immédiat de son prédécesseur (cette dernière règle évite l'introduction d'instances inutiles de ε dans l'arbre syntaxique).

Definition (Mot généré par un arbre syntaxique)

Un mot généré par un arbre syntaxique est obtenu par la concaténation des feuilles de l'arbre prises de gauche à droite.

Langage engendré par une grammaire algébrique

Definition

Soit $G = (N,\ T,\ \rightarrow,\ X)$ une grammaire algébrique, le langage $L(G)$ engendré par G est

1. $L(G) = \{w; w \in T^* \text{ et } X \overset{*}{\rightarrowtail} w\}$
2. $L(G) = \{w; w \in T^* \text{ et } w \text{ est un mot généré par un arbre syntaxique}\}$

Definition (Grammaire ambiguë)

Une grammaire G est dite **ambiguë** s'il existe un mot de $L(G)$ ayant plusieurs arbres syntaxiques.

Exemples : dérivations et arbres syntaxiques

Example

Soit la grammaire $G = (\{S,\ T\},\ \{a,\ b,\ c\},\ \longrightarrow,\ S)$, définie par les règles
$S \longrightarrow aTb \mid c$
$T \longrightarrow cSS \mid S$

- $S \rightarrowtail aTb \rightarrowtail acSSb \rightarrowtail accSb \rightarrowtail accaSbb \rightarrowtail accacbb$
  est une dérivation à gauche (à chaque étape le non terminal le plus à gauche est réécrit)
- $S \rightarrowtail aTb \rightarrowtail acSSb \rightarrowtail acSaTbb \rightarrowtail accaTbb \rightarrowtail accaSbb \rightarrowtail accacbb$ est une dérivation quelconque

le mot *accacbb* a deux dérivations différentes et un arbre syntaxique.

Représentation des arbres syntaxiques forme postfixée

Definition

Soit une grammaire $G = (N,\ T,\ \rightarrow,\ S)$ une grammaire algébrique, on ajoute un numéro de règle à chaque règle de la grammaire, on considére les représentation postfixée des arbres en faisant figurer les éléments de T et les numéros des règles.

Example

Soit la grammaire $G = (\{S,\ T\},\ \{a,\ b,\ c\},\ \rightarrow,\ S)$, définie par les règles
$S \rightarrow aTb\ |1|\ c|2|$
$T \rightarrow cSS\ |3|\ S\ |4|$
l'arbre syntaxique du mot $accacbb$ s'écrit $acc2ac24b13b1$ sous forme postfixée

Avant de commencer l'analyse syntaxique d'une grammaire :
s'assurer que la grammaire est réduite (sinon la réduire)

Analyse syntaxique descendante

- tentative de construire l'arbre syntaxique d'un mot du haut (c'est à dire de la racine (ou de l'axiome)) vers le bas (les feuilles (les nœuds étiquetés par les terminaux))
- tentative de trouver une dérivation à gauche pour le mot donné
- tentative de construire un arbre syntaxique pour le mot donné en construisant les nœuds de l'arbre en préordre
- détermination de classes de grammaires pour lesquelles il n'est pas utile de faire des retours arrières (les grammaires LL(1))

Analyse descendante : exemples introductifs

On essaie de construire un arbre syntaxique du mot donné de haut en bas, c'est à dire en partant de l'axiome de la grammaire et en parcourant le mot donné de gauche à droite.

Example (1)

$G = (\{S,\ T\},\ \{a,\ b,\ c,\ d\},\ \rightarrow,\ S)$ une grammaire définie par
$S \rightarrow aSbT \mid cT \mid d$
$T \rightarrow aT \mid bS \mid c$
on considère le mot $w = accbbadbc$

Example (2)

$G = (\{S,\ A\},\ \{a,\ b,\ c,\ d\},\ \rightarrow,\ S)$ une grammaire définie par
$S \rightarrow aAb$
$A \rightarrow cd \mid c$
on considère le mot $w = acb$

Example (3)

$G = (\{S\},\ \{a,\ b,\ c,\ d\},\ \rightarrow,\ S)$ une grammaire définie par
$S \rightarrow aSb \mid aSc \mid d$
on considère le mot $w = aaaaadbcbcb$

Analyse descendante : exemples introductifs (suite)

Example (4)

$G = (\{E,\ T,\ F,\ E',\ T'\},\ \{+,\ -,\ *,\ (,\ ),\ nb\ \},\ \},\ \rightarrow,\ E)$ une grammaire définie par
$E \rightarrow TE'$
$E' \rightarrow +TE' \mid -TE' \mid \varepsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT' \mid /FT' \mid \varepsilon$
$F \rightarrow (E) \mid nb$
on considère le mot $w = 3 * 4 + 10 * (5 + 11)/34 + 12$

Descente récursive

- écriture d'un programme qui étant une grammaire et un mot donné, détermine si le mot est engendré par la grammaire et si c'est le cas construit son arbre syntaxique
- à chaque non terminal de la grammaire correspond une procédure
- pour les terminaux on associe une procédure qui teste si le terminal lu est celui attendu
- cette démarche est utilisable pour les grammaires $LL(1)$ et adaptable aux grammaires $LL(k)$ $(k > 1)$

Mise en œuvre

- Soit $G = (N,\ T,\ \rightarrow,\ S)$ une grammaire

  Données : un mot α de T^* suivi du caractère \$, appelé marqueur de fin

  Résultats : un booléen *erreur* et une chaîne de caractères *res*,

  A la fin du programme si *erreur* $=$ *vrai* alors α appartient à $L(G)$ et *res* est l'arbre syntaxique de α
- Variables du programme : *res* : chaîne de caractères, *erreur* : booléen, *y* : le caractère courant du mot d'entrée α
- Programme principal :

  erreur $\leftarrow$ *faux*;
  res $\leftarrow \varepsilon$;
  lire(*y*); –lecture du premier caractère de α qui est affecté à *y*

  Ana-S ; –appel de la procédure *Ana-S* correspondant à l'axiome de la grammaire

  si *erreur* alors "mot non engendré par la grammaire"
  sinon si $y =$ \$ alors "mot engendré par la grammaire" , *res*
  sinon "mot non engendré par la grammaire"
  fsi
  fsi

- procédure associée à un terminal de G:

  Procédure Ana(x : T) ;
  si $\neg$*erreur* alors
  si $y = x$ alors
  debut
  $res \leftarrow res \oplus x$; *lire*(y); $-\oplus$ concaténe deux chaînes de caractères
  fin
  sinon *erreur* $\leftarrow$ *vrai*
  fsi
  fsi

  Procédure qui teste si le caractère en argument est égal au caractère courant du mot donné.
- à chaque non terminal X, on associe une procédure *Ana-X* (voir exemple)

Exemple

Soit la grammaire $G = (\{S,\ A\},\ \{a,\ b,\ c,\ d\},\ \rightarrow,\ S)$ où
$S \rightarrow aAb\ |1|$
$A \rightarrow cd\ |2|c\ |3|$

- procédure pour le non terminal S : *Ana-S*

  procédure *Ana-S*;
  si $\neg$*erreur* alors
  debut
  Ana(a); *Ana-A*; *Ana*(b); *res* $\leftarrow$ *res* $\oplus$ 1
  fin

- procédure pour le non terminal A : *Ana-A*

  procédure *Ana-A*;
  si $\neg$*erreur* alors
  debut
  Ana(c);
  si $y = d$ alors
  debut *Ana*(d) ; *res* $\leftarrow$ *res* $\oplus$ 2 fin
  sinon si $y = b$ alors
  res $\leftarrow$ *res* $\oplus$ 3
  sinon *erreur* $\leftarrow$ *vrai*
  fsi
  fsi
  fin

Exercice : exécuter les procédures pour les mots *acdb* et *acb*

"l'approche descente récursive marche-t-elle dans tous les cas?"

Problèmes :

- Règles de la forme $X \longrightarrow X_1\alpha_1$ et $X \longrightarrow X_2\alpha_2$
  quelle est la règle à appliquer : idée : connaître par quelle lettre peut commencer X_1 et X_2
  Idée : notion de *Premier*
  et si $X_1 \overset{*}{\longmapsto} \varepsilon$?
- Règles de la forme $X \longrightarrow \alpha$ et $X \longrightarrow \varepsilon$
  quelle règle à appliquer : idée : connaître par quelle lettre peut commencer α (notion de *Premier*) et par quelle lettre peut être suivi X dans une dérivation (notion de *Suivant*)

Non terminaux produisant le vide

Definition

Soit $G = (N,\ T,\ \rightarrow,\ S)$ une grammaire algébrique et $X \in N$, $Vide = \{X\ ;\ x \in N \text{ et } X \overset{*}{\rightarrowtail} \varepsilon\}$.
Vide est l'ensemble des terminaux qui produisent ε

Example

$G = (\{E,\ T,\ F,\ E',\ T'\},\ \{+,\ -,\ *,\ (,\ ),\ nb\ \},\ \},\ \rightarrow,\ E)$ une grammaire définie par
$E \rightarrow TE'$
$E' \rightarrow +TE' \mid -TE' \mid \varepsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT' \mid /FT' \mid \varepsilon$
$F \rightarrow (E) \mid nb$

$Vide = \{E',\ T'\}$

Définition des premiers

Definition (Premiers)

Soit $G = (N,\ T,\ \rightarrow,\ S)$ une grammaire algébrique et $\alpha \in (N \cup T)^*$,
$Premier(\alpha) = \{x\ ;\ x \in T \text{ et } \alpha \overset{*}{\rightarrowtail} xw\}$.

Calcul des premiers pour les non terminaux

1. pour tout non terminal X de la grammaire G, initialiser $Premier(X)$ à l'ensemble vide fpour
2. pour toute règle de la forme $X \rightarrow Y_1 \ldots Y_n$
   si $Y_1 \in T$ alors -ajouter Y_1 à $Premier(X)$
   sinon -ajouter $Premier(Y_1)$ à $Premier(X)$;
   -pour tout $j \in \{2, \ldots, n\}$ tq $\forall i \in \{1, \ldots, j\text{-}1\}$ tq $Y_i \in Vide$
   ajouter $Premier(Y_j)$ à $Premier(X)$
   fpour
   fsi

   fpour
3. recommencer l'étape 2 jusqu'à ce qu'il n'y ait plus de changement

Exemple(1) : calcul des premiers

Example (Calcul des premiers (1))

$G = (\{E,\ T,\ F,\ E',\ T'\},\ \{+,\ -,\ *,\ (,\ ),\ nb\ \},\ \ \},\ \ \rightarrow,\ E)$ la grammaire définie par

$E \rightarrow TE'$
$E' \rightarrow +TE' \mid -TE' \mid \varepsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT' \mid /FT' \mid \varepsilon$
$F \rightarrow (E) \mid nb$

on a $Vide = \{E',\ T'\ \}$

$Premier(E)$	$Premier(E')$	$Premier(T)$	$Premier(T')$	$Premier(F)$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\emptyset$	$\{+,\ -\}$	$\emptyset$	$\{*,\ /\}$	$\{(,\ nb\}$
$\emptyset$	$\{+,\ -\}$	$\{(,\ nb\}$	$\{*,\ /\}$	$\{(,\ nb\}$
$\{(,\ nb\}$	$\{+,\ -\}$	$\{(,\ nb\}$	$\{*,\ /\}$	$\{(,\ nb\}$
idem	*idem*	*idem*	*idem*	*idem*

Exemple(2) : calcul des premiers

Example (Calcul des premiers (1))

$G = (\{S,\ A,\ B,\ C\},\ \{a,\ b,\ c,\ d,\ e\},\ \rightarrow,\ S\})$ une grammaire définie par

$S \rightarrow ABCe$
$A \rightarrow aA \mid \varepsilon$
$B \rightarrow bB \mid cB \mid \varepsilon$
$C \rightarrow de \mid da \mid dA$

on a $Vide = \{A,\ B\}$

$Premier(S)$	$Premier(A)$	$Premier(B)$	$Premier(C)$
$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\emptyset$	$\{a\}$	$\{b,\ c\}$	$\{d\}$
$\{a,\ b,\ c,\ d\}$	$\{a\}$	$\{b,\ c\}$	$\{d\}$
idem	*idem*	*idem*	*idem*

Extension : définition des premiers pour les éléments de $(N \cup T)^*$

Definition

Soit $\alpha \in (N \cup T)^*$, on étend la fonction $premier(\alpha)$ aux éléments de $(N \cup T)^*$ de la façon suivante :

- $premier(a\beta) = \{a\}$ si $a \in T$
- $premier(X)$ est déjà défini pour $X \in N$
- $premier(X\beta) =$ si $X \notin Vide$ alors $premier(X)$ sinon $premier(X) \cup premier(\beta)$ fsi où $\beta \neq \varepsilon$

Définition des suivants

Definition (Définition des suivants)

Soit la grammaire $G = (N,\ T,\ \rightarrow,\ S)$ et soit $A \in N$, $Suivant(A)$ est l'ensemble des éléments a de $T \cup \{\$\}$ qui peuvent apparaître immédiatement après A dans une dérivation (c'est à dire les éléments a tel que $S \overset{*}{\rightarrowtail} \alpha A a \beta$)

Calcul des suivants des symboles non terminaux

1. initialiser $Suivant(S)$ à $\{\$\}$;
   pour tout non terminal $X \neq S$ de la grammaire G, initialiser $Suivant(X)$ à l'ensemble vide
   fpour
2. pour chaque règle de la forme $A \rightarrow \alpha B \beta$ où $B \in N$ ajouter $Premier(\beta)$ à $Suivant(B)$
3. pour chaque règle $A \rightarrow \alpha B$, ajouter $Suivant(A)$ à $Suivant(B)$
4. pour chaque règle $A \rightarrow \alpha B \beta$ tel que $\beta \overset{*}{\rightarrowtail} \varepsilon$ ajouter $Suivant(A)$ à $Suivant(B)$

recommencer à partir de l'étape 3 jusqu'à ce que l'on n'ajoute rien de nouveau dans les ensembles *Suivant*

Exemple de calcul de suivants

Example (1)

$G = (\{E,\ T,\ F,\ E',\ T'\},\ \{+,\ -,\ *,\ (,\ ),\ nb\ \},\ \},\ \rightarrow,\ E)$ la grammaire définie par

$E \rightarrow TE'$
$E' \rightarrow +TE' \mid -TE' \mid \varepsilon$
$T \rightarrow FT'$
$T' \rightarrow *FT' \mid /FT' \mid \varepsilon$
$F \rightarrow (E) \mid nb$

$Vide = \{E',\ T'\ \}$

$Premier(E) = \{(,\ nb\}$, $Premier(E') = \{+,\ -\}$, $Premier(T) = \{(,\ nb\}$, $Premier(T') = \{*,\ /\}$ et $Premier(F) = \{(,\ nb\}$

$Suivant(E)$	$Suivant(E')$	$Suivant(T)$	$Suivant(T')$	$Suivant(F)$
\$	$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
\$)	\$	+ − \$	+ − \$	* / + − \$
\$)	\$)	+ − \$)	+ − \$)	* / + − \$)

Conditions de terminaison des procédures de la descente récursive

Théorème

Les procédures d'analyse syntaxique d'une grammaire G s'arrêtent pour toute donnée α si et seulement si la grammaire n'est pas récursive gauche.

C'est à dire s'il n'existe pas de non terminal A vérifiant $A \overset{+}{\rightarrowtail} A\beta$.

Idée de démonstration : pour que les procédures d'analyse syntaxique ne terminent pas, il est nécessaire que le texte en entrée ne soit plus consommé à partir d'un certain temps. Cela ne peut se produire que dans le cas où l'on a des règles de la forme suivante :
$A_1 \rightarrow A_2\alpha_2$, $A_2 \rightarrow A_3\alpha_3$, ..., $A_k \rightarrow A_{k+1}\alpha_{k+1}$, ...
où les A_i sont des non terminaux.
Comme les A_i sont finis, il existe un k suffisamment grand tel que $A \overset{*}{\rightarrowtail} A\beta$

Remarque : cette condition est une condition générale nécessaire à toutes les démarches concernant l'analyse descendante.