TD5 - Grammaires de type 2 et algorithme CKY

Objectifs

- Notions de langage algébrique
- Transformer une grammaire sous forme normalisée de Chomsky
- Tester l'appartenance d'un mot à une grammaire sous forme normalisée de Chomsky
  - Algorithme CKY
- Application au traitement des langages naturels

Rappels et notations

Les termes *algébrique* et *hors-contexte* sont synonymes quand il s'agit de grammaire ou de langage. Une grammaire $G =< T, N, S, P >$ est algébrique quand toutes les règles de P sont de la forme :

- $X \to a$ où $a \in T$ et $X \in N$
- $X \to Y$ où $Y \in (N \cup T)^*$ et $X \in N$

Une grammaire $G =< T, N, S, P >$ hors-contexte **qui ne produit pas** ε est dite sous forme normale de Chomsky si et seulement si toutes les règles de P sont de la forme :

- $A \to a$ où $a \in T$
- $A \to BC$ où $B, C \in N$

Le mot vide, ε, peut également être noté λ (c'est notamment le cas en JFLAP).

Procédure de mise sous forme normalisée de Chomsky

1. Remplacer tous les terminaux x en partie droite des règles par des non terminaux en ajoutant des règles de la forme $X \to x$
2. Transformer les parties droites des règles comme suit.

   $X \to YZW$ est remplacée par deux régles : $X \to YV$ et $V \to ZW$

3. Transformer les parties droites des règles comme suit :

   $X \to Y$ remplacée par $X \to WZ$ pour chaque W et Z tels que $Y \to WZ$

Algorithme CKY

On utilisera l'algorithme de Coke, Younger et Kasami (CKY) pour tester si un mot w est reconnu par une grammaire sous forme normale de Chomsky.

Données : $G = < T, N, P, S >$ et w un mot de T^*
$n \leftarrow |w|$
$v \leftarrow \text{matrix}(n, n)$
pour $i = 1$ ***à*** n **faire**
  $v[i, 1] =$
  $\{A$ tel que A est le membre gauche d'une règle $A \rightarrow a$ et a est le i-ème symbole du mot w $\}$
pour $j = 2$ ***à*** n **faire**
  pour $i = 1$ ***à*** $n - j + 1$ **faire**
    $v[i, j] \leftarrow \emptyset$
    pour $k = 1$ ***à*** $j - 1$ **faire**
      $v[i, j] = v[i, j] \cup$
      $\{A$ tel que A est le membre gauche d'une règle $A \rightarrow BC$ avec $B \in v[i, k]$ et $C \in v[i + k, j - k]\}$

Algorithme 1 : Algorithm CKY

Le mot w est reconnu par la grammaire ssi $S \in v[1, n]$.

Exercice 1. Reconnaissance d'un mot par l'algorithme CKY

Soit $G = < T, N, S, P >$ une grammaire avec

$$\begin{aligned} T &= \{a, b\} \\ N &= \{S, A, B\} \\ S &= S \\ P &= \left\{ \begin{array}{l} S \rightarrow bA \mid aB \\ A \rightarrow a \mid aS \mid bAA \\ B \rightarrow b \mid bS \mid aBB \end{array} \right\} \end{aligned}$$

Question 1. Est-ce que cette grammaire est algébrique ? rationnelle ?

Question 2. Mettre cette grammaire sous forme normalisée de Chomsky.

Question 3. En utilisant l'algorithme CKY, montrer que le mot *aabbab* appartient au langage engendré par la grammaire.

Exercice 2. Des propositions très relatives

On s'intéresse aux constructions de phrases avec des propositions relatives.

Soit $G =< T, N, S, P >$ une grammaire avec

$$T = \{\textbf{que}, \textbf{qui}, \textbf{regarde}, \textbf{regardent}, \textbf{mange}, \textbf{mangent}, \textbf{dort}, \textbf{dorment}, \textbf{tombe}, \textbf{tombent}, \\ \textbf{une}, \textbf{un}, \textbf{la}, \textbf{le}, \textbf{des}, \textbf{les}, \textbf{pommes}, \textbf{pomme}, \textbf{femme}, \textbf{femmes}, \textbf{Pierre}, \textbf{Marie}\}$$

$$N = \{\text{s}, \text{sn}, \text{reln}, \text{rela}, \text{sv}, \text{proa}, \text{pron}, \text{vt}, \text{vi}, \text{det}, \text{n}, \text{np}\}$$

$$S = \text{s}$$

$$P = \left\{ \begin{array}{rl} \text{s} \rightarrow & \text{sn sv} \\ \text{sn} \rightarrow & \text{det n reln} \mid \text{det n rela} \mid \text{np reln} \mid \text{np rela} \mid \text{det n} \mid \text{np} \\ \text{reln} \rightarrow & \text{pron sv} \\ \text{rela} \rightarrow & \text{proa sn vt} \\ \text{sv} \rightarrow & \text{vi}|\text{vt sn} \\ \text{proa} \rightarrow & \textbf{que} \\ \text{pron} \rightarrow & \textbf{qui} \\ \text{vt} \rightarrow & \textbf{regarde} \mid \textbf{regardent} \mid \textbf{mange} \mid \textbf{mangent} \\ \text{vi} \rightarrow & \textbf{dort} \mid \textbf{dorment} \mid \textbf{tombe} \mid \textbf{tombent} \\ \text{det} \rightarrow & \textbf{une} \mid \textbf{un} \mid \textbf{la} \mid \textbf{le} \mid \textbf{des} \mid \textbf{les} \\ \text{n} \rightarrow & \textbf{pommes} \mid \textbf{pomme} \mid \textbf{femme} \mid \textbf{femmes} \\ \text{np} \rightarrow & \textbf{Pierre} \mid \textbf{Marie} \end{array} \right\}$$

Légende :

- reln $\Longleftrightarrow$ proposition relative nominative
- rela $\Longleftrightarrow$ proposition relative accusative
- proa $\Longleftrightarrow$ pronom relatif accusatif
- pron $\Longleftrightarrow$ pronom relatif nominatif
- vi $\Longleftrightarrow$ verbe intransitif
- vt $\Longleftrightarrow$ verbe transitif
- det $\Longleftrightarrow$ déterminant
- n $\Longleftrightarrow$ nom commun
- np $\Longleftrightarrow$ nom propre

Question 1. Est-ce que cette grammaire est algébrique ? rationnelle ?

Question 2. Vérifier l'appartenance de la phrase **une pomme que Pierre regarde tombe** au langage reconnu par la grammaire en utilisant un arbre syntaxique.

Question 3. Appliquer l'algorithme CKY pour vérifier :

- l'appartenance de la phrase **une pomme que Pierre regarde tombe** au langage engendré par la grammaire
- le rejet de la phrase **une pomme qui Pierre regarde tombe** du langage engendré par la grammaire

Exercice 3. JFLAP

L'analyse CKY peut être effectuée par le logiciel JFLAP (http://jflap.org/jflaptmp).

Pour cela, on se place en mode **grammaire** (*Grammar*). Ensuite, on rentre les différentes règles de la grammaire non normalisée.

On peut alors vérifier qu'il s'agit bien d'une grammaire algébrique (*context-free*) grâce à l'item *Test for Grammar Type* de l'onglet *Test*.

On peut ensuite normaliser la grammaire grâce à l'item (*Transform Grammar*) de l'onglet *Convert*.

On peut ensuite réaliser l'analyse CKY grâce à la commande *CYK Parse* de l'onglet *Input*.