Grammar Cheatsheet

TLP

2019

Contents

List of Tables

List of Figures

Build GCL for a given language

Reduce a CFG

Dada una gramática $G = (N, T, S, P)$:

- Un símbolo útil $\in N \cup T$ es aquel:
  - $X \in N \cup T$ accesible si: $S \Rightarrow^* \alpha X \beta$
  - $X \in N$ co-accesible si: $X \Rightarrow^* \omega, \omega \in T^*$
- El orden importa, primero calcular co-accesibles y luego accesibles.

Algoritmo para calcular símbolos co-accesibles

Símbolos co-accesibles: $S_{co} = \{A \in N \mid A \to \alpha, \alpha \in T^*\}$

$S_{co_i+1} = S_{[}co_i]\{A \in N \mid A \to \alpha \in P, \alpha \in (S_{[}co_i] \cup T)^*\}$

STOP WHEN: $S_{co_i} = S_{co_i+1}$

Algoritmo para calcular símbolos accesibles

Se construye un grafo:

- Los nodos son símbolos(dependencias)
- $X \to Y$ si $X \to \alpha Y \beta \in P$

X es accesible si $\exists$ un camino de S hasta X.

Algorithm for:

Given a CFG ...

CFG is finite

1. Reduce the grammar.
2. Transform into CNF.
3. Look for loops in the dependency graph.

GCL is empty

1. Calculate co-accesible symbols.
2. If $S \in S_c \rightarrow L(G) \neq \emptyset$ else $L(G) = \emptyset$

A word belongs to L(G)

CYK

Brute force

Normal Forms

Chomsky

Greibach

LL(k) Grammars

CFG to NPDA

NPDA to CFG