Expressing Type-0 Languages in Terms of Context-Free Ambiguity

Stanislav Láznička

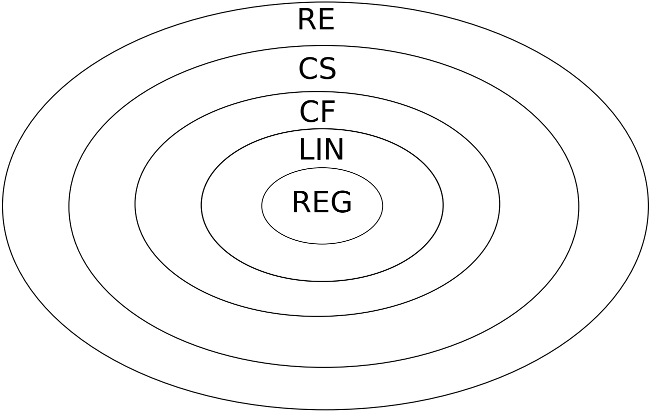
BUT Faculty of Information Technology, UIFS

6. ledna 2016

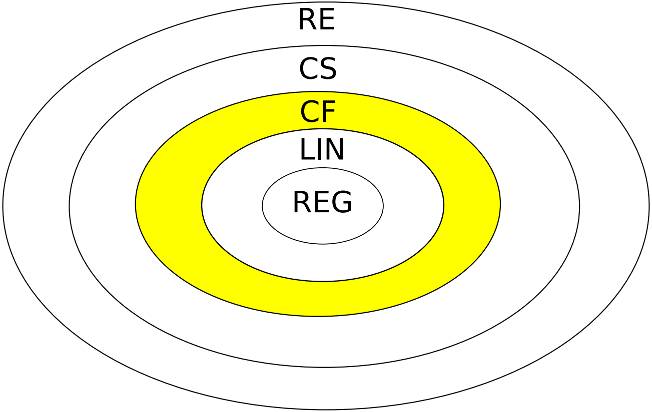
Content

- Basic Terms
- Ambiguity
- Context-Free and Type-0 Languages

The Chomsky Hierarchy of Formal Languages



The Chomsky Hierarchy of Formal Languages



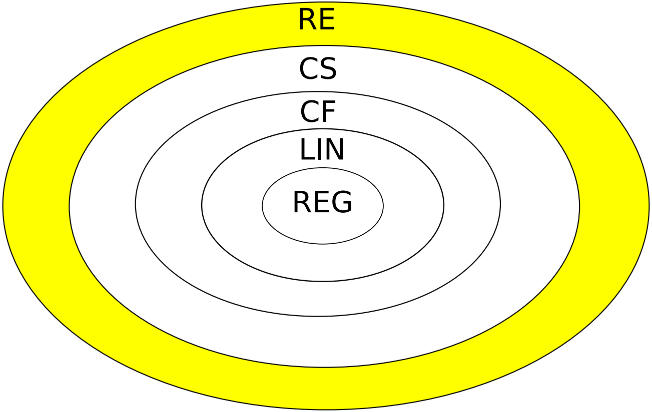
Context-Free Languages

Definition

A context-free grammar (CFG) G is defined by a tuple $G = (N, \Sigma, P, S)$ where:

1. N is a finite set of nonterminals.
2. Σ is an alphabet of terminals.
3. P is a set of productions of form $A \rightarrow (A \cup a)^*$ where $A \in N, a \in \Sigma$.
4. $S \in N$ is the start nonterminal.

The Chomsky Hierarchy of Formal Languages



Recursively Enumerable Languages

Definition

An unrestricted grammar G is defined by a tuple $G = (N, \Sigma, P, S)$ where:

1. N is a finite set of nonterminals.
2. Σ is an alphabet of terminals.
3. P is a set of productions of form $\alpha \to \beta$ where $\alpha \in (N \cup \Sigma)^+, \beta \in (N \cup \Sigma)^*$.
4. $S \in N$ is the start nonterminal.

Ambiguity

Definition

Let G be a CFG and $x \in L(G)$. Therefore there is a derivation sequence $S = \Phi_0 \Rightarrow \Phi_1 \Rightarrow \Phi_1 \Rightarrow \ldots \Rightarrow \Phi_n = x$ in G. Such a sequence gives a rise to a derivation tree where each node is labeled with a symbol from E ($E = \Sigma \cup N$) with S as the root node. G is **ambiguous** if there exists a string x in $L(G)$ with multiple derivation trees.

Decidability

The problem of grammar ambiguousness is undecidable. However, there exists an algorithm that is able to decide whether a grammar is unambiguous for some grammars.

Unambiguity

Vertical unambiguity

Given a CFG G, two sentential forms $\alpha, \alpha' \in (\Sigma \cup N)^*$ are vertically unambiguous, written $\Vdash_G \alpha; \alpha'$, iff:

$$L_G(\alpha) \cap L_G(\alpha') = \emptyset$$

A grammar is vertically unambiguous, written $\Vdash G$, if and only if for each two different sequential forms α, α' reachable in G $\Vdash_G \alpha; \alpha'$

Unambiguity 2

Horizontal unambiguity

Given a CFG G, two sentential forms $\alpha, \alpha' \in (\Sigma \cup N)^*$ are horizontally unambiguous, written $\models_G \alpha; \alpha'$, iff:

$$L_G(\alpha) \bowtie L_G(\alpha') = \emptyset$$

where $\bowtie$ is the language overlap operator defined by
$X \bowtie Y = \{xay | x, y \in \Sigma^* \wedge a \in \Sigma^+ \wedge x, xa \in X \wedge y, ay \in Y\}$
A grammar is horizontally unambiguous, written $\models G$, if and only if for every sentential form $\alpha\alpha'$ reachable in G $\models_G \alpha; \alpha'$

If both $\Vdash G$ and $\models G$ we write $|\!\models G$.
$|\!\models G \leftrightarrow G$ is unambiguous.

Ambiguity examples

1. Vertical ambiguous grammar

   $S \rightarrow Ay$

   $| \quad xB$

   $A \rightarrow xa$

   $B \rightarrow ay$

   There are two ways to parse the string *xay*.

2. Horizontal ambiguous grammar

   $S \rightarrow xAB$

   $A \rightarrow a$

   $| \quad \epsilon$

   $B \rightarrow ay$

   $| \quad y$

   Again, two possible derivation trees for *xay*.

Ambiguity Questions

Can CFG ambiguity be used to describe Type-0 languages?

Possibly.

Can it be used to get out of CFL class?

Yes.

Outside CFL

Let G_1 and G_2 be CFGs, $G_1 = (V_1, \Sigma, P_1, S_1)$, $G_2 = (V_2, \Sigma, P_2, S_2)$ where

$P_1 = \{S_1 \to A_1C_1 \quad A_1 \to aA_1b \quad A_1 \to ab \quad C_1 \to cC_1 \quad C_1 \to c\}$

$P_2 = \{S_2 \to A_2C_2 \quad A_2 \to aA_2 \quad A_2 \to a \quad C_2 \to bC_2c \quad C_2 \to bc\}$

$\Sigma = \{a, b, c\}$

$$L(G_1) \cap L(G_2) = \{a^n b^n c^n \mid n \in N\}$$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$S_1 \rightarrow A_1 C_1$

$aaabbbccc$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_1 \rightarrow aA_1b$

$aaabbbccc$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_1 \rightarrow aA_1b$

aaabbbccc

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_1 \rightarrow ab$

aaabbbccc

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_1 \rightarrow cC_1$

$aaabbbccc$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_1 \to cC_1$

$aaabbbccc$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_1 \to cC_1$

aaabbbccc

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_1 \rightarrow cC_1$

aaabbbcccc

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_1 \to cC_1$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$S_2 \rightarrow A_2 C_2$

aaabbbccccc...

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_2 \rightarrow bC_2c$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_2 \rightarrow bC_2c$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$C_2 \rightarrow bc$

$aaabbbcccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_2 \rightarrow aA_2$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_2 \to aA_2$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_2 \to aA_2$

$aaabbbccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_2 \to aA_2$

$aaaabbbcccccc\ldots$

Outside CFL 2

Idea

What if we analyzed $L(G_1) \cap L(G_2)$ separately?

$A_2 \rightarrow aA_2$

$\ldots aaaaabbbccccc\ldots$

Getting Type-0 Language

Theorem

For each recursively enumerable set $E \subseteq \Sigma^*$, there exist deterministic context-free languages L_1 and L_2, and a homomorphism h such that

$$E = h(L_1 \cap L_2)$$

Proof of the theorem by Ginsburg, Greibach and Harrison.

Getting Type-0 Language

Definition

A context-free grammar G is unambiguous if for every sentence $w \in L(G)$, there is exactly one derivation tree t with $\mathit{frontier}(t) = w$, where $\mathit{frontier}(x)$ is the sequence of the edge nodes of tree x.
All **deterministic** CFGs are unambiguous.

Definition

Let $\mathit{forest}(G)$ be a set of trees with edge nodes labeled by terminals from G. A tree $t \in \mathit{forest}(G)$ is a cut-frontier ambiguous tree (CFAT) if there is a $d \in \mathit{forest}(G)$ such that $d \neq t$ and $\mathit{frontier}(d) = \mathit{frontier}(t)$. Let CFAT(G) denote the set of all CFATs for G.

By the theorem from the previous slide, the pre-homomorphism language is a language of ambiguous tree frontiers.

Getting Type-0 Language

Theorem

Let L be RE language. Then, there is a CFG K such that $L = \{frontier(t) | t \in CFAT(K)\}$.

Getting Type-0 Language

Construction

Let G and H be two deterministic CFGs over Σ, $G = (N_G, \Sigma, P_G, S_G)$ and $H = (N_H, \Sigma, P_H, S_H)$; h is a homomorphism $h : \Sigma^* \to \Sigma_L^*$. Σ_L^* is an alphabet of terminals for L.
We construct a context-free grammar $K = (N, \Sigma_L^*, P, Z)$ such that $L = \{frontier(t) | t \in CFAT(K)\} = h(L(G) \cap L(H))$.
Z is a new nonterminal.
We set $N = \{Z, Z'\} \cup N_G \cup N_H \cup \Sigma$, all elements of this union are mutually disjoint (without loss of generality).
We also set
$P = \{Z \to Z' S_G, Z \to S_H Z', Z' \to \epsilon\} \cup P_G \cup P_H \cup \{a \to h(a) | a \in \Sigma\}$.

Note

Without loss of generality, we assume G and H use same set of terminals and different set of nonterminals.

References

- Based on idea of Alexander Meduna and Zbyněk Křivka

- GINSBURG, Seymour, Sheila A. GREIBACH, Michael A. HARRISON. One-Way Stack Automata. *Journal of the Association for Computing Machinery.* 1967, Vol. 14, No. 2, pg. 389-418. ISSN 0004-5411.
- BRABRAND, Claus, Robert GIEGERICH, Anders MØLLER. Analyzing Ambiguity of Context-Free Grammars. *Implementation and Application of Automata*[online]. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pg. 214 [cit. 2015-12-09]. DOI: 10.1007978-3-540-76336-9_21. ISBN 978-3-540-76335-2. Available from: http:link.springer.com10.1007978-3-540-76336-9_21

Last slide

Many thanks for your attention.