TREE CONTROLLED GRAMMARS

Part One: **Controlling Levels**

Jiří Koutný

ikoutny@fit.vutbr.cz

Department of Information Systems
Faculty of Information Technology
Brno University of Technology
Božetěchova 2
612 66 Brno, CZ

Contents

Language Classes

Classical classes
(by Chomsky)

- $\mathscr{L}$(REG)
- $\mathscr{L}$(LIN)
- $\mathscr{L}$(CF)
- $\mathscr{L}$(CS)
- $\mathscr{L}$(RE)
- $\mathscr{L}$(ALL)

Other classes
(by Controlled Grammars)

- $\mathscr{L}$(RC)
- $\mathscr{L}$(M)
- $\mathscr{L}$(P)
- $\mathscr{L}$(PER)
- $\mathscr{L}$(FOR)
- $\mathscr{L}$(TC)
- ...etc.

Typically
CFG + something

Why TC Grammars?

- Generates languages beyond CF
  - Like other controlled grammars
- Simple and natural extensions of CF grammars
  - Simpler than other controlled grammars?
- Derivation tree exactly like in CF case
- Parsing methods working in time $O(n^2)$

Prerequisites

1. Context Free Grammars (CFG)
2. Linear, Regular grammar
3. Derivation Step ($\Rightarrow$) ,
   Refleive and Transitive Closure of $\Rightarrow$ ($\Rightarrow^*$)
4. Language Generated by CFG ($L(G)$)
5. Unambigous, Ambigous, Inherently Ambigous CFG

Tree Controlled Grammar

TC Grammar is a pair (G,R), where:

- $G = \{N,T,P,S\}$ is a context-free grammar
- $R \subseteq (N \cup T)^*$ is a regular

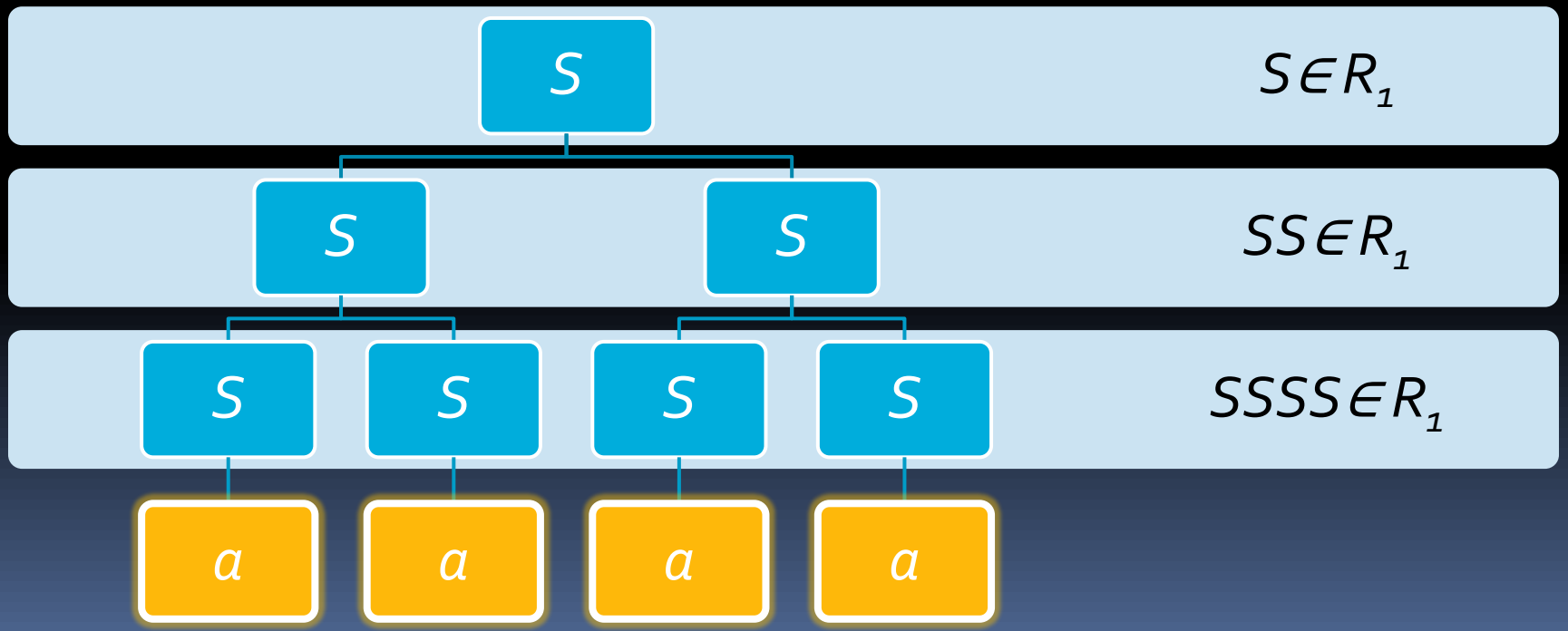
$L(G,R) = \{x \in L(G) \mid$ there exists a derivation tree of x such that each word obtained by concatenating all symbols at any level (except the last one) from left to right is in $R\}$

Example I: $\{a^{2^n} | n \geq 0\}$

$G_1 = (\{S\}, \{a\}, \{S \rightarrow SS | a\}, S)$

$R_1 = S^*$

$L(G_1, R_1) = (a^{2^n} | n \geq 0)$



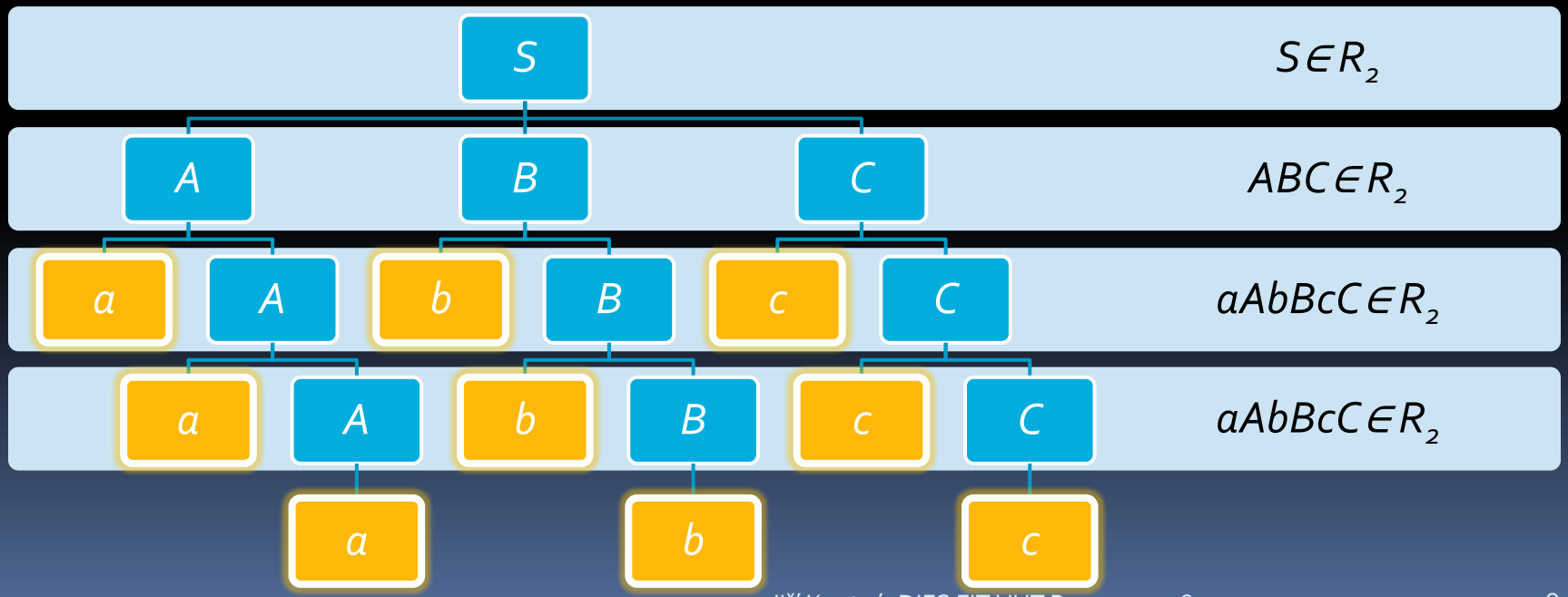
Example II: $\{a^n b^n c^n | n \geq 1\}$

$G_2 = (\{S,A,B,C\}, \{a,b,c\}, P, S)$

$P = \{S \rightarrow ABC,\ A \rightarrow aA|a,\ B \rightarrow bB|b,\ C \rightarrow cC|c\}$

$R_2 = \{S|ABC|aAbBcC\}$

$L(G_2,R_2) = \{a^n b^n c^n | n \geq 1\}$



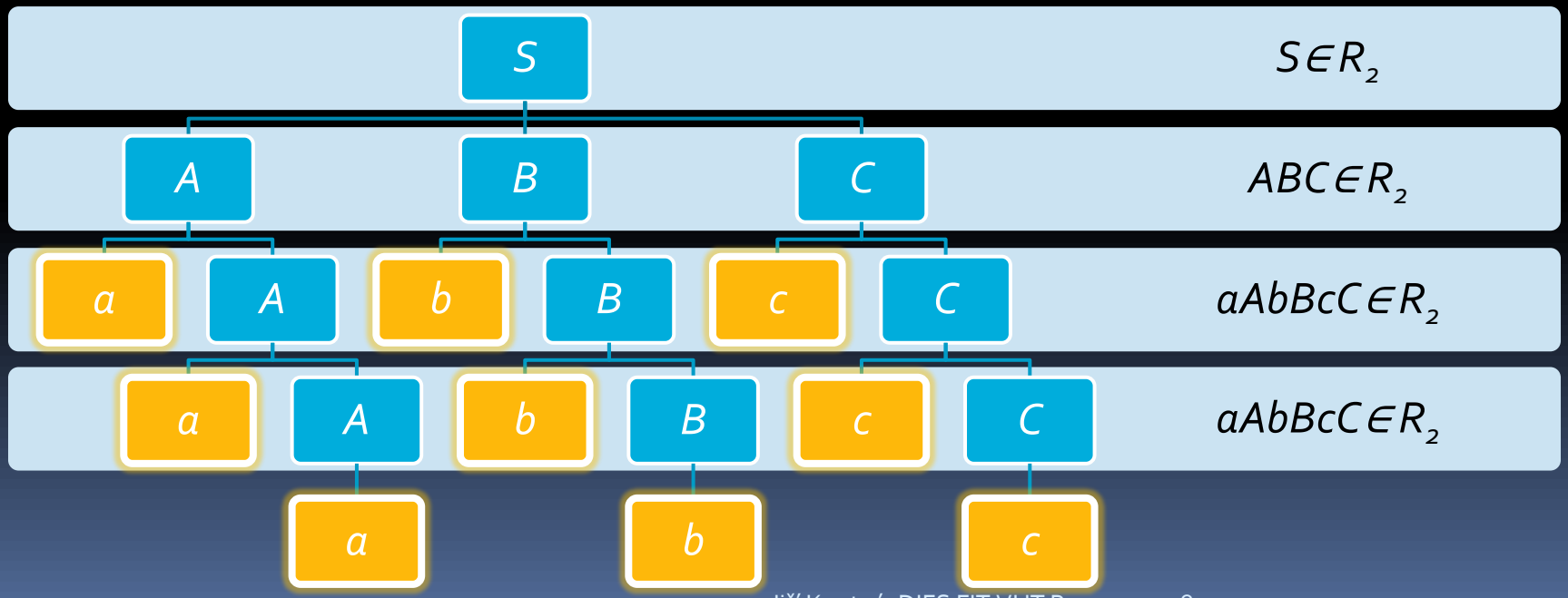
Example II: $\{a^n b^n c^n | n \geq 1\}$

$G_2 = (\{S,A,B,C\}, \{a,b,c\}, P, S)$

$P = \{S \rightarrow ABC,\ A \rightarrow aA|a,\ B \rightarrow bB|b,\ C \rightarrow cC|c\}$

$R_2 = \{(S)^*(ABC)^*(aAbBcC)^*\}$

$L(G_2,R_2) = \{a^n b^n c^n | n \geq 1\}$



Determine if $x \in L(G,R)$

(G,R) is a TC grammar, where G is unambigious $\Rightarrow$

- We can determine if $x \in L(G, R)$ in $O(n^2)$, $n=|x|$

Proof

- We can determine if $x \in L(G)$ in $O(n^2)$, $n=|x|$
- $x \notin L(G) \Rightarrow$
  - $x \notin L(G,R)$
- $x \in L(G) \Rightarrow$
  - We can construct unique derivation tree of x in $O(n^2)$
  - Number of levels of derivation tree is $\leq k.n$ (k depends on G)
  - Each level y is of the length $\leq n$, determine if $y \in R$ in $O(|y|)$
  - Thus whole tree in $O(n^2)$ steps

Determine if $x \in L(G,R)$

This parsing metod exists for

- All unambigous CF languages
- Some inherently abigous CF languages (see Example III)
- Some non CF languages.

Example III: $a^m b^m c^n \cup a^m b^n c^n$

For some inherently ambigous language L there exists TC grammar (G,R) such L(G,R) = L.

$L = \{a^m b^m c^n \mid m,n \geq 1\} \cup \{a^m b^n c^n \mid m,n \geq 1\}$

- Is inherently ambigous CF language
- Exists TC grammar (G,R), L = L(G,R), in which G is unambigous.

Example III: $a^m b^m c^n \cup a^m b^n c^n$

Let $G = (\{S\} \cup N_1 \cup N_2 \cup N_3, T, P_0 \cup P_1 \cup P_2 \cup P_3, S)$, where

- $N_1 = \{A_1, B_1\}$ $N_2 = \{A_2, B_2, E_2\}$ $N_3 = \{A_3, B_3, E_3\}$
- $T = \{a, b, c\}$
- $P_0 = \{S \rightarrow A_1B_1 \mid A_2B_2 \mid A_3B_3\}$
- $P_1 = \{A_1 \rightarrow aA_1 \mid a,\ B_1 \rightarrow bB_1c \mid bc\}$
- $P_2 = \{A_2 \rightarrow aA_2 \mid a,\ B_2 \rightarrow bB_2c \mid bE_2c,\ E_2 \rightarrow cE_2 \mid c\}$
- $P_3 = \{A_3 \rightarrow aA_3 \mid a,\ B_3 \rightarrow bB_3c \mid bE_3c,\ E_3 \rightarrow bE_3 \mid b\}$

And let $R = \{S\} \cup R_1 \cup R_2 \cup R_3$, where

- $R_1 = a^*A_1{}^*b^*B_1{}^*c^*$
- $R_2 = a^*A_2b^*B_2c^* \cup a^*b^*E_2c^*$
- $R_3 = a^*A_3b^*B_3c^* \cup a^*A_3b^*E_3c^*$

Example III: $a^m b^m c^n \cup a^m b^n c^n$

- Define $G_i = (\{S\} \cup N_i, T, \{S \to A_i B_i\} \cup P_i, S)$, for i=1,2,3
- Evidently $L(G) = L(G_1) \cup L(G_2) \cup L(G_3)$ and
  - $L(G_1) = \{a^m b^n c^n \mid m,n \geq 1\}$
  - $L(G_2) = \{a^m b^n c^p \mid m,n \geq 1, p > n\}$
  - $L(G_3) = \{a^m b^n c^p \mid m,p \geq 1, p < n\}$
- Each G_i is unambigous and $L(G_1)$, $L(G_2)$, $L(G_3)$ are mutually disjoint $\Rightarrow$ G is unambigous

Example III: $a^m b^m c^n \cup a^m b^n c^n$

To see that $L = L(G,R)$ observe that

- $L(G,R) = L(G_1, \{S \cup R_1\}) \cup L(G_2, \{S \cup R_2\}) \cup L(G_3, \{S \cup R_3\})$
- Choice of first production $S \to A_i B_i \Rightarrow$ only the production of P_i with control R_i can be applied.

Clearly $L(G_1, \{S\} \cup R_1) = L(G_1) = \{a^m b^n c^n \mid m,n \geq 1\}$.

Observe

- $L(G_2, \{S\} \cup R_2) = \{a^n b^n c^p \mid n,p \geq 1, p > n\}$
- $L(G_3, \{S\} \cup R_3) = \{a^n b^n c^p \mid n,p \geq 1, p < n\}$
- R_2 and R_3 guarantees the same number of a and b

Thus $L(G,R) = L$

ε-free TC Grammar

For every TC grammar (G,R) without ε-rules, the language L(G,R) is recursive.

Proof

- Let G=(N,T,P,S) and let $x \in T^*$, $|x| = n$
- (note it's not necessary that $x \in L(G)$)
- Let $T^{(N)} = \{y \in T^* \mid 1 \leq |y| \leq n\}$
- Let $F = \{x_1, x_2, \ldots, x_m \mid x_j \in R$ for $1 \leq j \leq m$, $1 \leq |x_1| \leq |x_2| \leq \ldots \leq |x_m| \leq n$, $x_j \neq x_k$ for $j \neq k\}$
- Note that F is finite
- $x \in L(G,R) \Leftrightarrow \exists F: x_1 \Rightarrow_G x_2 \Rightarrow_G \ldots \Rightarrow_G x_m$, $x_1 = S$, $x_m = x$
- It suffices to check all F and thus L(G,R) is recursive.

TC Grammars and Chomsky hierarchy

Language L is

a) Regular
b) Linear
c) Context-Free

if and only if there is a CF G=(N,T,P,S) such that

a) $L = L(G, T^*N)$
b) $L = L(G, T^*NT^*)$
c) $L = L(G, (N \cup T)^*)$

L is regular $\Leftrightarrow$ L = L(G,T*N)

Proof of a)

- $L(G,T^*N) \subseteq \mathscr{L}(REG)$
  - Let $L = L(G, T^*N)$ for some $G = (N,T,P,S)$
  - In any derivation of (G, T^*N) it can be used rules of the form $A \to wB \mid w$ $(A,B \in N, w \in T^*)$
  - Let $P' = P \cap N \times T^*(N \cup \{\varepsilon\})$ and let $G'=(N,T,P',S)$.
  - Then $L(G, T^*N) = L(G')$
- $\mathscr{L}(REG) \subseteq L(G,T^*N)$
  - Let L be regular and let $G = (N,T,P,S)$ be a right linear grammar generating L
  - Then $L(G) = L(G,T^*N)$
- Thus L is regular $\Leftrightarrow$ $L = L(G,T^*N)$

Proof of b) is analogous, proof of c) is trivial.

TC Grammars and $\mathscr{L}$(RE)

- Let T be an alphabet. Then there exists a CF $G_T = (N,T,P,S)$ (depending only on T) such that for each type 0 language L, $L \subseteq T^*$, there exists regular set R_L, $R_L \subseteq (T \cup N)^*$, so that $L = L(G_T, R_L)$.
- Every type 0 language can be generated by a TC grammar (G,R), where G=(N,T,P,S) and $P \subseteq N \times (N^* \cup T)$.
- Every type 0 language can be generated by a TC grammar (G,R), where G=(N,T,P,S) and $R \subseteq N^*$.

Proof can be found in [1].

Parsing Based on TC Grammars

- To determine if $x \in L(G,R)$
- Two phases
  1. To determine if $x \in L(G)$
     - Bottom-Up Parsing based on G in CNF
     - Don't need to generate derivation tree
     - Create sets with left sides of used rules
  2. To determine if $x \in L(G,R)$
     - Top-Down Parsing based on sets from phase 1
     - Construct derivation tree of x in L(G)
     - Test if each level of derivation tree is in R

See [2] for details and example.

Future Research

- Modification of TC Grammars
  - Controling Path in Derivation Tree (topic of my 2nd presentation)
- Controling by
  - Regular Language
  - Linear Language

References

1. K. Culik and H. A. Maurer:
   Tree controlled grammars,
   Computing,
   Vol. 19, pp. 129-139, 1977.

2. Navrátil Petr:
   Parsing Based on Regulated Grammars,
   diplomová práce,
   Brno, FIT VUT v Brně,
   2003

Questions

THANK YOU
FOR YOUR ATTENTION