CS51: Models of Computation

Lecture 26

Parsing Context-Free Languages

Review

- Definitions of phrase structure, context-free and regular languages
- Proof that regular languages and languages recognized by finite-state machines are the same.

Today's Topics

- **Chomsky normal form**
- **Simple parsing algorithm**

Parse Trees

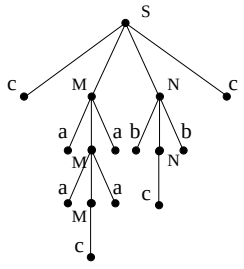
Example: $G_3 = (\mathcal{N}_3, \mathcal{T}_3, \mathcal{R}_3, S)$

(*a*)	S	→	*c*M*N**c*
(*b*)	M	→	a*M*a
(*c*)	M	→	c

(*d*)	N	→	*b*N*b*
(*e*)	N	→	*c*

A derivation of caacaabcbc and its parse tree.

$$S \Rightarrow cMNc \Rightarrow caMaNc \Rightarrow ca^2Ma^2Nc$$
$$\Rightarrow ca^2ca^2Nc \Rightarrow ca^2ca^2bNbc \Rightarrow ca^2ca^2bcbc$$



Yield of tree is string of leaves, **height** is length of longest path. In **leftmost derivation,** rules invoked in depth-first left to right order. **Rightmost** similar.

Context-Free Languages

- A context-free grammar is a phrase-structure grammar $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, s)$ in which each rule has only a single non-terminal on the left. Context-free languages are generated by context-free grammars.

Example: Let $\mathcal{N}_2 = \{s\}$, $\mathcal{T}_2 = \{\varepsilon, a, b\}$, $\mathcal{R}_2 = \{s \rightarrow asb, s \rightarrow \varepsilon\}$. Then, $G_2 = (\mathcal{N}_2, \mathcal{T}_2, \mathcal{R}_2, s)$ is context-free.

Chomsky Normal Form

- A CFG $G = (\mathcal{N}, \mathcal{T}, \mathcal{R}, S)$ is in **Chomsky normal form** if every rule is of the form $A \rightarrow BC$ or $A \rightarrow b$, $b \in \mathcal{T}$, except if $\varepsilon \in L(G)$ in which case $S \rightarrow \varepsilon$ is also a rule.

Theorem: Every context-free language L can be generated by a CFG in Chomsky normal form.

Example: $G_3 = (\mathcal{N}_3, \mathcal{T}_3, \mathcal{R}_3, S)$

(*a*)	S	→	*cMNc*
(*b*)	M	→	aMa
(*c*)	M	→	c

(*d*)	N	→	*bNb*
(*e*)	N	→	*c*

A Chomsky normal form grammar generating same language uses (*c*) & (*e*) and replaces others by:

(*a*) $S \rightarrow CD$, $C \rightarrow c$, $D \rightarrow ME$, $E \rightarrow NC$,

(*b*) $M \rightarrow AF$, $A \rightarrow a$, $F \rightarrow MA$

(*d*) $N \rightarrow BG$, $B \rightarrow b$, $G \rightarrow NB$

Putting CFG in Chomsky Normal Form

Theorem: Every context-free language L can be generated by a CFG in Chomsky normal form.

Proof: If $\varepsilon \in L$, add $S \rightarrow \varepsilon$. Let L be generated by G. Convert G to G' in Chomsky normal form in stages.

a) eliminate from G ε-rules of the form $B \rightarrow \varepsilon$ (except for $S \rightarrow \varepsilon$) as follows:for each rule with ≥ 1 B in right-hand side, e.g. $A \rightarrow \alpha B \beta B \gamma$ (α, β,γ are strings), add all possible rules formed by replacing B by ε in all possible ways e.g. $A \rightarrow \alpha\beta B\gamma$, $A \rightarrow \alpha B\beta\gamma$, $A \rightarrow \alpha\beta\gamma$, giving four rules for one original rule.

b) For rules $A \rightarrow \alpha w_i \beta$ (α, β are strings) with $w_i \in \mathcal{T}$, replace it by $A \rightarrow \alpha Z_i \beta$ & add rule $Z_i \rightarrow w_i$, where Z_i is a new non-terminal. Continue until all rules have a single terminal on right or a string of non-terminals. This new grammar also generates L.

Chomsky Normal Form (cont.)

Proof (**cont.**) Rules are now of the form: a) $A \rightarrow b$ for $b \in \mathcal{T}$, b) $S \rightarrow \varepsilon$, c) $A \rightarrow Z_1 Z_2 \ldots Z_k$, for $Z_i \in \mathcal{N}$.

Consider rules of type c) with k = 1. Cascading such rules gives derivations $A \stackrel{*}{\Rightarrow} B$; delete all rules of type c) with k = 1 and replace them with $A \rightarrow B$ if $A \stackrel{*}{\Rightarrow} B$. The same language is generated.

If $C \rightarrow D$ and $D \rightarrow b$, add $C \rightarrow b$, deleting all rules of the form $A \rightarrow B$. This generates same language; all remaining rules are of the form $S \rightarrow \varepsilon$, $A \rightarrow b$ or $A \rightarrow Z_1 Z_2 \ldots Z_k$ with $k \geq 2$, $Z_i \in \mathcal{N}$.

Now replace all rules of the form $A \rightarrow Z_1 Z_2 \ldots Z_k$ by the rules $A \rightarrow Z_1 N_1$, $N_1 \rightarrow Z_2 N_2$, ..., $N_{K-3} \rightarrow Z_{K-2} N_{K-2}$, $N_{K-2} \rightarrow Z_{K-1} Z_K$ where each N_i is a new non-terminal.

This new grammar is in correct form and generates L.

Example

Let $G = (\mathcal{N},\mathcal{T},\mathcal{R},\mathrm{E})$ be the grammar with $\mathcal{N}$= {E, T, F}, $\mathcal{T}$= {*a*, *b*, +, *, (,)} and let $\mathcal{R}$ have following rules:

(*a*)	E	→	E+T
(*b*)	E	→	T
(*c*)	T	→	T*F
(*d*)	T	→	F

(*e*)	F	→	(E)
(*f*)	F	→	*a*
(*g*)	F	→	*b*

E, T, and F denote *expressions*, *terms* & *factors*. Easy to show that $E \stackrel{*}{\Rightarrow} (a*b+a)*(a+b)$ and $E \stackrel{*}{\Rightarrow} a*b+a$.

Conversion: 1) no ε-rules, 2) in $A \to w_1 w_2 ... w_k$ replace (by **(**,) by **)**, + by **+**, and * by *****, 3) cascade non-terminals.

(*1*)	E	→	E**+**T
(*2*)	E	→	T*****F
(*3*)	E	→	**(**E**)**
(*4*)	E	→	*a*
(*5*)	E	→	*b*
(*6*)	T	→	T*****F
(*7*)	T	→	**(**E**)**
(*8*)	T	→	*a*

(*9*)	T	→	*b*
(*10*)	F	→	**(**E**)**
(*11*)	F	→	*a*
(*12*)	F	→	*b*
(13)	**(**	→	(
(14)	**)**	→	)
(15)	**+**	→	+
(16)	*****	→	*

The last step is to reduce to two the number of non-terminals on the right-hand side of a rule.

Example

$G = (\mathcal{N},\mathcal{T},\mathcal{R},\mathrm{E})$, where $\mathcal{N}$= {A,B,C,D,E,F,G,T,**(**,**)**,**+**, *****}, $\mathcal{T}$= {*a*, *b*, +, *, (,)} and let $\mathcal{R}$ have following rules:

(*A*)	E	→	EA
(B)	A	→	**+**T
(*C*)	E	→	TB
(*D*)	B	→	*****F
(*E*)	E	→	**(**C
(*F*)	C	→	E**)**
(*G*)	E	→	*a*
(*H*)	E	→	*b*
(*I*)	T	→	TD
(*J*)	D	→	*****F
(*K*)	T	→	**(**G

(*L*)	G	→	E**)**
(*M*)	T	→	*a*
(*N*)	T	→	*b*
(*P*)	F	→	**(**H
(*Q*)	H	→	E**)**
(*R*)	F	→	*a*
(*S*)	F	→	*b*
(*T*)	**(**	→	(
(*U*)	**)**	→	)
(*V*)	**+**	→	+
(*W*)	*****	→	*

This grammar is in Chomsky normal form.

We illustrate bottom-up parsing.

Parsing Using Chomsky Normal Form

- Let's parse **w** = $w_1w_2w_3w_4w_5 = a*b+a$ where $w_1=a$, w_2=*, $w_3=b$, w_4=+, $w_5=a$.
  - — w_i is derived by rules $A \to b$ where A is in $N_{i,i+1} = \{N \mid N \to w_i\}$.
  - — Thus, $N_{1,2}$={E,T,F},$N_{2,3}$ ={*****}, $N_{3,4}$={E,T,F}, $N_{4,5}$ ={**+**}, $N_{5,6}$ ={E,T,F}.
  - — All other derivations are of form $A \to BC$.
  - — To derive w_iw_{i+1}, we use a nonterminal in $N_{i,i+2} = \{N \mid N \to MP$, M in $N_{i,i+1}$, P in $N_{i+1,i+2}\}$
  - — Not knowing which pair w_iw_{i+1} is derived last we try all ways & compute $N_{i,i+2}$ for i =1,2,3.
  - — We try all ways to derive $w_iw_{i+1}w_{i+2}$ comptng $N_{i,i+3} = \{N \mid N \to MP$, M in $N_{i,i+2}$, P in $N_{i+2,i+3}\}$ $\cup$ $\{N \mid N \to MP$, M in $N_{i,i+1}$, P in $N_{i+1,i+3}\}$
  - — Continuing, we compute $N_{1,4}$. If it contains the start symbol, **w** is in our language and we have a parse for it.

More on Parsing

(*A*)	E	→	EA
(B)	A	→	**+**T
(*C*)	E	→	TB
(*D*)	B	→	*****F
(*E*)	E	→	**(**C
(*F*)	C	→	E**)**
(*G*)	E	→	*a*
(*H*)	E	→	*b*
(*I*)	T	→	TD
(*J*)	D	→	*****F
(*K*)	T	→	**(**G

(*L*)	G	→	E**)**
(*M*)	T	→	*a*
(*N*)	T	→	*b*
(*P*)	F	→	**(**H
(*Q*)	H	→	E**)**
(*R*)	F	→	*a*
(*S*)	F	→	*b*
(*T*)	**(**	→	(
(*U*)	**)**	→	)
(*V*)	**+**	→	+
(*W*)	*****	→	*

- $N_{1,2}$={E,T,F}, $N_{2,3}$ ={*****}, $N_{3,4}$ ={E,T,F}, $N_{4,5}$={**+**}, $N_{5,6}$ ={E,T,F}
- $N_{1,3} = \varnothing$, $N_{2,4}$ ={B, D}, $N_{3,5} = \varnothing$, $N_{4,6}$ ={A}
- $N_{1,4}$ ={E,T}, $N_{2,5} = \varnothing$, $N_{3,6}$ = {E}
- $N_{1,5} = \varnothing$, $N_{2,6} = \varnothing$
- $N_{1,6}$ ={E}

Summary

- Chomsky normal form
- Multiplication and closure of set matrices – basis for parsing of contex-free languages.