Algorithms and Decision Procedures for Context-Free Languages

Chapter 14

Decision Procedures for CFLs

Membership: Given a language *L* and a string *w*, is *w* in *L*?

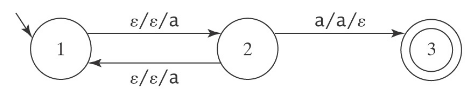
Two approaches:

- If *L* is context-free, then there exists some context-free grammar *G* that generates it. Try derivations in *G* and see whether any of them generates *w*.

Problem: $S \rightarrow S\,T \mid$ `a` Try to derive `aaa`

- If *L* is context-free, then there exists some PDA *M* that accepts it. Run *M* on *w*.

Problem:



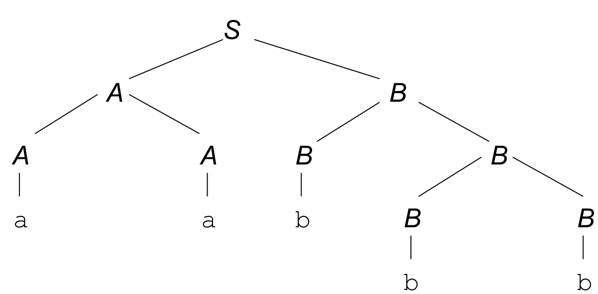
Normal Forms for Grammars

Chomsky Normal Form, in which all rules are of one of the following two forms:

- $X \rightarrow a$, where $a \in \Sigma$, or
- $X \rightarrow BC$, where B and C are elements of $V - \Sigma$.

Advantages:

- Parsers can use binary trees.
- Exact length of derivations is known:



Conversion to Chomsky Normal Form

1. Remove all ε-rules, using the algorithm *removeEps*. (We have seen this before.)

2. Remove all unit productions (rules of the form $A \rightarrow B$).

3. Remove all rules whose right hand sides have length greater than 1 and include a terminal:

   (e.g., $A \rightarrow$ aB or $A \rightarrow B$aC)

4. Remove all rules whose right hand sides have length greater than 2:

   (e.g., $A \rightarrow BCDE$)

Removing ε-Productions

Definition: a rule is ***modifiable*** iff it is of the form:

$P \rightarrow \alpha Q \beta$, for some nullable Q, $P \neq \alpha\beta \neq \varepsilon$

removeEps(G: cfg) =

1. Let $G' = G$.
2. Find the set N of nullable variables in G'.
3. For each modifiable rule $P \rightarrow \alpha Q \beta$ of G do
   Add the rule $P \rightarrow \alpha\beta$.
4. Delete from G' all rules of the form $X \rightarrow \varepsilon$.
5. Return G'.

$L(G') = L(G) - \{\varepsilon\}$

Example

Example:

$S \rightarrow \mathtt{a}AC\mathtt{a}$
$A \rightarrow B \mid \mathtt{a}$
$B \rightarrow C \mid \mathtt{c}$
$C \rightarrow \mathtt{c}C \mid \varepsilon$

Nullable: *A,B,C*

$S \rightarrow \mathtt{a}AC\mathtt{a} \mid \mathtt{a}A\mathtt{a} \mid \mathtt{a}C\mathtt{a} \mid \mathtt{aa}$
$A \rightarrow B \mid \mathtt{a}$
$B \rightarrow C \mid \mathtt{c}$
$C \rightarrow \mathtt{c}C \mid \mathtt{c}$

Unit Productions

A ***unit production*** is a rule $A \rightarrow B$ (right-hand side consists of a single nonterminal symbol)

removeUnits(G)

1. Let $G' = G$.
2. Until no unit productions remain in G' do:
   2.1 Choose some unit production $X \rightarrow Y$.
   2.2 Remove it from G'.
   2.3 Consider only rules that still remain. For every rule $Y \rightarrow \beta$, where $\beta \in V^*$, do:
   Add to G' the rule $X \rightarrow \beta$ unless it is a rule that has already been removed once.
3. Return G'.

Example

$S \rightarrow \mathtt{a}AC\mathtt{a} \mid \mathtt{a}A\mathtt{a} \mid \mathtt{a}C\mathtt{a} \mid \mathtt{aa}$
$A \rightarrow B \mid \mathtt{a}$
$B \rightarrow C \mid \mathtt{c}$
$C \rightarrow \mathtt{c}C \mid \mathtt{c}$

Remove $A \rightarrow B$. Add $A \rightarrow C \mid \mathtt{c}$.
Remove $B \rightarrow C$. Add $B \rightarrow \mathtt{c}C$ ($B \rightarrow \mathtt{c}$, already there).
Remove $A \rightarrow C$. Add $A \rightarrow \mathtt{c}C$ ($A \rightarrow \mathtt{c}$, already there).

So *removeUnits* returns:

$S \rightarrow \mathtt{a}AC\mathtt{a} \mid \mathtt{a}A\mathtt{a} \mid \mathtt{a}C\mathtt{a} \mid \mathtt{aa}$
$A \rightarrow \mathtt{a} \mid \mathtt{c} \mid \mathtt{c}C$
$B \rightarrow \mathtt{c} \mid \mathtt{c}C$
$C \rightarrow \mathtt{c}C \mid \mathtt{c}$

Mixed Rules

removeMixed(G) =
1. Let $G' = G$.
2. Create a new nonterminal T_a for each terminal a in Σ.
3. Modify each rule whose right-hand side has length greater than 1 and that contains a terminal symbol by substituting T_a for each occurrence of the terminal a.
4. Add to G, for each T_a, the rule $T_a \rightarrow a$.
5. Return G'.

Example

$S \rightarrow \mathrm{a}AC\mathrm{a} \mid \mathrm{a}A\mathrm{a} \mid \mathrm{a}C\mathrm{a} \mid \mathrm{aa}$
$A \rightarrow \mathrm{a} \mid \mathrm{c} \mid \mathrm{c}C$
$B \rightarrow \mathrm{c} \mid \mathrm{c}C$
$C \rightarrow \mathrm{c}C \mid \mathrm{c}$

removeMixed returns:

$S \rightarrow T_\mathrm{a}ACT_\mathrm{a} \mid T_\mathrm{a}AT_\mathrm{a} \mid T_\mathrm{a}CT_\mathrm{a} \mid T_\mathrm{a}T_\mathrm{a}$
$A \rightarrow \mathrm{a} \mid \mathrm{c} \mid T_\mathrm{c}C$
$B \rightarrow \mathrm{c} \mid T_\mathrm{c}C$
$C \rightarrow T_\mathrm{c}C \mid \mathrm{c}$
$T_\mathrm{a} \rightarrow \mathrm{a}$
$T_\mathrm{c} \rightarrow \mathrm{c}$

Long Rules

removeLong(G) =
1. Let $G' = G$.
2. For each rule r of the form:

$$A \rightarrow N_1N_2N_3N_4\ldots N_n,\ n > 2$$

create new nonterminals $M_2, M_3, \ldots M_{n-1}$.

3. Replace r with the rule $A \rightarrow N_1M_2$.
4. Add the rules:

$M_2 \rightarrow N_2M_3,$
$M_3 \rightarrow N_3M_4,$
$\ldots$
$M_{n-1} \rightarrow N_{n-1}N_n.$

5. Return G'.

Example

$S \rightarrow T_\mathrm{a}ACT_\mathrm{a} \mid T_\mathrm{a}AT_\mathrm{a} \mid T_\mathrm{a}CT_\mathrm{a} \mid T_\mathrm{a}T_\mathrm{a}$
$A \rightarrow \mathrm{a} \mid \mathrm{c} \mid T_\mathrm{c}C$
$B \rightarrow \mathrm{c} \mid T_\mathrm{c}C$
$C \rightarrow T_\mathrm{c}C \mid \mathrm{c}$
$T_\mathrm{a} \rightarrow \mathrm{a}$
$T_\mathrm{c} \rightarrow \mathrm{c}$

removeLong returns:

$S \rightarrow T_\mathrm{a}S_1$	$S \rightarrow T_\mathrm{a}S_3$	$S \rightarrow T_\mathrm{a}S_4$	$S \rightarrow T_\mathrm{a}T_\mathrm{a}$
$S_1 \rightarrow AS_2$	$S_3 \rightarrow AT_\mathrm{a}$	$S_4 \rightarrow CT_\mathrm{a}$	
$S_2 \rightarrow CT_\mathrm{a}$			

$A \rightarrow \mathrm{a} \mid \mathrm{c} \mid T_\mathrm{c}C$
$B \rightarrow \mathrm{c} \mid T_\mathrm{c}C$
$C \rightarrow T_\mathrm{c}C \mid \mathrm{c}$
$T_\mathrm{a} \rightarrow \mathrm{a}$
$T_\mathrm{c} \rightarrow \mathrm{c}$

Using a Grammar

decideCFLusingGrammar(L: CFL, w: string) =

1. If given a PDA, build G so that $L(G) = L(M)$.

2. If $w = \varepsilon$ then if S_G is nullable then accept, else reject.

3. If $w \neq \varepsilon$ then:
   3.1 Construct G' in Chomsky normal form such that $L(G') = L(G) - \{\varepsilon\}$.

   3.2 If G derives w, it does so in $2 \cdot |w| - 1$ steps. Try all derivations in G of $2 \cdot |w| - 1$ steps. If one of them derives w, accept. Otherwise reject.

Emptiness

Given a context-free language L, is $L = \varnothing$?

decideCFLempty(G: context-free grammar) =

1. Let G' = *removeunproductive*(G).

2. If S is not present in G' then return *True*
   else return *False*.

Finiteness

Given a context-free language L, is L infinite?

decideCFLinfinite(G: context-free grammar) =

1. Lexicographically enumerate all strings in Σ^* of length greater than b^n and less than or equal to $b^{n+1} + b^n$.

2. If, for any such string w, *decideCFL*(L, w) returns *True* then return *True*. L is infinite.

3. If, for all such strings w, *decideCFL*(L, w) returns *False* then return *False*. L is not infinite.

Why these bounds?

The Undecidable Questions about CFLs

- Is $L = \Sigma^*$?
- Is the complement of L context-free?
- Is L regular?
- Is $L_1 = L_2$?
- Is $L_1 \subseteq L_2$?
- Is $L_1 \cap L_2 = \varnothing$?
- Is L inherently ambiguous?
- Is G ambiguous?