5 Context-Free Grammars

5.1 Context-free grammars

Reading: Sipser 2.1 (pp. 100-105)

- CFG $G = (V, \Sigma, R, S)$, where V is the set of variables, Σ is the set of terminals (alphabet), R is the set of rules in the form of $V \to (V \cup \Sigma)^*$ (head→body), and $S \in V$ is the start variable.
- The CFG that generates all palindromes (strings that read the same forward and backward) over $\{0,1\}$ is $G = (\{S\}, \{0,1\}, R, S)$, where R contains $S \to \varepsilon|0|1|0S0|1S1$.
- Let u, v, w be strings in $(V \cup \Sigma)^*$. If $A \to w$ is a rule, then uAv yields uwv, written $uAv \Longrightarrow uwv$. We say u derives v, written $u \overset{*}{\Rightarrow} v$, if $\exists u_1, \ldots, u_k \in (V \cup \Sigma)^*$ such that $u \Longrightarrow u_1 \Longrightarrow \cdots \Longrightarrow u_k \Longrightarrow v$. Here, $\Longrightarrow$ means one step and $\overset{*}{\Rightarrow}$ means zero or more steps.
- Leftmost and rightmost derivations: $\underset{lm}{\Rightarrow}$, $\overset{*}{\underset{lm}{\Rightarrow}}$, $\underset{rm}{\Rightarrow}$, $\overset{*}{\underset{rm}{\Rightarrow}}$.
- The language of a CFG G, $L(G) = \{w \in \Sigma^* | S \overset{*}{\Rightarrow} w\}$. $L(G)$ is said to be a CFL.
- Why called "context-free"?

Example: Is $L = \{0^n 1^n | n \geq 0\}$ a context-free language? Yes since it is the language of the context-free grammar $S \to 0S1|\varepsilon$.
Example: What is the language for grammar G with $S \to AA$ and $A \to AAA|bA|Ab|a$? $L(G) = \{w \in \{a,b\}^* | w$ has an even (nonzero) number of as$\}$.
Example: A CFG for simple expressions in programming languages: $E \to E + E|E * E|(E)|I$ and $I \to Ia|Ib|I0|I1|a|b$.

5.2 Parse trees

- A parse tree is a tree representation for a derivation, in which each interior node is a variable, each leaf node is either a variable, a terminal, or ε, and if an interior node is a variable A and its children are $X_1, \ldots, X_k$, then there must be a rule $A \to X_1 \cdots X_k$.
- Yield of a parse tree: Concatenation of the leaf nodes in a parse tree rooted at the start variable.
- Four equivalent notions:
  1. $A \overset{*}{\Rightarrow} w$;
  2. $A \overset{*}{\underset{lm}{\Rightarrow}} w$;
  3. $A \overset{*}{\underset{rm}{\Rightarrow}} w$; and
  4. A parse tree with root A and yield w.

5.3 Regular grammars

- A regular grammar (RG) is a special CFG where the body of each rule contains at most one variable which, if present, must be the last symbol in the body.
- For each regular language L there is a RG G such that $L = L(G)$.
- Given RG $G = (V, \Sigma, R, S)$, construct an equivalent NFA $N = (Q, \Sigma, \delta, q_0, F)$, where $Q = V \cup \{f\}$, $q_0 = S$, $F = \{f\}$, and for each rule $A \to wB$, $\delta(A, w) = B$ and for each rule $A \to w$, $\delta(A, w) = f$.
- Given DFA $A = (Q, \Sigma, \delta, q_0, F)$, define an equivalent RG $G = (V, \Sigma, R, S)$, where $V = Q$, $S = q_0$, and for each transition $\delta(q, a) = p$, $q \to ap$ and for each final state $q \in F$, $q \to \varepsilon$.

Example: Given RG: $S \to aS|bA|aB$, $A \to aS|\varepsilon$, and $B \to bS|\varepsilon$, what is its NFA?
Example: Given a four-state DFA for language aa^*bb^*a, what is its regular grammar?

5.4 Ambiguity in grammars and languages

Reading: Sipser 2.1 (pp. 105-106)

- A CFG $G = (V, \Sigma, R, S)$ is ambiguous if there is $w \in \Sigma^*$ for which there are at least two parse trees (or leftmost derivations).
- Grammar G: $E \to E + E | E * E | (E) | I$ and $I \to Ia|Ib|I0|I1|a|b$ is ambiguous since a+b*a has two parse trees.
- Some ambiguous grammars have an equivalent unambiguous grammar. For example, an unambiguous grammar for the simple expressions is G': $E \to E + T | T$, $T \to T * F | F$, $F \to (E) | I$, and $I \to Ia|Ib|I0|I1|a|b$.
- A context-free language is said to be inherently ambiguous if all its grammars are ambiguous. For example, $\{a^n b^n c^m d^m | m, n \geq 1\} \cup \{a^n b^m c^m d^n | m, n \geq 1\}$. Its grammar is $S \to S_1 | S_2$, $S_1 \to AB$, $A \to aAb|ab$, $B \to cBd|cd$, $S_2 \to aS_2d|aCd$, and $C \to bCc|bc$. The grammar is ambiguous (considering $abcd$). There is a not so easy proof that all grammars for the language are ambiguous, thus it is inherently ambiguous.
- There is no algorithm to determine whether a given CFG is ambiguous. There is no algorithm to remove ambiguity from an ambiguous CFG. There is no algorithm to determine whether a given CFL is inherently ambiguous.

5.5 Chomsky normal form

Reading: Sipser 2.1 (pp. 106-109)
For a CFL, different people may come up with different but equivalent CFGs. There are several steps to simplify a CFG.

- Eliminating ε rules for $L(G) - \{\varepsilon\}$ by finding nullable variables ($A \stackrel{*}{\Rightarrow} \varepsilon$). For example, $S \to AB$, $A \to aAA|\varepsilon$, and $B \to bBB|\varepsilon$ can be changed to $S \to AB|A|B$, $A \to aAA|aA|a$, and $B \to bBB|bB|b$.
- Eliminating unit rules. For example, $E \to T|E + T$, $T \to F|T * F$, $F \to I|(E)$, and $I \to Ia|Ib|I0|I1|a|b$ can be changed to $E \to E + T|T * F|(E)|Ia|Ib|I0|I1|a|b$, $T \to T * F|(E)|Ia|Ib|I0|I1|a|b$, $F \to (E)|Ia|Ib|I0|I1|a|b$, and $I \to Ia|Ib|I0|I1|a|b$.
- Eliminating useless variables (and thus associated rules) by finding nongenerating and unreachable variables. For example, $S \to AB|a$ and $A \to b$ can be simplified to $S \to a$.

The Chomsky Normal Form (CNF): Any nonempty CFL without ε has a CFG G in which all rules are in one of the following two forms: $A \to BC$ and $A \to a$, where A, B, C are variables, and a is a terminal. Note that one of the uses of CNF is to turn parse trees into binary trees.
To convert a CFG to a grammar in CNF:

- Add a new start variable S_0 in the case when the old start variable S appears in the body of some rules.
- Simplify the grammar by removing ε rules, unit rules, and useless variables.
- Convert the rules in the simplified grammar into the proper forms of CNF by adding additional variables and rules.

Example (Sipser p. 108): Given a CFG G, construct a CNF G' such that $L(G) - \{\varepsilon\} = L(G')$.