cs302 Theory of Computation UVa Spring 2008

Notes: Context-Free Languages

Thursday, 14 February

Upcoming Schedule

Monday, 18 February: Office Hours (Olsson 236A, 2-3pm); Help Session (Olsson 228E, 5:30-6:30pm)
Monday, 18 February (3:30pm): Annie Anton, North Carolina State University, *Designing Legally Compliant Software Systems that Contain Sensitive Information* (Department Colloquim in Olsson 009)
Tuesday, 19 February (2:02pm): Problem Set 3
Thursday, 28 February: Exam 1 (in class)

Context-Free Grammars

A *grammar* is context-free if all production rules have the form: $A \rightarrow \alpha\gamma\beta$ (that is, the left side of a rule can only be a single variable; the right side is unrestricted and can be any sequence of terminals and variables).

We can define a grammar as a 4-tuple (V, Σ, R, S) where V is a finite set (variables), Σ is a finite set (terminals), S is the start variable, and R is a finite set of rules, each of which is a mapping $V \rightarrow (V \cup \Sigma)^*$.

Example. (Similar to Sipser's Exercise 2.9) Give a context-free grammar that generates the language $\left\{a^i b^j c^k | i = j \text{ or } i = k \text{ where } i, j, k \geq 0\right\}$. Show how $abbc$ is derived by your grammar. Show why $aaabbc$ could not be derived by your grammar.

Model of Computation for CFGs

First, we describe the model of computation for CFGs using a function notation (not the traditional $\Rightarrow$ notation).

Note that the grammar rules may have the same variable on the left side of may rules in R, so we cannot interpret R as a function. Instead, we define the function δ which captures the set of all right sides of rules for a given variable. The transition function $\delta : V \rightarrow \mathcal{P}((V \cup \Sigma)^*)$ (note the powerset operator - the output is a set of $(V \cup \Sigma)^*$ strings) is defined by:

$$\delta(A) = \{\alpha | \alpha \in (V \cup \Sigma)^* \wedge A \rightarrow \alpha \in R\}$$

Then, as with DFAs, we can define the extended transition function δ^* recursively:

$$\delta^*(\alpha) = \{\alpha\} \cup \bigcup_{\beta \in \delta(\alpha)} \delta^*(\beta)$$

A string w is in $G = (V, \Sigma, R, S)$ iff $w \in \delta^*(S)$.

Derivation. A more traditional way to define the model of computation for CFGs is using *derivation*. A grammar G derives a string w if there is a way to produce w starting from S following the rules in R. $S \Rightarrow^* w$ means G derives w. We define the $\Rightarrow^*$ function somewhat similarly to the δ^*.

First, we define $\Rightarrow$, the one step derivation function in terms of R, the rules of the CFG:

If $A \rightarrow \gamma$ is in R, then $\alpha A \beta \Rightarrow \alpha\gamma\beta$ for $\alpha, \beta, \gamma \in (V \cup \Sigma)^*$.

That is, if there is a rule $A \rightarrow \gamma$ in R, anywhere A appears in a sequence of variables and symbols, we can replace the A with γ, leaving the rest of the string unchanged.

Now, we can define $\Rightarrow^*$: $(V \cup \Sigma)^* \rightarrow (V \cup \Sigma)^*$, to mean that there is some way to produce the right side following zero of more steps starting from the input string (we can think of $\Rightarrow^*$ as outputing a set of strings, but define it using just single strings on the output side; the actual value of $derives(\alpha)$ is the set of all strings:

$\alpha \Rightarrow^* \alpha$ — any string derives itself (no replacements done).
$\alpha \Rightarrow^* \gamma$ if $\alpha \Rightarrow^* \beta$ and $\beta \Rightarrow \gamma$ — if we can go from α to β in zero or more steps, and from β to γ in one step, then we can derive γ from α.

The string w is in the language defined by the context-free grammar $G = (V, \Sigma, R, S)$ iff:

$$S \Rightarrow^* w$$

Proving Non-Context-Freeness

To show a language is not context-free, we need to prove there is no Context-Free Grammar that can generate the language. The strategy is similar to how we used the pumping lemma to show a language is non-regular. The pumping lemma for context-free languages gives us a property that must be true of any context-free language. We get a contradiction, but showing that there is no way to satisfy the properties of the pumping lemma for the given (non-context-free) language.

Pumping lemma for context-free languages. For any context-free language A, there is a pumping length p where all strings $s \in A$ with $|s| \geq p$ may be divided into 5 pieces, $s = uvxyz$ satsifying these conditions:

1. for each $i \geq 0, uv^ixy^iz \in A$
2. $|vy| > 0$
3. $|vxy| \leq p$

Suppose there is a CFG G that generates A. Then any string $s \in A$ can be derived using G. Since G is a context-free grammar, each production rule has a single variable on the left side. That means in a derivation of k steps (where each step involves replacing one variable with the right hand side of a corresponding rule) if $k \geq |V|$ then some variable $R \in V$ must be replaced twice:

$$S \Rightarrow^* uRz \Rightarrow^* uvRyz \Rightarrow^* uvxyz$$

The first replacement is $R \Rightarrow^* vRy$, which can be repeated any number of times, producing v^iRy^i.

Example. $D = \{ww | w \in \{0, 1\}^*\}$.

> Assume D is a context-free language. Then, there must be a CFG G that produces D, and the pumping lemma for context-free languages applies with pumping length p. As with pumping lemma for regular languages, we need to find *one* string w where $|w| \geq p$, and show that it cannot be pumped.
>
> Pick $w = \boxed{}$
>
> Show that all possible ways of dividing $w = uvxyz$ fail to satisfy the pumping lemma for CFLs requirements.

Tricky Example. Is $X = \{w | w \in \{0, 1\}^* \wedge$ there is no $z \in \{0, 1\}^*$ such that $w = zz\}$ context-free?