Homework 5: Context-Free Grammars

CSCI 161

Due 9pm, Thursday, October 22, 2020

Ben Dreyer

Overview. This week's homework gives you more practice designing and interpreting the languages of context-free grammars. Questions 1 and 2 have you design CFGs for two new languages, Question 3 touches on ambiguous strings and Chomsky normal form (3c will be easier later in the week), and Question 4 has you think about CFGs as a generalization of regular expressions. Hints are available in office hours!

Question 1. Design a CFG for the language over $\Sigma = \{1, \#\}$ whose elements consist of every pair of distinct, #-separated unary values:

$$L = \{x_1 \# x_2 \mid x_1, x_2 \in 1^*, x_1 \neq x_2\}.$$

Note: This L is a subset of the language from your HW3 proof of nonregularity. For extra practice with the pumping lemma from Unit 1, you could show (not for a grade) that this language is nonregular. Along with the CFG you submit for this question, this classifies L as nonregular and context-free (like $\{0^n 1^n \mid n \geq 0\}$).

Solution.

$$\begin{aligned} S &\to \#CM \mid CM\# \mid CSC \\ M &\to CM \mid \epsilon \\ C &\to 1 \end{aligned}$$

Question 2. Design a CFG for the language of binary strings that contain at least one 1 in their second half:

$$L = \{uv \mid u \in (0 \cup 1)^*, v \in (0 \cup 1)^* 1 (0 \cup 1)^*, |u| \geq |v|\}.$$

Solution.

$$\begin{aligned} S &\to XM \\ M &\to 1 \mid 0M1 \mid 0M0 \mid 1M0 \mid 1M1 \\ X &\to 1X \mid 0X \mid \epsilon \end{aligned}$$

Question 3. This multi-part question concerns the following CFG:

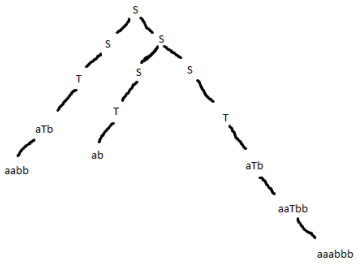
$$\begin{aligned} S &\to SS \mid T \\ T &\to aTb \mid ab \end{aligned}$$

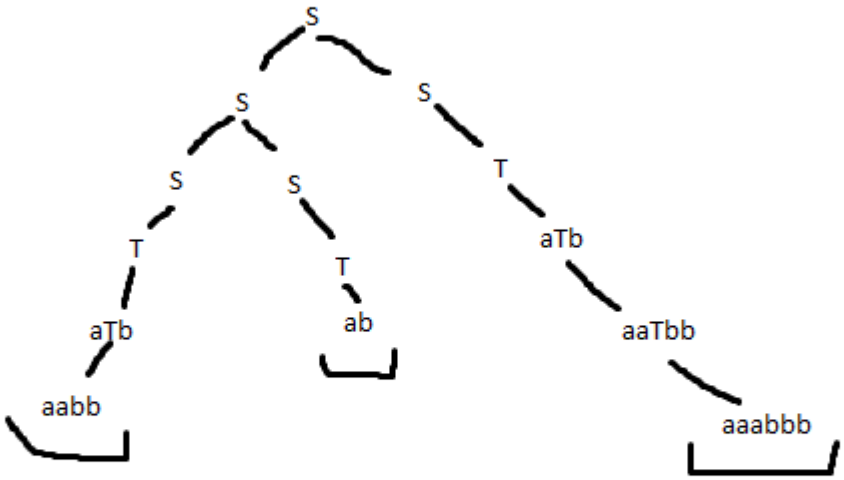
a) Give an English description of the language associated with the grammar.

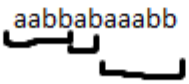
Solution. The language of equal number of a's and b's in which each sub-string of a(s) is followed by a equal length sub-string of b(s).

b) Provide a string and two leftmost derivations of it to show the grammar above is ambiguous.

Solution. String: aabbabaaabbb







Derivation 1:

c) Convert the grammar above to an equivalent CFG in Chomsky normal form.

Solution.

$$
\begin{aligned}
S0 &\to SS \mid YX \mid YZ \\
S &\to SS \mid YX \mid YZ \\
T &\to YX \mid YZ \\
X &\to TZ \\
Y &\to a \\
Z &\to b
\end{aligned}
$$

Question 4. This question will have you complete a proof that all regular languages are context-free by showing that any regular expression can be equivalently described by a context-free grammar. The core *content* of this question was discussed in class on Friday. The goal of this question is for you to review and understand how that content fits into a proof about the relationship between the two classes of languages.

The NFA→PDA arrow was discussed on Wednesday as being a consequence of the PDA's memory stack generalizing NFAs (since any NFA can be written as a PDA that never touches its stack), and the PDA↔CFG equivalence will be discussed later this unit. This question gives a different illustration of how context-free languages subsume regular languages, by providing a proof of the dotted regex→CFG arrow that is inspired by the regex→NFA construction we saw last unit. Most of the proof is below. Fill in what's missing!

Claim: Any regular expression has an equivalent context-free grammar.

Proof overview: For any regex R, we will give rules for a CFG over the same alphabet Σ associated with the same language. The proof follows the inductive structure of the definition of a regex, constructing a CFG for each type of regex and showing that for any regex length, there is an equivalent CFG.

Base cases: We give an equivalent CFG for each regex of length 1. The regex definition gives three cases.

1. $R = a$ for some $a \in \Sigma$. This is equivalent to the CFG with a single rule $S \to a$.
2. $R = \epsilon$. This is equivalent to the CFG with a single rule $S \to \epsilon$.
3. $R = \emptyset$. This is equivalent to the trivial CFG with no rules.

Inductive step: Fix any regex length $k > 1$. We will show that if any regex of length less than k has an equivalent CFG, then we can construct an equivalent CFG for any regex of length k. By the definition of a regex and the inductive hypothesis, any regex R of length k is the union, concatenation, or star of one or two regexes R_1 and R_2 of length less than k, which by inductive hypothesis have equivalent CFGs. In the following case analysis, assume the CFG associated with R_1 is called G_1 and has start variable S_1, and assume the CFG associated with R_2 is called G_2 and has start variable S_2. Assume without that the variables of the two CFGs are distinct; this assumption is without loss of generality, because overlapping variable names can be renamed in one of the grammars. We handle each case separately:

1. $R = (R_1 \cup R_2)$. This is equivalent to the CFG that consists of all rules and variables from each of G_1 and G_2 with a new start variable S and two additional rules $\boxed{S \to S_1 \mid S_2}$.

   Equivalence is because S derives a string derived from either S_1 or S_2; the presence of each rule in the two underlying grammars ensures that any string derivable by G_1 or G_2 is still derivable in the new grammar, and there are no other rules to derive any other strings.

2. $R = (R_1 \circ R_2)$. This is equivalent to the CFG that consists of all rules and variables from each G_1 and G_2 with a new start variable S and an additional rule $\boxed{S \to S_1 S_2}$.

   Equivalence is because S derives a string derived from S_1 followed by S_2; the presence of each rule in the two underlying grammars ensures that any string derivable by G_1 followed by G_2 is still derivable in the new grammar, and there are no other rules to derive any other strings.

3. $R = (R_1^*)$. This is equivalent to the CFG that consists of all rules and variables from G1 with a new start variable S and the new rule $\boxed{S \to S_1 S_1 \mid \epsilon}$.

   Equivalence is because S derives a string from S1, which can be repeated any number of times; the presence of each rule in the grammar ensures that any connected strings that are separably derivable by G1 is derivable by the new grammar.