CS 181: Formal Languages and Automata Theory **Spring 2021**

Homework 8 Solution

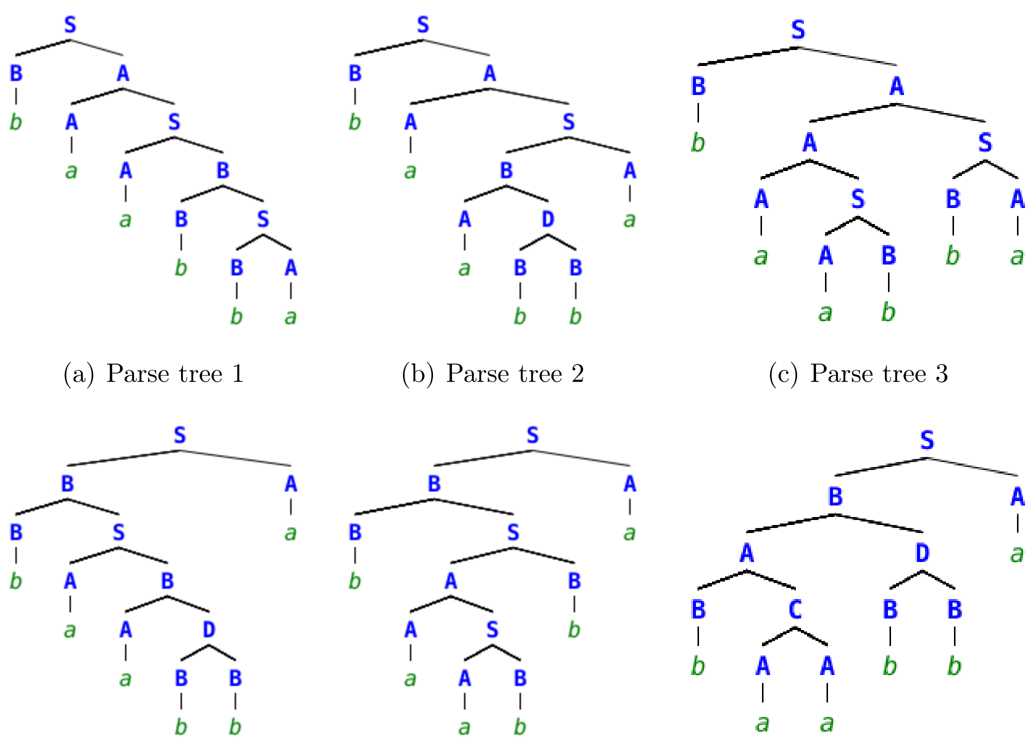
Assigned: Thursday 20 May *Due: Thursday 27 May 3:00pm PDT*

Problem 1

Let $\Sigma = \{a, b\}$. Show two different left-most reductions of the string "baabba" in the following context-free grammar:

$$
\begin{aligned}
S &\longrightarrow AB \mid BA \\
A &\longrightarrow AS \mid BC \mid a \\
B &\longrightarrow BS \mid AD \mid b \\
C &\longrightarrow AA \\
D &\longrightarrow BB
\end{aligned}
$$

Here, the variable set is $\{S, A, B, C, D\}$, and the start variable is S.
Solution:
For instructional purpose, we first show six parse trees for the string *baabba* in Figure 1:



(a) Parse tree 1 (b) Parse tree 2 (c) Parse tree 3

(d) Parse tree 4 (e) Parse tree 5 (f) Parse tree 6

Figure 1: Parse trees for the string *baabba*

Next, we show the left-most reduction for each parse tree, as follows:

1. The left-most reduction for the parse tree 1 in Figure 1(a):

$$\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow BAA\underline{b}ba \\
&\longrightarrow BAAB\underline{b}a \\
&\longrightarrow BAABB\underline{a} \\
&\longrightarrow BAAB\underline{BA} \\
&\longrightarrow BAA\underline{BS} \\
&\longrightarrow BA\underline{AB} \\
&\longrightarrow B\underline{AS} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}$$

2. The left-most reduction for the parse tree 2 in Figure 1(b):

$$\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow BAA\underline{b}ba \\
&\longrightarrow BAAB\underline{b}a \\
&\longrightarrow BAA\underline{BB}a \\
&\longrightarrow BA\underline{AD}a \\
&\longrightarrow BAB\underline{a} \\
&\longrightarrow BA\underline{BA} \\
&\longrightarrow B\underline{AS} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}$$

3. The left-most reduction for the parse tree 3 in Figure 1(c):

$$\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow BAA\underline{b}ba \\
&\longrightarrow BA\underline{AB}ba \\
&\longrightarrow B\underline{AS}ba \\
&\longrightarrow BA\underline{b}a \\
&\longrightarrow BAB\underline{a} \\
&\longrightarrow BA\underline{BA} \\
&\longrightarrow B\underline{AS} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}$$

4. The left-most reduction for the parse tree 4 in Figure 1(d):

$$\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow BAA\underline{b}ba \\
&\longrightarrow BAAB\underline{b}a \\
&\longrightarrow BAA\underline{BB}a \\
&\longrightarrow BA\underline{AD}a \\
&\longrightarrow B\underline{AB}a \\
&\longrightarrow \underline{BS}a \\
&\longrightarrow B\underline{a} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}$$

5. The left-most reduction for the parse tree 5 in Figure 1(e):

$$\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow BAA\underline{b}ba \\
&\longrightarrow BA\underline{AB}ba \\
&\longrightarrow B\underline{AS}ba \\
&\longrightarrow BA\underline{b}a \\
&\longrightarrow B\underline{AB}a \\
&\longrightarrow \underline{BS}a \\
&\longrightarrow B\underline{a} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}$$

6. The left-most reduction for the parse tree 6 in Figure 1(f):

$$
\begin{aligned}
\underline{b}aabba &\longrightarrow B\underline{a}abba \\
&\longrightarrow BA\underline{a}bba \\
&\longrightarrow B\underline{AA}bba \\
&\longrightarrow \underline{BC}bba \\
&\longrightarrow A\underline{b}ba \\
&\longrightarrow AB\underline{b}a \\
&\longrightarrow A\underline{BB}a \\
&\longrightarrow \underline{AD}a \\
&\longrightarrow B\underline{a} \\
&\longrightarrow \underline{BA} \\
&\longrightarrow S
\end{aligned}
$$

Problem 2

Let L_P be a Recursively Enumerable (R.E.) language, and let L_A be a Recursive language.

a. Prove that $L_P \cup L_A$ is R.E. by showing a construction using the Universal TM (UTM) for a TM procedure that recognizes the union.

b. *Briefly* explain why we cannot say that the union is Recursive. You may use the style of argument discussed in class. You do *not* have to prove it.

Solution:

a. Pseudo-code for TM procedure, M, using UTM to simulate M_A and then M_P:

0. On input string w, M will:

1. Use UTM to simulate M_A on input w.

2. If simulation of M_A halts and accepts, then M halts and accepts.

3. If simulation of M_A halts and rejects, then continue to next step.

4. Use UTM to simulate M_P on input w.

5. If simulation of M_P halts and accepts, then M halts and accepts.

6. If simulation of M_P halts and rejects, then M halts and rejects.

If the second simulation never halts, then M will correctly not accept input w.

b. Brief explanation: Since we are given only that L_P is R.E., we do not know whether it is Recursive. If L_P is R.E & Not Recursive, then M_P can only be a procedure, not an algorithm. On inputs which are not in L_A and not in L_P, the simulation of M_A will halt and reject; so M will try simulating M_P. It is possible that M_P will never halt, since it is a procedure. Thus, M will go into an infinite loop. This cannot be avoided because there is no way, in general, to turn a procedure into an algorithm.