CS/ECE 374  Fall 2018
Homework 3
Due Wednesday, September 26, 2018 at 10am

Groups of up to three people can submit joint solutions. Each problem should be submitted by exactly one person, and the beginning of the homework should clearly state the Gradescope names and email addresses of each group member. In addition, whoever submits the homework must tell Gradescope who their other group members are.

The following unnumbered problems are not for submission or grading. No solutions for them will be provided but you can discuss them on Piazza.

- Let $L$ be an arbitrary regular language.
  - Prove that the language $\text{palin}(L)\{w | ww^R \in L\}$ is also regular.
  - Prove that the language $\text{drome}(L)\{w | w^R w \in L\}$ is also regular.

- Suppose $F$ is a fooling set for a language $L$. Argue that $F$ cannot contain two distinct string $x, y$ where both are not prefixes of strings in $L$.

- Prove that the language $\{0^i1^j | \gcd(i,j) = 1\}$ is not regular.

- Consider the language $L = \{w : |w| = 1 \mod 5\}$. We have already seen that this language is regular. Prove that any DFA that accepts this language needs at least 5 states.

- Consider all regular expressions over an alphabet $\Sigma$. Each regular expression is a string over a larger alphabet $\Sigma' = \Sigma \cup \{\text{-Symbol}, \epsilon\text{-Symbol}, +, (, )\}$. We use $\text{-Symbol}$ and $\epsilon$-Symbol in place of $;$ and $\epsilon$ to avoid confusion with overloading; technically one should do it with $+, (, )$ as well. Let $R_{\Sigma'}$ be the language of regular expressions over $\Sigma$.
  1. Prove that $R_{\Sigma'}$ is not regular.
  2. Prove that $R_{\Sigma'}$ is a CFL by giving a CFG for it.

1. (a) Prove that the following languages are not regular by providing a fooling set. You need to provide an infinite set and also prove that it is a valid fooling set for the given language.
   i. $L = \{0^i1^j2^k | i + j = k + 1\}$.
   ii. Recall that a block in a string is a maximal non-empty substring of indentical symbols.
      Let $L$ be the set of all strings in $\{0, 1\}^*$ that contain two non-empty blocks of 1s of unequal length. For example, $L$ contains the strings 01101111 and 010011100010 but does not contain the strings 000110011011 and 0000000111.
   iii. $L = \{0^n^3 | n \geq 0\}$.

(b) Suppose $L$ is not regular. Prove that $L \setminus L'$ is not regular for any finite language $L'$. Give a simple example of a non-regular language $L$ and a regular language $L'$ such that $L \setminus L'$ is regular.
2. Describe a context-free grammar for the following languages. Clearly explain how they work and the role of each non-terminal. Unclear grammars will receive little to no credit.

(a) \{a^ib^jc^k \mid k = 3(i + j)\}.
(b) \{a^ib^jc^kd^\ell \mid i,j,k,\ell \geq 0 \text{ and } i + \ell = j + k\}.
(c) \(L = \{0,1\}^* \setminus \{0^n12^n \mid n \geq 0\}\). In other words the complement of the language \(\{0^n12^n \mid n \geq 0\}\).

3. Given languages \(L_1\) and \(L_2\) we define \(insert(L_1,L_2)\) to be the language \(\{uvw \mid v \in L_1, uw \in L_2\}\) to be the set of strings obtained by “inserting” a string of \(L_1\) into a string of \(L_2\). For example if \(L_1 = \{isfun\}\) and \(L_2 = \{0,CS\}\) then

\[insert(L_1,L_2) = \{isfun0,0isfun,0isfunCS,CSisfun,CSisfun\}\]

- The goal is to show that if \(L_1\) and \(L_2\) are regular languages then \(insert(L_1,L_2)\) is also regular. In particular you should describe how to construct an NFA \(N = (Q,\Sigma,\delta,s,A)\) from two DFAs \(M_1 = (Q_1,\Sigma,\delta_1,s_1,A_1)\) and \(M_2 = (Q_2,\Sigma,\delta_2,s_2,A_2)\) such that \(L(N) = insert(L(M_1),L(M_2))\). You do not need to prove the correctness of your construction but you should explain the ideas behind the construction (see lab 3 solutions).

- Not to submit: Describe an algorithm that given regular expressions \(r_1\) and \(r_2\) constructs a regular expression \(r\) such that \(L(r) = insert(L(r_1),L(r_2))\). Note that you would need to do this from the inductive definitions of \(r_1\) and \(r_2\).

Solved problem

4. Let \(L\) be the set of all strings over \(\{0,1\}^*\) with exactly twice as many 0s as 1s.

(a) Describe a CFG for the language \(L\).

[Hint: For any string \(u\) define \(\Delta(u) = \#(0,u) - 2\#(1,u)\). Introduce intermediate variables that derive strings with \(\Delta(u) = 1\) and \(\Delta(u) = -1\) and use them to define a non-terminal that generates \(L\).]

**Solution:**

\[S \rightarrow \varepsilon \mid SS \mid 0S1 \mid 01S0 \mid 1S00\]

(b) Prove that your grammar \(G\) is correct. As usual, you need to prove both \(L \subseteq L(G)\) and \(L(G) \subseteq L\).

[Hint: Let \(u_{\leq i}\) denote the prefix of \(u\) of length \(i\). If \(\Delta(u) = 1\), what can you say about the smallest \(i\) for which \(\Delta(u_{\leq i}) = 1\)? How does \(u\) split up at that position? If \(\Delta(u) = -1\), what can you say about the smallest \(i\) such that \(\Delta(u_{\leq i}) = -1\)?]

**Solution:** We separately prove \(L \subseteq L(G)\) and \(L(G) \subseteq L\) as follows:

**Claim 1.** \(L(G) \subseteq L\), that is, every string in \(L(G)\) has exactly twice as many 0s as 1s.
Proof: As suggested by the hint, for any string \( u \), let \( \Delta(u) = \#(0,u) - 2\#(1,u) \). We need to prove that \( \Delta(w) = 0 \) for every string \( w \in L(G) \).

Let \( w \) be an arbitrary string in \( L(G) \), and consider an arbitrary derivation of \( w \) of length \( k \). Assume that \( \Delta(x) = 0 \) for every string \( x \in L(G) \) that can be derived with fewer than \( k \) productions.\(^1\) There are five cases to consider, depending on the first production in the derivation of \( w \).

- If \( w = \varepsilon \), then \( \#(\varepsilon) = 0 \) and \( \#(1,\varepsilon) = 0 \) by definition, so \( \Delta(w) = 0 \).
- Suppose the derivation begins \( S \rightarrow SS \rightarrow^* w \). Then \( w = xy \) for some strings \( x, y \in L(G) \), each of which can be derived with fewer than \( k \) productions. The inductive hypothesis implies \( \Delta(x) = \Delta(y) = 0 \). It immediately follows that \( \Delta(w) = 0 \).\(^2\)
- Suppose the derivation begins \( S \rightarrow 00S1 \rightarrow^* w \). Then \( w = 00x1 \) for some string \( x \in L(G) \).
- Suppose the derivation begins \( S \rightarrow 1S00 \rightarrow^* w \). Then \( w = 1x00 \) for some string \( x \in L(G) \).
- Suppose the derivation begins \( S \rightarrow 0S1S1 \rightarrow^* w \). Then \( w = 0x1y0 \) for some strings \( x, y \in L(G) \). The inductive hypothesis implies \( \Delta(x) = \Delta(y) = 0 \). It immediately follows that \( \Delta(w) = 0 \).

In all cases, we conclude that \( \Delta(w) = 0 \), as required. \(\square\)

Claim 2. \( L \subseteq L(G) \); that is, \( G \) generates every binary string with exactly twice as many \( 0 \)s as \( 1 \)s.

Proof: As suggested by the hint, for any string \( u \), let \( \Delta(u) = \#(0,u) - 2\#(1,u) \). For any string \( u \) and any integer \( 0 \leq i \leq |u| \), let \( u_i \) denote the \( i \)th symbol in \( u \), and let \( u_{\leq i} \) denote the prefix of \( u \) of length \( i \).

Let \( w \) be an arbitrary binary string with twice as many \( 0 \)s as \( 1 \)s. Assume that \( G \) generates every binary string \( x \) that is shorter than \( w \) and has twice as many \( 0 \)s as \( 1 \)s. There are two cases to consider:

- If \( w = \varepsilon \), then \( \varepsilon \in L(G) \) because of the production \( S \rightarrow \varepsilon \).
- Suppose \( w \) is non-empty. To simplify notation, let \( \Delta_i = \Delta(w_{\leq i}) \) for every index \( i \), and observe that \( \Delta_0 = \Delta_{|w|} = 0 \). There are several subcases to consider:
  - Suppose \( \Delta_i = 0 \) for some index \( 0 < i < |w| \). Then we can write \( w = xy \), where \( x \) and \( y \) are non-empty strings with \( \Delta(x) = \Delta(y) = 0 \). The induction hypothesis implies that \( x, y \in L(G) \), and the production rule \( S \rightarrow SS \) implies that \( w \in L(G) \).
  - Suppose \( \Delta_i > 0 \) for all \( 0 < i < |w| \). Then \( w \) must begin with \( 00 \), since otherwise \( \Delta_1 = -2 \) or \( \Delta_2 = -1 \), and the last symbol in \( w \) must be \( 1 \), since otherwise \( \Delta_{|w|-1} = -1 \). Thus, we can write \( w = 00x1 \) for some binary string \( x \). We easily observe that \( \Delta(x) = 0 \), so the induction hypothesis implies \( x \in L(G) \), and thus the production rule \( S \rightarrow 00S1 \) implies \( w \in L(G) \).
  - Suppose \( \Delta_i < 0 \) for all \( 0 < i < |w| \). A symmetric argument to the previous case implies \( w = 1x00 \) for some binary string \( x \) with \( \Delta(x) = 0 \). The induction hypothesis implies \( x \in L(G) \), and thus the production rule \( S \rightarrow 1S00 \) implies \( w \in L(G) \).

\(^1\)Alternatively: Consider the shortest derivation of \( w \), and assume \( \Delta(x) = 0 \) for every string \( x \in L(G) \) such that \( |x| < |w| \).
\(^2\)Alternatively: Suppose the shortest derivation of \( w \) begins \( S \rightarrow SS \rightarrow^* w \). Then \( w = xy \) for some strings \( x, y \in L(G) \). Neither \( x \) or \( y \) can be empty, because otherwise we could shorten the derivation of \( w \). Thus, \( x \) and \( y \) are both shorter than \( w \), so the induction hypothesis implies... We need some way to deal with the decompositions \( w = \varepsilon \cdot w \) and \( w = w \cdot \varepsilon \), which are both consistent with the production \( S \rightarrow SS \), without falling into an infinite loop.
Finally, suppose none of the previous cases applies: $\Delta_i < 0$ and $\Delta_j > 0$ for some indices $i$ and $j$, but $\Delta_j \neq 0$ for all $0 < i < |w|$.

Let $i$ be the smallest index such that $\Delta_i < 0$. Because $\Delta_j$ either increases by 1 or decreases by 2 when we increment $j$, for all indices $0 < j < |w|$, we must have $\Delta_j > 0$ if $j < i$ and $\Delta_j < 0$ if $j \geq i$.

In other words, there is a unique index $i$ such that $\Delta_{i-1} > 0$ and $\Delta_i < 0$. In particular, we have $\Delta_1 > 0$ and $\Delta_{|w|-1} < 0$. Thus, we can write $w = 0x1y0$ for some binary strings $x$ and $y$, where $|0x1| = i$.

We easily observe that $\Delta(x) = \Delta(y) = 0$, so the inductive hypothesis implies $x, y \in L(G)$, and thus the production rule $S \rightarrow 0S1S0$ implies $w \in L(G)$.

In all cases, we conclude that $G$ generates $w$. ☐

Together, Claim 1 and Claim 2 imply $L = L(G)$. ■

**Rubric:** 10 points:
- part (a) = 4 points. As usual, this is not the only correct grammar.
- part (b) = 6 points = 3 points for $\subseteq$ + 3 points for $\supseteq$, each using the standard induction template (scaled).