CS 181 Homework 3

Charles Zhang, 305-413-659

April 17, 2021

Problem 1

Proof (by contradiction):

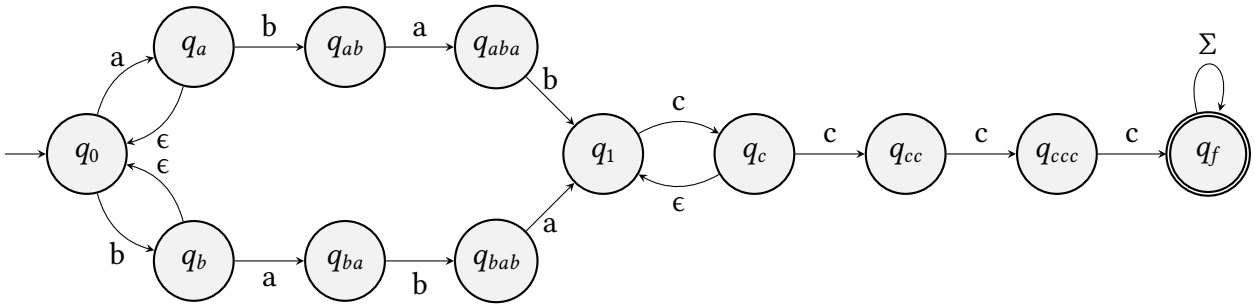
- Assume L is regular.
- Let p be the pumping length given by the pumping lemma.
- Let s be the string $a^{2p}b^p$.
- With s being a member of C and having length more than p, the pumping lemma guarantees that s can be split into three pieces, $s = xyz$, where for any $i \geq 0$, the string xy^iz is in C.
- Sipser's condition 3 of the pumping lemma states that when pumping s, it must be divided so that $|xy| \leq p$.
- Since s begins with $2p$ occurrences of a, this means that xy must be made up of entirely as, otherwise $|xy| \geq p$.
- Since xy is made up entirely of as, it logically follows that y must be made up entirely of as.
- Since the language is specified as $a^{(2n)}b^n$, where n must represent the same number in both occurrences, we can conclude that a string w must contain exactly twice as many as as bs for it to be a member of L.
- Since y is guaranteed to be made up of all as, a string s' that is pumped such that $i = 2$ will not be a member of the language, as the number of as will increase, but the number of bs will not, so it can be said that the number of occurrences of b is not equal to exactly 2 times the number of occurrences of a.
- Therefore, the string s cannot be pumped, which contradicts the original assumption that L is regular by the pumping lemma.$\Longrightarrow\Longleftarrow$

Problem 2

$$(a(a \cup b \cup c)^*(b \cup c)) \cup (b(a \cup b \cup c)^*(a \cup c)) \cup (c(a \cup b \cup c)^*(a \cup b))$$

This regular expression splits L_2 into three parts: strings that begin with a, strings that begin with b, and strings that begin with c. In each of these parts, it takes in the starting symbol S, 0 or more occurrences of any symbol in Σ, followed by an occurrence of E, where $E \in \Sigma - S$. The union of these three cases is then used to form the regular expression. This ensures both that the starting and ending symbol cannot match and that the string has a length greater than 1.

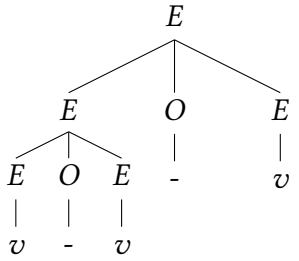
Problem 3



This state diagram essentially splits L_3 into two parts: the part that detects the substrings $abab$ and $baba$, and the part that detects the substring $cccc$. The part on the left detects either $abab$ or $baba$, with ϵ-edges returning from q_a and q_b to q_0. These ϵ-edges allow us to use nondeterminism to continually "guess and check" where the $abab$ or $baba$ appear. This same logic is then repeated for the right half of the state diagram, but checking the substring $cccc$ instead. Since this substring can occur anytime after one of $abab$ or $baba$ is found, we use another ϵ-edge to return to q_1 in order to guess and check where the $cccc$ starts. Once $cccc$ is found, the rest of the string doesn't matter, as all requirements for the string being in the language have been met, so all following inputs will continuously loop into the accept state q_f.

Problem 4

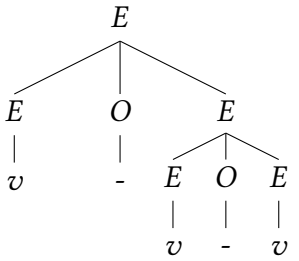
a)



b)

$\underline{E}$
$\underline{E}OE$
$\underline{E}OEOE$
$v\underline{O}EOE$
$v - \underline{E}OE$
$v - v\underline{O}E$
$v - v - \underline{E}$
$v - v - v$

c)



d)

$\underline{E}$
$\underline{E}OE$
$v\underline{O}E$
$v - \underline{E}$
$v - \underline{E}OE$
$v - v\underline{O}E$
$v - v - \underline{E}$
$v - v - v$

Problem 5

$$A \longrightarrow bBe;$$

$$B \longrightarrow s; C \mid bBe; C$$

$$C \longrightarrow s; C \mid bBe; C \mid \epsilon$$

Problem 6

$$A \longrightarrow bBe$$

$$B \longrightarrow sC \mid bBeC$$

$$C \longrightarrow, sC \mid , bBeC \mid \epsilon$$