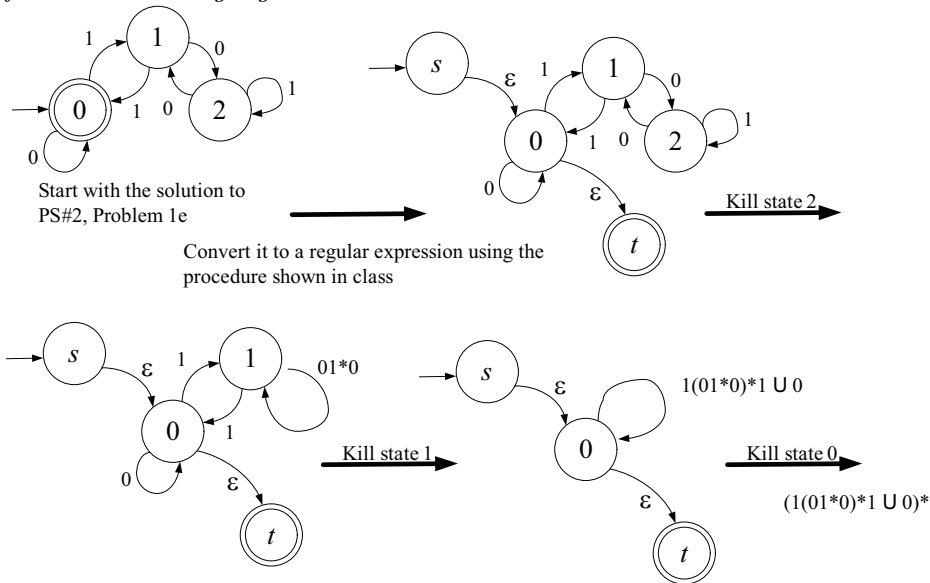


Problem Set 4 Solutions

Problem 1. Find a regular expression representing the encoding of binary numbers divisible by 3. Show your work in systematically devising this regular expression, starting from a DFA for the same language.



Problem 2. Suppose you have a (fully parenthesized, concatenation-explicit) regular expression of α of length n over the binary alphabet. Exhibit (and justify) an explicit bound $b(n)$ such that there is a regular expression β , $|\beta| \leq b(n)$, such that $L(\beta) = \overline{L(\alpha)}$.

This problem is an accounting headache (the tighter a bound you want, the worse the headache). In the solution below I will, for clarity, keep renaming “ n ” as the size of the thing I am starting with. I’ll write a function f_i , for the blowup for step i . At the end we’ll compose all of these f_i ’s to get our upper bound.

First convert the length- n regular expression α into an NFA. How many states will that NFA have? Each character $a \in \Sigma$ of α accounts for one character of α and yields two states; each character ε (the formal symbol) in α accounts for one character of α and yields two states; each character \emptyset (the formal symbol) in α accounts for one character in α and yields one state; each $(\beta \cup \gamma)$ in α accounts for three symbols (the left parenthesis, the right parenthesis, and the union symbols) and adds one state; each $(\beta \circ \gamma)$ accounts for three symbols of α and adds no states; and each (β^*) in α accounts for three symbols and adds one state. So each character of α gives rise to 0, 1/3, 1, or 2 states of the NFA. So, at worst, each character of α gives rise to two states in the NFA. So a length n regular expression gives rise to an NFA having at most $2n$ states. (This bound can of course be improved but, for simplicity, we won’t do so.) Let $f_1(n) = 2n$, the first piece of the blowup.

Renaming, convert an n -state NFA into a DFA for the same language. This blows up the number of states to 2^n . Let $f_2(n) = 2^n$.

Next, convert the n -state DFA into a DFA for the complement of the language. This preserves the number of states.

Next we need to convert our DFA into a regular expression. Renaming, our initial DFA has n states and the labels have length 1. We add two new states, to get an $n + 2$ state NFA with labels of maximal length 1. Let $f_3(n) = n + 2$.

Again renaming, let n be the number of states in the current GNFA. We now kill at most n states. Let's consider what happens when we kill a state.

Let ℓ be the current maximal length of a label on an arc. When we kill a state we add in at most n^2 new arcs before collapsing parallel arcs. After adding in all these shortcuts the maximal label length will be $3\ell + 9$ since AB^*C means $((A \circ (B^*)) \circ C)$, which requires 9 extra symbols. After adding in all of these arcs we must combine parallel arcs. Since there will be at most n^2 parallel arcs to combine, and since combining arcs adds the lengths of their labels plus 3 more, we'll end up with labels that are at most $n^2 \cdot (3\ell + 9 + 3) = 3n^2\ell + 12n^2 \leq 15n^2\ell$. That is, killing a state increases the maximal length of a label by (at most) a multiplicative factor of $15n^2$. So if we do this n times the maximal label length will be at most $15n^3$. Let $f_4(n) = 15n^3$.

The label that we're left with when there is only a single arc is the regular expression we're after. So its length is at most $f_4(f_3(f_2(f_1(n)))) = 15(2^{2n} + 2)^3$.

Of course your bound will vary according to how you did the accounting.

Problem 3. (Assigned last week) For $n \geq 0$, let $L_n = \{1^i : 0 \leq i < n\}$ (where $1^0 = \varepsilon$). Prove that there is a DFA M_n having n final states that accepts L_n . Then prove that L_n cannot be accepted by any DFA having fewer accept states.

This problem was postponed for one week; solution to be supplied next week. The first part is easy (draw the machine). For the second, we must show that L_n is not accepted by any DFA having only $n - 1$ accept states. This is proven by contradiction, using the pigeonhole principle. Suppose that, for some n , there exists a DFA $M = (Q, \Sigma, \delta, q_0, F)$ such that $L(M) = L_n$ and $|F| \leq n - 1$. Consider the n strings $\{\varepsilon, 1, 11, \dots, 1^{n-1}\}$. All of these strings must be accepted by M : $\delta^*(q_0, 1^i) \in F$ for each $0 \leq i \leq n - 1$. By the pigeonhole principle, there exists some $0 \leq i < I \leq n - 1$ such that $\delta^*(q_0, 1^i) = \delta^*(q_0, 1^I)$. But in that case $\delta^*(q_0, 1^{I+k(I-i)}) \in F$, for any k , so $L(M)$ is infinite and $L(M) \neq L_n$.

Problem 4. Show that the following languages are not regular.

Part A. $L = \{www : w \in \{a, b\}^*\}$.

Assume for contradiction that L were regular. Let N be the pumping length, as guaranteed by the pumping lemma. Let $s = a^N b a^N b a^N b$. Then $s \in L$ and $|s| \geq N$ so, by the pumping lemma, there exists x, y, z such that $s = xyz$ and $|xy| \leq N$ and $y \neq \varepsilon$ and $xy^i z \in L$ for all $i \geq 0$. In particular, pumping down, $xy^0 z = xz = a^{N-\alpha} b a^N b a^N b$ must be in L , where $\alpha = |y| > 0$. But this string is clearly not in L , a contradiction.

Part B. $L = \{a^{2^n} : n \geq 0\}$.

Assume for contradiction that L were regular. Let N be the pumping length, as guaranteed by the pumping lemma. Let $s = a^{2^N}$. Then $s \in L$ and $|s| \geq N$ so, by the pumping lemma,

there exists x, y, z such that $s = xyz$ and $y \neq \varepsilon$ and $xy^iz \in L$ for all $i \geq 0$. In particular, for some $\alpha \geq 1$ (namely, $\alpha = |y|$), we have that $a^{2^N+i\alpha} \in L$ for all $i \geq 0$, which means that $2^N + i\alpha$ is always a power of two, for any $i \geq 0$. Thus (looking at $i = 1$) we have that $2^N + \alpha$ is a power of two, and (looking at $i = 2$) we have that $2^N + 2\alpha$ is a bigger power of two, so it must be at least twice $2^N + \alpha$; that is, $2^N + 2\alpha \geq 2(2^N + \alpha)$, which means that $2^N + 2\alpha \geq 2^N + 2^N + 2\alpha$, so $0 \geq 2^N$, which is impossible.

Part C. $L = \{0^n 1^m 0^n : m, n \geq 0\}$.

Not regular. Assume for contradiction that L were regular. Let N be the pumping length, as guaranteed by the pumping lemma. Let $s = 0^N 10^N$. Then $s \in L$ and $|s| \geq N$ so, by the pumping lemma, there exists x, y, z such that $s = xyz$ and $|xy| \leq N$ and $y \neq \varepsilon$ and $xy^iz \in L$ for all $i \geq 0$. In particular, pumping down, for some $\alpha \geq 1$ (namely, $\alpha = |y|$), we have that $0^{N-\alpha} 10^N \in L$. But this string is not in L , a contradiction.