Harvard University
Computer Science 121

Problem Set 2

Due Tuesday, September 24, 2013 at 11:59 PM.
Submit your solutions electronically to cs121+ps2@seas.harvard.edu with "ps2 submission" in the subject line. The solutions to each part should be attached as separate PDF files, called lastname+ps2a.pdf, lastname+ps2b.pdf, and lastname+ps2c.pdf.
Late problem sets may be turned in until Friday, September 27, 2013 at 11:59 PM with a 20% penalty.

Problem set by ** ENTER YOUR NAME HERE **

Collaboration Statement: **FILL IN YOUR COLLABORATION STATEMENT HERE (See the syllabus for information)**

See syllabus for collaboration policy.

PART C (Graded by Joy)

PROBLEM 1 (1+7+7 points)

Let Σ and Δ be alphabets. Consider a function $\varphi : \Sigma \to \Delta^*$. Extend φ to a function from $\Sigma^* \to \Delta^*$ such that:

$$\begin{aligned} \varphi(\varepsilon) &= \varepsilon \\ \varphi(w\sigma) &= \varphi(w)\varphi(\sigma), \text{ for any } w \in \Sigma^*, \sigma \in \Sigma \end{aligned}$$

Any function $\varphi : \Sigma^* \to \Delta^*$ defined in this way from a function $\varphi : \Sigma \to \Delta^*$ is called a **homomorphism**. We can extend this definition to languages as follows: for any language L and homomorphism φ, let

$$\varphi(L) = \{\varphi(w) : w \in L\}.$$

(A) Let $\Sigma = \{a, b\}$, $\Delta = \{a, b, c, d\}$, and $\varphi(a) = abc$ and $\varphi(b) = cbd$. If $L = \{aa, abb, bab\}$, what is $\varphi(L)$?

(B) Prove that the set of regular languages is closed under homomorphism. Specifically, prove that $\varphi(L)$ is regular for any regular language L and homomorphism φ.

(C) Prove that the set of regular language is also closed under inverse homomorphism. Specifically, prove that $\varphi^{-1}(L) = \{w : \varphi(w) \in L\}$ is regular for any regular language L and homomorphism φ.

(A) $\varphi(L) = \{abcabc, abccbdcbd, cbdabccbd\}$.
(B) These proofs can be done with either regular expressions or finite automata. We will use regular expressions for this part and finite automata for the next part, as examples.

Let R be a regular expression for L. Then, we construct a new regular expression R' by replacing each $\sigma \in \Sigma$ in R by $\varphi(\sigma)$. We claim that $L(R) = \varphi(L')$, which shows that $\varphi(L)$ is regular. Our proof is by structural induction:

- $R = \varepsilon$. In this case, $R' = \varphi(\varepsilon) = \varepsilon$, and $L(R') = \{\varepsilon\} = \varphi(L(R))$.
- $R = \emptyset$. In this case, $R' = \emptyset$ and $L(R') = \emptyset = \varphi(L(R))$.
- $R = \sigma$, where $\sigma \in \Sigma$. In this case, $R' = \varphi(\sigma)$ and $L(R') = \{\varphi(\sigma)\} = \varphi(L(R))$.
- $R = S_1 \cup S_2$, where S_1 and S_2 are some two regular expressions. Suppose for induction that, if S_1' is the regular expression resulting from applying φ to each σ in S_1, then $L(S_1') = \varphi(L(S_1))$. Suppose the same holds for S_2 and S_2'. In this case, $R' = S_1' \cup S_2'$, so

$$\begin{aligned} L(R') &= L(S_1') \cup L(S_2') \\ &= \varphi(L(S_1)) \cup \varphi(L(S_2)) \\ &= \varphi(L(S_1) \cup \varphi(L(S_2)) \\ &= \psi(L(R)). \end{aligned}$$

- $R = S_1 \cdot S_2$. Make the same inductive assumption and let $R' = S_1' \cdot S_2'$. Then

$$\begin{aligned} L(R') &= \{wx : x \in L(S_1'), y \in L(S_2')\} \\ &= \{\varphi(y)\varphi(z) : w \in L(S_1), z \in L(S_2)\} \\ &= \{\varphi(u) : u \in L(S_1 \cdot S_2)\} \\ &= \varphi(L(R)). \end{aligned}$$

- $R = S_1^*$. Make the same inductive assumption and let $R' = S_1'^*$. Then,

$$\begin{aligned} L(R) &= \{w : w \in S_1'^*\} \\ &= \{\varphi(x) : x \in S_1\}^* \\ &= \psi(L(R)). \end{aligned}$$

(C) Since L is regular, there exists some DFA for L. We construct a DFA M for $\varphi^{-1}(L)$ as follows:

- Use the same starting state, set of states, and set of final states as in the DFA for L.
- For a state q and string s, denote by $\delta(q, s)$ the state we reach in the DFA for L after processing the string s, starting from the state q.

  Then, to construct M, for every state q and character σ, we set $\delta_M(q, \sigma) = \delta(q, \varphi(\sigma))$.

Now, we show that this NFA accepts exactly $\varphi^{-1}(L)$. In particular, note that by the second set of our construction, $\delta(q, \varphi(s)) = \delta_M(q, s)$ for any state q and string s.

First, we show that M accepts every string in $\varphi^{-1}(L)$. Let s be a string in $\varphi^{-1}(L)$. Then, we know that $\varphi(s) \in L$. Consequently, $\delta_M(q_0, s) = \delta(q_0, \varphi(s)) \in F$, so M accepts s.

Next, we show that every string accepted by M is in $\varphi^{-1}(L)$. Suppose that s is a string accepted by M. Then, we know that $\delta(q_0, \varphi(s)) = \delta_M(q_0, s) \in F$, so the DFA for L accepts $\varphi(s)$. Hence, $s \in \varphi_{-1}(L)$.

PROBLEM 2 (2 bonus points)

CHALLENGE: If L is a subset of $\{a\}^*$, show that L^* is regular.

If $L = \emptyset$ or $L = \{e\}$, then $L^* = \{e\}$, which is regular.

Otherwise, L has a string of length ≥ 1. Let $N = \{n : a^n \in L^*\}$, which will contain at least one non-zero number. Let k be the smallest non-zero number in N.

The idea here is that all multiples of k will have to be in N, and if anything that gives a remainder of j when divided by k is in N, everything larger that gives a remainder of j when divided by k will also have to be in N (we can keep concatenating a^k on without leaving L^*). This means that we will be able to "exhaust" L^* by a finite union of regular expressions. More formally:

For each $0 < j < k$, let m_j be the smallest $n \in N$ that gives a remainder of j when divided by k, 0 if there are no such n.

Let $\alpha = (a^k)^* \bigcup a^{m_1}(a^k)^* \bigcup a^{m_2}(a^k)^* \bigcup \cdots \bigcup a^{m_{k-1}}(a^k)^*$.

Claim: $L(\alpha) = (L^*)^*$

Proof:

($\Rightarrow$) Given $w \in L(\alpha)$, we know that $w \in L((a^k)^*)$ or $w \in L(a^{m_j}(a^k)^*)$ for some j. So, either $w = a^k a^k \cdots a^k$ (0 or more times) or $w = a^{m_j} a^k a^k \cdots a^k$. In either case, w is a number of things in L^* concatenated together and is thereby in $(L^*)^*$.

($\Leftarrow$) Given $w \in (L^*)^*$. w is also in L^*, because for any language, $L^* = (L^*)^*$. Thus, $w = a^n$ for some $n \in N$. Let j be the remainder resulting when n is divided by k. If j=0, then $w \in L((a^k)^*)$, and thereby in $L(\alpha)$. Otherwise, $n = m_j + ik$ for some $i \geq 0$, because m_j is the smallest $n \in N$ that gives a remainder of j when divided by k (since n gives a remainder of j when divided by j, m_j can't be 0). So, $w = a^n = a^{m_j+ik} = a^{m_j}(a^k)^i \in L(a^{m_j}(a^k)^*)$, which implies that $w \in L(\alpha)$.