Midterm 1

CS 373: Theory of Computation

Date: Thursday, September 30, 2010.

Instructions:

- This is a closed book exam. No notes, cheat sheets, textbook, or printed material allowed.
- You have 120 minutes to solve this exam.
- This exam has 5 problems each worth 10 points. However, not all problems are of equal difficulty.
- Please write your name on the top of *every* page in the space provided.
- If your solution does not fit in the space provided, and continues onto one of the back sheets, please indicate clearly where we should look for the solution.
- Unless otherwise stated, recall that "prove that", "show that" for a problem means you need to formally prove what you are claiming.
- Answering "I don't know" for a problem *does not receive any points.*
- You may use, without proof, any result that you were asked to prove in the homework or was proved in the lecture. If you use such a result, please explicitly state the result you are using (like " 'Reverse of a regular language is regular' was proved in a homework", rather than saying "this was shown in a homework").

Name	SOLUTIONS
Netid	solutions

Problem	Maximum Points	Points Earned	Grader
1	10		
2	10		
3	10		
4	10		
5	10		
Total	50		

Problem 1. [Category: Comprehension] **True/False.** Decide for each statement whether it is true or false. Circle **T** if the statement is *necessarily true*; circle **F** if it it is not necessarily true. Each correct answer is worth **1 point**.

(a) For languages $L_1 = L(0^*11^*)$ and $L_2 = L(0^*)$, define $L_1/L_2 = \{w \mid \exists x.\ wx \in L_1 \text{ and } x \in L_2\}$. Then $010 \in L_1/L_2$.
T **F**
False. There is no x such that $010x \in L_1$.

(b) $L = \{0^n1^m0^n \mid m < 12 < n\}$ is regular.
T **F**
False. Observe that $L \cap 0^*10^* = \{0^n10^n \mid n > 12\}$. Further, $L_1 = \{0^n10^n \mid n \leq 12\}$ is a finite language (and hence regular). Finally, take $h(0) = a$ and $h(1) = b$, then $h((L \cap 0^*10^*) \cup L_1) = \{a^nba^n \mid n \geq 0\}$ which was proved to be non-regular in the lectures.

(c) For language L_1 and L_2 over the alphabet Σ, $L_1 \setminus L_2$ denotes the difference between the two sets, i.e., it is the set of all strings that belong to L_1 but not L_2. If L_1 and L_2 are regular then $L_1 \setminus L_2$ is regular.
T **F**
True. $L_1 \setminus L_2 = L_1 \cap \overline{L_2}$ and so is regular by closure properties.

(d) A language $L \subseteq \Sigma^*$ is said to be *co-finite* if its complement $(\Sigma^* \setminus L)$ is a finite set. If L is co-finite then L is regular.
T **F**
True. Since $\overline{L}$ is finite, it is regular, and so $\overline{\overline{L}} = L$ is regular.

(e) If $L \subseteq \{0\}^*$ then L is regular.
T **F**
False. $L = \{0^p \mid p \text{ is a prime}\}$ was shown to be non-regular in class.

(f) Though we proved in homework 2 that the reverse of a regular language is regular, *non-regular* languages may or may not be closed under this operation. That is, if L is *not* regular then L^R may or may not be regular.
T **F**
False. If L^R is regular, then ${L^R}^R = L$ must be regular.

(g) Though regular languages are closed under homomorphism, *non-regular* languages may or may not be closed under homomorphisms. That is, if L is *not* regular and h is a homomorphism then $h(L)$ may or may not be regular.
T **F**
True. Consider $L = \{0^n1^n \mid n \geq 0\}$. Take $h_1(0) = a$ and $h_1(1) = \epsilon$ then $h_1(L) = L(a^*)$ which is regular. On the other hand, for $h_2(0) = 0$ and $h_2(1) = 1$, $h_2(L) = L$ which is non-regular.

(h) Let $L = L_1 \cup L_2$ where $L_1 \cap L_2 = \emptyset$. If L_1 is regular and L_2 is not regular, then L is not regular.
T **F**
True. Observe that $L \cap \overline{L_1} = L_2$ is not regular, and L is not regular by closure properties.

(i) The class of *finite* languages is closed under homomorphism. That is, for any homomorphism $h : \Sigma^* \to \Delta^*$, if L is a finite language in Σ^*, then $h(L)$ is a finite language.
T **F**
True. If $L = \{w_1, \ldots w_n\}$ then $h(L) = \{h(w_1), \ldots h(w_n)\}$ and so has the same cardinality as L.

(j) Suppose $L_1, L_2, \ldots$ is an infinite sequence of regular languages over alphabet Σ. Then their union, i.e., $\cup_i L_i$, is also regular.
T **F**
False. Take $L_i = \{0^i 1^i\}$. L_i is regular because it is finite, but $\cup_i L_i = \{0^n 1^n \mid n \geq 1\}$ is not regular. Note, in class what we showed was that the union of *finite* number of languages is regular.

Problem 2. [Category: Comprehension+Design] Let $\Sigma = \{T, F, p, q, \neg, \vee, \wedge\}$. The set of *well formed (unparenthesized) Boolean formulas* (WFF) is a language over Σ defined inductively as follows:

- T, F, p, and q are elements of WFF
- If x is in WFF then $\neg x$ is in WFF
- If x and y are in WFF then $x \vee y$ and $x \wedge y$ are in WFF

(a) Which of the following strings are in WFF: ϵ, $\neg\neg\neg p \vee \neg q \wedge F$, $\neg p \wedge T \wedge T \vee F$, $p \wedge \neg$. **[2 Points]**

(b) Design a DFA with at most 3 states recognizing WFF. You need not prove the correctness of your construction, but your construction should be clear. **[4 Points]**

(c) Construct a regular expression for the language WFF. You can either translate the DFA constructed in the previous part, showing each step clearly, or write down the regular expression directly, but with a clear explanation for why it might be correct. **[4 Points]**

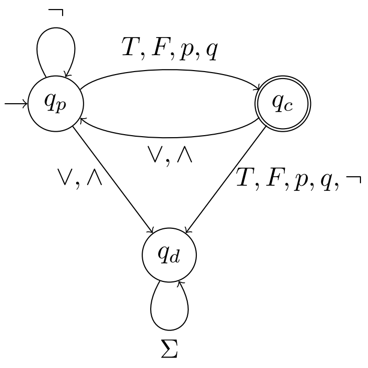
Solution:

(a) $\neg\neg\neg p \vee \neg q \wedge F$, and $\neg p \wedge T \wedge T \vee F$ are in WFF.

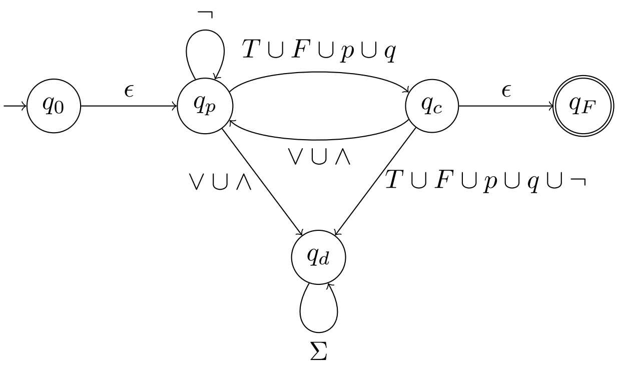
(b) The 3 states will be

- q_c to remember that what has been seen so far is a formula
- q_p to remember that what has been so far is a prefix of a formula
- q_d to remember that we have seen an error

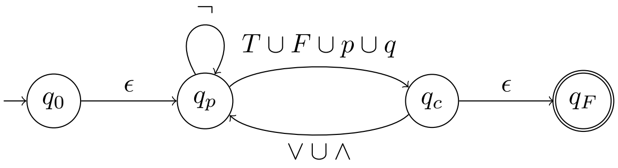
The DFA is shown below.



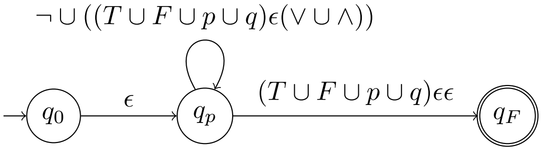
(c) First we will convert the above to a GNFA as follows.



We first remove q_d to get



Next we remove q_c to get


$\neg \cup ((T \cup F \cup p \cup q)\epsilon(\vee \cup \wedge))$
q_0
ϵ
q_p
$(T \cup F \cup p \cup q)\epsilon\epsilon$
q_F

Finally, we remove q_p to get the expression $r = (\neg \cup ((T \cup F \cup p \cup q)(\vee \cup \wedge)))^*(T \cup F \cup p \cup q)$.

■

Problem 3. [Category: Comprehension+Design+Proof] Given languages L_1, L_2, and L_3, $\text{maj}(L_1, L_2, L_3)$ is the collection of all strings that belong to at least 2 out of L_1, L_2, and L_3. We can prove that if L_1, L_2, and L_3 are regular then $L = \text{maj}(L_1, L_2, L_3)$ is also regular as follows: Given DFAs M_1, M_2, and M_3 recognizing L_1, L_2, and L_3, respectively, the DFA for L will run M_1, M_2, and M_3 simultaneously (as in the cross-product construction for intersection) on input w, and accept if at least two out of M_1, M_2, and M_3 accept.

Complete the following proof of this closure property based on the above intuition. Let $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, with $L(M_1) = L_1$; $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$, with $L(M_2) = L_2$; and, $M_3 = (Q_3, \Sigma, \delta_3, q_3, F_3)$, with $L(M_3) = L_3$. The DFA recognizing $L = \text{maj}(L_1, L_2, L_3)$ is given by $M = (Q, \Sigma, \delta, q_0, F)$, where

(a) $Q = \underline{Q_1 \times Q_2 \times Q_3}$ **[1 Point]**

(b) $q_0 = \underline{(q_1, q_2, q_3)}$ **[1 Point]**

(c) $F = \underline{\{(p_1, p_2, p_3) \mid \text{either } (p_1 \in F_1 \text{ and } p_2 \in F_2) \text{ or } (p_1 \in F_1 \text{ and } p_3 \in F_3) \text{ or } (p_2 \in F_2 \text{ and } p_3 \in F_3)\}}$ **[1 Point]**

(d) For a state (p_1, p_2, p_3) of M and $a \in \Sigma$, $\delta((p_1, p_2, p_3), a) = \underline{(\delta_1(p_1, a), \delta_2(p_2, a), \delta_3(p_3, a))}$ **[1 Point]**

(e) Taking $\hat{\delta} : Q \times \Sigma^* \to Q$, $\hat{\delta}_1 : Q_1 \times \Sigma^* \to Q_1$, $\hat{\delta}_2 : Q_2 \times \Sigma^* \to Q_2$, $\hat{\delta}_3 : Q_3 \times \Sigma^* \to Q_3$, to be the functions capturing the computations of M, M_1, M_2, and M_3, respectively on a string, the correctness of this construction can then be established by proving that for any $w \in \Sigma^*$ **[1 Point]**

$$\hat{\delta}(q_0, w) = \underline{(\hat{\delta}_1(q_1, w), \hat{\delta}_2(q_2, w), \hat{\delta}_3(q_3, w))}$$

(f) Prove by induction on the length of w, the statement in part (e). **[4 Points]**

Base Case: When $|w| = 0$, we have

$$\hat{\delta}(q_0, \epsilon) = q_0 = (q_1, q_2, q_3) = (\hat{\delta}_1(q_1, \epsilon), \hat{\delta}_2(q_2, \epsilon), \hat{\delta}_3(q_3, \epsilon))$$

Induction Step: Suppose the statement in part (e) holds for all w of length n. Consider $w = ua$, where $a \in \Sigma$ and $u \in \Sigma^*$ of length n. Then,

$$\begin{aligned}
\hat{\delta}(q_0, ua) &= \delta(\hat{\delta}(q_0, u), a) && \text{(defn. of } \hat{\delta}\text{)} \\
&= \delta((\hat{\delta}_1(q_1, u), \hat{\delta}_2(q_2, u), \hat{\delta}(q_3, u)), a) && \text{(ind. hyp.)} \\
&= (\delta_1(\hat{\delta}_1(q_1, u), a), \delta_2(\hat{\delta}_2(q_2, u), a), \delta_3(\hat{\delta}_3(q_3, u), a)) && \text{(defn. of } \delta\text{)} \\
&= (\hat{\delta}_1(q_1, ua), \hat{\delta}_2(q_2, ua), \hat{\delta}_3(q_3, ua)) && \text{(defn. of } \hat{\delta}_1, \hat{\delta}_2, \hat{\delta}_3\text{)}
\end{aligned}$$

(g) Prove that $L(M) = \text{maj}(L_1, L_2, L_3)$. **[1 Point]**

$$\begin{aligned}
w \in L(M) \quad &\text{iff} \quad \hat{\delta}(q_0, w) = (p_1, p_2, p_3) \text{ and } (p_1, p_2, p_3) \in F && \text{(defn. of acceptance)} \\
&\text{iff} \quad \hat{\delta}(q_0, w) = (p_1, p_2, p_3) \text{ and either } (p_1 \in F_1 \text{ and } p_2 \in F_2) \\
&\qquad \text{or } (p_1 \in F_1 \text{ and } p_3 \in F_3) \text{ or } (p_2 \in F_2 \text{ and } p_3 \in F_3) && \text{(defn. of } F\text{)} \\
&\text{iff} \quad \hat{\delta}_1(q_1, w) = p_1 \text{ and } \hat{\delta}_2(q_2, w) = p_2 \text{ and } \hat{\delta}_3(q_3, w) = p_3 \text{ and} \\
&\qquad \text{either } (p_1 \in F_1 \text{ and } p_2 \in F_2) \text{ or } (p_1 \in F_1 \text{ and } p_3 \in F_3) \\
&\qquad \text{or } (p_2 \in F_2 \text{ and } p_3 \in F_3) && \text{(part (e))} \\
&\text{iff} \quad \text{either } (w \in L_1 \text{ and } w \in L_2) \text{ or } (w \in L_1 \text{ and } w \in L_3) \\
&\qquad \text{or } w \in L_2 \text{ and } w \in L_3
\end{aligned}$$

Problem 4. [Category: Proof] Consider the following language L over the alphabet $\{0, 1, \#\}$.

$$L = \{x\#y \mid x, y \in \{0, 1\}^+ \text{ and } y \text{ is the binary number equal to } x + 1\}$$

(Recall, $\{0, 1\}^+$ is the set of all binary strings not including ϵ.) Thus, $1110\#1111 \in L$, because $1111 = 15 = 1 + 14 = 1 + 1110$, and $0000\#1 \in L$ because $1 = 1 + 0 = 1 + 0000$. On the other hand, $111\#110 \notin L$ because $110 = 6 \neq 1 + 7 = 1 + 111$. (Also, badly formed strings like $\#, 1\#1\#$, etc. are not in L.) Prove that L is not regular. **[10 Points]**

Solution: Pumping Lemma Proof: Let p be the pumping length. Consider $w = 10^p\#10^{p-1}1$. Observe that $|w| > p$ and $w \in L$. Let $x, y, z \in \{0, 1, \#\}^*$ be such that $w = xyz$, $|xy| \leq p$ and $|y| > 0$. We consider two cases; in both cases we will show that $xy^0z \notin L$.

- *Case $x = \epsilon$:* In this case we have $x = \epsilon$, $y = 10^r$, and $z = 0^s\#10^{p-1}1$. Now, $xy^0z = 0^s\#10^{p-1}1 \notin L$.
- *Case $x \neq \epsilon$:* In this case we have $x = 10^r$, $y = 0^s$ and $z = 0^t\#10^{p-1}1$, with $s > 0$. Then, $xy^0z = 10^{r+t}\#10^{p-1}1 \notin L$ as $r + t < p$ and so $10^{r+t} + 1 = 10^{r+t-1}1 \neq 10^{p-1}1$.

Closure Properties Proof: Let $L_1 = L \cap L(10^+\#10^*1) = \{10^n\#10^{n-1}1 \mid n \geq 1\}$. Take $h_1 : \{a, b, c, d\}^* \to \{0, 1, \#\}^*$ such that $h_1(a) = h_1(b) = 0$, $h_1(c) = 1$ and $h_1(\#) = d$. Then, $L_2 = h^{-1}(L_1) = \{c(a \cup b)^n dc(a \cup b)^{n-1}c | n \geq 1\}$. Now $L_3 = L_2 \cap L(ca^*dcb^*c) = \{ca^n dcb^{n-1}c | n \geq 1\}$. Finally, taking $h_2 : \{a, b, c, d\}^* \to \{0, 1\}^*$, where $h_2(a) = 0$, $h_2(b) = h_2(d) = 1$, and $h_2(c) = \epsilon$, we get that $h_2(L_3) \cup \{\epsilon\} = \{0^n1^n \mid n \geq 0\}$ which is known to non-regular. ■

Problem 5. [Category: Design] For languages A and B over alphabet Σ, let the *shuffle* of A and B be the language

$$\{w \mid w = a_1b_1a_2b_2\cdots a_nb_n \text{ where } a_i, b_i \in \Sigma^* \text{ and } a_1a_2\cdots a_n \in A,\ b_1b_2\cdots b_n \in B\}$$

Show that the class of regular languages is closed under shuffle. While you need not prove the correctness of your construction, it should be clearly explained with a formal definition. **[10 Points]**

Note: In homework 4, you proved that regular languages are closed under *perfect shuffle*. In perfect shuffle, symbols from a string in A strictly alternate with symbols from a string in B. In shuffle, however, *sub-strings* (possibly empty) of a string from A alternate with *sub-strings* (possibly empty) of a string from B. For example, if $A = \{11\}$ and $B = \{00\}$ then the perfect shuffle of A and B is $\{1010\}$ while the shuffle of A and B is $\{1100, 1010, 1001, 0011, 0110, 0101\}$.

Solution: There are two possible ways to solve this problem — using closure properties, or by an explicit construction of an automaton for shuffle(A, B).

Closure Properties: Let $\widehat{\Sigma} = \{a, \bar{a} \mid a \in \Sigma\}$. So $\widehat{\Sigma}$ contains for each symbol a of Σ, the symbols itself and a "copy" (which is $\bar{a}$). For example $\widehat{\{0,1\}} = \{0, \bar{0}, 1, \bar{1}\}$. Consider homomorphism, $h_A : \widehat{\Sigma}^* \to \Sigma^*$ such that $h(a) = a$ and $h(\bar{a}) = \epsilon$ for $a \in \Sigma$. Now let us consider $L_A = h_A^{-1}(A)$. The strings in L_A are such that if we look at the "unbarred" symbols then they form a string in A. Similarly, define $h_B : \widehat{\Sigma}^* \to \Sigma^*$ such that $h(a) = \epsilon$ and $h(\bar{a}) = a$ for $a \in \Sigma$. Then, take $L_B = h_B^{-1}(B)$; so L_B contains strings such that if you look at the "barred" symbols and remove the bars then you get a string in B.

Let $L = L_A \cap L_B = \{w \mid h_A(w) \in A \text{ and } h_B(w) \in B\}$; that is, the unbarred symbols in w form a string in A, and the barred symbols form a string in B (after removing the bar). L is almost the shuffle of A and B — the only difference it has the barred symbols, which we will remove next. Consider $h : \widehat{\Sigma}^* \to \Sigma^*$ such that $h(a) = h(\bar{a}) = a$ for $a \in \Sigma$. Now it is easy to see that $h(L) = \text{shuffle}(A, B)$. Thus, shuffle of A and B is regular because we obtained it from A and B by applying regularity preserving operations.

By construction: Let $M_A = (Q_A, \Sigma, \delta_A, q_A, F_A)$ be a DFA recognizing A and $M_B = (Q_B, \Sigma, \delta_B, q_B, F_B)$ be a DFA recognizing B. The NFA for shuffle of A and B will simulate both M_A and M_B on the input, while nondeterministically choosing which machine to run on a particular input symbol; note, unlike in the perfect shuffle case, we don't need to strictly alternate running the machines and so the construction is actually simpler. So the NFA will be obtained by a "modified" cross-product construction.

Formally, let $N = (Q, \Sigma, \delta, q_0, F)$ where

- $Q = Q_A \times Q_B$
- $q_0 = (q_A, q_B)$
- $F = F_A \times F_B$
- For $a \in \Sigma$, δ is given as

$$\delta((p_A, p_B), a) = \{(\delta_A(p_A, a), p_B), (p_A, \delta_B(p_B, a))\}$$

In all other cases, δ is $\emptyset$.

The correctness can be established by showing that if N on an input w reaches a state (p_A, p_B) then there is a way to break up w so that running M_A on some of the substrings reaches p_A and running M_B on the remaining substrings reaches p_B. Formally,

$$\begin{aligned}\hat{\Delta}(q_0, w) = \{(p_A, p_B) \mid \quad & \exists a_1, b_1, a_2, b_2, \ldots a_k, b_k.\ a_i, b_i \in \Sigma^*, \\ & \hat{\delta}_A(q_A, a_1 a_2 \cdots a_k) = p_A \text{ and } \hat{\delta}_B(q_B, b_1 b_2 \cdots b_k) = p_B\}\end{aligned}$$

The above observation can be proved by induction on the length of w and can be used to prove the correctness of the construction.

■