Harvard University
Computer Science 121
Midterm Solution 15-Oct-98

Do all problems. The points on the exam total 80. You may use any theorems proven in class or on the homeworks. Unless otherwise specified, let $\Sigma = \{a, b\}$. Good luck!

PROBLEM 1 (4+4+4+4+4+4 points)

True or false? Justify your answer in four or fewer sentences.
(A) $P(\{n \in \mathbb{N} : n \leq 2002\})$ is uncountable. (Recall that $P(A)$ is the power set of A.)
(B) $\{w : w$ is a regular expression for $\{a^n b^m : n + m \leq 2002\}\}$ is a finite language.
(C) The concatenation of a regular language and a non-regular language is necessarily non-regular.
(D) $(L^+)^+ = L^+L^+$.
(E) $\emptyset \subseteq \{\emptyset, a\}$.
(F) If a DFA M contains a self-loop on some state q, then M must accept an infinite language.
(G) Your favorite TF has blonde hair.

(A) False. $|\{n \in \mathbb{N} : n \leq 2002\}|$ is finite. Therefore its power set (all the subsets of it) is also finite. Anything that is finite is countable and therefore not uncountable.

(B) False. The language is finite, and therefore regular, so there exists some regular expression for the language, call it α. But then $(\alpha \cup \alpha)$ is another regular expression for the language. We can union α to itself an arbtrary number of times, to get arbitrarily many different regular expressions for the language.

(C) False. $L = \{a^n b^n : n \geq 0\}$ is not regular, but $\emptyset L = \emptyset$ is.

(D) False. $L^+ = LL^*$. Note that $LL^* = L^*L$ and $(L^*)^* = L^*$. So $(L^+)^+ = (L^+)(L^+)^* = (LL^*)(LL^*)^* = LL^*L^*L^* = LL^*$. But $L^+L^+ = LL^*LL^* = LLL^*$. So let $L = \{a\}$. Then $a \in LL^* = (L^+)^+$, but $a \notin LLL^* = L^+L^+$.

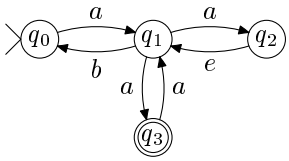
(E) True. $\emptyset$ is the subset of any set. Formally $A \subseteq B$ means that every element of A belongs to B. Since $\emptyset$ has no elements, this is vacuously true.

(F) False. Create a DFA that has no final states, yet has a self loop. Clearly this accepts the language $\emptyset$ which is not infinite.

(G) Free point!

PROBLEM 2 (2+1+6 points)

(A) Consider the adjacent NDFA. Does this NDFA accept a? aa? $abaaa$? aba? [You do not need to show your work.]
(B) Trace an accepting computation of this NDFA on the string aaa.



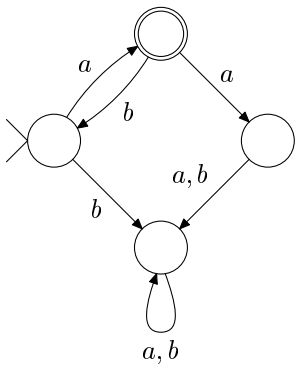
(C) Consider the relation "Is reachable from in one transition or more" over the set of states of this NDFA. Is this relation reflexive? Symmetric? Transitive? An equivalence relation? Which, if any, of these answers would change if the relation were over sets of states of an *arbitrary* NDFA?

(A) No, yes, yes, no.
(B) $(q_0, aaa) \vdash (q_1, aa) \vdash (q_2, a) \vdash (q_1, a) \vdash (q_3, e)$.
(C) It is reflexive, symmetric, and transitive, and thus an equivalence relation. Note that the relation is not the same as the graph of the NDFA. Over an arbitrary NDFA, only transitivity must hold. If state q_c is reachable in one or more transitions from q_b, and q_b is reachable in one or more transitions from q_a, then q_c is reachable in one or more transitions from q_a.

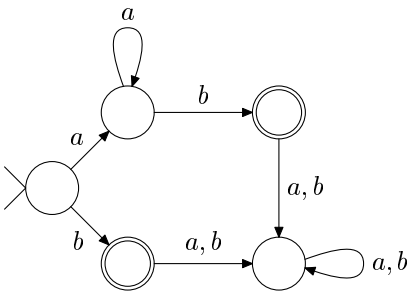
PROBLEM 3 (4+4+4+4 points)

For each of the DFAs drawn below, give both an informal description of and a regular expression for the language it accepts. Do not use the construction described in class: just build the regular expression from your informal description. Your descriptions should be short, and need not be backed by proofs.

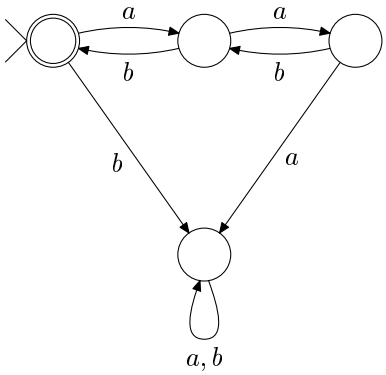
(A)



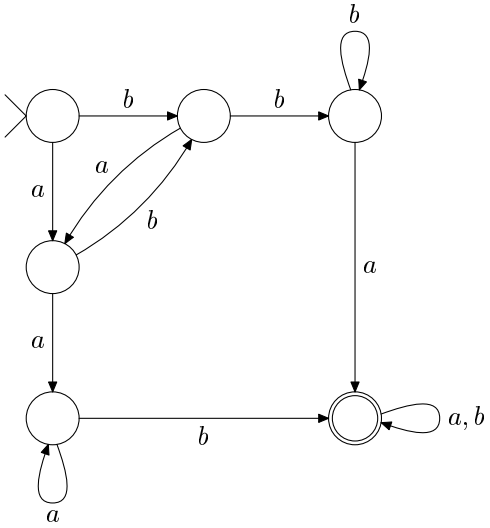
(B)



(C)



(D)



(A) The language accepted is the set of all strings starting with an a followed by any number of ba-pairs. Valid regular expressions include $a(ba)^*$ and $(ab)^*a$.

(B) The language accepted is any number of a's followed by a single b. The regular expression is

a^*b. Note that this is not the smallest DFA that accepts this language.

(C) The language accepted is the set of strings w with the same number of a's and b's and the additional requirement that any prefix of w cannot have more b's than a's, nor can it have more than two more a's than b's. Some regular expressions are $(ab \cup aa(ba)^*bb)^*$ or $(ab \cup a(ab)^*b)^*$, or more concisely, just $(a(ab)^*b)^*$.

(D) The language accepted is all strings containing aab or bba as a substring. The regular expression is $(a \cup b)^*(aab \cup bba)(a \cup b)^*$.