CS 5311 HW 1: Closure Results and the Pumping Lemma Due: Tues., 2/5/13

Read the section about "Closure Under the Regular Operations" at the end of Section 1.2. (Sipser refers to the union, concatenation, and Kleene star of languages as regular operations). Since languages are sets, we may also consider operations such as intersection and complement (with respect to Σ^*), for example.

Note that in Theorem 1.45, Sipser proves that the union of two regular languages is also a regular language by constructing a *nondeterministic* finite automaton N that accepts the union of the languages accepted by two arbitrary deterministic finite automata N_1 and N_2. (We see that the *NFA* constructed in this proof is nondeterministic because of the use of ε-transitions and the transition $\delta(q,a) = \{q_1, q_2\}$ in step 4 of the construction.)

We now consider how to prove the same theorem by constructing a *deterministic* finite automaton and thus avoid appealing to Theorem 1.39 to complete the proof of Theorem 1.45 as given in the text. Let us start with A_1, A_2, N_1, and N_2 as in Theorem 1.45. We construct a deterministic *FA* N' (so named to distinguish it from the *NFA* N constructed in Theorem 1.45) that accepts $A_1 \cup A_2$ as follows.

Proof Idea: In Theorem 1.45 the *NFA* N is constructed so that on its first move it "guesses" whether its input is in A_1 or A_2 by making an ε-move (see Figure 1.46) that starts a simulation of the *DFA* N_1 or N_2, respectively. The idea for our *DFA* N' is to have it simulate the computations of N_1 and N_2 *in parallel*. This is done by making the states of N' ordered pairs of states of N_1 and N_2. Each "move" of N' will advance the computations of N_1 and N_2 by one move. In this way, when N' completes its computation, if either of the states in its "ordered pair" state is a final state of N_1 or N_2, then one of those machines must accept the input string and so the input is in the union of A_1 and A_2 and so N' should accept. If neither of the states in the ordered pair is a final state of N_1 or N_2, then N' will reject.

Proof: Let A_1, A_2, N_1, and N_2 be as in Theorem 1.45. We construct a deterministic *FA* $N' = (Q, \Sigma, q_0, \delta, F)$ that accepts $A_1 \cup A_2$.

1. $Q = Q_1 \times Q_2$. (If you have forgotten what "$\times$" means, see Example 0.5.)

2. The start state of N' is $q_0 = (q_1, q_2)$, that is, the ordered pair consisting of the start states of N_1 and N_2, respectively. (Note that q_0 is the right "data type" because it is an ordered pair and thus a "legal" member of Q.)

3. The set of final states is $F = \{\ (q', q'') \mid q' \in F_1 \text{ or } q'' \in F_2\ \}$. That is, each member of F is an ordered pair of states from N_1 and N_2 where at least one state is a final state of the respective *DFA*.

4. The transition function is $\delta((q', q''), a) = (\delta_1(q', a), \delta_2(q'', a))$ for all $(q', q'') \in Q$ and $a \in \Sigma$. (Mind the parentheses!) Note that the output of δ is an ordered pair of states of N_1 and N_2, respectively, so, again, we are correctly handling the data types that are inolved. In

this manner, N' simulates, in parallel, the computations of N_1 and N_2.

Combining the details of the construction of N' and the explanation given in the Proof Idea, we should be able to see that if N' is given an input $w \in \Sigma^*$, its last state will be in F exactly when either N_1 or N_2 accepts w.

1. Let N_1 and N_2 be the *DFA*s given in Figures 1.6 and 1.8, respectively. Draw the directed graph of the *DFA* N' that is the result of the construction procedure above.

2. In the style of the procedure above, show that the regular languages are closed under the operation of intersection. *Hint:* Only a slight modification of the proof above is needed to solve this problem.

3. Do Problem 1.41 in the style of the procedure above.

4. Use the pumping lemma for regular languages to prove that $L = \{ 0^n 1 0^n \mid n \geq 0 \}$ is not regular.

5. Problem 1.46(c).