CSCI 383 Fall 2009  Proof that a Function of Regular Languages is Regular

Suppose you are asked to prove a statement such as the following:

Show that if \( A \) is regular, then \( f(A) \) is also regular, for some function \( f \).

Show that if \( A \) and \( B \) are regular, then \( f(A, B) \) is also regular, for some function \( f \).

Then you need to show that \( f(A) \) is regular, by defining a finite automata that accepts \( f(A) \).

Main Steps

The main steps for proving a language \( f(A) \) (or, \( f(A, B) \)) is regular is as follows:

**Step 1: Provide notation.** You will need to construct your finite automata for \( f(A) \) from the automaton for \( A \). Therefore, you should provide notation with which to talk about \( A \). That is, you should include the following sentence: “Since \( A \) is regular, by definition there exists a dfa \( M_A = (Q_A, \Sigma, \delta_A, s_A, F_A) \) such that \( L(M_A) = A \).”

**Step 2: Define your new FA.** To show that \( f(A) \) is regular, you need to define an automaton \( M = (Q, \Sigma, \delta, s, F) \) (or, possibly an nfa \( N = (Q, \Sigma, \Delta, S, F) \)) that recognizes \( f(A) \). Here is where you should define the 5 elements of the 5-tuple. Be especially careful that the types match; that is, that \( s \in Q \), \( F \subseteq Q \), \( \delta : Q \times \Sigma \rightarrow Q \), etc.

**Step 3: Give intuition.** As lovely as your definition in step 2 is, it could probably use with a sentence or two of English intuition. If you can, provide that here.

**Step 4: Prove your FA is correct.** For every \( x \in \Sigma^* \), show that \( x \in L(M) \iff x \in f(A) \). This shows that your finite automaton accepts the desired language.

**Step 5: Prove Intermediate Steps.** Did you use a generalized version of \( \hat{\delta} \) that we haven’t actually defined? Did you use some concatenation property that we haven’t proven yet? To be complete, all such intermediate assumptions should have formal proofs, unless specified otherwise.

Comments

- You can possibly avoid this whole process by using closure properties of regular languages. In particular, do you know anything about the complement of \( f(A) \)? Is \( f(A) \) the intersection of two languages you already know are regular? Think about this before you delve into the steps above.

- Be explicit and precise in your definitions. The more complete you are in constructing the FA, the easier your proof will be.

- It may help to informally describe your finite automaton in advance, using pebbles or other techniques. This can help focus what you’re trying to do, and give me an idea as well. An example is helpful, too.
Example

Show that if $A$ and $B$ are regular, then $A \cup B$ is also regular.

(Note: in this situation, $f(A, B) = A \cup B$.)

Proof. If $A$ and $B$ are regular, then there are dfas $M_A = (Q_A, \Sigma, \delta_A, s_A, F_A)$ and $M_B = (Q_B, \Sigma, \delta_B, s_B, F_B)$ that accept $A$ and $B$, respectively, that is, $L(M_A) = A$ and $L(M_B) = B$. We will show $A \cup B$ is regular by constructing a dfa $M = (Q, \Sigma, \delta, s, F)$ that accepts $A \cap B$.

Here is an informal description of $M$. We have two pebbles; one will stay on the dfa $M_A$ and one on the dfa $M_B$. We start with the two pebbles placed on the two start states $s_A$ and $s_B$. On an input string $x$, we move the two pebbles according to the transition function of the respective dfa, moving them in parallel as they would move in their own dfa. At the end of the string, we accept $x$ if and only if at least one of the two pebbles is in a finish state, that is, if at least one of the dfas would have accepted $x$ in its language.

Now let’s do this more formally. Define the dfa $M$ as follows.

$$Q = Q_A \times Q_B$$
$$\delta((p,q),a) = (\delta_A(p,a), \delta_B(q,a)) \quad \forall (p,q) \in Q, a \in \Sigma$$
$$s = (s_A, s_B)$$
$$F = \{(p,q) : p \in F_A \text{ or } q \in F_B\}$$

To complete the proof, we need to show this dfa actually recognizes the language $A \cup B$. We do this by showing that a string $x$ is accepted by $M$ iff $x \in A \cup B$.

$$x \in L(M) \iff \hat{\delta}(s,x) \in F \quad \text{by definition of dfa acceptance}$$
$$\iff \hat{\delta}(s_A, s_B, x) \in F \quad \text{by definition of our dfa } M$$
$$\iff \hat{\delta}(s_A, x) \cup \hat{\delta}(s_B, x) \in F \quad \text{by the lemma below}$$
$$\iff \hat{\delta}_A(s_A, x) \in F_A \text{ or } \hat{\delta}_B(s_B, x) \in F_B \quad \text{by definition of } F$$
$$\iff x \in L(M_A) \text{ or } x \in L(M_B) \quad \text{by definition of dfa acceptance}$$
$$\iff x \in A \cup B \quad \text{by definition of } M_A \text{ and } M_B$$

Lemma 1. For all $x \in \Sigma^*$, $\hat{\delta}(p, q, x) = (\hat{\delta}_A(p, x), \hat{\delta}_B(q, x))$.

Proof. We’ll do this proof by (structural) induction on $x$. As a base case, consider when $x = \epsilon$.

$$\hat{\delta}(p, q, \epsilon) = (p, q) \quad \text{by the recursive definition of } \hat{\delta}$$
$$= (\delta_A(p, \epsilon), \delta_B(q, \epsilon)) \quad \text{by the recursive definition of } \hat{\delta}$$

For the inductive step, suppose the claim is true for some $x \in \Sigma^*$; consider $xa$ where $a \in \Sigma$.

$$\hat{\delta}(p, q, xa) = \delta(\hat{\delta}(p, q, x), a) \quad \text{by the definition of } \hat{\delta}$$
$$= \delta((\hat{\delta}_A(p, x), \hat{\delta}_B(q, x)), a) \quad \text{by the IH}$$
$$= (\delta_A(\hat{\delta}_A(p, x), a), \delta_B(\hat{\delta}_B(q, x), a)) \quad \text{by the definition of } \delta$$
$$= (\hat{\delta}_A(p, xa), \hat{\delta}_B(q, xa)) \quad \text{by the definition of } \hat{\delta}$$