Finite Automata and Regular Expressions - II

Theorem Let r be a regular expression. Then there exists an NFA with $\epsilon-$ transitions accepting $L(r)$

Theorem If L is accepted by a DFA, then L can be represented by a regular expression.

The first theorem is relatively easy. The second theorem uses Dynamic Programming. So, suppose we are given the transition diagram of a DFA. We would like to represent the set of strings accepted by a regular expression. Suppose the states of the DFA are $1, 2, \ldots n$. It suffices to represent for each pair of states i, j, the set of strings which take the DFA from state i to state j (call this set R_{ij}). Now, how does one represent R_{ij} ? If there is a direct transition from i to j, R_{ij} clearly includes this one-letter string. More generally, we define R_{ij}^k to be the set of strings that take the DFA from state i to state j going through no **intermediate** state higher than k. Then the dynamic programming recursion is

$$R_{ij}^k = R_{i,k}^{k-1}(R_{kk}^{k-1})^* R_{kj}^{k-1} \cup R_{ij}^{k-1}.$$

The base case R_{ij}^0 is easy to compute and so one may "compute" (or write a reg. expression) for R_{ij}^k....

Decision algorithms for Finite Automata

Membership Given a DFA and a string x and a DFA M, decide if M accepts x.

Emptiness Given a DFA M decide if it accepts ant string at all.

Finiteness Given a DFA M, decide if $L(M)$ is finite.

All these three questions can be decided in polynomial time - we are now thinking of one fixed algorithm (C-Program) which takes as input a description of (any) DFA M (and any string x) and answers the question in time polynomially bounded in the length of the input. We will later see that such questions will be undecidable for more complicated models. So, we have a big win if we can model questions about Verifying a family of circuits or similar objects as questions regarding Finite Automata or similar restricted models of computation; this is what is often done.

The decision algorithms for DFA are easy to see : For Membership, one may just simulate in quadratic time. For emptiness, we just have to answer the question of whether there is a path in the transition graph from the start state to some final state, a path problem. For finiteness : $L(M)$ is infinite iff there is a cycle in the trasition graph which is reachable from the start state and from which, we can reach a final state.