THOMPSON CONSTRUCTION

Y MANOJ KUMAR
(11CS10055)

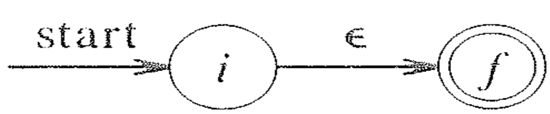
Construction of an NFA from a Regular Expression

INPUT: A regular expression r over alphabet C.

OUTPUT: An NFA N accepting L(r)

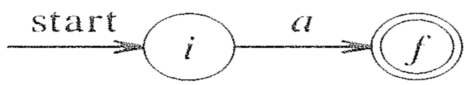
METHOD: Begin by parsing r into its constituent subexpressions. The rules for constructing an NFA consist of basis rules for handling subexpressions with no operators, and inductive rules for constructing larger NFA's from the NFA's for the immediate subexpressions of a given expression.

BASIS: For expression e construct the NFA



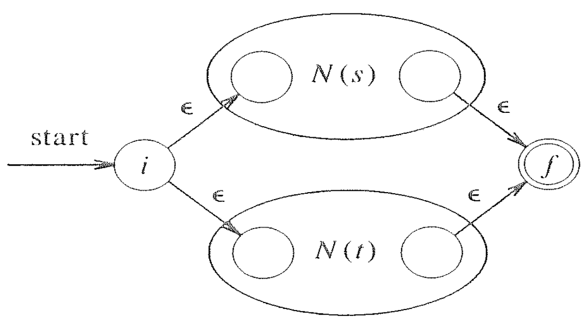
Here, i is a new state, the start state of this NFA, and f is another new state, the accepting state for the NFA.

For any subexpression a in C, construct the NFA

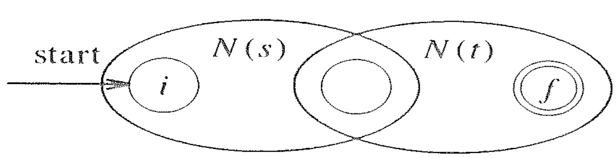


INDUCTION: Suppose N(s) and N (t) are NFA's for regular expressions s and t, respectively.

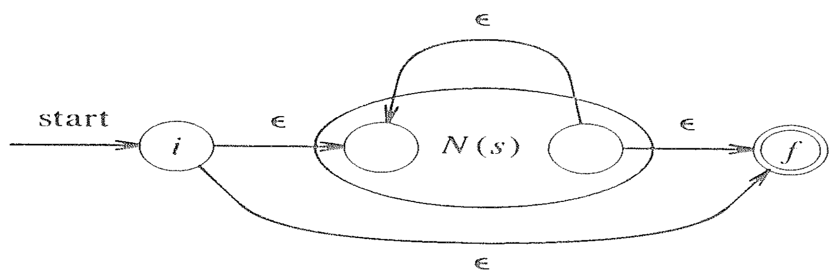
a) For the regular expression s|t,



b) For the regular expression st,



c) For the regular expression S*,

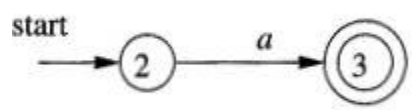


d) Finally, suppose r = (s). Then L(r) = L(s), and we can use the NFA, N(s) as N(r).

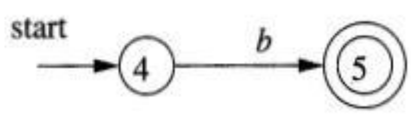
1) N(r) has at most twice as many states as there are operators and operands in *r*. This bound follows from the fact that each step of the algorithm creates at most two new states.
2) N(r) has one start state and one accepting state. The accepting state has no outgoing transitions, and the start state has no incoming transitions.
3) Each state of N (r) other than the accepting state has either one outgoing transition on a symbol in C or two outgoing transitions, both on E.

construct an NFA for r = (a|b)*a

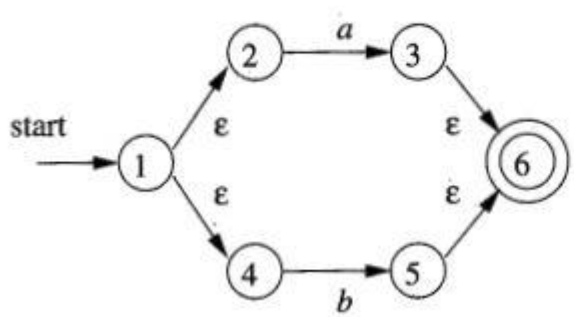
For r_1 = a,



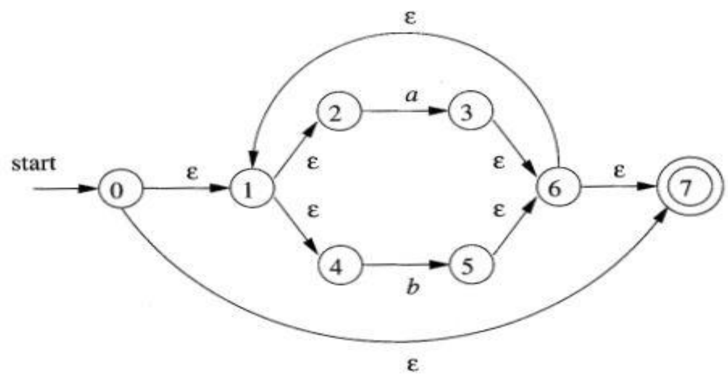
For r_1 = b,



For r_3 = a| b



The NFA for $r_5 = (r_3)$ is the same as that for r3. The NFA for $r_6 = (r_3)^*$



Finally NFA for r = (a|b)*a