SAN JOSE STATE UNIVERSITY
DEPARTMENT OF ELECTRICAL ENGINEERING

CS 154 Formal Languages and Computability Spring 2012
Section 1 Room MH 225 Class 5: 02-08-12

T. Howell .

Regular Expressions

Regular expressions take advantage of the fact that regular languages are closed under the operations of concatenation, union, and asterate (Kleene star). It turns out that these are all the operations we need to describe any regular language, starting with symbols for each member of our alphabet and **ε** and ∅, representing the empty string and the empty set. We write AB for the concatenation of languages A and B and A + B for their union. The asterate of A is A*. Note that **∅* =** {ε}, and **ε*** = {ε}.

Let's talk about these closures. Closure under concatenation means that the concatenation of two regular languages is a regular language. Concatenation means that you take a member of one language and concatenate it with a member of the other. The new language is the set of all strings you can form in this way. To prove that regular languages are closed under concatenation, we just connect NFAs for them, M and N in series. Put ε-transitions from the final states of M to the start state of N.

Closure under union means the union of two regular languages is a regular language. We can build a DFA for the union using the product construction. The product automaton tracks the states of the two component automata in parallel. The start state is the pair (s_M, s_N) where s_M and s_N are the start states for DFAs M and N, respectively. The final states are all pairs (p_M, q_N) where p_M or q_N is a final (accepting) state for DFA M or N, respectively. (Change "or" to "and" in this construction to prove the intersection of regular languages is regular.) The transition function for the product automaton takes (p_M, q_N) to (r_M, s_N) on input a whenever δ_M (p_M, a) = r_M and δ_N (q_N, a) = s_N.

Asterate is a little trickier. The Kleene star (*) of a language L is the infinite union $L^0 \cup L^1 \cup L^2 \cup L^3 \cup L^4 \cup \ldots$. Recall that $L^0 = \varepsilon$. We can make an NFA for L* by adding a new start state. Connect ε-transitions from the new start state to the old one and from the (old) accepting states back to the new start state. A string from L^j is accepted by looping j times through the old NFA for L.

Pattern Matching and Regular Expressions

Previously we had patterns over $\Sigma \cup$ **{ε**, ∅, #, @, +, ∩, ~, *, $^+$, (,)}.

Many of these operators are redundant. This can be useful for writing patterns efficiently, but proving things becomes more difficult where there are many operators.

We can slim down to just concatenation, union, and star. All the others are nice but unnecessary.

We will show that atomic patterns **{ε**, ∅,} and the symbols of Σ along with operators +, ·, and * are sufficient. (Even **ε** could be eliminated since $\varnothing^*$ is equivalent, but we won't do that.) Expressions using these operators are called <u>regular expressions</u>.

Precedence of operators:
To minimize the need for parentheses, we give * highest precedence, concatenation next, and + lowest. This is similar to ordinary arithmetic expressions.

Theorem Let $A \subseteq \Sigma^*$. The following statements are equivalent.

(i) A is regular. This means A = L(M) for some FA M.

(ii) A = L(α) for some regular expression α.

Proof:
We prove (ii) ⇒ (i) now. Next class we prove (i) ⇒ (ii).
All singleton patterns match regular sets. This is the basis for an induction proof. Since regular sets are closed under +, ·, and *, the sets matching expressions built from them are regular. The induction is on the structure of the regular expression, and a fourth case has to be dealt with: (expr).

Example 1:
Convert regular expression to NFA:
$(aaa)^* + (aaaaa)^*$
The NFA resulting from the construction for * is the one we already saw for strings of a's whose length is divisible by 3 or 5.

Example 2: $(11 + 0)^*(00 + 1)^*$
The machine has four states, although the mechanical procedures we have learned would give more.