Regular Expressions and DFAs

See section 3.2 of the text

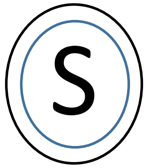
We have already seen the language of Regular Expressions.

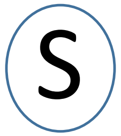
1. The language represented by ε is $\{\varepsilon\}$; the language represented by ϕ is ϕ; any letter a in Σ represents the language {a}
2. If E is a regular expression then so is (E) and it represents the same language as E.
3. If expressions E and F represent languages $\mathcal{L}_1$ and $\mathcal{L}_2$ then expression E+F represents $\mathcal{L}_1 \cup \mathcal{L}_2$.
4. If expressions E and F represent languages $\mathcal{L}_1$ and $\mathcal{L}_2$ then expression EF represents the language of strings formed by concatenating a string from $\mathcal{L}_2$ onto the end of a string from $\mathcal{L}_1$.
5. If expression E represents language $\mathcal{L}$ then expression E^* represents the language of strings formed by concatenating 0 or more strings from $\mathcal{L}$ together.
6. If expression E represents language $\mathcal{L}$ then expression E^+ represents the language of strings formed by concatenating 1 or more strings from $\mathcal{L}$ together. $E^+=EE^*$

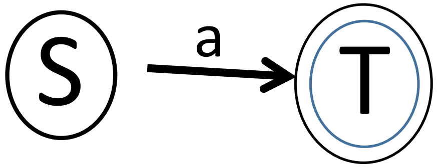
Note that our definition of the language represented by regular expressions is recursive.

Theorem: If E is a regular expression then there is a DFA that accepts the language represented by E.
Proof. Structural induction!!

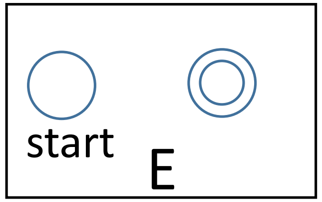
Here are the base cases:

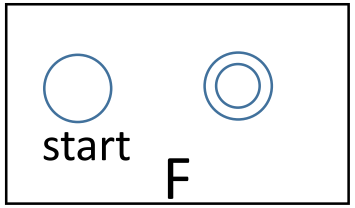
ε: 

φ: 

For any a in Σ: 

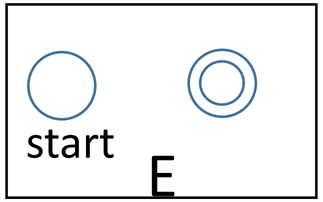
For the inductive cases, suppose E and F are regular expressions whose languages are accepted by the ε-NFAs



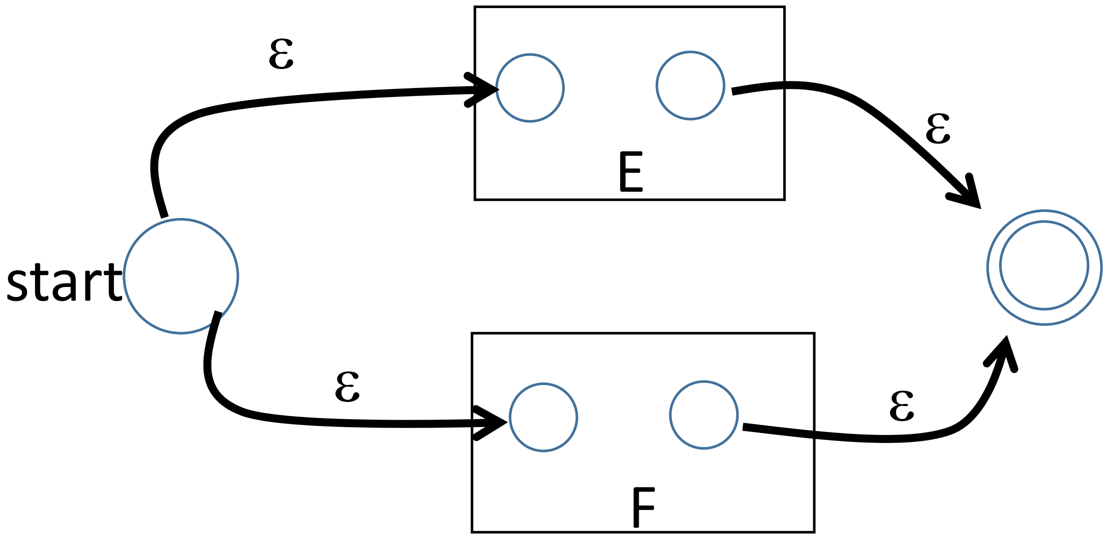


Since these are ε-NFAs we can assume there is only one final state i each automaton and there are no transitions out of it. Here are automata for the expressions we can build from E and F:

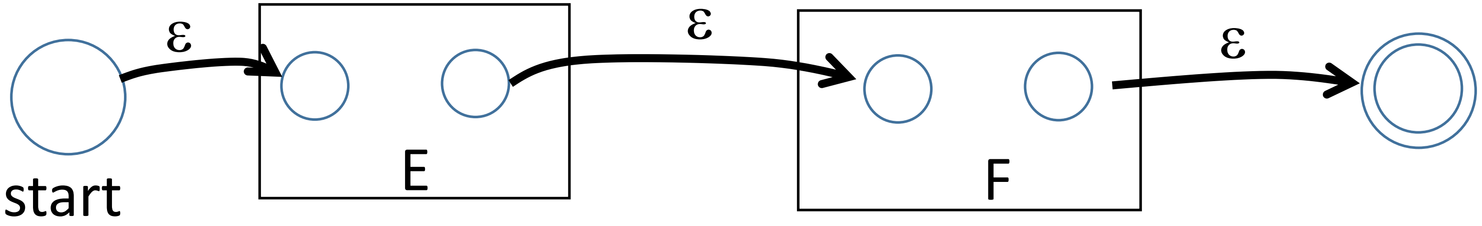
(E):



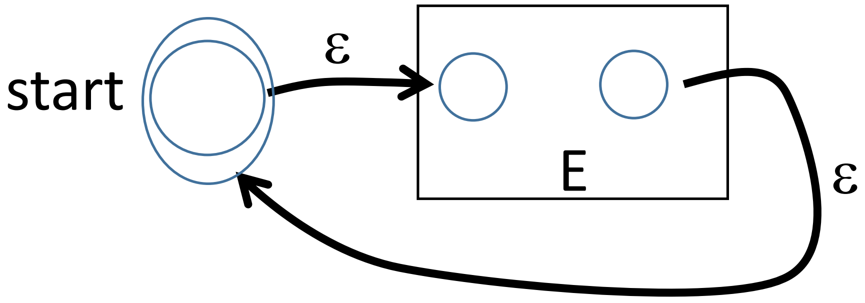
E+F:



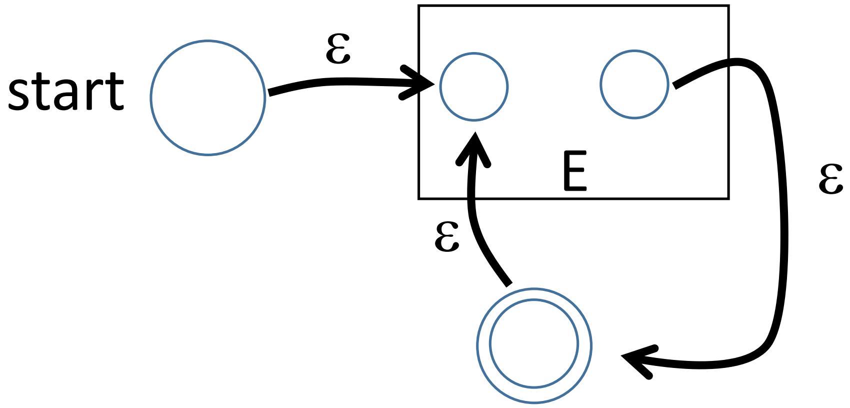
EF:



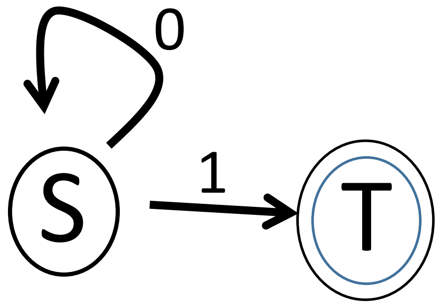
E^*:



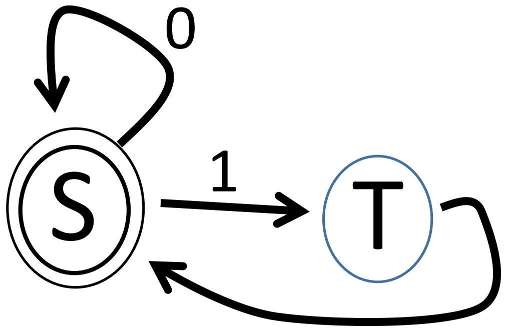
E^+:



For the E^* automaton note that we need a new start state; it isn't enough to just make the start state final:

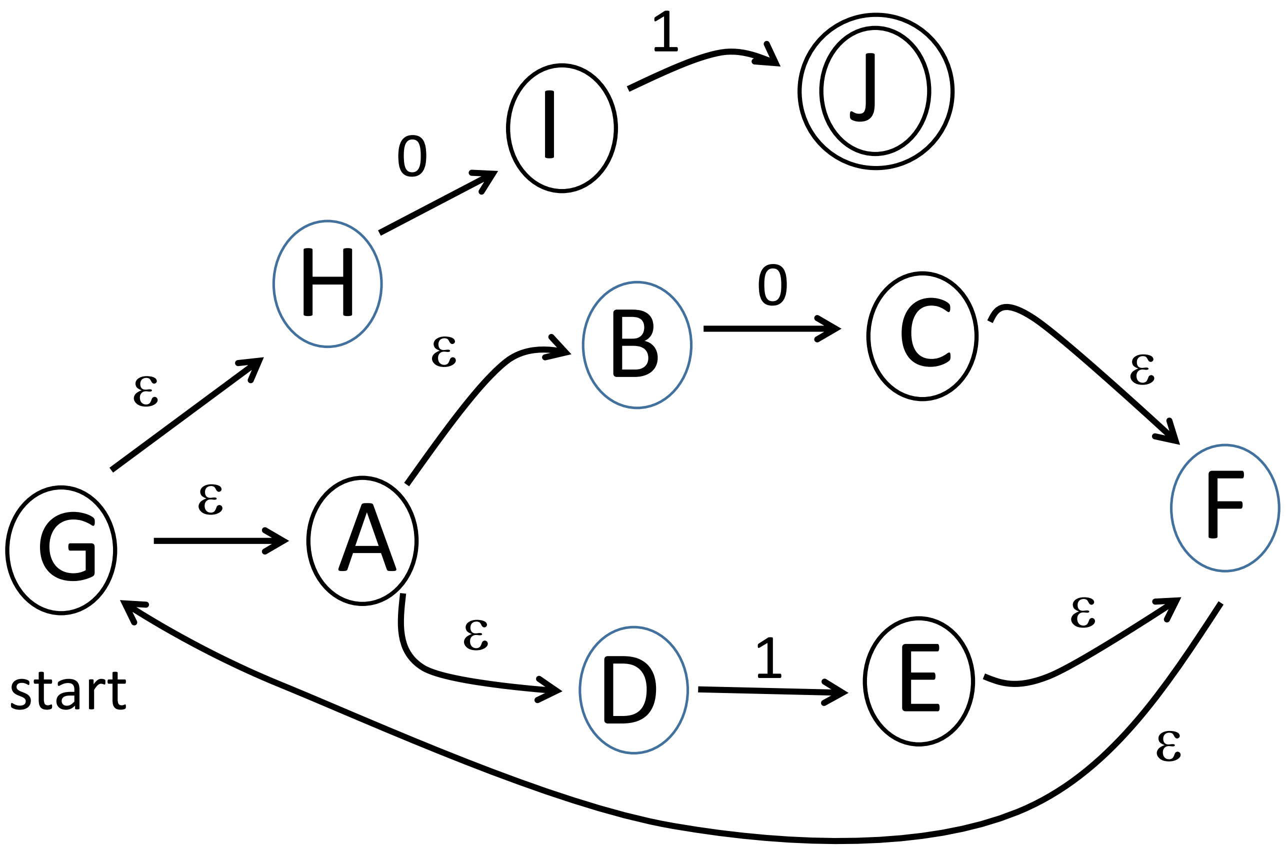


This accepts 0^*1

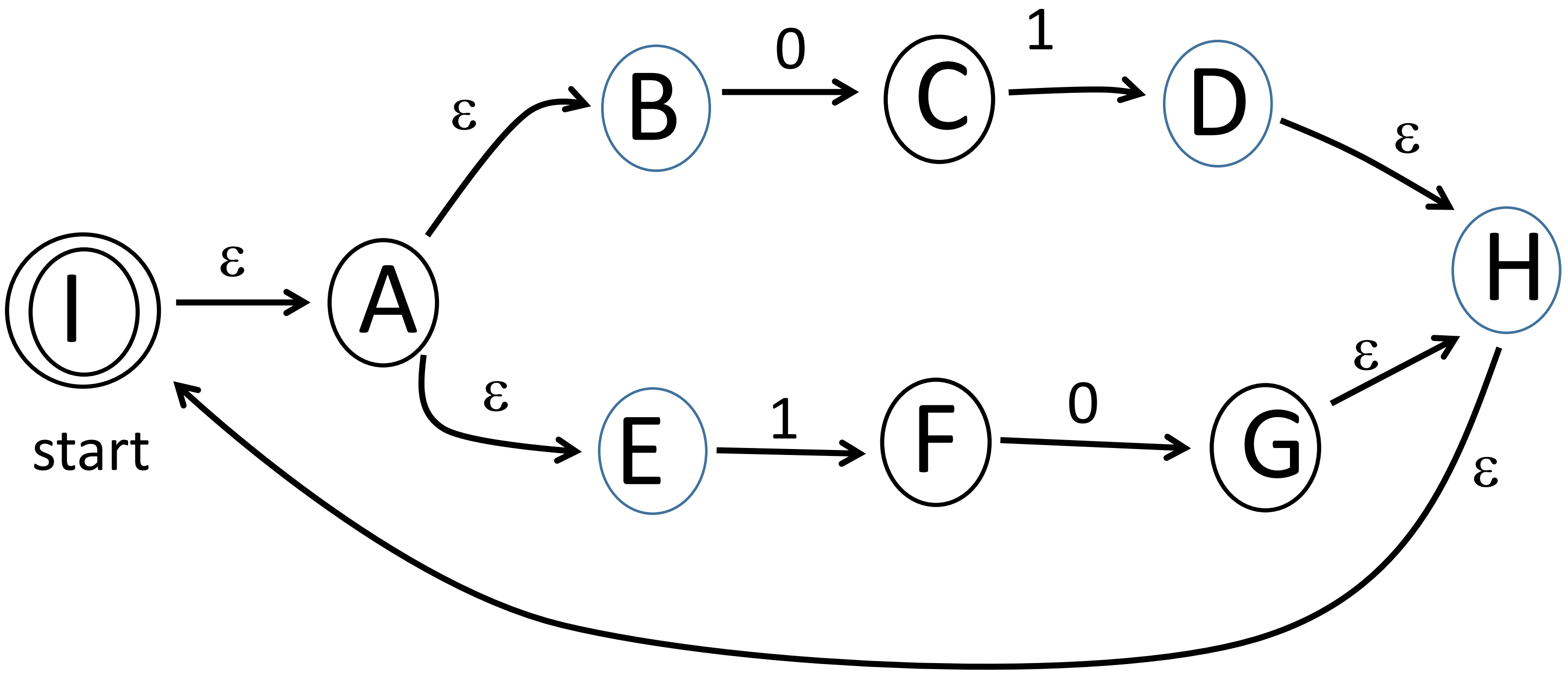


This accepts 000 and many other strings not in $(0^*1)^*$

Example: Find a finite automaton that accepts the language represented by $(0+1)^*01$



Example: Find a finite automaton that accepts the language represented by $(01+10)^*$

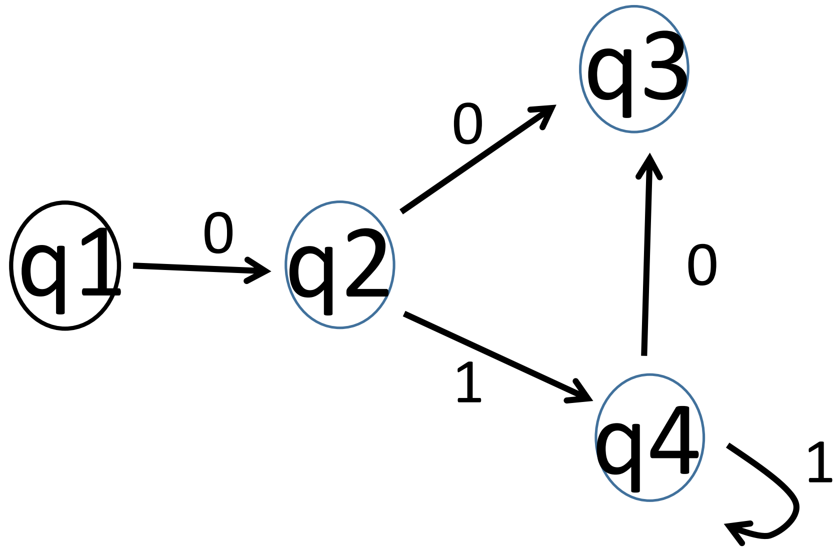


Theorem: Any language accepted by a DFA is also denoted by a regular expression.
Proof: This is more difficult because we don't have a recursive definition of a DFA for induction. We need to start with an arbitrary DFA and construct a regular expression for it.

Setup:

1. Number the states of the DFA $q_1, q_2, \ldots q_n$ where q_1 is the start state. Note that we start indexing at 1, not 0.
2. Define R_{ij}^{k} to be the set of all strings that take the automaton from state q_i to state q_j without passing through any states numbered higher than k (where "passing through" means first entering, then leaving).

For example, consider:



Here $R_{13}^{2} = \{00\}$
$R_{12}^{0} = \{0\}$
$R_{13}^{4} = \{00, 010, 0110, \ldots\} = 01^{*}0$

Note that if the automaton has n states then $\bigcup_{q_i \in F} R_{1j}^{n}$ is the set of strings accepted by the automaton. We will use recursion on k to show that each of the R_{ij}^{k} sets is denoted by a regular expression.

For the base case, k=0. If $i \neq j$ then R_{ij}^{0} is empty if there is no transition from q_i to q_j; if there is such a transition then $R_{ij}^{0} = \{a | \delta(q_i, a) = q_j\}$ If i and j are equal $R_{ii}^{0} = \{a | \delta(q_i, a) = q_i\} \cup \{\varepsilon\}$
In all of these cases R_{ij}^{0} is finite and so is represented by a regular expression.

For the inductive case, note that for any k > 0

$$R_{ij}^{k} = R_{ij}^{k-1} \cup R_{ik}^{k-1}\left(R_{kk}^{k-1}\right)^{*}R_{kj}^{k-1}$$

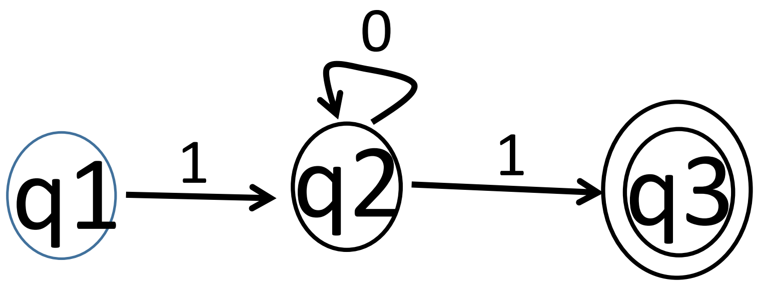
don't pass thru q_k	first trip to state k	repeated trips to q_k	from q_k to q_j

This means we can represent R_{ij}^{k} by the regular expression

$$r_{ij}^{k} = r_{ij}^{k-1} + r_{ik}^{k-1}\left(r_{kk}^{k-1}\right)^{*}r_{kj}^{k-1}$$

Finally, $r = \sum_{q_j \in F} r_{1j}^{n}$ is a regular expression that denotes the language accepted by the automaton.

Example:



$$r_{ij}^{1} = r_{ij}^{0} + r_{i1}^{0}(r_{11}^{0})^{*}r_{1j}^{0}$$
$$r_{ij}^{2} = r_{ij}^{1} + r_{i2}^{1}(r_{22}^{1})^{*}r_{2j}^{1}$$

	k=0	k=1	k=2
r_{11}^{k}	ε	ε	ε
r_{12}^{k}	1	1	1+1(0+ε)*(0+ε)=10*
r_{13}^{k}	ϕ	ϕ	1(0+ε)*1=10*1
r_{21}^{k}	ϕ	ϕ	ϕ
r_{22}^{k}	0+ε	0+ε	(0+ε)+(0+ε)(0+ε)*(0+ε)=0*
r_{23}^{k}	1	1	1+(0+ε)(0+ε)*1=0*1
r_{31}^{k}	ϕ	ϕ	ϕ
r_{32}^{k}	ϕ	ϕ	ϕ
r_{33}^{k}	ε	ε	ε

Finally, we are only interested in r_{13}^3.

$$r_{13}^3 = r_{13}^2 + r_{13}^2 (r_{33}^2)^* r_{33}^2$$

= 10*1+(10*1)ε*ε

$$= 10^*1$$