NFA’s with $\Lambda$–Transitions

• We extend the class of NFAs by allowing instantaneous transitions:

1. The automaton may be allowed to change its state without reading the input symbol.
2. In diagrams, such transitions are depicted by labeling the appropriate arcs with $\Lambda$.
3. Note that this does not mean that $\Lambda$ has become an input symbol. On the contrary, we assume that the symbol $\Lambda$ does not belong to any alphabet.
example

\[ \{ a^n \mid n \text{ is even or divisible by 3} \} \]
Definition

• A $\varepsilon$-NFA is a quintuple $A = (Q, \Sigma, s, F, \delta)$, where

  - $Q$ is a set of states
  - $\Sigma$ is the alphabet of input symbols
  - $s$ is an element of $Q$ --- the initial state
  - $F$ is a subset of $Q$ --- the set of final states
  - $\delta: Q \times (\Sigma \cup \Lambda) \rightarrow Q$ is the transition function

• Note $\Lambda$ is never a member of $\Sigma$
• ε-NFAs add a convenient feature but (in a sense) they bring us nothing new: they do not extend the class of representable languages.

• **Theorem.** Every language accepted by an ε-NFA is also accepted by some DFA.

• The proof requires a modification of the subset construction. To describe it, we need the notion of Λ-closure.

• Λ-transitions are a convenient feature: try to design an NFA for the even or divisible by 3 language that does not use them!
Λ-Closure

• Λ-closure of a state
• The Λ-closure of the state q, denoted ECLOSE(q), is the set that contains q, together with all states that can be reached starting at q by following only Λ-transitions.

In the above example:
• ECLOSE(p) = {p,q,r}
• ECLOSE(x) = {x} for any of the remaining five states, x.
Elimination of $\Lambda$-Transitions

• Given an $\varepsilon$-NFA $N$, this construction produces an NFA $N'$ such that $L(N')=L(N)$.
• Then we can apply the subset construction to $N$ and obtain a DFA, $D$, such that $L(D)=L(N')=L(N)$. This would prove the Theorem (page 6) above.

• The construction of $N'$ begins with $N$ as input, and takes 3 steps:
  
  1. Make $p$ an accepting state of $N'$ iff $ECLOSE(p)$ contains an accepting state of $N$.
  2. Add an arc from $p$ to $q$ labeled $a$ iff there is an arc labeled $a$ in $N$ from some state in $ECLOSE(p)$ to $q$.
  3. Delete all arcs labeled $\Lambda$.  

We illustrate the procedure on the following $\varepsilon$-NFA $N$, accepting the strings over $\{a,b,c\}$ of the form $a^i b^j c^k$ ($i,j,k \geq 0$).
1) Make p an accepting state iff ECLOSE(p) contains an accepting state of N

2) Add an arc from p to q labeled a iff there is an arc labeled a from some state in ECLOSE(p) to q

3) Delete all arcs labeled $\Lambda$
Why does it work?

• The language accepted by the automaton is being preserved during the three steps of the construction: $L(N)=L(N_1)=L(N_2)=L(N_3)$

• Each step here requires a proof. A Good exercise for you to do!
Automata and Languages

• We have seen that the three types of automata we've considered all define the same class of languages:
  
  \[
  \{ L(A) \mid A \text{ is a DFA} \} 
  \]

  = \[
  \{ L(A) \mid A \text{ is an NFA} \} 
  \]

  = \[
  \{ L(A) \mid A \text{ is an } \varepsilon\text{-NFA} \} 
  \]
Remarkable facts

• A remarkable fact is that this class of languages is closed under boolean operations (union, intersection, complement) and Kleene star.

• i.e. if $A, B$ are DFA's

• Then so are $A^*$, $A \cup B$, $A \cap B$ and $A$

• The converse is just as amazing: every language that can be obtained starting with finite subsets of an alphabet by applying these operations is a language of some DFA.

• It is important to understand these facts.