Where We Have Been

- Proved regular languages are closed under intersection/union/complement with DFAs
- Wanted to prove closure under concatenation
- Introduced a new type of machine: the nondeterministic finite automaton (NFA)
- NFAs are easier to work with to prove closure properties
Languages Recognized by NFAs Closed under Concatenation?
Languages Recognized by NFAs Closed under Concatenation?
Languages Recognized by NFAs Closed under Union?
Languages Recognized by NFAs Closed under Union?
Do NFAs recognize a larger class of languages? That is, are NFAs more powerful than DFAs?
The Equivalence of NFAs and DFAs
What do we Mean by Equivalence?

**Definition.** Two machines are *equivalent* if they recognize the same language.
Outline
Part 1: DFA to NFA

Lemma. Every deterministic finite automaton has an equivalent nondeterministic finite automaton.

Proof. Let $M = (Q, \Sigma, \delta, q_0, F)$ be an arbitrary DFA. We need to show existence of an NFA $N$ that recognizes the same language as $M$. 
Recall: DFA

A finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\), where

- \(Q\) is a finite set called the **states**,  
- \(\Sigma\) is a finite set called the **alphabet**,  
- \(\delta : Q \times \Sigma \rightarrow Q\) is the **transition function**,  
- \(q_0 \in Q\) is the **start state**, and  
- \(F \subseteq Q\) is the set of **accept states**.
Recall: NFA

A nondeterministic finite automaton is a 5-tuple \((Q, \Sigma, \delta, q_0, F)\) where

- \(Q\) is a finite set called the states,
- \(\Sigma\) is a finite set called the alphabet,
- \(\delta : Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)\) is the transition function, where \(\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}\)
- \(q_0 \in Q\) is the start state, and
- \(F \subseteq Q\) is the set of accept states.
Lemma. Every nondeterministic finite automaton has an equivalent deterministic finite automaton.

Proof. Let $N = (Q, \Sigma, \delta, q_0, F)$ be an arbitrary NFA. We need to show existence of a DFA $M$ that recognizes the same language as $N$. 
Removing Choice

**Proof. (Cont)** Let $N = (Q, \Sigma, \delta, q_0, F)$ be an NFA.

We construct a DFA $M$ that “simulates” $N$.

1. $Q' = P(Q)$
2. For $R \in Q'$ and $a \in \Sigma$, let $\delta'(R, a) = \bigcup_{r \in R} \delta(r, a)$
3. $q'_0 = \{q_0\}$
4. $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$
Equivalent DFA?
Equivalent DFA?
Equivalent DFA?
What about $\varepsilon$ arrows?
Modifying Our Construction

**Proof.** Let $N = (Q, \Sigma, \delta, q_0, F')$ be an NFA recognizing language $A$.

We construct a DFA $M$ recognizing $A$.

1. $Q' = P(Q)$
2. For $R \in Q'$ and $a \in \Sigma$, let $\delta'(R, a) = \bigcup_{r \in R} E(\delta(r, a))$ where $E(R) = \{q \mid q \text{ can be reached under } R \text{ along } 0 \text{ or more } \varepsilon \text{ arrows}\}$
3. $q'_0 = E(\{q_0\})$
4. $F' = \{R \in Q' \mid R \text{ contains an accept state of } N\}$
Equivalent DFA?
Equivalent DFA?
Equivalent DFA?
Alternate Way to Define a Regular Language!

**Corollary.** A language is regular if and only if some NFA recognizes it.

**Proof.**
(=>) Let $L$ be a regular language. Then there exists a DFA $M$ such that $L(M) = L$ but every DFA is also an NFA, so $M$ is the NFA recognizing $L$.

(<=) Let $L$ be a language recognized by a NFA $N$. We showed that $N$ can be converted to an equivalent DFA $M$ such that $L(M) = L$. 