Today's learning goals

- Design NFA recognizing a given language
- Use algorithms from closure proofs to build an NFA recognizing a given language step-by-step
- Convert a NFA (with or without spontaneous moves) to a DFA recognizing the same language

Reminders:
- Group HW1 due Saturday
- Review Quiz 2 "due" Sunday
- HW2 due Tuesday
NFA

\[ \Sigma = \{0, 1\} \]
\[ \varepsilon \notin \Sigma \]

L(M1) = \{ \varepsilon \} \cup \{0^* 0 1^*\}
= L(\varepsilon) \cup L(0^* 0 1^*)

L(M2) = L(\Sigma^* (00 \cup 11))
= L(\Sigma^* 0 0) \cup L(\Sigma^* 1 1)
NFA

Formal definition of M1:

\( (Q, \Sigma, \delta, q_0, F) \)

\( \delta : Q \times \Sigma \times \varepsilon \rightarrow \mathcal{P}(Q) \)

\( Q = \{ q_0, q_1, q_2 \} \)

\( \Sigma = \{ 0, 1 \} \)

\( \delta \) need to specify vals

\( W = 1011 \)

(2, 3, 5, 8, 13)
Concatenation, formally

- "Guess" some stage of input at which switch modes

Given $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, $M_2 = (Q_2, \Sigma, \delta_2, q_2, F_2)$ build $N = (\ldots, \Sigma, \delta, q_1, F_2)$ with $\delta$…
More differences between NFA and DFA

- **DFA**: unique computation path for each input
- **NFA**: allow several (or zero) alternative computations on same input
  - $\delta(q,x)$ may specify more than one possible next states
  - $\varepsilon$ transitions allow the machine to transition between states spontaneously, without consuming any input symbols

Types of components of formal definition

<table>
<thead>
<tr>
<th>Type</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>DFA</td>
<td>$\delta : Q \times \Sigma \rightarrow Q$</td>
</tr>
<tr>
<td>NFA</td>
<td>$\delta : Q \times \Sigma_{\varepsilon} \rightarrow \mathcal{P}(Q)$</td>
</tr>
</tbody>
</table>
Concatenation, formally

\[ \delta( (q, x) ) = \begin{cases} \{ \delta_1((q, x)) \} \quad & \text{if } q \text{ is in } Q_1, x \text{ is in } \Sigma \\ \{ \delta_2((q, x)) \} \quad & \text{if } q \text{ is in } Q_2, x \text{ is in } \Sigma \\ \varnothing \quad & \text{if } q \text{ is in } F_1, x = \varepsilon \\ \varnothing \quad & \text{if } q \text{ is in } Q_1 - F_1, x = \varepsilon \\ \varnothing \quad & \text{if } q \text{ is in } Q_2, x = \varepsilon \end{cases} \]
New – easier – construction for union

• "Guess" one of finite list of criteria to meet

Accept if either (or both) accepts
Given $M_1 = (Q_1, \Sigma, \delta_1, q_1, F_1)$, build

$$N = (Q_1 \cup \{q_0\}, \Sigma, \delta, q_0, F_1 \cup \{q_0\})$$

and $\delta(q, x) = \ldots$

*Construction in the book (page 63)*
NFA

Concatenation?
Union?
Kleene star?
Simulating NFA with DFA

Not quite a closure proof, but …

**Proof:**

**Given** name variables for sets, machines assumed to exist.  
**WTS** state goal and outline plan.  
**Construction** using objects previously defined + new tools working towards goal. Give formal definition and explain.  
**Correctness** prove that construction works.  
**Conclusion** recap what you've proved.
Simulating NFA with DFA

For any language recognized by an NFA, there is a DFA that recognizes this language.

Proof:

Given A, a language recognized by $N = (Q, \Sigma, \delta, q_0, F)$ a NFA

WTS there is some DFA $M$ with $L(M) = A$

Construction

Correctness

Conclusion
Idea of construction

Track set of possible states NFA might be in.

keep going...
From NFA to DFA

Which states can this NFA be in before first input symbol is read?

A. q0
B. any state
C. q0, q1
D. q0, q4
E. q0, q1, q4

From alphabet
Subset construction

Given $A$, a language recognized by $N = (Q, \Sigma, \delta, q_0, F)$ a NFA

**WTS** there is some DFA $M$ with $L(M) = A$

**Construction** Define $M = (Q', \Sigma, \delta', q_0', F')$ with

- $Q' = \text{the power set of } Q = \{ X | X \text{ is a subset of } Q \}$
- $q_0' = \{ \text{states } N \text{ can be in before first input symbol read} \}$
- $F' = \{ \}$
- $\delta' (\ , ) =$
Subset construction

Given A, a language recognized by N = (Q, Σ, δ, q0, F) a NFA

WTS there is some DFA M with L(M) = A

Construction Define M = (Q', Σ, δ', q0', F') with

- Q' = the power set of Q = \{ X | X is a subset of Q \}
- q0' = \{ q0 \} U δ((q0, ε)) ...
- F' = { ... }
- δ' ( ... ) =
Subset construction

Given $A$, a language recognized by $N = (Q, \Sigma, \delta, q_0, F)$ a NFA

WTS there is some DFA $M$ with $L(M) = A$

Construction Define $M = (Q', \Sigma, \delta', q_0', F')$ with

- $Q' = \text{the power set of } Q = \{ X \mid X \text{ is a subset of } Q \}$
- $q_0' = \{ q_0 \} \cup \delta((q_0, \varepsilon)) \ldots$
- $F' = \{ \ldots \}$
- $\delta'((X, x)) = \{ q \in Q \mid q \text{ is in } \delta((r, x)) \text{ for some } r \in X \text{ or is accessible via spontaneous moves} \}$
From NFA to DFA
Subset construction

Given $A$, a language recognized by $N = (Q, \Sigma, \delta, q_0, F)$ a NFA, there is some DFA $M$ with $L(M) = A$.

Construction Define $M = (Q', \Sigma, \delta', q_0', F')$ with:
- $Q' = \text{the power set of } Q = \{ X \mid X \text{ is a subset of } Q \}$
- $q_0' = \{q_0\} \cup \delta((q_0, \varepsilon))$ …
- $F' = \{\text{guarantee at least one computation is successful}\}$
- $\delta' ((X, x)) = \{q \in Q \mid q \text{ is in } \delta((r, x)) \text{ for some } r \in X \text{ or is accessible via spontaneous moves}\}$
What does it mean for a set of states $X$ to guarantee at least one computation is successful?

A. $X$ is a subset of $F$
B. $X = F$
C. $X \cap F$ is nonempty
D. $X$ is an element of $F$
E. None of the above.
Subset construction examples
DFA equiv NFA

**Theorem**: For each language $L$, 

$L$ is recognizable by some DFA 

iff 

$L$ is recognizable by some NFA
Theorem: For each language L,

L is recognizable by some DFA
iff
L is recognizable by some NFA
iff
L is describable by some regular expression
For next time

- Work on Individual HW2 due Tuesday

Pre class-reading for Monday: Example 1.56 on page 68