Today's learning goals

- Design NFA recognizing a given language
- Convert an NFA (with or without spontaneous moves) to a DFA recognizing the same language
- Decide whether or not a string is described by a given regular expression
- Design a regular expression to describe a given language
- Convert between regular expressions and automata
Nondeterministic finite automata

- "Guess" some stage of input at which switch modes
- "Guess" one of finite list of criteria to meet

Accept if either (or both) accepts
Acceptance in an NFA

An NFA \((Q, \Sigma, \delta, q_0, F)\) accepts a string \(w\) in \(\Sigma^*\) iff we can write \(w = y_1 y_2 \cdots y_m\) where each \(y_i \in \Sigma_e\) and there is a sequence of states \(r_0, \ldots, r_m \in Q\) such that

1. \(r_0 = q_0\)
2. \(r_{i+1} \in \delta(r_i, y_{i+1})\) for each \(i = 0, \ldots, m - 1\)
3. \(r_m \in F\).
NFA

\[ L = \{ w \mid |w| = 3n+1, \text{ for any } n \geq 0 \text{ and last symbol of } w \text{ is } 0 \} \]

\[ = \{ 1 \}^* \cup \{ 0 \}^* \]
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
Which of the following strings is accepted by this NFA?
A. The empty string
B. 01
C. 111
D. 0
E. More than one of the above.
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

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

- $Q' = \text{the power set of } Q = \{ X \mid X \text{ is a subset of } Q \}$
- $q_0' = \{ q_0 \} \cup \delta((q_0, \epsilon)) \ldots$
- $F' = \{ \}$
- $\delta' ( ) = \ldots$
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

- \( Q' = \) the power set of \( Q = \{ X \mid X \text{ is a subset of } Q \} \)
- \( q_0' = \{ q_0 \} \cup \delta((q_0, \varepsilon)) \) …
- \( F' = \{ \text{guarantee at least one computation is successful} \} \)
- \( \delta' \) ( ) =
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

**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

- \( Q' = \) the power set of \( Q = \{ X | X \) is a subset of \( Q \} \)
- \( q_0' = \{ q_0 \} \cup \delta((q_0, \varepsilon)) \)  
- \( F' = \{ X | X \) is a subset of \( Q \) and \( X \cap F \) is nonempty \}
- \( \delta'( (X, x) ) = \{ \} \)
Subset construction

**Given** A, a language recognized by $N = (Q, \Sigma, \delta, q₀, F)$ a NFA

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

**Construction** Define $M = (Q', \Sigma, \delta', q₀', F')$ with

- $Q' = \text{the power set of } Q = \{ X \mid X \text{ is a subset of } Q \}$
- $q₀' = \{ q₀ \} \cup \delta((q₀, \varepsilon))$ ... 
- $F' = \{ X \mid X \text{ is a subset of } Q \text{ and } X \cap F \text{ is nonempty} \}$
- $\delta'((X, x)) = \{ q \in Q \mid q \text{ is in } \delta(r, x) \text{ for some } r \in X \text{ or accessible via spontaneous moves} \}$

Types? $X \subseteq Q$ i.e. $X \in \wp(Q)$ i.e. $X \in Q$
Subset construction warmup examples
Subset construction example
Recap

Regular sets of strings i.e., those that aren't too complicated are those that are

• the language of some DFA, or equivalently,
• the language of some NFA.

If a set of strings can be expressed as the result of complement, union, intersection, concatenation, Kleene star of regular languages, then it itself is regular.
Regular expressions

- Can all regular languages be expressed as those that are built up from very simple languages using the regular operations?
Regular expressions in practice

- **Compilers**: first phase of compiling transforms Strings to Tokens *keywords, operators, identifiers, literals*
  - One regular expression for each token type

- **Other software tools**: grep, Perl, Python, Java, Ruby, …