1. Deterministic Finite Automata. Homeworks are p. 78: 2.1.2, 2.1.3, 2.1.7; p. 79: 2.2.1, 2.2.3, 2.3.1, 2.3.2, 2.3.3, 2.4, 2.5, 2.6; p. 81: 2.9, 2.11, 2.12, 2.13; p. 82: 2.14.1, 2.15, 2.16, 2.21, 2.22

- parity of incoming ones:

- even count of zeros after rightmost one:

- Definition: A finite automaton is a 5-tuple $M = (Q, \Sigma, \delta, q, F)$, where
  - $Q$ is a finite set of states
  - $\Sigma$ is a finite alphabet of symbols
  - $\delta : Q \times \Sigma \rightarrow Q$ is a function (the transition function)
  - $q \in Q$ is the start state
  - $F \subseteq Q$ is a collection of accepting states.

- The transition function is a program.

- Note tabular display of the transition function.

- For a set of symbols, $\Sigma$, we let $\Sigma^*$ denote the set of finite strings composed of symbols from $\Sigma$.

- A language over $\Sigma$ is a subset of $\Sigma^*$.

- Definition: Let $M = (Q, \Sigma, \delta, q, F)$ be an DFA, and let $w = w_1, w_2, \ldots, w_n \in \Sigma^*$. Let $r_0, r_1, \ldots, r_n$ be a sequences of states from $Q$ such that $r_0 = q$ (from the machine definition), and $r_{i+1} = \delta(r_i, w_i)$ for $i = 1, 2, \ldots, n$. If $r_n \in F$, we say that $M$ accepts $w$. If $r_n \notin F$, we say that $M$ rejects $w$.

- Definition: Let $M = (Q, \Sigma, \delta, q, F)$ be an DFA. Then, $L(M) = \{ w \in \Sigma^* : w$ is accepted by $M \}$ is the language accepted by $M$.

- Definition: A language $A \subseteq \Sigma^*$ is a regular language if there exists an DFA $M$ such that $A = L(M)$. Note two examples of regular languages at top: “even count of ones” and “even count of zeros after rightmost one.”

- Given DFA $M = (Q, \Sigma, \delta, q, F)$, define $\bar{\delta} : Q \times \Sigma^* \rightarrow Q$ via

$$\bar{\delta}(r, w) = \begin{cases} r, & w = \epsilon \\ \delta(\bar{\delta}(r, v), a), & w = va, \text{ for } v \in \Sigma^* \text{ and } a \in \Sigma \end{cases}$$

- In terms of $\bar{\delta}$, $L(M) = \{ w \in \Sigma^* : \bar{\delta}(w) \in F \}$. 

• Binary string contains 101 as substring:

• Binary string that contains a one in the third position from the right. Use \( q_{000}, \ldots, q_{111} \) to track last three symbols at any given point in the scan...

• Operations on languages. For languages \( A, B \subseteq \Sigma^* \):
  - (union) \( A \cup B = \{ w : w \in A \text{ or } w \in B \} \)
  - (concatenation) \( AB = \{ w : w = uv \text{ for } u \in A, v \in B \} \)
  - (Kleen closure, star) \( A^* = \{ w_1, w_2, \ldots, w_k : k \geq 0, w_i \in A \text{ for } 1 \leq i \leq k \} \)

• Kleene closure notation consistent with \( \Sigma^* \) notation, where we regard \( \Sigma \) as a language containing a finite number of one-character strings.

• Note Kleene closure of any language contains \( \epsilon \). So, if \( A = \phi \), then \( A^* = \{ \epsilon \} \)

• Examples using \( A = \{0, 01\}, B = \{1, 10\} \)...

• For any language \( A \subseteq \Sigma^* \), define
  \[
  A^0 = \{ \epsilon \} \\
  A^1 = A \\
  A^2 = AA \\
  \text{and so forth. In general } A^k = AA^{k-1} \text{ and } A^* = \bigcup_{k=0}^{\infty} A^k.
  \]

• Theorem: The set of regular languages is closed under union operations. That is, if \( A, B \) are regular languages over the same alphabet, then \( A \cup B \) is a regular language.
  
  Proof: Note: cannot “run” on \( A \) then, having failed, run on “B.” A machine gets a single pass over the input string.
  
  But, can construct a Cartesian product machine...

• Regular languages closed under complementation: Exchange accepting and non-accepting status for states in accepting DFA for \( L \). New machine accept \( L^c \).

• Note problems in proving closure under concatenation and Kleene closure...

• In a non-deterministic machine \( NFA \), a machine has several possibilities (including none) at each state
  - The machine may choose to move to one or more states on consuming a given input symbol
  - The machine may choose to move to one or more states without consuming the input symbol
  - The machine may have no choices for a particular input symbol (computation “dies” on that input)

• Binary string contain 101 or 11:

• Trace clones of NFA above for input 010110.
• Binary string with 1 in third place from right:
  DFA had eight states.

• Binary string with zero count = 0 mod 2 or mod 3:

• Definition: A nondeterministic finite automaton (NFA) is a 5-tuple $M = (Q, \Sigma, \delta, q, F)$ with
  
  – $Q$ is a finite set of states
  
  – $\Sigma$ is a finite set (alphabet) of symbols
  
  – $\delta : Q \times (\Sigma \cup \{\epsilon\}) \rightarrow P(Q)$
  
  – $q \in Q$ is the start state
  
  – $F \subseteq Q$ is a set of accepting states.

• Earlier example

• Definition: Let $M = (Q, \Sigma, \delta, q, F)$ be an NFA and let $w = w_1w_2 \ldots w_m \in \Sigma^*$. We say that $M$ accepts $w$ if there exists a state sequence $r_0, r_1, \ldots, r_m \in Q$ with
    
    – $r_0 = q$
    
    – For $0 \leq i < m$, $r_{i+1} \in \delta(r_i, y_{i+1})$
    
    – $r_m \in F$. Otherwise, we say $M$ rejects $w$.

• Definition: $M = (Q, \Sigma, \delta, q, F)$ be an NFA. Then the language $L(M)$ accepted by $M$ is
  
  $L(M) = \{w \in \Sigma^* : M$ accepts $w\}$.

• Definition: Let $M = (Q, \Sigma, \delta, q, F)$ be an NFA. For $q \in Q$, the $\epsilon$-closure of $q$ is
  
  $C_\epsilon(q) = \{r \in Q : r$ can be reached from $q$ by one or more $\epsilon$-transitions\}.

• Example: $C_\epsilon(q_1) = \{q_1, q_2\}$:

• Theorem: If $N = (Q, \Sigma, \delta, q, F)$ is an NFA, then there exists a DFA $M$ such that $L(M) = L(N)$.
  Also, if $M = (Q, \Sigma, \delta, q, F)$ is a DFA, then there exists an NFA $N$ such that $L(N) = L(M)$. 

![](image)
Proof: Note DFA to NFA is trivial: convert \( \delta \) by replacing all entries in the tabulation with singleton sets and add an \( \epsilon \) columns with all \( \phi \) entries.

To convert an NFA to a DFA, we will construct a DFA whose state space is the power set of the NFA state space.

Intuition: The constructed DFA, after reading a prefix, say \( w \), of the input will is a subset, which is precisely those states that the NFA could occupy after reading the same prefix...

Here is the construction:

Given \( NFA = (Q, \Sigma, \delta, q, F) \), we construct \( DFA = (Q', \Sigma', \delta', q', F') \), where

- \( Q' = P(Q) \)
- \( q' = C_\epsilon(q) \)
- \( F' = \{ A \in Q' : A \cap F \neq \phi \} \)
- For \( A \in P(Q) \), define \( \delta'(A, a) = \cup_{r \in A} C_\epsilon(\delta(r, a)) \)

Trace accepting path in NFA as accepting path in DFA — ditto for non-accepting paths

Trace accepting path in DFA as accepting path in NFA — ditto for non-accepting paths. •

• Theorem: Language \( A \) is regular if and only if there exists an NFA \( N \) with \( A = L(N) \).

\[ \]

• Example: Convert to DFA:

\[ \]

Short conversion:

\[ \]

<table>
<thead>
<tr>
<th>[ \rightarrow *{1, 2} ]</th>
<th>[ a ]</th>
<th>[ b ]</th>
</tr>
</thead>
<tbody>
<tr>
<td>[ *{1, 2, 3} ]</td>
<td>[ {1, 2, 3} ]</td>
<td>[ \phi ]</td>
</tr>
<tr>
<td>[ *{2, 3} ]</td>
<td>[ {1, 2} ]</td>
<td>[ {2, 3} ]</td>
</tr>
<tr>
<td>[ \phi ]</td>
<td>[ \phi ]</td>
<td>[ \phi ]</td>
</tr>
</tbody>
</table>
2. Theorem: Regular language are closed under union. That is, $A, B$ regular implies $A \cup B$ regular.

Proof: New start state with $\epsilon$-transitions to start states of $A, B$ NFAs. Keep accepting states in both. 

3. Theorem: Regular languages are closed under concatenation. That is, $A, B$ regular implies $AB$ regular.

Proof: New NFA has start state from $A$. All accepting states from $A$ get $\epsilon$-transitions to start state of $B$. Remove accepting status from all such states in $A$. Keep accepting status for all such states in $B$.

4. Theorem: Regular languages are close under Kleene closure.

Proof: Modify NFA $N$ that accepts language $L$. New starting/accepting states — to accept $\epsilon$ string. New start gets $\epsilon$-transition to start state of $M$. All accepting states of $M$ get $\epsilon$-transition to start state of $M$. 

Long conversion:

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$b$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\phi$</td>
<td>$\phi$</td>
<td>$\phi$</td>
</tr>
<tr>
<td>${1}$</td>
<td>${3}$</td>
<td>$\phi$</td>
</tr>
<tr>
<td>$*{2}$</td>
<td>${1,2}$</td>
<td>$\phi$</td>
</tr>
<tr>
<td>${3}$</td>
<td>${2}$</td>
<td>${2,3}$</td>
</tr>
</tbody>
</table>

$\rightarrow *\{1,2\}$ $\rightarrow *\{1,2\}$ $\rightarrow *\{1,2\}$ $\rightarrow *\{1,2\}$ $\rightarrow *\{1,2\}$