Announcements

Bonus Readings
- **Lovelace** - The Origin graphic novel, *Logicomix*, and Research Paper *(Computer solution to the 17-point Erdos-Szekeres problem)* up
  - Questions to answer
    - What is the paper/novel trying to do?
    - Does it work well?

- **Extreme Algorithms at 12:15 in ROOM # 2.228**

Start of Class
All pics are from the book in CH.1.2

Definition: $M = (Q, \Sigma, q_s, F)$ as an NFA
The language $L(m)$ accepted by $m$
$L(m) = \{w \in \Sigma^* : M \text{ accepts } w\}$

Example of a NFA

How does an NFA compute? Suppose that we are running an NFA on an input string and come to a state with multiple ways to proceed. For example, say that we are in state $q_1$ in NFA $N_1$ and that the next input symbol is a 1. After reading that symbol, the machine splits into multiple copies of itself and follows all the possibilities in parallel. Each copy of the machine takes one of the possible ways.

$$
\begin{align*}
q_1 \xrightarrow{0,1} q_2 \xrightarrow{1} q_3 \xrightarrow{0,\varepsilon} q_4 \xrightarrow{1} q_4
\end{align*}
$$
Let's consider some sample runs of the NFA $N_1$ shown in Figure 1.27. The computation of $N_1$ on input 010110 is depicted in the following figure.

![Diagram of NFA $N_1$](image)

**Figure 1.29**
The computation of $N_1$ on input 010110

**Helpful tip with NFA/DFA:**
- “When the machine is in a given state and reads the next input symbol, we know what the next state will be—it is determined. We call this deterministic computation (DFA). In a nondeterministic machine, several choices may exist for the next state at any point (NFA). Nondeterminism is a generalization of determinism, so every deterministic finite automaton is automatically a nondeterministic finite automaton”

**Equivalence of NFA/DFA:**
Example 2: NFA to DFA

\[ E = \{0, 1\} \]

\[ A = \{w \in E^*: w \text{ contains a } 1 \text{ in the 3rd position from the end}\} \]

Our NFA: Generalize outcome

![NFA Diagram]

Turned into a “Small” DFA: Every possible outcome

![DFA Diagram]

Def ‘Tip:

\{1, 4\} Represents the possible transition from state 1 to 4 or \(Q_1 \rightarrow Q_4\)

\(Q_1 \rightarrow Q_3\) ← Both are the same as traversing states

\{1,3\} ←

DFA ⇔ NFA

Convert a DFA to a NFA Example:

\((m41) (s55)\)

Theorem:

- Every NFA has an equivalent DFA

Proof
• Idea for proof:
  ○ Construct a DFA that runs all NFA computations simultaneously
  ○ If we have $n$ states, we then have $2^n$ states in the DFA,
  ○ Each subset corresponds to 1 state in the DFA

• Let $N = (Q, \Sigma, \delta, q_s, F)$ be an NFA recognizing the language $A$ we will construct a DFA and
• $M = (Q', \Sigma', \delta', q'_s, F')$ that also recognizes $A$
• First. Let's assume they are no epsilon arrows;
• Assume $N$ has no epsilon arrows

$Q' = \mathcal{P}(Q)$, every state of $M$ is a subset of states of $N$
For $R \in Q'$ and $a \in \text{Epsilon}$,
Let $S'(R, a) = \{ q \in Q : q \in S(r,a) \text{ for some } r \in R \}$
1. \( Q' = \mathcal{P}(Q) \).
   Every state of \( M \) is a set of states of \( N \). Recall that \( \mathcal{P}(Q) \) is the set of subsets of \( Q \).

2. For \( R \in Q' \) and \( a \in \Sigma \), let \( \delta'(R, a) = \{ q \in Q \mid q \in \delta(r, a) \text{ for some } r \in R \} \).
   If \( R \) is a state of \( M \), it is also a set of states of \( N \). When \( M \) reads a symbol \( a \) in state \( R \), it shows where \( a \) takes each state in \( R \). Because each state may go to a set of states, we take the union of all these sets. Another way to write this expression is
   \[ \delta'(R, a) = \bigcup_{r \in R} \delta(r, a). \]

3. \( q_0' = \{ q_0 \} \).
   \( M \) starts in the state corresponding to the collection containing just the start state of \( N \).

4. \( F' = \{ R \in Q' \mid R \text{ contains an accept state of } N \} \).
   The machine \( M \) accepts if one of the possible states that \( N \) could be in at this point is an accept state.

---

The notation \( \bigcup_{r \in R} \delta(r, a) \) means: the union of the sets \( \delta(r, a) \) for each possible \( r \) in \( R \).

Now we need to consider the \( \epsilon \) arrows. To do so, we set up an extra bit of notation. For any state \( R \) of \( M \), we define \( E(R) \) to be the collection of states that can be reached from members of \( R \) by going only along \( \epsilon \) arrows, including the members of \( R \) themselves. Formally, for \( R \subseteq Q \) let

\[ E(R) = \{ q \mid q \text{ can be reached from } R \text{ by traveling along } 0 \text{ or more } \epsilon \text{ arrows} \} \]

Then we modify the transition function of \( M \) to place additional fingers on all states that can be reached by going along \( \epsilon \) arrows after every step. Replacing \( \delta(r, a) \) by \( E(\delta(r, a)) \) achieves this effect. Thus

\[ \delta'(R, a) = \{ q \in Q \mid q \in E(\delta(r, a)) \text{ for some } r \in R \} \].

---

**Regular Language tip**

- A language is regular iff some NFA recognizes it

Example. \( s57, m45 \)
N = (Q, \{a,b\}, \delta, 1, \{1\})

Below shows the construction of our NFA and transformed into its equivalent DFA in figure 1.43.

Image:

To construct a DFA $D$ that is equivalent to $N_4$, we first determine $D$'s states. $N_4$ has three states, \{1,2,3\}, so we construct $D$ with eight states, one for each subset of $N_4$'s states. We label each of $D$'s states with the corresponding subset. Thus $D$'s state set is \[
\{\emptyset, \{1\}, \{2\}, \{3\}, \{1,2\}, \{1,3\}, \{2,3\}, \{1,2,3\}\}.
\]
Next, we determine the start and accept states of $D$. The start state is $E(\{1\})$, the set of states that are reachable from 1 by traveling along $\varepsilon$ arrows, plus 1 itself. An $\varepsilon$ arrow goes from 1 to 3, so $E(\{1\}) = \{1, 3\}$. The new accept states are those containing $N_4$'s accept state; thus $\{\{1\}, \{1,2\}, \{1,3\}, \{1,2,3\}\}$.

Finally, we determine $D$'s transition function. Each of $D$'s states goes to one place on input $a$ and one place on input $b$. We illustrate the process of determining the placement of $D$'s transition arrows with a few examples.

In $D$, state $\{2\}$ goes to $\{2,3\}$ on input $a$ because in $N_4$, state 2 goes to both 2 and 3 on input $a$ and we can't go farther from 2 or 3 along $\varepsilon$ arrows. State $\{2\}$ goes to state $\{3\}$ on input $b$ because in $N_4$, state 2 goes only to state 3 on input $b$ and we can't go farther from 3 along $\varepsilon$ arrows.

State $\{1\}$ goes to $\emptyset$ on $a$ because no $a$ arrows exit it. It goes to $\{2\}$ on $b$. Note that the procedure in Theorem 1.39 specifies that we follow the $\varepsilon$ arrows after each input symbol is read. An alternative procedure based on following the $\varepsilon$ arrows before reading each input symbol works equally well, but that method is not illustrated in this example.

State $\{3\}$ goes to $\{1,3\}$ on $a$ because in $N_4$, state 3 goes to 1 on $a$ and 1 in turn goes to 3 with an $\varepsilon$ arrow. State $\{3\}$ on $b$ goes to $\emptyset$.

State $\{1,2\}$ on $a$ goes to $\{2,3\}$ because 1 points at no states with $a$ arrows, 2 points at both 2 and 3 with $a$ arrows, and neither points anywhere with $\varepsilon$ arrows. State $\{1,2\}$ on $b$ goes to $\{2,3\}$. Continuing in this way, we obtain the diagram for $D$ in Figure 1.43.
\[ S'(R, a) = \bigcup_{r \in R} S(r, a) \]

Start state \( q_0' = \{q\} \)

F' = \( \{R \in Q': R \text{ contains an accept state of } N\} \)

DFA M accepts if one of the possible states that \( N \) could be in is an accept state

If a state in our DFA is one of the accept states, that means that \( R \) contains an accept state of \( N \)