Outline

• Nondeterministic Finite Automata
• Closure under regular operations
• NFA, FA equivalence
• Regular Expressions
• FA and Regular Expressions

NFA diagrams

• (single) start state
• states
• transitions:
  • may have several with a given label (or none)
  • may be labeled with ε
• At each step, several choices for next state

NFA operation

• One way to think of NFA operation:
  • string x = x₁x₂x₃...xₙ accepted if and only if
    – there exists a way of inserting ε’s into x
      x₁εεx₂x₃...εxₙ
    – so that there exists a path of transitions from
      the start state to an accept state

NFA formal definition

A nondeterministic FA

(Q, Σ, δ, q₀, F)

– Q is a finite set called the states
– Σ is a finite set called the alphabet
– δ: Q x (Σ u {ε}) → P(Q) is a function called the transition function
– q₀ is an element of Q called the start state
– F is a subset of Q called the accept states

Transitions labeled with alphabet symbols or ε

Power set of Q: the set of all subsets of Q

NFA formal definition

• Specification of this NFA in formal terms:
  – Q = {s₁, s₂, s₃, s₄}
  – Σ = {0, 1}
  – q₀ = s₁
  – F = {s₄}

  δ(s₁, 0) = {s₁}
  δ(s₁, 1) = {s₁, s₃}
  δ(s₁, ε) = {s₁}
  δ(s₂, 0) = {s₃}
  δ(s₂, 1) = {s₃}
  δ(s₂, ε) = {s₃}
  δ(s₃, 0) = {s₄}
  δ(s₃, 1) = {s₄}
  δ(s₃, ε) = {s₄}
Formal description of NFA operation

NFA \( M = (Q, \Sigma, \delta, q_0, F) \) accepts a string \( w = w_1w_2w_3\ldots w_n \in \Sigma^* \) if \( w \) can be written (by inserting \( \varepsilon \)'s) as:

\[
y = y_1y_2y_3\ldots y_m \in (\Sigma \cup \{\varepsilon\})^*
\]

and \( \exists \) sequence \( r_0, r_1, \ldots, r_m \) of states for which:

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

Closures

- Recall: want to show the set of languages recognized by NFA is closed under:
  - union \( "C = (A \cup B)" \)
  - concatenation \( "C = (A \circ B)" \)
  - star \( "C = A^*" \)

Closure under union

\[
C = (A \cup B) = \{x : x \in A \text{ or } x \in B\}
\]

Closure under concatenation

\[
C = (A \circ B) = \{xy : x \in A \text{ and } y \in B\}
\]

Closure under star

\[
C = A^* = \{x_1x_2x_3\ldots x_k : k \geq 0 \text{ and each } x_i \in A\}
\]

NFA, FA equivalence

**Theorem:** a language \( L \) is recognized by a FA if and only if \( L \) is recognized by a NFA.

Must prove two directions:

\( \Rightarrow \) L is recognized by a FA \implies L is recognized by a NFA.

\( \Leftarrow \) L is recognized by a NFA \implies L is recognized by a FA.

(usually one is easy, the other more difficult)
NFA, FA equivalence

\((\Rightarrow)\) L is recognized by a FA \iff L is recognized by a NFA

**Proof:** a finite automaton is a nondeterministic finite automaton that happens to have no \(\varepsilon\)-transitions, and for which each state has exactly one outgoing transition for each symbol.

NFA, FA equivalence

\((\Leftarrow)\) L is recognized by a NFA \iff L is recognized by a FA.

**Proof:** we will build a FA that simulates the NFA (and thus recognizes the same language).

– alphabet will be the same
– what are the states of the FA?

– given NFA \(M = (Q, \Sigma, \delta, q_0, F)\)
– construct FA \(M' = (Q', \Sigma', \delta', q'_0, F')\)
– same alphabet: \(\Sigma' = \Sigma\)
– states are subsets of M's states: \(Q' = \mathcal{P}(Q)\)
– if we are in state \(R \subseteq Q'\) and we read symbol \(a \in \Sigma'\), what is the new state?

– new start state: \(q'_0 = E(\{q_0\})\)
– new accept states:
\[F' = \{R \in Q' : R contains an accept state of M\}\]

Formally we should also prove that the construction works, by induction on the number of steps of the computation.

– at each step, the state of the FA \(M'\) is exactly the set of reachable states of the NFA \(M\)...
So far…

**Theorem:** the set of languages recognized by NFA is closed under union, concatenation, and star.

**Theorem:** a language L is recognized by a FA if and only if L is recognized by a NFA.

**Theorem:** the set of languages recognized by FA is closed under union, concatenation, and star.

Next…

• Describe the set of languages that can be built up from:
  – unions
  – concatenations
  – star operations

• Called “patterns” or regular expressions

• **Theorem:** a language L is recognized by a FA if and only if L is described by a regular expression.

Regular expressions

• R is a regular expression if R is
  – a, for some a ∈ Σ
  – ε, the empty string
  – Ø, the empty set
  – (R₁ ∪ R₂), where R₁ and R₂ are reg. exprs.
  – (R₁ · R₂), where R₁ and R₂ are reg. exprs.
  – (R₁*), where R₁ is a regular expression

A reg. expression R describes the language L(R).

Some examples

• alphabet Σ = {0,1}

  • \( \{ w : w \text{ has at least one } 1 \} \)
    \[ = \Sigma^*1\Sigma^* \]

  • \( \{ w : w \text{ starts and ends with same symbol} \} \)
    \[ = 0\Sigma^*0 \cup 1\Sigma^*1 \cup 0 \cup 1 \]

  • \( \{ w : |w| \leq 5 \} \)
    \[ = (\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma)(\varepsilon \cup \Sigma) \]

  • \( \{ w : \text{ every } 3^{rd} \text{ position of } w \text{ is } 1 \} \)
    \[ = (1\Sigma^*)(\varepsilon \cup 1 \cup 1\Sigma) \]