Regular expressions

- used for describing string patterns, e.g.

\[(0 \cup 1)0^* \quad \rightarrow \quad \text{start with } 0 \text{ or } 1, \text{ followed by any number of } 0\text{'s}\]

\[(0 \cup 1)^* \quad \rightarrow \quad \text{any string over } \{0,1\}\]
Regular expressions

Formal definition:

R is a **regular expression** if R is one of the following:

1. a for some $a \in \Sigma$,
2. $\epsilon$
3. $\emptyset$
4. $(R_1 \cup R_2)$, where $R_1, R_2$ are regular expressions
5. $(R_1.R_2)$, where $R_1, R_2$ are regular expressions
6. $(R_1)^*$, where $R_1$ is a regular expression.

Note: this type of definition is called a **recursive/inductive definition** (i.e. the definition is a recursive algorithm).
Regular expressions

For convenience: \( R^+ = RR^* \)

Examples: give regular expressions for the following languages:

- \( \{ w \in \{0,1\}^* \mid w \text{ contains the substring 001 } \} \)
  \((0u1)^*001(0u1)^*\)

- \( \{ w \in \{0,1\}^* \mid w \text{ does not contain two consecutive 0's } \} \)
  \(1^*(01^+)^*(0u\varepsilon)\)

- \( \{ w \in \{0,1\}^* \mid |w| \text{ is divisible by 2 or 3 } \} \)
  \(((0u1)(0u1)(0u1))^* \cup (0u1)(0u1)^* \quad \text{can write} \quad ((0u1)^3)^* \cup (0u1)^2)^*\)

- \( \{ w \in \{0,1\}^* \mid |w| < 4 \} \)
  \((0u1)^3 \cup (0u1)^2 \cup (0u1) \cup \varepsilon \quad = \quad (0u1\varepsilon)^3\)
Examples: let $R$ be any regular expression

- $R \cdot \emptyset = ? \emptyset$
- $R \cdot \{\varepsilon\} = ? R$
- $\emptyset^* = ? \{\varepsilon\}$
- $\{\varepsilon\}^* = ? \{\varepsilon\}$

The language defined by $R$ is denoted $L(R)$. We’ll often abuse notation and use $R$ to denote the language $L(R)$. 
Thm 1.54: A language is regular iff some regular expression describes it.

Lemma 1.55: Given a regular expression $R$, there exists a FA $M$ such that $L(M) = L(R)$.

Lemma 1.60: Given a FA $M$, there exists a regular expression $R$ such that $L(R) = L(M)$. 
Lemma 1.55: Given a regular expression $R$, there exists a FA $M$ such that $L(M) = L(R)$.

**Idea:** use constructions for $\epsilon$, $\cdot$, $*$ from last class (Section 1.3)

**Base Case:**
1. If $R = a$ for some $a \in \Sigma$, then $M$: $\rightarrow a \rightarrow \bullet$
2. If $R = \epsilon$, then $M$: $\rightarrow \bullet$
3. If $R = \phi$, then $M$: $\rightarrow \bullet$

**Inductive Case:**
4. If $R = (R_1 \cup R_2)$ then, by the IH (inductive hypothesis), we have a FA $M_1$ for $R_1$ and $M_2$ for $R_2$:
   - use the union construction $\rightarrow \bullet$ to get $M$ for $R$
5. If $R = (R_1 \cdot R_2)$ then, by the IH, we have $M_1$ for $R_1$, $M_2$ for $R_2$:
   - use the concatenation construction
6. If $R = (R_1)^*$ then, by IH, we have $M_1$ for $R_1$:
   - use the star construction
Lemma 1.60: Given a FA $M$, there exists a regular expression $R$ such that $L(R) = L(M)$.

Proof idea:

Generalized NFA (GNFA)
- transitions may be marked by reg:expr. (not just $\Sigma \cup \{\varepsilon\}$)
- single accept state that a) has arrows coming in from every other state, b) does not have any outgoing arrows
- start state that a) has arrows to every other state, b) does not have any incoming arrows
- all other states have arrows to all other states

proof by picture that every NFA has an equivalent GNFA
Equivalence of reg. expr. and FA's

Lemma 1.60: Given a FA $M$, there exists a regular expression $R$ such that $L(R) = L(M)$.

Proof idea:

Generalized NFA (GNFA) $(Q, \Sigma, \delta, q_{\text{start}}, q_{\text{accept}})$ where all as usual except $\delta : (Q - \{q_{\text{accept}}\}) \times (Q - \{q_{\text{start}}\}) \to R$ where $R$ is the set of all regular expressions over $\Sigma$.

Idea: start with a GNFA, remove states one by one and redraw arrows as necessary.

How to get a GNFA:

- After removing $q_2$:

  

- Want to remove $q_0$.

  

  This label becomes $1 \cup 0^1*$.

Example:

- $\sigma u 01^* 0^+ 01^* 0^+ 1$: this label becomes $1 \cup 0^1*$.

- $q_0 \rightarrow q_2 \rightarrow q_3$: we need to account for $q_0 \rightarrow q_2 (\text{loop}) \rightarrow q_3$.

- $q_0 \rightarrow 0^1 \epsilon \rightarrow q_3$.
Lemma 1.60: Given a FA $M$, there exists a regular expression $R$ such that $L(R) = L(M)$.

Proof idea:

How to construct an equivalent GNFA with one fewer state?

Suppose we have a GNFA $N = (Q, \Sigma, \delta, q_{\text{start}}, q_{\text{accept}})$ with $|Q| \geq 3$.

Want a GNFA $N' = (Q', \Sigma, \delta', q_{\text{start}}, q_{\text{accept}})$ where $|Q'| = |Q| - 1$ and $L(N') = L(N)$.

Let $q_r \in Q - \{q_{\text{start}}, q_{\text{accept}}\}$ then:

- $Q' = Q - \{q_{\text{remove}}\}$
- $\delta'(q_1, q_2) = \delta(q_1, q_2) \cup \delta(q_1, q_r) \cdot \delta(q_r, q_2)^* \cdot \delta(q_r, q_r)$

$\forall q, c \in Q' - \{q_{\text{accept}}\}$

$\forall q_2 \in Q' - \{q_{\text{start}}\}$