Regular expressions

- used for describing string patterns, e.g.
  
  \((0 \cup 1)0^*\) - a zero or one, followed by any sequence of zeros
  
  \((0 \cup 1)^*\) - 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. $\varepsilon$
3. $\emptyset$
4. $(R_1 \cup R_2)$, where $R_1, R_2$ are regular expressions
5. $(R_1 \cdot R_2)$, 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:

- $\{ w \in \{0,1\}^* \mid w \text{ contains the substring 001} \}$
  
  $$(01^*)001(01^*)$$

- $\{ w \in \{0,1\}^* \mid w \text{ does not contain two consecutive 0's} \}$
  
  $1^* \cup 1^*0(1^0)^*1^*$

- $\{ w \in \{0,1\}^* \mid |w| \text{ is divisible by 2 or 3} \}$
  
  $$((01)(01))^* \cup ((01)(01)(01))^*$$
  
  $$\cup \ldots$$

- $\{ w \in \{0,1\}^* \mid |w| < 4 \}$
  
  $$\cup (01)^* \cup (01)^4 \cup (01)^5 \cup (01)^6 \cup (01)^7 \cup (01)^8 \cup (01)^9 = (01^*01^5)^3$$
Examples: let $R$ be any regular expression

- $R \cdot \emptyset = \emptyset$

- $R \cdot \epsilon = R$

- $\emptyset^* = \epsilon$

- $\epsilon^* = \epsilon$

The language defined by $R$ is denoted $L(R)$. We’ll often abuse notation and use $R$ to denote the language $L(R)$. 

- see the reasoning from a couple of weeks ago where we said that for any language $L$, $L^0 = \{\epsilon\}$
Equivalence of reg. expr. and FA’s

Kleene

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 \( L(M) = L(R) \).

- **BASE CASE:**
  1. If \( R = a \) for some \( a \in \Sigma \), then NFA: \( \rightarrow a \rightarrow \) (Example: \( 0 \cup (1 \cdot (100)^* \))
  2. If \( R = \varepsilon \), then NFA: \( \rightarrow 0 \)
  3. If \( R = \emptyset \), then NFA: \( \rightarrow 0 \)

- **INDUCTIVE CASE:**
  4. If \( R = (R_1 \cup R_2) \) for some reg. expr. \( R_1, R_2 \), then by IH (inductive hypothesis), assume we have NFAs \( N_1, N_2 \) for \( R_1, R_2 \).
  5. Analogous inductive cases for concat. and star.

We can create an NFA \( N \) for \( R \) by these steps.
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 $\Sigma \cup \{\epsilon\}$)
- 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 (including self-loops)
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 $\delta : (Q - \{q_{\text{accept}}\}) \times (Q - \{q_{\text{start}}\}) \rightarrow \mathcal{R}$ where $\mathcal{R}$ 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:

Example:

\begin{align*}
\text{NFA, convert to NFA:} & \quad \text{start state with} \quad \\
& \quad \text{a) all outgoing,} \quad \\
& \quad \text{b) none incoming.} \quad \text{NFA + arrows with } \varnothing
\end{align*}
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?

Example:

\[ \emptyset \cup (100)^* \epsilon = 0(100)^* \]

We are going to remove one of the inner states, e.g., the top one.

Simplifying:

Need to update all non-eliminated transitions.

Eliminate the bottom state.

Then eliminate the top state:

\[ \emptyset \cup (101^*(01)^* \epsilon \cdot 1^* \epsilon \cdot \emptyset = \emptyset \]

& had reg. expr.
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:

How to construct an equivalent GNFA with one fewer state?

Example:

we are going to remove one of the inner states, e.g. the top one

simplifying

need to update all non-eliminated transitions

Algo/construction:

1. for every internal state q in the GNFA (we’ll eliminate q):
2. for every pair of states r, s from the remaining states of the GNFA, r, s ≠ q:
3. update the transition from r to s to:
4. (when left with only 2 states)
   return δ(q_{start}, q_{accept})