CSE 322,  Fall 2010

Regular Expressions
Regular expressions over \( \mathcal{E} \):

\[
\theta \quad \text{is an \( \epsilon \text{-r.e.} \)}
\]

\[
\epsilon \quad \text{is an \( \epsilon \text{-r.e.} \)}
\]

\[
a \quad \text{is an \( \epsilon \text{-r.e.} \)}
\]

\[
\text{for each \( a \in \mathcal{E} \)}
\]

If \( R_1 \) and \( R_2 \) are \( \epsilon \text{-r.e.} \), then so are:

\[
(R_1 \cup R_2)
\]

\[
(R_1 \circ R_2)
\]

\[
(R_1^*)
\]

The language denoted by \( R \cup \emptyset \) is:

\[
L(\emptyset) = \emptyset
\]

\[
L(\epsilon) = \{ \epsilon \}
\]

\[
L(R_1 \cup R_2) = L(R_1) \cup L(R_2)
\]
\[ L(\phi^*) = L(\phi)^* \]
\[ \mathbb{R}^* = \phi^* \]
\[ \{\emptyset\} = \{\emptyset\} \]

**Short hands**

\[ \Sigma = \{a, b, c\} \]

\[ L((a \cup b) \cup c) = \Sigma \]

\[ (\Sigma^* \cup \Sigma) \cdot a \]

\[ (((a \cup b) \cup c) \cdot \Sigma^* \cup \Sigma) \cdot a \]

**precedence & associativity**

\[ (a \cup b \cup c) \]

\[ a \cup b \cdot c^* \]

\[ (a \cup (b \cdot (c^*))) \]
"words ending with ".TXT" + 
\( \Sigma^* .TXT \) 
\((a_u b_u \ldots u \Sigma) \cdot (a_u u_2 u_a u_a\ldots)^* \)
\( \Sigma^* \cdot \{ a_u d \}^* \)

(S3)*
0* 10*
(S3) \( \Sigma \)

\( \Sigma^* \cdot \Sigma^+ \)
\( a a^* \rightarrow a^+ \)

00 \in 0^* (10^* 10^*)^*
00 ^* \in (0^* 10^* 10^*)^*
(0^* 10^* 1)^* 0^*

\( (d^* .d^+ u d^+.d^*) \cdot (\Sigma u \Sigma) (u u \Sigma) (u + u -) d^+ \)
**Theorem:**
A regular expression $R \in \mathbb{R}$ has an NFA $M_R$ such that $L(M_R) = L(CM_R)$

**Proof:**
By induction on $k$, the number of operators in $R$

Base cases ($k = 0$):
Then $R$ is "$\emptyset$", "$\varepsilon$", or "$a$" for $a \in \Sigma$ explicitly give simple NFAs recognizing $\emptyset$, $\{\varepsilon\}$, and $\{a\}$ for each $a \in \Sigma$ (details omitted)

Induction Step ($R$ has $k > 0$ operators)

**I.H.:** Assume that for all regular expressions $R'$ with $\leq k$ operators, there exists an NFA $M_{R'}$ recognizing $L(CM_{R'})$

If $R$ has $k > 0$ operators, it can be expressed as $R = (R_1 \cup R_2)$ or $(R_1 \cdot R_2)$ or $(R_1)^*$, where $R_1$, $R_2$ (if any) have $\leq k - 1$ operators. By I.H., there exist NFAs $M_{R_1}$, $M_{R_2}$ such that $L(M_{R_i}) = L(CM_{R_i})$, $i = 1, 2$. Modify/combine these NFAs as in previous proofs of closure under $\cup$, $\cdot$, or $(\cdot)^*$ to get $M_R$ such that $L(M_R) = L(CM_R)$. 

Example

\[(ab)^* \cup a\]
Converse?

For every D/NFA $M$ reg expr defining the same language

\[ (ab)^+ \]
Every regular language can be described by a regular expression
Note: No loss in assuming no edges into go / out of F / only one if ∈ F

no longer finish.
GNFA

\[ G = (Q, \Sigma, \delta, q_0, F) \]

\( Q, \Sigma, q_0, F \) as usual

\[ \delta: (Q \times \Sigma) \times (Q \times \Sigma) \to \mathcal{P}(\Sigma) \]

**Define**

- **G can be in state** \( q \in Q \) **after reading** \( \chi \in \Sigma^* \) **if** \( \exists k \geq 0, \exists r_0, r_1, \ldots, r_k \in Q \)

\[ \exists \chi_1, \ldots, \chi_k \in \Sigma^* \]

**Such that**

\[ \begin{align*}
\text{(i)} & \quad \chi = \chi_1 \cdot \chi_2 \cdot \ldots \cdot \chi_k \\
\text{(ii)} & \quad r_0 = q_0 \\
\text{(iii)} & \quad r_k = q \\
\text{(iv)} & \quad \forall 1 \leq i \leq k, \chi_i \in L(\delta(r_{i-1}, r_i))
\end{align*} \]

- **L(G) = \{ \chi \mid G \text{ can be in state } q_0 \ldots q \}**

**Note:** \( \delta \) syntax a little different; maps state pair to label (reg. exp.) rather than state x symbol -> new state.
Theorem

If \( L \) is accepted by a CNFA, then \( L \) is regular.

Proof sketch:
Replace edge labeled "r" by NFA equivalent to \( r \) based on previous theorem.
If $L$ is regular, then $L = L(R)$ for some regular expression $R$.

Proof will take FA for $L$, & reduce it to a (G)NFA for same $L$ with progressively fewer states until $R$ becomes obvious.
Q: What strings accepted by $L_0 \rightarrow L_1 \rightarrow L_f$?

\[ \{ \mathbf{w} \mid \mathbf{w} = \mathbf{x}_1 \mathbf{x}_2 \text{ with } \mathbf{x}_1 \in L_1, \mathbf{x}_2 \in L_5 \} \]

\[ = L_1 \cdot L_5 \]

\[ \mathbf{b}_0 \rightarrow \mathbf{b}_1 \rightarrow \mathbf{b}_2 \rightarrow \mathbf{b}_3 \mathbf{b}_4 \rightarrow \mathbf{b}_5 \]

\[ L = \bigcup_{\text{path } p} \text{concatent } L_0 \text{ on path } p \]
\[(\emptyset \cup \varepsilon 0^*\varepsilon) \cup (\emptyset \cup \varepsilon 0^*1)(0 \cup 10^*1)^*(\emptyset \cup 10^*\varepsilon) = 0^* \cup (0^*1)(0 \cup 10^*1)^*(10^*)\]
Given \( \mathcal{G} = (Q, \Sigma, \delta, q_0, F) \)
with \( > 2 \) states

**Notation** \( \forall \delta_i \neq \delta_f, \delta_j \neq \delta_f \)
\[ r_{ij} = \delta(\delta_i, \delta_f) \]

**Pick** any state \( \delta_k \neq \delta_f \)

**Build** \( \mathcal{G}' = (Q', \Sigma, \delta', q_0, F) \)
with one less state as follows:
\[ Q' = Q - \{ \delta_k \} \]

\[ \delta'(\delta_i, \delta_f) = r_{ij} = r_{ij} \cup \{ \delta_k \} \cup r_{ij} \]

Claim: \( \mathcal{G} \) & \( \mathcal{G}' \) are equivalent

To prove this, it is useful to focus on a subproblem: how do edges in \( \mathcal{G}' \) relate to paths in \( \mathcal{G} \)?
In a nutshell, delete state $k$ from $G$, but enlarge language on each edge to compensate, so that potential contribution of $k$ is added to each edge in $G'$.
Relating edges of $G'$ to paths of $G$

A path in $G$: any sequence of states
A simple path in $G$: any sequence of
32 states s.t. 1st & last are not $k$,
and all intermediate ones (if any) are $k$.

\[
\begin{align*}
&i \rightarrow j \\
&i \rightarrow k \rightarrow j' \\
&i \rightarrow k \rightarrow k \rightarrow j' \\
&\vdots
\end{align*}
\]

The Point:
(a) every path in $G$ can be decomposed into simple paths
(b) every edge in $G'$, say $i \rightarrow j$, corresponds to the set of all simple paths in $G$ with those end points

other edges
ignored/forbidden
Claim 2

\[ L(r_{ij}) = \{ w \mid G \text{ can move from } i \text{ to } j \text{ reading } w \text{ and passing through no intermediate states except possibly } k \}. \]

Equivalently:

\[ L(r_{ij}) = \{ w \mid G \text{ can move from } i \text{ to } j \text{ reading } w \text{ along a simple path } \exists \]. \]

\[ = L(r_{ij} \cup r_{ik} \cup r_{kj} \cup r_{ij}) \]
Claim 11: A NFA is equivalent to a regular expression.

Proof: $NFA \rightarrow GNFA \rightarrow 2$-stat $\rightarrow P.E.$

by induction

using claim 1
(a)

(b)

(c)

(d)

(e)

\[(a(aa \cup b)^*ab \cup bb)((ba \cup a)(aa \cup b)^*ab \cup bb)^*((ba \cup a)(aa \cup b)^* \cup \varepsilon) \cup a(aa \cup b)^*\]
Summary

$L$ is regular $\iff$

$L \in L(M)$ for some DFA $M$

$L = L(C(N))$ $\iff$ NFA $N$

$L \in L(C(G))$ $\iff$ GNFAG

$L \in L(CR)$ $\iff$ Reg.exp. $R$