CSC236 fall 2012

regular languages, regular expressions

Danny Heap
heap@cs.toronto.edu
BA4270 (behind elevators)
http://www.cdf.toronto.edu/~heap/236/F12/
416-978-5899

Using Introduction to the Theory of Computation,
Chapter 7
Outline

regular expressions, regular languages

notes
they’re equivalent:

\[ L = L(M) \] for some DFSA \( M \) \( \iff \) \( L = L(M') \) for some NFSA \( M' \) \( \iff \)

\[ L = L(R) \] for some regular expression \( R \)

step 1: convert \( L(M) \) to \( L(R) \), eliminate states
they’re equivalent:

\[ L = L(M) \] for some DFSA \( M \) \( \Leftrightarrow \)
\[ L = L(M') \] for some NFSA \( M' \) \( \Leftrightarrow \)
\[ L = L(R) \] for some regular expression \( R \)

step 1: convert \( L(M) \) to \( L(R) \), eliminate states

\[
R^* S (Q + TR^* S)^* \]
dynamic (table based) program
equivalence...

state elimination recipe for state $q$ 

1. $s_1 \ldots s_m$ are states with transitions to $q$, with labels $S_1 \ldots S_m$
2. $t_1 \ldots t_n$ are states with transitions from $q$, with labels $T_1 \ldots T_n$
3. $Q$ is any self-loop on $q$
4. Eliminate $q$, and add (union) transition label $S_i Q^* T_j$ from $s_i$ to $t_j$. 

\[
\text{regex} \quad R^* S (Q + T R^* S)^* 
\]
equivalence:
step 2: convert \( L(R) \) to \( L(M) \):
start with \( \emptyset, \varepsilon, a \in \Sigma \)

\[
\Sigma = \{ 0, 3 \}
\]

\[
s \rightarrow \bigcirc
\]

accepts \( L(\emptyset) \)

\[
s \rightarrow \bigcirc \rightarrow \bigcirc
\]

accepts \( L(\varepsilon) \)

\[
s \rightarrow \bigcirc \rightarrow \bigcirc \rightarrow \bigcirc
\]

accepts \( L(3) \)
equivalence...

step 2.5: convert $L(R)$ to $L(M)$:
union, concatenation, stars

$M_1$ accepts $L(R)$, $M_2$ accepts $L(S)$

$M$ accepts $L(R+S)$

$M$ accepts $L(RS)$

$M$ accepts $L(R^*)$
Language $L = \{ s \in \{0,1\}^* \mid s = 1^n0^n, n \in \mathbb{N} \}$

M1 accepts 0
$S \rightarrow O \quad M_1$
$S \rightarrow O \quad M_2$

E.g.
$\text{grep}$

summer
socks
sacks

notes