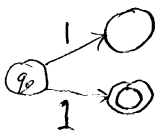
Kevin Matulef
Office hours: Wed 4-6, 2-349
In applied math common room
Office: State G622
Phone: 617-275-6958

9/15/05

Review

- DFAs, NFAs, showed they are equivalent
- Regular Langs (DFA/NFAs) closed under
  - Union
  - Concatenation
  - *
  - Complement (just switch accept/reject states of a DFA (not an NFA))
  - Intersection
- Regular Expressions equivalent to Reg languages
- Pumping Lemma



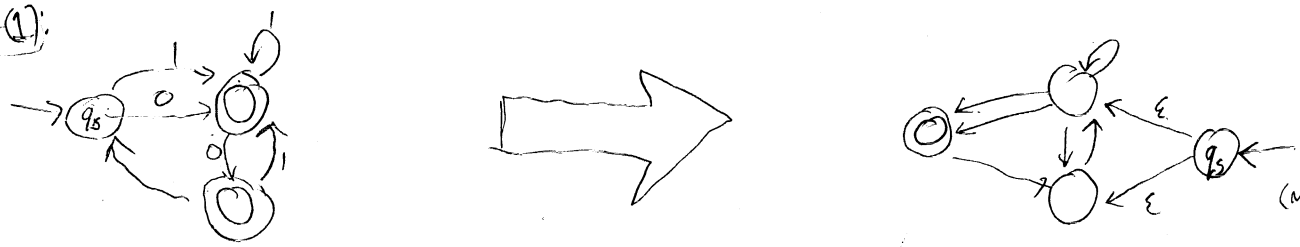
accepts 1

accepts 1

Q1 For a language A, let A^R be the "reversal" of A

ie. $A^R = \{ w_1 \cdots w_n \mid w_n w_{n-1} \cdots w_1 \in A \}$. Show that if A regular $\Rightarrow A^R$ regular

Proof (1):



(Note: new machine is NFA)

If M recognizes A, reverse M's transition arrows, add new start state, with ε-transition to old accept states. Make old start state an accept state

Proof (2): A is represented by a reg exp. Just reverse that exp' (eg. $0(1\cup01)^*$ --> $1(1\cup10)^*0$)

Q (Addition) $\Sigma = \left\{ \begin{bmatrix}0\\0\end{bmatrix}, \begin{bmatrix}0\\1\end{bmatrix}, \begin{bmatrix}1\\0\end{bmatrix}, \ldots \right\}$ all 8 combos

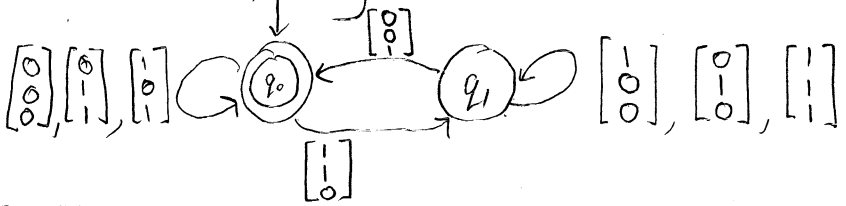
$B = \{ w \in \Sigma^* \mid$ the bottom row of w is the sum of the top two rows $\}$

e.g. $\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} \in B$, $\begin{bmatrix}0\\1\end{bmatrix}\begin{bmatrix}1\\0\end{bmatrix} \notin B$

Show B is regular.

Proof: Actually, we'll show B^R is regular.

To do addition, only need to remember carry bit (q_0 - state w/o carry; q_1 - state where 1 is carried)



(Note: machine is ND)

Pumping Lemma: For every reg lang A, there is a number p, such that for any string $s \in A$ with $|s| \geq p$, we can divide s into three parts, $s = xyz$ such that

1) $xy^iz \in A$ for every $i \geq 0$
2) $|y| > 0$
3) $|xy| \leq p$

Example, $\{0^n1^n \mid n \geq 0\}$ is not regular, take string 0^p1^p

---

Q| Show $\{0^i1^j \mid i > j\}$ is not regular

A| Take string $0^{p+1}1^p$, and "pump down" (at most p) to get something like 0^i1^p, contradiction!

Converting DFA's to Reg Exps

Build a GNFA, which has a reg-exp on each transition

1) add new start state, connect with ε-transition to old start state
2) add new accept state, connect with ε-transitions from old accept states
3) Put ϕ transitions between states not already connected)
4) "Rip" states out 1 at a time, updating all remaining transitions to include the "ripped out" paths

See examples 1.67, 1.68 on pages 75-76 of the book