MA/CSSE 474

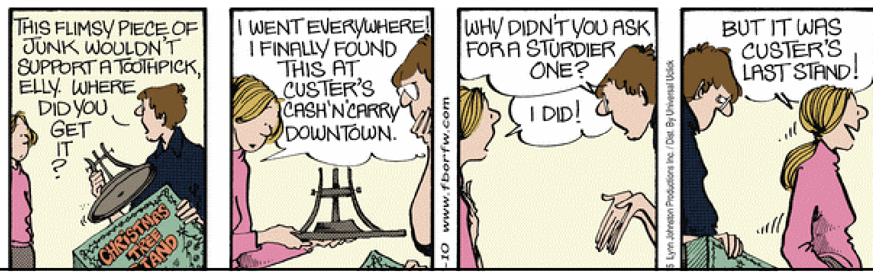
Theory of Computation

DFSM to RE

Exam discussion

Your Questions?

- Previous class days' material
- Reading Assignments
- HW6 problems
- Anything else



Recap: Kleene's Theorem

Finite state machines and regular expressions define the same class of languages.

To prove this, we must show:

Theorem*:* Any language that can be defined by a regular expression can be accepted by some FSM and so is regular. **Done last time.**

Theorem: Every regular language (i.e., every language that can be accepted by some DFSM) can be defined with a regular expression.

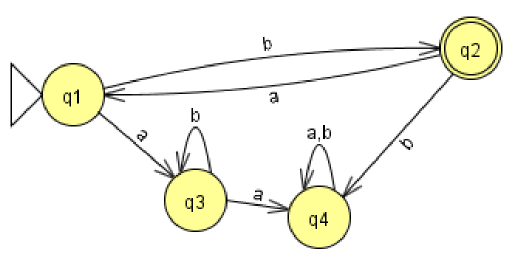
For Every FSM There is a Corresponding Regular Expression

- We'll show this by construction.
  The construction is different than the textbook's.
- Let $M = (\{q_1, \ldots, q_n\}, \Sigma, \delta, q_1, A)$ be a DFSM.
  Define **R_{ijk}** to be the set of all strings $x \in \Sigma^*$ such that
  - $(q_i, x) \vdash_M^* (q_j, \varepsilon)$, and
  - if $(q_i, y) \vdash_M^* (q_\ell, \varepsilon)$, for any prefix y of x
    (except $y = \varepsilon$ and $y = x$), then $\ell \leq k$
- That is, R_{ijk} is the set of all strings that take us from q_i to q_j without passing through any intermediate states numbered higher than k.
  - In this case, "passing through" means both entering and leaving.
  - Note that either i or j (or both) may be greater than k.

Example: R_{ijk}

- R_{ijk} is the set of all strings that take us from q_i to q_j without passing through any intermediate states numbered higher than k.
  - In this case, "passing through" means both entering and leaving.
  - Note that either i or j (or both) may be greater than k.

R_{110}
R_{111}
R_{112}
R_{131}
R_{132}
R_{330}
R_{333}
R_{142}
R_{143}



DFSM→Reg. Exp. construction

- R_{ijk} is the set of all strings that take M from q_i to q_j without passing through any intermediate states numbered higher than k.
- Examples: R_{ij0} R_{ijn}
- We will show that for all $i,j \in \{1, \ldots, n\}$ and all $k \in \{0, \ldots, n\}$, R_{ijk} there is a regular expression r_{ijk} that defines R_{ijk}.
- Also note that L(M) is the union of R_{1jn} over all q_j in A.
  - We know that the union of languages defined by reg. exps. is defined by a reg. exp.

DFSM→Reg. Exp. continued

- R_{ijk} is the set of all strings that take M from q_i to q_j without passing through any intermediate states numbered higher than k.

  It can be computed recursively:
- Base cases (k = 0):
  - If $i \neq j$, $R_{ij0} = \{a \in \Sigma : \delta(q_i, a) = q_j\}$
  - If $i = j$, $R_{ii0} = \{a \in \Sigma : \delta(q_i, a) = q_i\} \cup \{\varepsilon\}$
- Recursive case (k > 0):

  R_{ijk} is $R_{ij(k-1)} \cup R_{ik(k-1)}(R_{kk(k-1)})^* R_{kj(k-1)}$
- We show by induction that each R_{ijk} is defined by some regular expression r_{ijk}.

DFSM→Reg. Exp. Proof pt. 1

- Base case definition (k = 0):
  - If $i \neq j$, $R_{ij0} = \{a \in \Sigma : \delta(q_i, a) = q_j\}$
  - If $i = j$, $R_{ii0} = \{a \in \Sigma : \delta(q_i, a) = q_i\} \cup \{\varepsilon\}$
- **Base case proof:**
  R_{ij0} is a finite set of symbols, each of which is either ε or a single symbol from Σ.
  So R_{ij0} can be defined by the reg. exp.
  $r_{ij0} = a_1 \cup a_2 \cup \ldots \cup a_p$ (or $a_1 \cup a_2 \cup \ldots \cup a_p \cup \varepsilon$ if i=j),
  where $\{a_1, a_2, \ldots, a_p\}$ is $\{a \in \Sigma : \delta(q_i, a) = q_j$.
- **Note** that if M has no direct transitions from q_i to q_j, then r_{ij0} is $\varnothing$ (it is ε if i=j and no "loop" on that state).

DFSM→Reg. Exp. Proof pt. 2

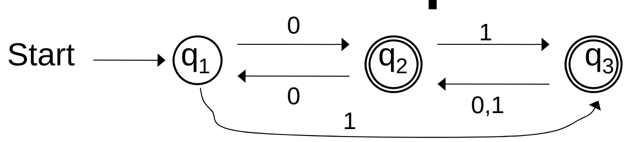
- Recursive definition (k > 0):
  R_{ijk} is $R_{ij(k-1)} \cup R_{ik(k-1)}(R_{kk(k-1)})^* R_{kj(k-1)}$
- **Induction hypothesis:** For each ℓ and m, there is a regular expression $r_{\ell m k-1}$ such that $L(r_{\ell m(k-1)}) = R_{\ell m(k-1)}$.
- **Induction step**. By the recursive parts of the definition of regular expressions and the languages they define, and by the above recursive definition of R_{ijk} :
  $R_{ijk} = L(r_{ij(k-1)} \cup r_{ik(k-1)}(r_{kk(k-1)})^* r_{kj(k-1)})$

DFA→Reg. Exp. Proof pt. 3

- We showed by induction that each R_{ijk} is defined by some regular expression r_{ijk}.
- In particular, for all $q_j \in A$, there is a regular expression r_{1jn} that defines R_{1jn}.
- Then $L(M) = L(r_{1j_1n} \cup \ldots \cup r_{1j_pn})$,

  where $A = \{q_{j_1}, \ldots, q_{j_p}\}$
- The union of finitely many regular expressions is a regular expression.

An Example



	k=0	k=1	k=2
r_{11k}	ε	ε	$(00)^*$
r_{12k}	0	0	$0(00)^*$
r_{13k}	1	1	0^*1
r_{21k}	0	0	$0(00)^*$
r_{22k}	ε	$\varepsilon \cup 00$	$(00)^*$
r_{23k}	1	$1 \cup 01$	0^*1
r_{31k}	$\varnothing$	$\varnothing$	$(0 \cup 1)(00)^*0$
r_{32k}	$0 \cup 1$	$0 \cup 1$	$(0 \cup 1)(00)^*$
r_{33k}	ε	ε	$\varepsilon \cup (0 \cup 1)0^*1$