A2 — posted by section (Lila, Colin, ...).

T2 spoiler:
B average,
back Friday.

CSC236 fall 2012

automata and languages

FSA

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

formal languages

FSAs

notes

some definitions

Suppose $\{0, 1, \varepsilon\}$ $\quad 0, \; 1, \; 1\varepsilon \stackrel{?}{\equiv} 1$

alphabet: finite, non-empty set of symbols, e.g. $\{a, b\}$ or $\{0, 1, -1\}$. Conventionally denoted Σ.

a, ε

string: finite (including empty) sequence of symbols over an alphabet: abba is a string over $\{a, b\}^*$. $\not\ni a+b$
Convention: $\boxed{\varepsilon}$ is the empty string, never an allowed symbol, Σ^* is set of all strings over Σ.

$\Lambda \quad \in$

$\Sigma = \{0, 1\}, \quad \Sigma^* = \{\varepsilon, 0, 1, 01\} \quad 0 \quad 01 \quad 0111101011001$

language: Subset of Σ^* for some alphabet Σ. Possibly empty, possibly infinite subset. E.g. $\{\}$, $\{aa, B's\}$.

$L = [""]$

$L2 = []$

N.B.: $\{\} \neq \{\varepsilon\}$.

Many problems can be reduced to languages: logical formulas, identifiers for compilation, natural language ~~recognition~~ processing. Key question is recognition:

Given language L and string s, is $s \in L$?

Languages may be described either by descriptive generators (for example, regular expressions) or procedurally (e.g. finite state automata)

more notation

string length: denoted $|s|$, is the number of symbols in s, e.g. $|bba| = 3$.

$s = t$: if and only if $|s| = |t|$, and $s_i = t_i$ for $1 \leq i \leq |s|$.

s^R: reversal of s is obtained by reversing symbols of s, e.g. $1011^R = 1101$. $\varepsilon^R = \varepsilon$ $\quad 1^R = 1$

$abaa^R = aaba$

st or $s \circ t$: contcatenation of s and t — all characters of s followed by all those of t, e.g. $bba \circ bb = bbabb$.

s^k: denotes s concatenated with itself k times. E.g., $ab^3 = ababab$, $101^0 = \varepsilon$.

Σ^n: all strings of length n over Σ, Σ^* denotes all strings over Σ.

$\Sigma^0 = \{\varepsilon\}$

language operations

Σ^* Σ^{*} $\tilde{L}$ $\sim L$ $L = \{0, 1, 01, 11, 10, 00\}$

$\overline{L}$: Complement of L, i.e. $\Sigma^* - L$. If L is language of strings over $\{0, 1\}$ that start with 0, then $\overline{L}$ is the language of strings that begin with 1 plus the empty string.

$L \cup L'$: union $\{a, b, ba\} = L'$

$L \cap L'$: intersection

$L - L'$: difference
$L \setminus L'$

states needed to classify a string

what state is a stingy vending machine in based on coins?
accepts only nickles (a), dimes (b), and quarters (c),
no change given, and everything costs 30 cents
useful toy (you'll need JRE)

n d n

δ	0	5	10	15	20	25	≥ 30
n	5	10	15	20	25	≥ 30	≥ 30
d	10	15	20	25	≥ 30	≥ 30	≥ 30
q	25	≥ 30	≥ 30	≥ 30	≥ 30	≥ 30	≥ 30

build an automaton with formalities...

devise transitions that don't commute

quintuple: $(Q, \Sigma, q_0, F, \delta)$

Q is set of states, Σ is finite, non-empty alphabet, q_0 is start state

F is set of accepting states, and $\delta : Q \times \Sigma \mapsto Q$ is transition function

e.g. {0¢, 5¢, 10¢, 15¢, 20¢, 25¢, ≥30¢}

We can extend $\delta : Q \times \Sigma \mapsto Q$ to a transition function that tells us what state a **string** takes the automaton to:

$$\delta^*(5, ndn) = \delta(\delta^*(5, nd), n)$$
$$= \delta(\delta(\delta^*(5, n), d), n)$$
$$= \delta(\delta(\delta(\delta^*(5, \varepsilon), n), d), n)$$

$$\delta^* : Q \times \Sigma^* \mapsto Q \qquad \delta^*(q, s) = \begin{cases} q & \text{if } s = \varepsilon \\ \delta(\delta^*(q, s'), a) & \text{if } s' \in \Sigma^*, a \in \Sigma \end{cases}$$

$s = s'a$

String s is accepted if and only iff $\delta^*(q_0, s) \in F$, it is rejected otherwise.

example — an odd machine

devise a machine that accepts strings over $\{a, b\}$ with an odd number of as

Formal proof requires inductive proof of invariant:

$$\delta^*(E, s) = \begin{cases} E & \text{if } s \text{ has even number of } a\text{s} \\ O & \text{if } s \text{has odd number of } a\text{s} \end{cases}$$

notes