Computers: What can’t they do!
Computers: What can’t they do?
Part 2: NFAs and Regexps; Turing Machines

November 22, 2011
CS 42: Principles and Practice of Computer Science
The Story So Far…

✓ If we want to know what can/cannot be computed, it’s enough to consider decision problems.

✓ Every decision problem corresponds to a language, a set of inputs (finite strings) where the answer is yes.

✓ DFAs (a.k.a. deterministic finite automata a.k.a. deterministic finite state machines) can model any deterministic mechanism with a finite number of configurations (finite memory).
A Jewel of Theoretical Computer Science

The following are equivalent:

✓ There is a DFA accepting the language $L$.
✓ [Rabin and Scott] There’s an NFA accepting $L$.
✓ [Kleene] $L$ can be described by a regular expression.
DFAs + Nondeterminism = NFAs

An NFA is a has the magic power to “guess” the right way to go, and can even change states without reading input (ε transitions).

NFA accepts the string if there exists at least one accepting path.
What’s Accepted?  (bb?  b?  aaab?  ba?)
Regular Expressions: Simple Language Specifications

By example:

✓ $1^* | 1(00)^*$

$$\{ w \in \{0, 1\}^* | w \text{ is a sequence of (zero or more) ones} \} \cup \{ w \in \{0, 1\}^* | w \text{ is a 1 followed by an even number of zeros.} \}$$

✓ $b(ea|a)(r|d)$

$$\{ \text{bear, bead, bar, bad } \}$$

✓ $(0|1)(0|1)1(0|1)^*$

$$\{ w \in \{0, 1\}^* | \text{w's third character is a 1} \}$$
Regular Expression Ingredients

A regular expression $s$ can be:

- $\emptyset$
- $\epsilon$ (or $\lambda$)
- A single character from the alphabet (e.g., $a$ or $0$)
- Concatenation: $r_1 r_2$
- Union: $r_1 \cup r_2$
- Repetition (Kleene Star): $r_1^*$.
Regular Expressions in Practice

Unix’s `egrep` command does line-by-line search for text matching a regular expression.

```bash
gegrep  'hh'       /usr/share/dict/words
gegrep  'y.*y'    /usr/share/dict/words
gegrep  '(xq|hq)' /usr/share/dict/words
gegrep  '^y.*y$'  /usr/share/dict/words
```

Many systems, including `egrep` add extra operations (mostly, but not entirely, abbreviations)
REs to the Rescue!

but how does regular expression matching actually **work**?
A Jewel of Theoretical Computer Science

The following are equivalent:

✓ There is a DFA accepting the language \( L \).
✓ [Rabin and Scott] There’s an NFA accepting \( L \).
✓ [Kleene] \( L \) can be described by a regular expression.
Can this be simulated by a DFA?
Can we turn this Regular Expression into an NFA?

\[ 1^* | 1(00)^* \]

Not covered in this class: DFAs to regular expressions
NONREGULAR LANGUAGE THEOREM

Given a language \( L \), two strings \( x \) and \( y \) are distinguishable if there is some string \( z \) such that \( xz \in L \) and \( yz \notin L \), or vice-versa.

If strings are distinguishable, running them through a state machine for \( L \) must lead you to different states.

If there is an infinite set of pairwise-distinguishable strings, then none of them can lead you to the same state-machine state. Thus, there cannot be a finite state machine recognizing \( L \).

A language is regular if it can be recognized by a finite state machine, and nonregular otherwise.
Turing Machines

Any computer is a finite state machine.

Hence, there are questions it provably cannot answer. (E.g., “is the input a palindrome?”)

But it sure seems like computers can answer these questions (until we run out of memory), so maybe this isn’t so helpful in practice?

Perhaps we should consider more powerful models of computation. (But not too powerful! Why?)
**Alan Turing**

**WWII**

Enigma machine ~ The axis's encryption engine

1946

**2007**

Bletchley Park
Turing Machines

A simple model of universal computation.

✓ Control/Program: a finite state machine

✓ Data: An infinite tape, with a read/write head.
   Initially blank except for the input string.

✓ Steps determined by current state + current symbol;
   Can overwrite current symbol, and move left or right.

✓ Can stop (and say yes/no) or not.
TURING MACHINE: ARTIST’S CONCEPTION

HMC CS!
TM Demo(s)

http://ironphoenix.org/tril/tm/

http://www.youtube.com/watch?v=cYw2ewo06c4