Chapter 4 (sort of): How Universal Are Turing Machines?

CS105: Great Insights in Computer Science
Homework 3 is out!

Due Wednesday, Oct 21st via Sakai
Harry Potter and the Goblet of Clickers
## Recap: Reduction

<table>
<thead>
<tr>
<th>Level</th>
<th>Examples</th>
<th>Alternatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>software libraries</td>
<td>graphics, animation, robotics</td>
<td>networking, security, mathematics</td>
</tr>
<tr>
<td>high-level language</td>
<td>Python</td>
<td>C, Java, C++, Logo, LISP, Fortran, ML</td>
</tr>
<tr>
<td>machine language</td>
<td>ML³</td>
<td>x86, CARDIAC, Z80</td>
</tr>
<tr>
<td>logic gates</td>
<td>equal, ifthenelse, add</td>
<td>memlookup, memwrite</td>
</tr>
<tr>
<td>basic logic gates</td>
<td>and, or, not</td>
<td>nor, nand, xor</td>
</tr>
<tr>
<td>physical bits</td>
<td>0,1 via high/low voltage</td>
<td>water pressure, kinetic energy</td>
</tr>
</tbody>
</table>
Computer Hierarchy

- Logic (and, or, not)
- Boolean Algebra
- Programing Languages (Scratch, Java, etc...)
- English

Low Complexity → High Complexity
The Element of Surprise

• A lot of CS research has gone into understanding the powers, limits, and philosophy of... coin flips!

• Later, I’ll show how computers use a bit of randomness to speed up their computations.

• For now, let’s take a brief sidebar into some probability theory that will be useful.
Using Random Bits

• Since numbers are made of bits, we can generate a random number using random bits.

• If there’s a way to create random bits (coin flips), how make a random number from 0 to 3 (dreidel)?

• How about 0 to 15?

• Tricky: How about 0 to 2?
Examples

when I receive randbit
set bit to pick random 0 to 1

when I receive rand4
broadcast randbit and wait
set bit1 to bit
broadcast randbit and wait
set bit2 to bit
set answer to $2 \times bit2 + bit1$

when I receive rand3
broadcast randbit and wait
set answer to 3
repeat until answer < 3
set bit1 to bit
broadcast randbit and wait
set bit2 to bit
set answer to $2 \times bit2 + bit1$

when I receive rand16
broadcast randbit and wait
set bit1 to bit
broadcast randbit and wait
set bit2 to bit
broadcast randbit and wait
set bit3 to bit
broadcast randbit and wait
set bit4 to bit
set answer to $8 \times bit4 + 4 \times bit3 + 2 \times bit2 + bit1$
State of the Art

- The ... Mersenne twister algorithm, by Makoto Matsumoto and Takuji Nishimura in 1997 ... has a colossal period of \(2^{19937}-1\) iterations (probably more than the number of computations which can be performed in the future existence of the universe), is proven to be equidistributed in 623 dimensions (for 32-bit values), and runs faster than all but the least statistically desirable generators.

- Python has a package for this generator.
Demo: Guess My Bit

- Six rounds:
  - Guess my bit. If wrong, you’re out.
  - Anyone who can survive all 6 rounds gets extra credit.
Some Probability Theory

• If event has probability $p$, $1/p$ tries before it happens (on average).

• If $n$ distinct events are equally likely, $\sim n \ln n$ tries before we see all $n$ of them (order independent) or $\sim n^2$ (in order).

• If we start with a number $n$, we can cut it in half log $n$ times before 1 is reached.
Here’s Where We Stand

• We discussed how a computer could be created starting from bits and wires and working up to a high-level language.

• In classic CS style (reduction!), we now take the lower levels for granted and build on them to create new capabilities.

• We will now operate on top of the hierarchy, asking questions about what we can do now that we have the entire computer.