Computer Theory

Michael J. Watts

http://mike.watts.net.nz

Lecture Outline

- Turing machines
- Computability
- Representation issues

Turing Machines

- Simple theoretical model
  - computability
- Basis of modern computers
- Finite state machines
- Equivalent to a digital computer
- Deal with an infinitely long tape
- Tape has a finite number of non-blank squares

Turing Machines

- Each square has a symbol from a finite alphabet
  - A datum
- Has a read-write head
- Reads a symbol
- Symbol + current state
  - Writes a new symbol
  - Moves left or right on the tape

Turing Machines

- Continues until it reaches an unknown condition
- All computer languages and architectures are equivalent to Turing machines
- Universal Turing machine
  - Generalisation
  - Reads instructions off of tape

Turing Machines

- Nondeterministic Turing machine
  - Adds a write-only head
  - Writes a guess at the solution
  - Based on internal “rule”

Computability

- "A function is computable if can be computed with a Turing machine"
  - http://www.ams.org/new-in-math/cover/turing.html
- Valid input -> algorithm -> correct output
- Some problems are not computable
  - Halting problem

Computability

- Polynomial time
- NP-Complete
  - Non-deterministic polynomial time
  - NP-Hard
- Many optimisation problems are NP-complete or NP-hard
- Hamilton path
  - Travelling salesman

Computability

- Exponential time
  - Number of steps is an exponential function of complexity
- Encryption breaking
- Factorial complexity
  - Don't bother

Representation

- Numbers in computers are represented in binary
  - Base two numbers
  - Integers / floating point
- Floating point
  - Single / double precision
- Problems
  - Recurring digits
  - Accuracy

Summary

- Turing machines are the basis of computer theory
- Any function that can be computed by a Turing machine in computable
- Some problems are not computable
- Some problems are infeasible
- Problems with representation of numbers in computers