Lecture 20: Course Review

6.006: Introduction to Algorithms

- Goals:
  1. Solve hard computational problems (with **non-constant-sized inputs**)
  2. Argue an algorithm is **correct** (Induction, Recursion)
  3. Argue an algorithm is **"good"** (Asymptotics, Model of Computation)

  – (effectively communicate all three above, to human or computer)
- Do there always exist "good" algorithms?
  – Most problems are not solvable efficiently, but many we think of are!
  – **Polynomial** means polynomial in size of input
  – **Pseudopolynomial** means polynomial in size of input AND size of numbers in input
  – NP: **Nondeterministic Polynomial** time, polynomially checkable certificates
  – NP-hard: set of problems that can be used to solve any problem in NP in poly-time
  – NP-complete: intersection of NP-hard and NP

How to solve an algorithms problem?

- Reduce to a **problem** you know how to solve
  – Search/Sort (Q1)
    * Search: Extrinsic (Sequence) and Intrinsic (Set) Data Structures
    * Sort: Comparison Model, Stability, In-place
  – Graphs (Q2)
    * Reachability, Connected Components, Cycle Detection, Topological Sort
    * Single-Source / All-Pairs Shortest Paths
- Design a new recursive algorithm
  – Brute Force
  – Divide & Conquer
  – Dynamic Programming (Q3)
  – Greedy/Incremental

Next Steps

- (U) 6.046: Design & Analysis of Algorithms
- (G) 6.851: Advanced Data Structures
- (G) 6.854: Advanced Algorithms

6.046

- Extension of 6.006
  - **Data Structures:** Union-Find, Amortization via potential analysis
  - **Graphs:** Minimum Spanning Trees, Network Flows/Cuts
  - **Algorithm Design (Paradigms):** Divide & Conquer, Dynamic Programming, Greedy
  - **Complexity:** Reductions
- Relax Problem (change definition of correct/efficient)
  - **Randomized Algorithms**
    * 6.006 mostly deterministic (hashing)
    * Las Vegas: always correct, probably fast (like hashing)
    * Monte Carlo: always fast, probably correct
    * Can generally get faster randomized algorithms on structured data
  - **Numerical Algorithms/Continuous Optimization**
    * 6.006 only deals with integers
    * Approximate real numbers! Pay time for precision
  - **Approximation Algorithms**
    * Input optimization problem (min/max over weighted outputs)
    * Many optimization problems NP-hard
    * How close can we get to an optimal solution in polynomial time?
- Change Model of Computation
  - Cache Models (memory hierarchy cost model)
  - Quantum Computer (exploiting quantum properties)
  - Parallel Processors (use multiple CPUs instead of just one)
    * Multicore, large shared memory
    * Distributed cores, message passing

Future Courses

Model

- Computation / Complexity (6.045, 6.840, 6.841)
- Randomness (6.842)
- Quantum (6.845)
- Distributed / message passing (6.852)
- Multicore / shared memory (6.816, 6.846)
- Graph and Matrix (6.890)
- Constant Factors / Performance (6.172)

Application

- Biology (6.047)
- Game Theory (6.853)
- Cryptography (6.875)
- Vision (6.819)
- Graphics (6.837)
- Geometry (6.850)
- Folding (6.849)