Where to Go From Here

Announcements

- Problem Set Six due right now with a late day.
  - Solutions released at the end of lecture.
- Final project due Saturday at 12:15PM.
  - Note that this is 12:15 in the afternoon rather than 12:15 at night.
  - At least one staff member will be in the Gates building to let you in if you need access.
- Limited OH for the rest of this week:
  - Andy's Wednesday OH canceled.
  - Keith's Thursday OH canceled.
  - Julie's Thursday OH canceled.
  - Feel free to email the staff list with questions!

Please evaluate this course on Axess.

Your feedback really makes a difference.

Where We've Been

Where We've Been

- Graph Algorithms
- Divide-and-Conquer Algorithms
- Randomized Algorithms
- Greedy Algorithms
- Dynamic Programming
- Intractability

Further Directions

- **New Algorithmic Techniques**
  - What are other approaches to problem-solving?
- **New Application Domains**
  - To what areas of study should we apply algorithmic techniques?

Further Directions in Algorithms

Approximation Algorithms

- How do you approximate intractable problems?
- What techniques are necessary for designing and analyzing approximation algorithms?
- ***Take CS261!***

Parallel Algorithms

- How do you solve standard algorithmic problems (searching, sorting, graph searches, etc.) quickly on parallel machines?
- How do you design data structures that support concurrent modification?
- How do you farm out work across multiple computers and aggregate the results?
- ***Take CS149!***

Cache-Oblivious Algorithms

- How do you design algorithms that are always cache-friendly regardless of cache size?
- How do you build data structures that provably minimize cache misses?
- This can make a *huge* difference given modern memory architectures!

Geometric Algorithms

- How do you model physical objects?
- How do you find objects near a test point in high-dimensional space?
- How do you triangulate a model elegantly and efficiently?
- What on earth is Pixar actually doing?
- ***Take CS268!***

String and Genomic Algorithms

- How do you find all matches of a dictionary inside a long text?
- How do you reconstruct an evolutionary history from a set of genomes?
- How do you rebuild a DNA strand given millions of overlapping fragments?
- How do we know when HIV entered the human population?
- ***Take CS262!***

Numerical Algorithms

- What's the best way to solve a linear system of equations?
- How do you formulate and solve linear programs?
- How do you approximate solutions to differential equations?
- What on earth is Pixar doing?
- ***Take CS205A!***

Bitwise Algorithms

- How can we speed up classical algorithms when working on integer data?
- How can we search and sort in $o(\log n)$ and $o(n \log n)$ time?
- How can we solve problems by iteratively refining approximate solutions?

Quantum Algorithms

- How do you program a quantum computer?
- How do you design quantum algorithms to solve classical problems?
- What are the theoretical limits of quantum computation?
- What do cryptographers mean by “science fiction attacks?”
- ***Take CS259Q!***

Algorithmic Game Theory

- How far from optimal can a system get if individuals greedily maximize their own profits?
- How do you design systems that encourage people to behave honestly and fairly?
- ***Take CS364A/B!***

Streaming Algorithms

- How do we compute properties of data presented one element at a time?
- What is the minimum amount of memory necessary to answer questions about streaming data?
- How does Google know what search queries are popular?

Heuristic Search

- What general techniques are useful for optimization problems?
- What frameworks exist for modeling search problems with no known efficient solutions?
- Why can computers beat humans at chess but not at Go?
- ***Take CS221!***

Pathfinding Algorithms

- What shortest-paths algorithms work well for real transportation networks?
- How well can we approximate distances between points in constrained space?
- What exactly is Google Maps doing?

Complexity Theory

- What are the limits of efficient computation?
- How do classical, randomized, and quantum algorithms interrelate?
- What lies beyond **P** and **NP**?
- ***Take CS254!***

Data Structures

- How can we exploit properties of data to store and access them more rapidly?
- What metrics on data can be easily and readily computed?
- How do we design and analyze complicated structures?
- ***Take CS166!***

Getting into Research

- Stanford undergrad?
- Interested in algorithms research?
- Take ***CS167*** next spring!
- Course focuses on transitioning from coursework-level algorithms to research-level algorithms.

What We've Covered

- Insertion sort
- Breadth-first search
- Dijkstra's algorithm
- Depth-first search
- Topological sorting
- Kosaraju's algorithm
- Mergesort
- Maximizing single-sell profit
- Binary search
- Binary heaps
- Heapsort
- Maximizing unimodal arrays
- Karatsuba Multiplication
- Median-of-Medians
- Quickselect
- Quicksort
- Introselect
- Introsort
- Karger's algorithm
- Approximating max-cut
- Chained hash tables
- Frog jumping
- Activity scheduling
- Prim's algorithm
- Kruskal's algorithm
- Disjoint-set forests
- Weighted activity selection
- Buying cell towers
- Sequence alignment
- Levenshtein distance
- Bellman-Ford
- Floyd-Warshall
- TSP DP
- Color-Coding
- 0/1 Knapsack DP
- 0/1 Knapsack FPTAS

And that's just from lecture!

What We've Covered

- **Abstract Problem Solving**
  - Reductions, recursion, randomness, etc.
- **Formalizing Intuition**
  - Induction, exchange arguments, cut-and-paste arguments, etc.
- **Expanding Vocabulary**
  - Efficient selection, SCCs, min cuts, etc.

You now have a wide array of tools for solving big, important problems.

Best of luck wherever they take you!