Admin: Final exam Mon 12/15 1:30-4:30
in Johnson
(See TAs if you have conflict!)

Relevant Readings: CLRS Chapters 5, 7, 34, 35

(This material will not be on final.)

Outline: "On Beyond 6,006"

- Courses
- Topics
Courses

6.046 "Intermediate Algorithms" (Leiserson/Goonum)
more advanced algs. & analysis techniques
less coding
randomized algs., data structures

6.047 Computational Biology (Kellis)
more string-based algorithms for genetic data
phylogeny

6.854 Advanced Algorithms (Karger)
intense survey of field

6.850 Geometric Computing (Indyk)
points/lines/polygons/meshes...

6.851 Advanced Data Structures (Demaine)
e.g. sublogarithmic performance

6.852 Distributed Algorithms (Lynch)
reaching consensus in a network with faults

6.855 Network Optimization (Scully)
optimization in a graph - beyond shortest paths
6.856: Randomized Algorithms (Karp)  
how randomness can make algo simpler & faster

6.857: Network & Computer Security (Rivest)  
(applied cryptography)

6.885: Geometric Folding Algorithms  
how to fold one shape into another

Other theory classes:

6.045: Automata, Computability, & Complexity →

6.840: Theory of Computing (Sipser)

6.841: Advanced Complexity Theory (Sudan)  
→ merging?

6.842: Randomness & Computation (Rubinfeld)?

(Aaronson)
Topics

1. Randomness

Flipping coins could help, right? Randomness in algorithm, not same as randomness in input. Aim for good expected running time.

Example: Quicksort

given $A[1..n]$ to sort
- Pick a random $i$, $1 \leq i \leq n$
- let $x = A[i] = \text{"pivot"}$
- partition $A$ so that $x$ (in time $\Theta(n)$)

```
< x  j  > x
```

- sort $A[1..j-1]$ and $A[j+1,n]$ recursively

Sizes of subproblems are random variables

but: this works extremely well!!

Good $\Theta(n \log n)$ expected running time.

Analysis in 6.046
Example 2: Primality testing

When testing a number $p$ for primality, there is a "color test" (details omitted) for integers $a$, $1 \leq a < p$ such that

- if $p$ is prime, then all such $a$'s are green:

```
1 2 3 4 5 6 7
0 0 0 0 0 0
```

- but if $p$ is not prime, at least $\frac{1}{2}$ the $a$'s are red

```
1 2 3 4 5 6 7 8 9 10
0 0 0 0 0 0 0 0 0 0
```

Trouble is: pattern looks pretty "random", no easy way to find a red $a$ deterministically.

So: try 50 $a$'s at random

all green $\Rightarrow$ say "prime" (we hope)
any reds $\Rightarrow$ say "not prime" (for sure!)

error controllable by adjusting "50"
Approximate algorithms

Once we open the door to algorithms that are not always right, we can do other things, e.g., "just get close" in other ways.

Consider again **Vertex Cover**: Given an undirected graph $G = (V, E)$, we want to find a small set of vertices to color red so that each edge touches at least one red vertex.

Consider following algorithm:

- Pick an edge arbitrarily (or at random)
- color both endpoints red
- remove from consideration both endpoints & all edges that touch them (they're covered now)
- repeat until no edges left
Claim:

Vertex cover found is at most twice optimal size

Post:

Each edge picked must have at least one endpoint in minimum cover. (edges picked don't touch)

Finding minimum cover is hard; what is best approximate cover? (can we best factor of 2)?

[seems very hard to do so!]

Whitefield of approximation algorithms :/
Parallel Computing

1. On an infinitely parallel computer, how long to add up n numbers?

\[ 3 \rightarrow 1 \rightarrow 4 \rightarrow 1 \rightarrow 5 \rightarrow 9 \rightarrow 2 \rightarrow 6 \]
\[ 4 \rightarrow 5 \rightarrow 14 \rightarrow 8 \]
\[ 9 \rightarrow 22 \rightarrow 31 \]
\[ \Theta(\lg n) \text{ time} \]

2. How long to compute dot product of two length n vectors \( x \cdot y = \sum_{i=1}^{n} x_i \cdot y_i \)?

\[ \text{Ans: } \Theta(\lg n) \text{ time (1 to mpy, } \lg n \text{ to add)} \]
3) How long to compute product of two \( n \times n \) matrices?

\[ A \times B = C \]

\textbf{Ans: } \( \Theta (\log n) \) time

(do \( n^2 \) dot-products in parallel...)

4) How long to compute shortest paths in an \( n \)-vertex graph?

\textbf{Ans: } \( \Theta (\log^2 n) \) time

Do \( \log n \) "matrix multiplies" \( A, A^2, A^4, \ldots, A^n \)

where inner product \( x \cdot y \) means \( \min_i (x_i + y_i) \)

starting from given edge-weight matrix (0's on diagonals) \( A \).

See you at the final!