Admin: mailing list 6.046-tas@csail.mit.edu
new students?

Outline:

- Divide & Conquer: Master Theorem
- Convex Hull (2D) in time $O(n \log n)$
- Medians in time $O(n)$
Divide & Conquer

Given a problem of size $n$:

- **Divide** it into a subproblems of size $\frac{n}{b}$
- **Solve** each subproblem recursively ("conquer")
- **Combine** solutions of subproblems to get overall solution

Notes:

- Most frequently $b = 2$, so problems are "half-size"
- In principle, subproblems may have quite different sizes, but usually they are equal-sized.
- May need more expressive notion of size, e.g. 
  \# vertices & \# edges for a graph problem
- If $n$ is not divisible by $b$, some subproblems may have size $\lceil \frac{n}{b} \rceil$, and some may have size $\lfloor \frac{n}{b} \rfloor$. (See book...)
Analysis of D&C ("Master Theorem") [Ref. ch. 4]

Suppose $a \geq 1$ and $b > 1$ are fixed integers.

Let $T(n)$ denote w.c. running time on input of size $n$.

Suppose $n = b^h$ (see book for general case; same answer)

Suppose

$$T(n) = \begin{cases} 
\Theta(1) & \text{if } n = 1 \\
aT(n/b) + \Theta(n^p \log^q n) & \text{if } n > 1
\end{cases}$$

recursion work for divide & combine

for some values $p \geq 0$ and $q \geq 0$.

Then

$$T(n) = \begin{cases} 
\Theta(n^p \log^q n) & \text{if } p > \log_b a \\
\Theta(n^p \log^{q+1} n) & \text{if } p = \log_b a \\
\Theta(n \log_b a) & \text{if } p < \log_b a
\end{cases}$$
Recursion Tree:

- Site work \( g = 0 \)
  \[ n^p \]
- \( a \cdot (n/b)^p \)
- \( a^2 \cdot (n/b^2)^p \)
- \( a^h \cdot (1)^p \) or \( a^h \cdot \Theta(1) \)

\[ n = b^h \]
\[ a = b^{\log_b a} \]
\[ a^h = (b^{\log_b a})^h = (b^h)^{\log_b a} = n^{\log_b a} \]

\[ \text{input size} \]

Assume \( g = 0 \) (see book for general case, Ex. 4.6-2)

\[ \text{Work} = \sum_{i=0}^{h} a^i \cdot (n/b^i)^p = n^p \cdot \sum_{i=0}^{h} \left( \frac{a}{b^p} \right)^i \]

\[ \text{sum of geometric series} \]

- **Root**
  \[ \Theta(n^p) \] if \( p > \log_b a \) (decreasing series)

- **Even**
  \[ \Theta(n^p \log n) \] if \( p = \log_b a \) (all terms \( h = \log_b n \))

- **Leaves**
  \[ \Theta \left( \text{last term} \right) = \Theta(n^p \cdot \frac{a^h}{b^{pn}}) = \Theta(n^p \cdot \frac{n^{\log_b a}}{n^p}) = \Theta(n^{\log_b a}) \]
**Convex Hull**

- Nice example of D&C

\[ T(n) = 2T(n/2) + \Theta(n) \]

(same soln as mergesort: \( T(n) = \Theta(n \log n) \))

- Given \( n \) points in plane \( S = \{ (x_i, y_i) | i = 1, 2, \ldots, n \} \)
  (Assume no two have same x coord, and
  no two have same y coord, and
  no three in a line, for convenience.)

- Define **convex hull of** \( S \), \( \text{CH}(S) \), as smallest
  polygon containing all pts in \( S \).

- If pts are "nails", then \( \text{CH}(S) \) is shape
  of rubber band around all the nails.

- Represent \( \text{CH}(S) \) as sequence of pts on boundary,
  in order (e.g. as doubly-linked list).

\[
D \leftarrow Q \leftarrow R \leftarrow S \leftarrow T
\]
Idea:

- Sort points by x-coord (once & for all; time $O(n\log n)$)
- For any set $S$ of points:
  - Divide into "left-half" & "right-half" (by x-coords)
  - Compute CH on each half (recursively)
  - Combine CH's of two halves ("merge")

Only tricky part is merge:

Given $(a_1, a_2, \ldots, a_p)$ & $(b_1, b_2, \ldots, b_q)$ [clockwise order]

Let $a_i \not\in C b_j$ mean "$a_i$ & $b_j$ can see each other"

Upper edge $=(a_i, b_j)$ s.t. $a_i \not\in C b_j$ & $a_i \not\in C b_{j+1}$ & $a_i \not\in C b_j$

(Lower edge similar)
• Pick $a_1$ to maximize $x$ within $A$ (CH(A))
• Pick $b_1$ to minimize $x$ within $B$ (CH(B))
• $i = 1$
  $j = 1$
  while $a_i \leq C b_{j+1}$ or $a_{i-1} \leq C b_j$:
    $\begin{cases} 
    i = i+1 \ (\text{mod } p) \\
    j = j+1 \ (\text{mod } q) 
    \end{cases}$
  return $(a_i, b_j)$ as upper tangent

• Find lower tangent similarly
• Splice doubly linked lists together
  in obvious manner, removing all points passed over.

Time for merge is $\Theta(n)$, since each
iteration of while loop passes over one point.

$\therefore$ total time $T(n) = 2T(n/2) + \Theta(n)$

$= \Theta(n \lg n)$

Ref: Preparata & Hong, CACM (1977)
Median-Finding

Ref: §9.3: Blum, Floyd, Pratt, Rivest, Tarjan (1973)

Given set of n numbers, define rank(x) as number of numbers in set that are ≤ x.

\[
\begin{array}{cccccc}
  x & 2 & 3 & 9 & 13 & 29 & 41 \\
  \text{rank}(x) & 1 & 2 & 3 & 4 & 5 & 6 \\
\end{array}
\]

(lower) median = element of rank \( \lfloor n/2 \rfloor \)  (e.g. 9)
(upper) median = " " " \( \lceil n/2 \rfloor \)  (e.g. 13)

Selection problem: Given set \( S \) of n (distinct) numbers, and \( i \) (1 ≤ i ≤ n), find \( x \in S \) s.t. \( \text{rank}(x) = i \). (i.e. \( i^{th} \) smallest)

Clearly sorting works, in time \( \Theta(n \lg n) \).
Can we beat \( \Theta(n \lg n) \)?
Yes, with D&C (& unusual recurrence!)
(No Master Theorem!)
Idea: (for computing elt of rank i from S)

**Select**(x, i):
- Pick x ∈ S (cleverly, somehow!)
- Compute \( k = \text{rank}(x) \)
  \[ B = \{ y \in S \mid y < x \} \]
  \[ C = \{ y \in S \mid y > x \} \]
- if \( k = i \) : return x
- if \( k > i \) : return Select(B, i)
- if \( k < i \) : return Select(C, i-k)

Let \( T(n, i) = \text{time to compute } i^{th} \text{ smallest from } S, \text{ when } |S| = n \)

\[
T(n) = \max_{1 \leq i \leq n} T(n, i)
\]

\[
T(n) \leq [\text{time to pick } x] + \max_{1 \leq k \leq n} T(\max(k, n-k))
\]

Need to pick x s.t. \( \text{rank}(x) \) is not extreme. How?
(Can't afford to use median; want something median-like)
To pick $x$:

- Arrange $S$ into columns of size 5 ($\lceil n/5 \rceil$ columns)
- Sort each column (big elts on top) [linear time!]
- Find "median of medians" as $x$

$\approx \frac{3n}{10}$ elts

$\leq x$

$\uparrow \approx \frac{3n}{10}$ elts

Recurrence:

$$T(n) = \begin{cases} 
\Theta(1) & \text{for } n \leq 140 \\
T(\lceil n/5 \rceil) + T\left(\frac{7n}{10} + 6\right) + \Theta(n) & \text{otherwise}
\end{cases}$$

Intuition: By doing recursion on $n/5$, get to discard $\geq 3n/10$ elts

- $\frac{n}{5} + \frac{7n}{10} \leq \frac{9n}{10}$ so recursing on proper fraction.
Prove $T(n) \leq c \cdot n$ by induction, for some large enough $c$.

True for $n \leq 140$ by choosing $c$ large.

$T(n) \leq c \cdot \lceil n/5 \rceil + c \left( \frac{7n}{10} + 6 \right) + a \cdot n$ (for a large enough $c$ to cover $\Theta(n)$ term)

\[
\leq \frac{cn}{5} + c + \frac{7nc}{10} + 6c + an
\]

\[
= cn + \left( -\frac{cn}{10} + 7c + an \right)
\]

if this is $\leq 0$, we are done with proof.

\[
c \geq \frac{70c}{n} + 10a
\]

Ok for $n \geq 140$ and $c \geq 20a$.