Admin: New students - meet TA's

Note bug in HW#1 (problem 4) assignment; correct version now posted.

Outline:
- Divide & Conquer; Master Theorem
- Convex Hull (in 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. $\#\text{vertices} \& \#\text{edges}$ for a graph problem

- If $n$ is not divisible by $b$, some subproblems may have size $\lfloor \frac{n}{b} \rfloor$, and some may have size $\lceil \frac{n}{b} \rceil$. (See book...)
Analysis of D&C ("Master Theorem") [Ref. ch. 4]

Suppose $a \geq 1$ & $b > 1$ 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

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

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

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:

\[
\begin{align*}
\text{Site} & \quad \text{work}(q=0) \\
& = n^p \\
& = a \cdot (n/b)^p \\
& = a^2 \cdot (n/b^2)^p \\
& = \ldots \\
& = a^h \cdot (1)^p \text{ or } a^h \cdot \Theta(1)
\end{align*}
\]

\[
\begin{align*}
\text{input size} & \quad n = b^h \\
a & = b^{\log_b a} \\
c & = b^{\log_b a} \\
& = (b^{\log_b a})^h = (b^h)^{\log_b a} = n^{\log_b a} \\
& = \text{# leaves}
\end{align*}
\]

Assume \( q = 0 \) (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 =, \( h = \log_bn \))

**Leaves** = \( \Theta(\text{last term}) = \Theta(n^p \cdot \frac{a^h}{b^h n}) = \Theta(n^p \cdot \frac{n^{\log_b a}}{n^p}) = \Theta(n^{\log_b a}) \)
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 \( S \), \( CH(S) \), as smallest
  polygon containing all pts in \( S \).
- If pts are "nails", then \( CH(S) \) is shape
  of rubber band around all the nails.

Represent \( CH(S) \) as sequence of pts on boundary,
  in order (e.g. as doubly-linked list). (clockwise)
Idea:
- Sort points by x-coord (once & for all; time $\Theta(n \log n)$)
- For input set $S$ of points:
  - Divide into "left-half" $A$ & "right-half" $B$ by x-coords
  - Compute $CH(A)$ & $CH(B)$, recursively
  - Combine $CH$'s of two halves ("merge")

- Algorithm is due to Preparata & Hong, CACM (1977)

- If we can do merge in time $\Theta(n)$, then overall running time $T(n)$ satisfies recurrence

$$T(n) = 2T(n/2) + \Theta(n)$$

$$= \Theta(n \log n)$$ via Master Thm

- How to do merge in linear time?

\[
\begin{align*}
  CH(A) &= \langle a_1, a_2, a_3, a_4 \rangle \\
  CH(B) &= \langle b_1, b_2, b_3 \rangle \\
  CH(S) &= \langle a_2, a_3, a_4, b_3, b_1 \rangle
\end{align*}
\]
To find upper tangent:

- Assume wlog: $a_i$ maximizes $x$ within $CH(A)$
  $b_i$ minimizes $x$ within $CH(B)$

- Define $L$ as vertical line separating $A$ & $B$ (see fig)

- Define $y(i,j)$ as $y$-coordinate of pt of intersection between $L$ & segment $(a_i, b_j)$

- Claim: $(a_i, b_j)$ is upper tangent iff it maximizes $y(i,j)$

  (Proof: exercise)

- Algorithm to find upper tangent:

  $\begin{align*}
  i &= 1 & \text{(left finger on $a_i$)} \\
  j &= 1 & \text{(right finger on $b_i$)} \\
  \text{while } y(i,j+1) > y(i,j) \text{ or } y(i-1,j) > y(i,j):} \\
  &\quad \text{if } y(i,j+1) > y(i,j): \\
  &\quad\quad j = j + 1 \pmod{8} & \text{(move right finger}) \\
  &\quad\text{else:} \\
  &\quad\quad i = i - 1 \pmod{p} & \text{(move left finger}) \\
  \text{return } (a_i, b_j) \text{ as upper tangent}
  \end{align*}$

- Similarly for lower tangent

- Note: Preparata & Hong give a slightly different method.
Median-Finding

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

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

\[ x: 2, 3, 9, 13, 29, 41 \]

\[ \text{range}(x): 1, 2, 3, 4, 5, 6 \]

\( \text{(lower) median = element of range} \) \[ \left\lceil \frac{n+1}{2} \right\rceil \] \( \text{(e.g. 9)} \)

\( \text{(upper) median = } \) \[ \left\lceil \frac{n+1}{2} \right\rceil \] \( \text{(e.g. 13)} \)

Selection problem: Given set S of n (distinct) numbers, and i (1 ≤ i ≤ n), find number \( x \in S \) s.t. \( \text{range}(x) = i \). (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 \((S, i)\):
- Pick \(x \in S\) (cleverly, somehow!)
- Compute \(k = \text{rank}(x)\)
  
  \[
  B = \{y \in S \mid y < x\} \\
  C = \{y \in S \mid y > x\}
  \]

\[
\begin{array}{ccc}
B & \rightarrow & x \\
\downarrow & & \downarrow \\
\text{elts} & & \text{elts} \\
\end{array}
\]

- 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 \(i^{th}\) smallest from } S, \text{ when } |S| = n\)

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

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

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 els on top) [linear time!]
- Find "median of medians" as $x$

Recurrence:

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

Intuition: By doing recursion on $n/5$, get to discard $\geq 3n/10$ els.
Prove $T(n) \leq c \cdot n$ by induction, for some large enough $c$

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

- $T(n) \leq c \cdot \left\lceil \frac{n}{5} \right\rceil + c \left( \frac{7n}{10} + 6 \right) + an$ (for a large enough term 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.

$\implies$

$c \geq \frac{70c}{n} + 10a$

Ok for $n > 140$ & $c \geq 20a$ √