**Divide & Conquer**

- Paradigm
- Convex Hull
- Median finding

**Paradigm**

Given a problem of size \( n \)

Divide it into subproblems of size \( \frac{n}{b} \)

where \( a > 1, b > 1 \)

Solve each subproblem recursively

Combine solutions of subproblems to get overall solution

\[ T(n) = a \cdot T\left(\frac{n}{b}\right) + \text{[work for merge]} \]
Convex Hull

Given \( n \) points in plane \( \text{Ref § 33.3} \)

\[
S = \{ (x_i, y_i) | i = 1, 2, \ldots, n \}
\]

Assume no two have same x coord, no two have same y coord, and no three in a line for convenience.

Convex Hull : Smallest polygon containing all points in \( S \)

\( \text{CH}(S) \)

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

\( \text{CH}(S) \) represented by the sequence of points on the boundary in order clockwise

as doubly linked list

\( p \leftrightarrow q \leftrightarrow r \leftrightarrow s \leftrightarrow t \)
Brute force for Convex Hull

Test each line segment to see if it makes up an edge of the convex hull.

→ If the rest of the points are on one side of the segment, the segment is on the convex hull — above.

→ Else the segment is not — above.

O(n^2) edges, O(n) tests ⇒ O(n^3) complexity

Can we do better?
D&C for Convex Hull

Sort points by x coord (once & for all, O(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)
- Combine CH's of two halves (merge step)

How to Merge?

Find upper tangent (a_i, b_j) (a_4, b_2) U.T.
Find lower tangent (a_k, b_m) (a_3, b_3) L.T.
Cut & paste in time Θ(n) (a_1, a_2, a_3, a_4, a_5) (b_1, b_2, b_3)
First link a_i to b_j, go down b list till you see b_m and link b_m to a_k
Continue along the a list until you return to a_i
FINDING TANGENTS

Assume \(a_i\) maximizes \(x\) within \(CH(A) = (a_1, a_2, \ldots, a_p)\)
\(b_i\) minimizes \(x\) within \(CH(B) = (b_1, b_2, \ldots, b_q)\)

Let \(L\) be the vertical line separating \(A\) & \(B\)

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 \(y(i;j)\)

If \(y(i;j)\) is not maximum, there will be points on both sides of \((a_i, b_j)\) and it can't be a tangent.

**Algorithm:** Obvious \(O(n^2)\) algorithm looks at all \(a_i, b_j\) pairs

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

\[
\begin{align*}
\text{\(\Theta(n)\)} & \quad \text{\(i = 1\)} \\
& \quad \text{\(j = 1\)} \\
& \quad \text{\(\text{while} \ (y(i,j+1) > y(i,j) \text{ or } y(i-1,j) > y(i,j)) :\)} \\
& \quad \quad \text{\(\text{if } y(i,j+1) > y(i,j) : \)} \\
& \quad \quad \quad \quad \text{\(j = j+1 \ (\text{mod } p)\)} \\
& \quad \quad \text{\(\text{else } i = i-1 \ (\text{mod } p)\)} \\
& \quad \quad \text{\(\text{return } (a_i, b_j) \text{ as upper tangent}\)} \\
& \quad \quad \text{\(\text{move right finger \(\uparrow\)}\)} \\
& \quad \quad \text{\(\text{move left finger \(\uparrow\)}\)} \\
& \quad \quad \text{\(\text{return } (a_i, b_j) \text{ as upper tangent}\)} \\
& \quad \quad \text{\(\text{Similarly for lower tangent}\)}
\end{align*}
\]

\[ T(n) = 2T(n/2) + \Theta(n) \quad \text{Master Theorem gives } \Theta(n \log n) \]
Intuition for why Merge works

\[ a_1, b_1 \] are right most & leftmost points.
We move anti-clockwise from \( a_1 \),
clockwise from \( b_1 \).
\( a_1, \ldots, a_q \) is a convex hull, as is \( b_1, b_2, \ldots, b_q \).
If \( a_i, b_j \) is such that moving from either
\( a_i \) or \( b_j \) decreases \( y(i, j) \) there are
no points above the \((a_i, b_j)\) line.

The formal proof is quite involved and won't be covered.
Median Finding

[Ref: §9.3]

Given a set of \( n \) numbers, define \( \text{rank}(x) \) as the number of numbers in the set that are \( \leq x \).

Find the element of rank \( \left\lfloor \frac{n+1}{2} \right\rfloor \): lower median
(Or element of rank \( i \))

Find the element of rank \( \left\lceil \frac{n+1}{2} \right\rceil \): upper median

Clearly sorting works in time \( \Theta(n \log n) \).

Can we do better?

Select \((S, i)\)

- Pick \( x \in S \) (cleverly)
- Compute \( k = \text{rank}(x) \)
- \( B = \{ y \in S \mid y < x \} \)
- \( C = \{ y \in S \mid y > x \} \)

\[
\begin{array}{c}
\leftarrow B \rightarrow \ x \leftarrow C \rightarrow \\
\hline
\text{k-1 elements} \quad \text{h-k elements}
\end{array}
\]

- If \( k = i \): return \( x \)
- Else if \( k > i \): return \( \text{Select}(B, i) \)
- Else if \( k < i \): return \( \text{Select}(C, i-k) \)
Picking $x$ Cleverly

Need to pick $x$ so $\text{rank}(x)$ is not extreme.

- Arrange $S$ into columns of size $5$ ($\frac{\sqrt{n}}{5}$ cols)
- Sort each column (big elements on top) (linear time)
- Find "median of medians" as $x$

How many elements are guaranteed to be $> x$?

Half of the $\frac{n}{5}$ groups contribute at least $3$ elements $> x$ except for $1$ group with less than $5$ elements & $1$ group that contains $x$.

At least $3\left(\frac{\sqrt{n}}{10} - 2\right)$ elements are $> x$.

Recurrence: $T(n) = \begin{cases} 
O(1) & \text{for } n \leq 140 \\
T\left(\frac{n}{5}\right) + T\left(\frac{7n}{10}\right) + O(n) & \text{median of medians, discard } \frac{3n}{6} \text{ elements, } \text{sorting each column} 
\end{cases}$
Solving the Recurrence

Master theorem does not apply

Prove $T(n) \leq c \cdot n$ by induction, for some large enough $c$

- True for $n \leq 140$ by choosing large $c$
- $T(n) \leq C \cdot \lceil n/5 \rceil + C \left( \frac{7n+6}{10} \right) + a \cdot n$
  (a needs to be large enough to cover $\Theta(n)$ term)

\[
\leq \frac{Cn}{5} + C + \frac{7nC}{10} + 6C + an
\]

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

If this is $\leq 0$, we are done

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

OK for $n \geq 140$ & $C \geq 20a$
Example

\[ a_3, b_1 \text{ is upper tangent} \]
\[ a_4 > a_3 \]
\[ b_2 > b_1 \]
\[ a_1, b_3 \text{ is lower tangent} \]
\[ a_2 < a_1 \]
\[ b_4 < b_3 \]

\( a_i, b_j \) is an upper tangent. Does not mean that \( a_i \) or \( b_j \) is the highest point.