August 30, 2024 1:17 PM

# Convex Hull in 2D

Alg'm 0 : brute-force ⇒ $O(n^3)$ time

Alg'm 1 : Jarvis' march ⇒ $O(n^2)$ time [similar to selectionsort]

Alg'm 2 : Graham's scan ⇒ $O(n \log n)$ time [similar to insertionsort & sweeping]

## Alg'm 3 : Preparata-Hong ('77)

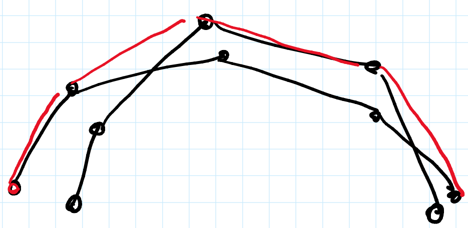
"merge-hull" [similar to mergesort]

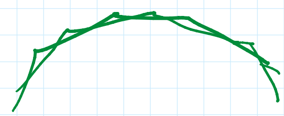
idea - divide & conquer

compute UH of $p_1, \ldots, p_{n/2}$ recursively

& UH of $p_{n/2+1}, \ldots, p_n$ recursively

merge ← $O(n)$ time not too difficult



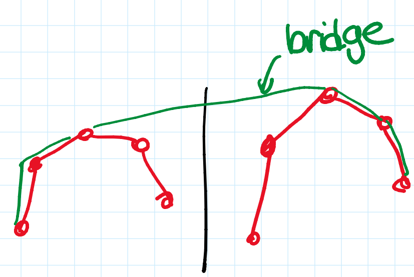


$$\Rightarrow \quad T(n) = 2T\left(\frac{n}{2}\right) + O(n)$$

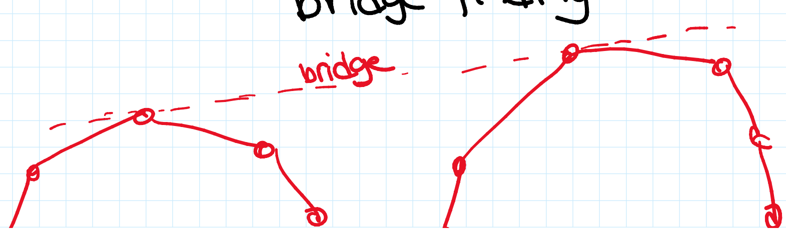
$$\Rightarrow \quad \boxed{O(n \log n)} \text{ time}$$

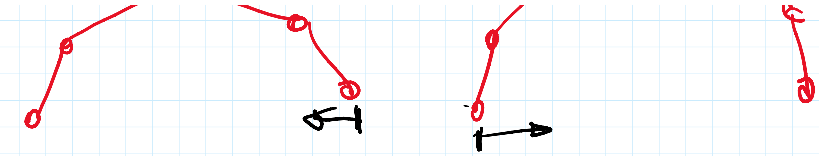
Alternative version:

pre-sort first by x




merging reduces to bridge finding



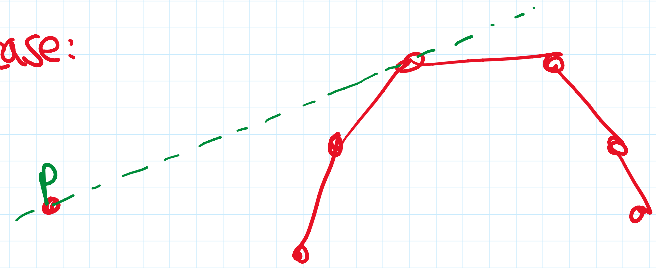


How to find bridge:

linear scan $\Rightarrow$ $\underline{O(n)}$ time

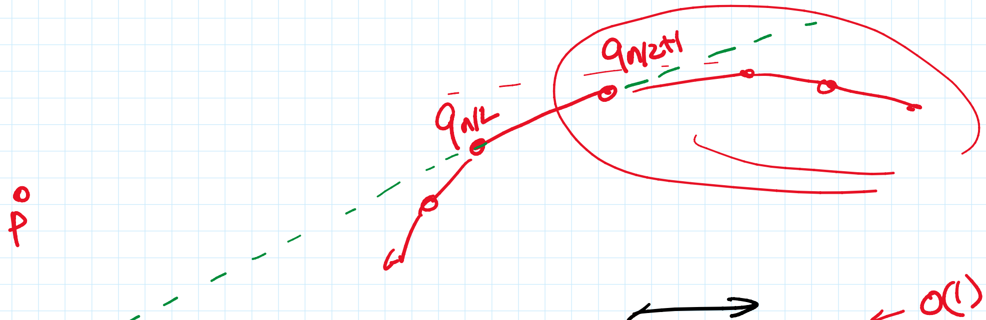
better?

Special case:




find tangent of point $p$ with UH.

idea - binary search




if $p$ above $\overleftrightarrow{q_{n/2}\, q_{n/2+1}}$ $\leftarrow O(1)$
search $q_{n/2+1} \cdots q_n$
else search $q_1 \cdots q_{n/2}$

$\Rightarrow$ $\underline{O(\log n)}$ time

general case:


search over the $p_i$'s

binary search over the $p_i$'s.

$O(\log n)$ time by special case

if $p_{n/2+1}$ above tangent line between $p_{n/2}$ and $q_1 \cdots q_n$

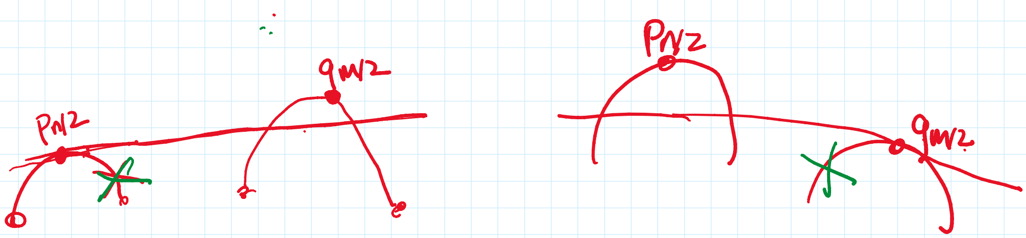
then search in $p_{n/2+1} \cdots p_n$

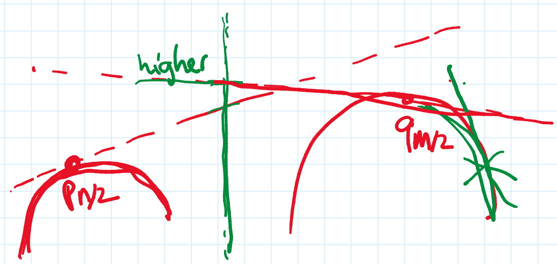
else search in $p_1 \cdots p_{n/2}$

$\Rightarrow$ total time $O(\log n \cdot \log n)$

$= \boxed{O(\log^2 n)}$.

Rmk - can be improved to $O(\log n)$




$$T(n) = 2T\left(\frac{n}{2}\right) + \underline{O(\log^2 n)}$$

$\Rightarrow$ by Master method $T(n) = \boxed{O(n)}$

excluding pre-sorting

$\Rightarrow$ total time $\boxed{O(n \log n)}$

August 30, 2024 2:06 PM

# Convex Hull in 2D

Alg'm 0: brute-force $\Rightarrow O(n^3)$ time

Alg'm 1: Jarvis' march $\Rightarrow O(n^2)$ time [similar to selectionsort]

Alg'm 2: Graham's scan $\Rightarrow O(n \log n)$ time [similar to insertionsort & sweeping]

Alg'm 3: "mergehull" (Preparata-Hong) $\Rightarrow O(n \log n)$ time

[similar to mergesort]

is it possible to do better than $O(n\log n)$?

---

NO, will prove lower bounds ...

**Lower Bd** Computing the CH (in cw order) of $n$ pts in $\mathbb{R}^2$ requires $\Omega(n\log n)$ worst-case time for any comparison-based alg'm.

Pf: idea - reduction from sorting

Suppose $\exists$ alg'm $\mathcal{A}$ for CH in better than $n \log n$ time.

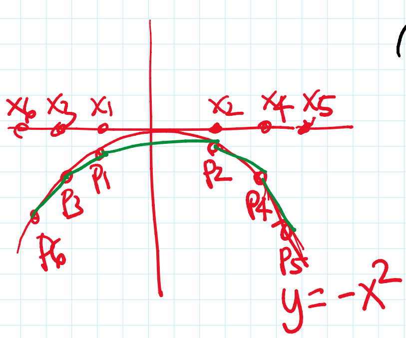
Here's a sorting alg'm:

to sort $x_1, \ldots, x_n$,

for $i = 1$ to $n$ do let $p_i = (x_i, -x_i^2)$

run $\mathcal{A}$ on $\{p_1, \ldots, p_n\}$

output $x$-coords of the output list of pts




$\Rightarrow$ can solve sorting better than $n\log n$ time

little-oh
($o(n\log n)$)

$\Rightarrow p_6, p_3, p_1, p_2, p_4, p_5$

$\Rightarrow$ contradict known sorting

$y = -x^2$ than $n \log n$ time

$\Rightarrow p_6, p_3, p_1, p_2, p_4, p_5 \quad \Rightarrow$ contradict known sorting lower bds! $\square$

What about weaker versions of UH problem?

e.g. (i) report UH in arbitrary order

(ii) count # of UH vertices

(iii) decide if # of UH vertices $= n$ (if all pts are UH vertices)

## Lower Bd (Ben-Or '83)

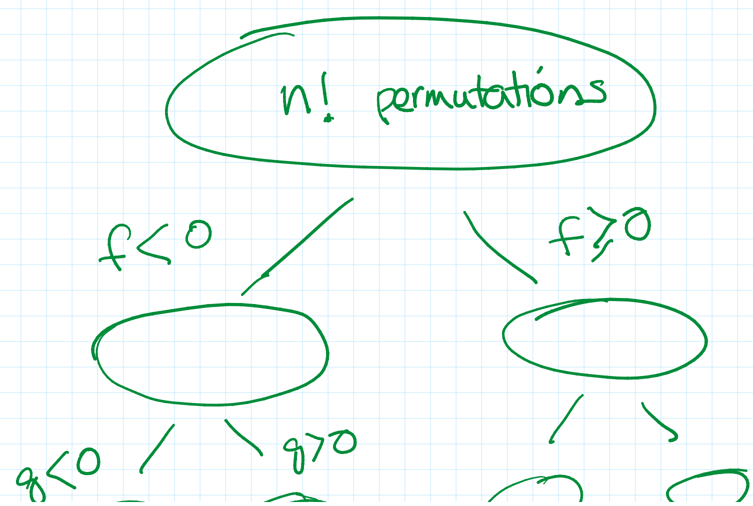
These weaker versions of problem still require $\Omega(n \log n)$ worst-case time for any alg'm that accesses input only thru algebraic primitive ops

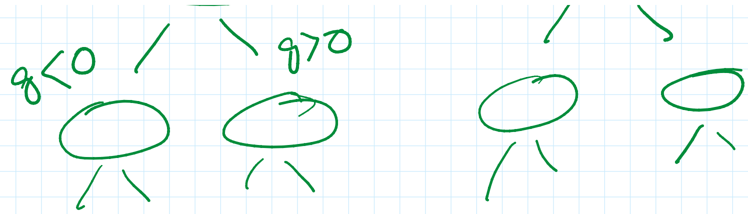
* test if $f(x_1, y_1, \ldots, x_n, y_n) < 0$ for some polynomial $f$ of const degree

e.g. $f = (x_j - x_i)(y_j - y_i) - (x_k - x_i)(y_k - y_i) < 0$?

Pf Idea -

idea - Sorting lower bd - decision tree





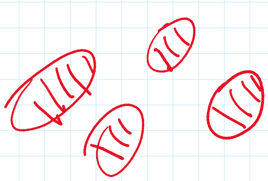

$$\# \text{ leaves} \geq n!$$
$$\# \text{ leaves} \leq 2^{\text{height}}$$
$$\Rightarrow \text{worst-case time} \geq \text{height} \geq \log(n!)$$
$$= \Omega(n \log n)$$

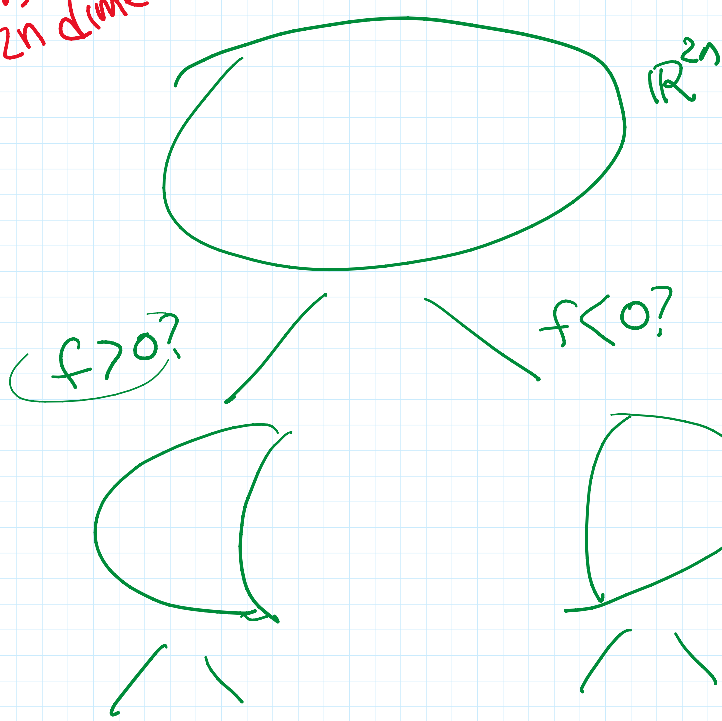
new idea - algebraic decision tree

view input as point $(x_1, y_1, \ldots, x_n, y_n) \in \mathbb{R}^{2n}$
in very high dims $2n$.

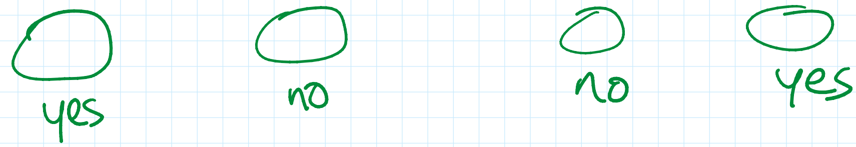
Let $S = \{ (x_1, y_1, \ldots, x_n, y_n) \in \mathbb{R}^{2n}$ : UH of $(x_1, y_1), \ldots, (x_n, y_n)$ has $n$ vertices $\}$

region in $2n$ dimensions





leaves



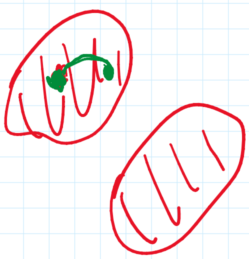

$$\#\text{ leaves} \le 2^{\text{height}}$$

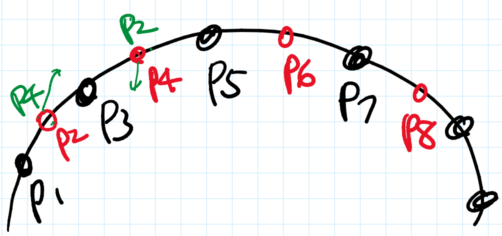
$$\text{worst case time} \ge \text{height} \ge \log(\#\text{leaves})$$

$$\ge \Omega\left(\log\left(\#\text{ connected components of } S\right)\right)$$

Ben-Or: use algebraic geometry (Milnor-Thom theorem) . .

In our case, $S$ has $\Omega((n/2)!)$ connected comps






fix $p_1, p_3, p_5, \ldots$ on $y = -x^2$

no two permutations of $p_2, p_4, p_6, \ldots$ are in same component

$$\Rightarrow \Omega\left((n/2)!\right)$$

$$\Rightarrow \Omega\left(\log (n/2)!\right) = \Omega(n \log n).$$

$\square$