The Convex Hull Prob
(Sorting Prob of CG)

Def \( A \subseteq \mathbb{R}^d \) is convex if closed under convex combinations.

Def \( \text{ConvexClosure}(A) = \text{CC}(A) = \text{smallest convex set} \supseteq A \)

2Defs of Convex Hull

Def 1 \( \text{CH}(A) = \text{JCC}(A) \)

Def 2 \( \text{CH}(A) = \text{CC}(A) \)

We will use Def 1.

A finite set

Thus in 2D \( \text{CH}(A) \) is a simple closed polygon.

(say CCW)

\[ \text{Diagram of convex polygon} \]
STITCH (L, R)
Lower bridge (L, R)
\[ a = \text{rightmost } (L) \]

\[ b = \text{leftmost } (L) \]

(Repeat 

*) While a ∈ Right(a, b) set a ← a

**) While b ∈ Right(a, b) set b ← b

Upper bridge (L, R) = ?

---

Correctness

*) generates triangles \((a, b, a)\)

**) " " \((a, b, b)\)

1) The \( \Delta \)'s are disjoint

They are ordered by their intersection with vertical line \( L \).

2) They are in \( \text{CC}(A) \).

Thus termination:

At termination \( a, \bar{a}, b, \bar{b} \) are all left of \((a, b)\)
Since \((a,a), (a,a)\), \((b,b), (b,b)\) are on \(CH(L) \& CH(R)\) respectively.

Done

---

**Timing:** Preprocess \(O(n \log n)\) to sort

STITCH in \(O(n)\)

\[ T(n) = 2T\left(\frac{n}{2}\right) + cn \]

\[ T(n) = O(n \log n) \]
Lower bounds

Sorting reducible to CH

Input: \( x_1, \ldots, x_n \)

\( CH((x_1, x_1^2), \ldots, (x_n, x_n^2)) \)

The CH will be \( x_i \)'s in sorted order.

An important use for CH

\( p_1, \ldots, p_n \in \mathbb{R}^2 \)

\( \bar{p}_i = (p_x, p_y, p_x^2 + p_y^2) \)

\( CH(\bar{p}_1, \ldots, \bar{p}_n) = \text{Triangulated surface} \)

The Delaunay Triangulation
Quick Sort & Backward Analysis

Consider

\[ QS(M) \] (distinct keys)

1) pick random \( a \in M \)
2) split \( M \): \( S < a < L \) \( (|M| - 1) \) comparisons
3) return \( QS(S) \times a \times QS(L) \)

Goal: Expect \# comparisons

Consider dart game:

Init: empty board

\[ \begin{array}{c}
1 \\
\hline
1
\end{array} \]

While non empty square

pick random empty sq

Cost = \# empty sqs to left & right of dart.

Claim: Expect cost of dart game = Expect cost \( QS \).
Backwards game:
Init: full board

While \( \exists \) dart remove random dart.
Cost: \# empty \( D \)s left & right.

Claim: \[ \text{Expect cost DG} = \text{Expect cost BW DG} \]

Analysis backwards game
Assume \( i \) darts on board
\[ T_i = \text{Expected cost to remove random dart} \]

Total Cost = \[ \sum_{i=1}^{n} \text{cost } d_i \]

\[ E(DG) = \sum T_i \leq 2nH_n \]
\[ = 0(n \log n) \]
Random Incremental CH

Procedure Random Incremental CH (P)

0) Make Δ = (p, p2, p3) pick c ∈ interior Δ
1) Construct ray from c to each p_i
2) Partition p_i by edge of Δ they cross.
3) Randomly permute p_1, ..., p_n.

For i = 1 to n

let e be edge crossed by ray c → p_i
BuildTent(p, e)

Procedure BuildTent(p, e)

1) Find edges of CH "visible" to p by searching out from e
2) Replace visible edges with 2 new edges
3) Assign rays to the new edges.
Correctness?

Timing

\( O(n) \) work other than BuildTest.

Consider steps 1 & 2 in BuildTest

1) At most 2N edges
2) Charging rule for line-side tests
   a) Not visible tests: we charge \( p_i \)
   b) Each visible test: we charge to the edge

Total 2N+2N on \( 4N \) tests.

Consider step 3 in BuildTest:

Ray-coats

Backwards analysis

2-3 points to pick from say \( p_i \)

\[ \text{Cost}(p_i) = \begin{cases} 0 & \text{if } p_i \text{ not on hull} \\ \# \text{ray crossing to left & right} & \text{else} \end{cases} \]
\[ C_i = \text{cost} \]

\[ E(C_i) \leq \frac{2(n-1)}{i-3} \]

\[ C = \text{total cost} \]

\[ E(C) = \sum_{i=4}^{n} E(C_i) \leq \sum_{i=4}^{n} \frac{2(n-1)}{i-3} \leq 2n \sum_{i=4}^{n} \frac{1}{i} \]