CS 516 -4
Computational Geometry & Graph Drawing
(Spring 2013)

Last class…

- Brief review of Asst0
- Convex hulls, halfspace intersections and duality
- Two equivalences concerning convex hulls:
    - Structural equivalence (via duality) with (origin-containing) half-space intersection
    - Algorithmic equivalence (via reducibility argument) with sorting

Reading

MountNotes Chapters 7,8
Chapters 9,10, 6

Today

- more convex hull algorithms
  - sensitivity to output size h
- “matching lower bounds”

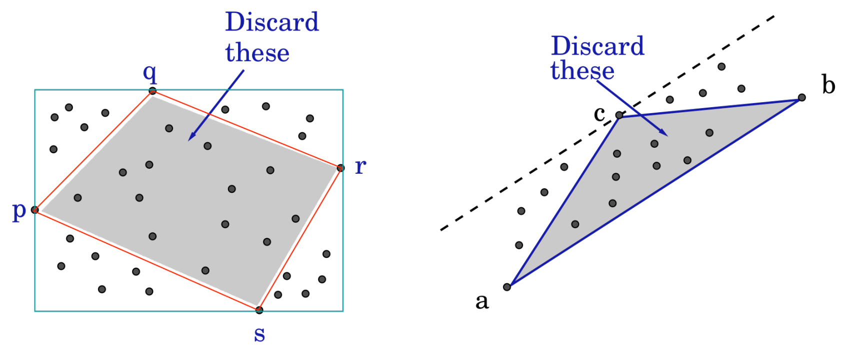
Ideas…

- More careful analysis of existing algorithms

Ideas…

- More careful analysis of existing algorithms
- Try to discard non-extreme points quickly

Quick Hull Algorithm



Initialization

Recursive Elimination

1. **Form initial quadrilateral Q, with left, right, top, bottom. Discard points inside Q.**

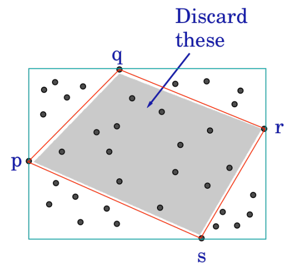
2. **Recursively, a convex polygon, with some points "outside" each edge.**

3. **For an edge ab, find the farthest outside point c. Discard points inside triangle abc.**

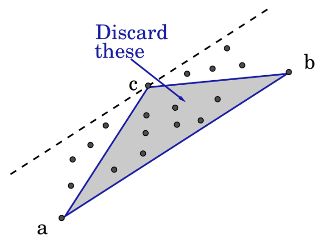
4. **Split remaining points into "outside" points for ac and bc.**

5. **Edge ab on CH when no point outside.**

Complexity of QuickHull



Initialization



Recursive Elimination

1. **Initial quadrilateral phase takes $O(n)$ time.**
2. **$T(n)$: time to solve the problem for an edge with n points outside.**
3. **Let n_1, n_2 be sizes of subproblems. Then,**

$$T(n) = \left\{ \begin{array}{ll} 1 & \textbf{if } n = 1 \\ n + T(n_1) + T(n_2) & \textbf{where } n_1 + n_2 \leq n \end{array} \right\}$$

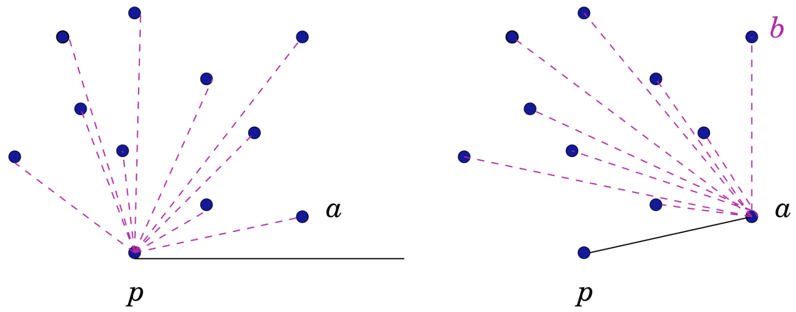
4. **Like QuickSort, this has expected running time $O(n \log n)$, but worst-case time $O(n^2)$.**

Ideas…

- More careful analysis of existing algorithms
- Try to discard non-extreme points quickly
- “wrap” around the extreme points

Efficient CH Algorithms

Gift Wrapping: **[Jarvis '73; Chand-Kapur '70]**

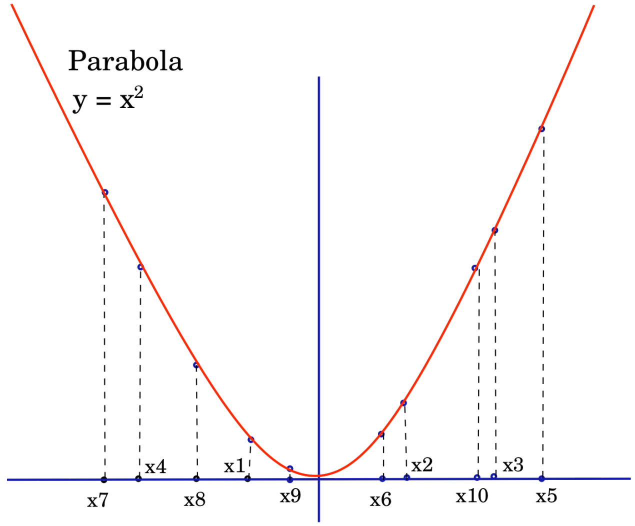


1. **Start with bottom point p.**
2. **Angularly sort all points around p.**
3. **Point a with smallest angle is on CH.**
4. **Repeat algorithm at a.**
5. **Complexity $O(Nh)$; $3 \leq h = |CH| \leq N$.**
   Worst case $O(N^2)$.

What is the complexity of finding 2-d convex hulls, in terms of *n* and *h*?

- Lower bound of $\Omega(n \lg n)$
- Jarvis' algorithm is $O(nh)$, beats the lower bound when *h* is small

Lower Bounds



- **Reduce sorting to convex hull.**
- **List of numbers to sort $\{x_1, x_2, \ldots, x_n\}$.**
- **Create point $p_i = (x_i, x_i^2)$, for each i.**
- **Convex hull of $\{p_1, p_2, \ldots, p_n\}$ has points in sorted x-order. $\Rightarrow$ CH of n points must take $\Omega(n \log n)$ in worst-case time.**
- **More refined lower bound is $\Omega(n \log h)$. LB holds even for identifying the CH vertices.**

Output-Sensitive CH

1. **Kirkpatrick-Seidel (1986) describe an $O(n \log h)$ worst-case algorithm. Always optimal—linear when $h = O(1)$ and $O(n \log n)$ when $h = \Omega(n)$.**

2. **T. Chan (1996) achieved the same result with a much simpler algorithm.**

3. **Remarkably, Chan's algorithm combines two slower algorithms (Jarvis and Graham) to get the faster algorithm.**

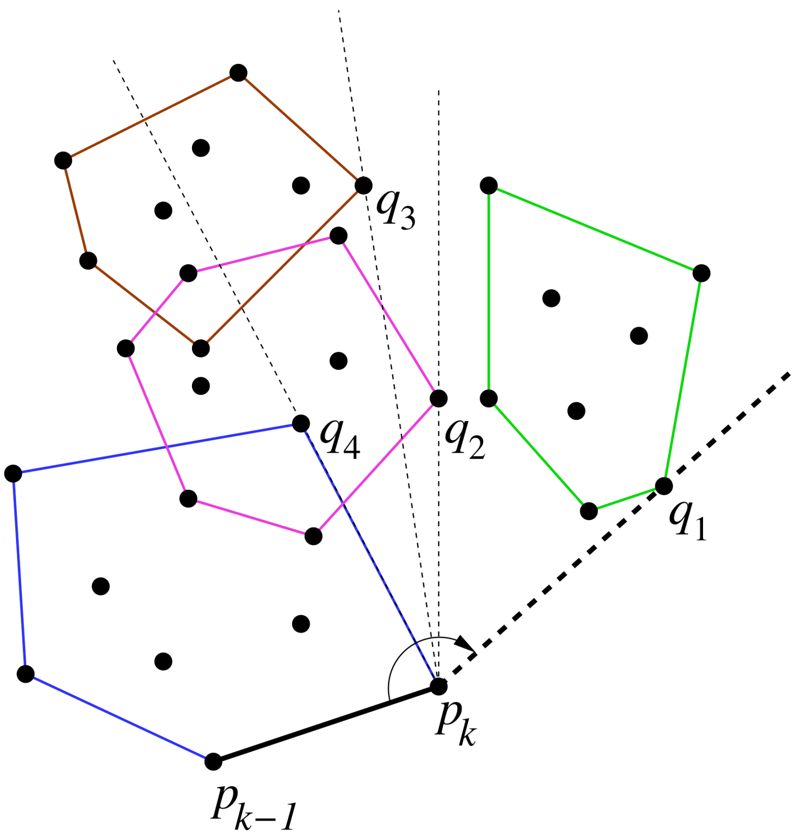
4. **Key idea of Chan is as follows.**

   (a) **Partition the n points into groups of size m; number of groups is $r = \lceil n/m \rceil$.**
   (b) **Compute hull of each group with Graham's scan.**
   (c) **Next, run Jarvis on the groups.**

Chan's Algorithm

1. **The algorithm requires knowledge of CH size h.**
2. **Use m as proxy for h. For the moment, assume we know $m = h$.**
3. **Partition P into r groups of m each.**
4. **Compute $\text{Hull}(P_i)$ using Graham scan, $i = 1, 2, \ldots, r$.**
5. **$p_0 = (-\infty, 0)$; p_1 bottom point of P.**
6. **For $k = 1$ to m do**
   - **Find $q_i \in P_i$ that maximizes the angle $\angle p_{k-1} p_k q_i$.**
   - **Let p_{k+1} be the point among q_i that maximizes the angle $\angle p_{k-1} p_k q$.**
   - **If $p_{k+1} = p_1$ then return $\langle p_1, \ldots, p_k \rangle$.**
7. **Return "m was too small, try again."**

Illustration

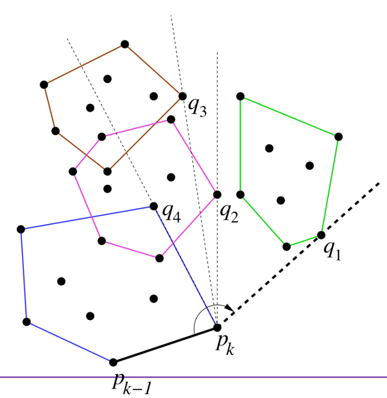


Time Complexity

- **Graham Scan:** $O(rm \log m) = O(n \log m)$.
- **Finding tangent from a point to a convex hull in** $O(\log n)$ **time.**
- **Cost of Jarvis on** r **convex hulls: Each step takes** $O(r \log m)$ **time; total** $O(hr \log m) = ((hn/m) \log m)$ **time.**
- **Thus, total complexity**

$$O\left(\left(n + \frac{hn}{m}\right)\log m\right)$$

- **If** $m = h$**, this gives** $O(n \log h)$ **bound.**
- **Problem: We don't know** h**.**



Finishing Chan

Hull(P)

- **for** $t = 1, 2, \ldots$ **do**

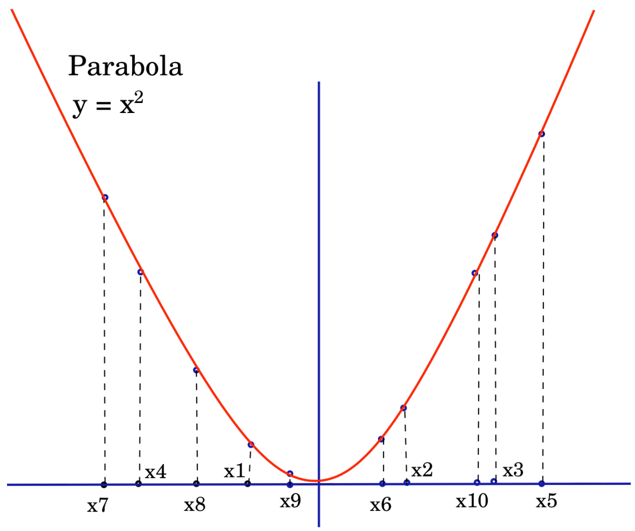
  1. **Let** $m = \min(2^{2^t}, n)$.
  2. **Run Chan with** m**, output to** L.
  3. **If** $L \neq$ **"try again" then return** L.

1. **Iteration** t **takes time** $O(n \log 2^{2^t}) = O(n 2^t)$.
2. **Max value of** $t = \log \log h$**, since we succeed as soon as** $2^{2^t} > h$.
3. **Running time (ignoring constant factors)**

$$\sum_{t=1}^{\lg \lg h} n 2^t = n \sum_{t=1}^{\lg \lg h} 2^t \leq n 2^{1 + \lg \lg h} = 2n \lg h$$

4. **2D convex hull computed in** $O(n \log h)$ **time.**

Lower Bounds



- **Reduce sorting to convex hull.**
- **List of numbers to sort $\{x_1, x_2, \ldots, x_n\}$.**
- **Create point $p_i = (x_i, x_i^2)$, for each i.**
- **Convex hull of $\{p_1, p_2, \ldots, p_n\}$ has points in sorted x-order. $\Rightarrow$ CH of n points must take $\Omega(n \log n)$ in worst-case time.**
- **More refined lower bound is $\Omega(n \log h)$. LB holds even for identifying the CH vertices.**

Lower bounds revisited

- recall limitations of the reduction from sorting argument
  - need a stronger model than pairwise comparisons

Lower bounds revisited

- recall limitations of the reduction from sorting argument
  - need a stronger model than pairwise comparisons
    - algebraic decision trees

Lower bounds revisited

- recall limitations of the reduction from sorting argument
  - need a stronger model than pairwise comparisons
    - algebraic decision trees
  - applies to strong version of CH problem (requires ordered output)
  - does not explain output-size sensitivity (dependence on h)

Lower bounds revisited

- recall limitations of the reduction from sorting argument
  - need a stronger model than pairwise comparisons
    - algebraic decision trees
  - applies to strong version of CH problem (requires ordered output)
  - does not explain output-size sensitivity (dependence on h)
    - formulate decision problems as point-classification problems