Chan’s Convex Hull Algorithm

Michael T. Goodrich
Review

• We learned about a binary search method for finding the common upper tangent for two convex hulls separated by a line in $O(\log n)$ time.

• This same method also works to find the upper tangent between a point and a convex polygon in $O(\log n)$ time.
More Review

• The upper-hull plane-sweep algorithm runs in $O(n \log n)$ time.
  – This algorithm is sometimes called "Graham Scan"

• The Gift Wrapping algorithm runs in $O(nh)$ time, where $h$ is the size of the hull.
  – This algorithm is sometimes called "Jarvis March"

• Which of these is best depends on $h$

• It would be nice to have one optimal algorithm for all values of $h$…
Optimal Output-Sensitive Convex Hull Algorithms in Two and Three Dimensions

T. M. Chan

Department of Computer Science, University of British Columbia, Vancouver, British Columbia, Canada V6T 1Z4

Abstract. We present simple output-sensitive algorithms that construct the convex hull of a set of $n$ points in two or three dimensions in worst-case optimal $O(n \log h)$ time and $O(n)$ space, where $h$ denotes the number of vertices of the convex hull.
Main Idea

• Assume, for now, we have an estimate, m, that is $O(h)$.

• Divide our set into $n/m$ groups of size $O(m)$ each.

• Find the convex hull of each group in $O(m \log m)$ time using Graham scan.

• Next, do a Jarvis march around all these “mini hulls.”
Jarvis March Steps

- Start with a point, $p_k$, on the convex hull
- Find the tangent for every mini hull with $p_k$
- Takes $O((n/m)\log m)$ time
- Pick the furthest one
- Repeat
Analysis

• Doing all the Graham scans to build the mini hulls takes $O((n/m)m \log m) = O(n \log m)$ time.

• Doing each Jarvis march step takes $O((n/m) \log m)$ time. There are $h \leq m$ such steps to find the convex hull. So all these steps take $O(n \log m)$ time.

• If $m$ is $O(h)$, the running time is $O(n \log h)$.

• But we don’t know $h$…
Pseudo Code

**Algorithm** Hull2D\((P, m, H)\), where \(P \subseteq E^2\), \(3 \leq m \leq n\), and \(H \geq 1\)

1. partition \(P\) into subsets \(P_1, \ldots, P_{\lceil n/m \rceil}\) each of size at most \(m\)
2. for \(i = 1, \ldots, \lceil n/m \rceil\) do
3. compute \(\text{conv}(P_i)\) by Graham’s scan and store its vertices in an array in ccw order
4. \(p_0 \leftarrow (0, -\infty)\)
5. \(p_1 \leftarrow \text{the rightmost point of } P\)
6. for \(k = 1, \ldots, H\) do
7.   for \(i = 1, \ldots, \lceil n/m \rceil\) do
8.     compute the point \(q_i \in P_i\) that maximizes \(\angle p_{k-1} p_k q_i\) (\(q_i \neq p_k\)) by performing a binary search on the vertices of \(\text{conv}(P_i)\)
9. \(p_{k+1} \leftarrow \text{the point } q \text{ from } \{q_1, \ldots, q_{\lceil n/m \rceil}\} \text{ that maximizes } \angle p_{k-1} p_k q\)
10. if \(p_{k+1} = p_1\) then return the list \(\langle p_1, \ldots, p_k \rangle\)
11. return *incomplete*
Guessing an estimate for $h$

- Start with $m = 4$.
- Run Chan’s algorithm. If it doesn’t return *incomplete*, we’re done.
- Otherwise, try again with $m = m^2$.
- Keep repeating this until we get a complete hull.

Could be $\bigO(n \log m) = \bigO(n \log h)$
The Complete Running Time

• The complete running time (adding up the terms in reverse order):

\[ O(n \log h + n \log h^{1/2} + n \log h^{1/4} + \ldots) = O(n \log h + (1/2)n \log h + (1/4)n \log h + \ldots) = O(n \log h). \]