CS6100: Topics in Design and Analysis of Algorithms

Convex Hull

John Augustine
**Convex Hull**

**Definition:** The convex hull of a set $P$ of points (or other objects!) is the smallest convex set containing $P$.

- Fundamental! Similar to sorting.
  - Order out of chaos.
  - Brings out algorithmic issues.
  - Randomization helps.
  - Requires geometric insights.

input = set of points: 
$p_1, p_2, p_3, p_4, p_5, p_6, p_7, p_8, p_9$

output = representation of the convex hull: 
$p_4, p_5, p_8, p_2, p_9$
Algorithm 1: Behold the Geometry

INPUT: set P of 2D points
0. Initialize set CH(P) to null set
1. For every ordered pair of points (p,q) from P
2. If all OTHER points lie to the right of directed line through p and q
3. include directed edge (p,q) to CH(P)
4. Sort the convex hull edges in clockwise order.

Is the algorithm correct for ANY input set of points?
Degeneracies

• “Pesky” input instances that can be handled by “simple” tricks.

• Typical examples:
  – 3 or more collinear points
  – 4 or more co-circular points
  – 2 points sharing the same $x$ or $y$ coordinate
  – etc.

• Important when implementing an algorithm.

• We ignore them in the lectures\(^1\). Why?

• However, we must state our assumptions precisely.

• And, our assumptions should not hide fundamental algorithmic issues.

\(^1\)Not in your project if it requires implementation! Also, dealing with degeneracies is fair game in assignments and exams.
Algorithm 1

Three collinear points is a degeneracy.

Requires $O(n^3)$ time.

Can we do better?
Algorithm 2: Gift Wrapping

Example of incremental algorithms.

**INPUT** Set $P$ of points in 2D.
Let $p_1$ be the left most point in $P$.
Let $p_2$ be the point such that all other points lie to the right of $\overline{p_1p_2}$.

\[ i \leftarrow 3 \]

\[ \text{CH}(P) \leftarrow \{(p_1, p_2)\} \]

**repeat**

\[ MAX = 0 \]

**for all** $p \in P$ **do**

\{Let $\angle(abc)$ denote the smaller angle formed by points $a$, $b$, and $c$.\}

\[ \text{if} \quad \angle(p_{i-2}p_{i-1}p) > MAX \quad \text{then} \]

\[ MAX \leftarrow \angle(p_{i-2}p_{i-1}p) \]

\[ p_i \leftarrow p \]

**end if**

**end for**

Apppend $(p_{i-1}, p_i)$ to $\text{CH}(P)$.

**until** $p_i \neq p_1$
Lower Bound

What is the lower bound on the running time of any convex hull algorithm?

Suppose you have a $o(\log n)$-time CH algorithm $\mathcal{A}$.

And, suppose you want to sort $\{a_1, a_2, \ldots, a_n\}$.

Construct $P = \{(a_1, a_1^2), (a_2, a_2^2), \ldots, (a_n, a_n^2)\}$.

Use $\mathcal{A}$ to construct the CH of $P$.

Violated lower bound on (comparison) sorting.

**Theorem 1.** $\Omega(n \log n)$ is a lower bound on the running time of any CH algorithm.

Can we close the gap between $\Omega(n \log n)$ and $O(n^2)$?
Algorithm 3

The connection to sorting leads us to ask:

“Will pre-sorting the points help”? 

Let's consider only the upper hull. See figure.

Suppose elements of $P$ are sorted in increasing order of $x$ value. And, suppose we have partially constructed the convex hull up to some $p_i$.

Notice that a point $p_j, j < i$, is no longer relevant if it is not in the partially constructed convex hull. What if $p_j$ is on the partially constructed convex hull?
Algorithm 3

Recall: we are only constructing upper hull.

Overview:

1. Sort the points in $L \rightarrow R$ order.

2. $CH(P) = \emptyset$

3. For each point $p$ taken in order
   (a) Add $p$ to $CH(P)$
   (b) Correct $CH(P)$ if any concavity (left turn) arises.
Algorithm CONVEXHULL(P)
Input. A set P of points in the plane.
Output. A list containing the vertices of CH(P) in clockwise order.
1. Sort the points by x-coordinate, resulting in a sequence p_1, \ldots, p_n.
2. Put the points p_1 and p_2 in a list L_{upper}, with p_1 as the first point.
3. for i ← 3 to n
   4. do Append p_i to L_{upper}.
   5. while L_{upper} contains more than two points and the last three points in L_{upper} do not make a right turn
   6. do Delete the middle of the last three points from L_{upper}.
7. Put the points p_n and p_{n−1} in a list L_{lower}, with p_n as the first point.
8. for i ← n−2 downto 1
   9. do Append p_i to L_{lower}.
10. while L_{lower} contains more than 2 points and the last three points in L_{lower} do not make a right turn
11. do Delete the middle of the last three points from L_{lower}.
12. Remove the first and the last point from L_{lower} to avoid duplication of the points where the upper and lower hull meet.
13. Append L_{lower} to L_{upper}, and call the resulting list L.
14. return L
Correctness and Time Complexity

Theorem 2. Algorithm 3 constructs the convex hull of a set of $n$ points in $O(n \log n)$ time.

Proof: We first prove correctness by induction on $n$, then time complexity.

Only consider upper hull. Lower hull by symmetry.
Base case is easy. The leftmost point is a upper hull of itself.
When adding a new point, note that old hull is “below” newly formed hull.
Sorting takes $O(n \log n)$ time.
Each point added and deleted at most once. Hence addition/deletion of points do not contribute.

QED.
Randomized Incremental Convex Hull Algorithm

Initialization:

Let $P = (p_1, p_2, p_3, \ldots, p_n)$ be a suitably shuffled (i.e., permuted uniformly at random) sequence of points.

Let $p$ be a point inside $\Delta p_1 p_2 p_3$.

Assign $CH(P) = \{(p_1, p_2), (p_2, p_3), (p_3, p_1)\}$ stored as a doubly linked list. Each edge in $CH(P)$ will maintain a conflict list (initialized to $\emptyset$).

For each $p_i \in P$,

If $p_i$ inside $\Delta p_1 p_2 p_3$, discard $p_i$.

Else if $p_i$ outside $\Delta p_1 p_2 p_3$,

1. Add $\text{ptr}(p_i)$ in the conflict list of the edge $\ell$ in $CH(P)$ that intersects $\overline{pp_i}$.
2. Add $\text{ptr}(\ell)$ to $p_i$. 
Randomized Incremental
Convex Hull Algorithm

/* As in randomized incremental sorting, we will
incrementally construct the convex hull $CH(P)$ while
maintaining bidirectional pointers between unprocessed
points and edges of $CH(P)$ */

Incremental Construction of $CH(P)$:

For $i = 4$ to $n$

Find the edge $\ell = (p_a, p_b)$ intersecting with $pp_i$.

If $p$ lies inside $CH(P)$, discard it and continue to
next $i$. Delete $\text{ptr}(p)$ from conflict list of $\ell$.

Else,

− Discard edge $\ell$ from $CH(P)$ and add $(p_a, p_i)$
  and $(p_i, p_b)$.

− If a concavity is induced in $CH(P)$ at $p_a$
  (symmetrically, at $p_b$), correct it (as we did in
  the deterministic algorithm).

− Correct pointers.
Backward Analysis

Each edge is created once and deleted once. In each step, 2 edges are created. So time for creations/deletions is at most $O(n)$.

We need to account for pointer reassignments.

Let $P_i$ be the first $i$ points. Each point in $P \setminus P_i$ is assigned to an edge in $CH(P_i)$, i.e., con. hull of $P_i$.

When $i$th iteration is run backward, some points in $P \setminus P_i$ are reassigned. Let $X_i$ be the number of reassignments.

**Exercise.** Can we use direct analysis to bound $X_i$?

Consider any $p \in P \setminus P_i$. Suppose further that $p$ is in the conflict list of edge $e$ in the $CH(P_i)$.

What is the probability denoted $Pr(p)$ that $p$ is reassigned?

**Answer:** Exactly the probability that $e$ is deleted!
At iteration $i$, one point $p' \in P_i$ (not necessarily in $CH(P_i)$) is deleted.

If $p'$ is part of the convex hull of $P_i$, then, the two edges incident on $p'$ are deleted. Therefore, probability that an edge $e$ is deleted is $\frac{2}{i} \in O(1/i)$.

Note that $X_i = \sum_{p \in P \setminus P_i} Pr(p) \in O\left(\frac{n-i}{i}\right)$.

Summing over all iterations using linearity of expectation, we will get the following theorem.

**Theorem 3.** The randomized incremental algorithm computes the convex hull of a set of $n$ points in expected $O(n \log n)$ time.

**Exercise.** What is the worst case running time of the randomized incremental algorithm?