1 Geometry

1.1 Convex Hull

Build upper hull:

- Sort points by x coord
- Sweep line from left to right
- maintain upper hull "so far"
- as encounter next point, check if hull turns right or left to it
- if right, fine
- if left, hull is concave. Fix by deleting some previous points on hull.
- just work backwards till no left turn.
- Each point deleted only once, so $O(n)$
- but $O(n \log n)$ since must sort by x coord.

1.2 Halfspace intersection

Duality.

- $(a, b) \to ax + by + 1 = 0$.
- line through two points becomes point at intersection of 2 lines
- point at distance d antipodal line at distance $1/d$.
- intersection of halfspace become convex hull.

So, $O(n \log n)$ time.

2 Voronoi Diagram

Goal: find nearest athena terminal to query point.
Definitions:

- point set p
- $V(p_i)$ is space closer to p_i than anything else
- for two points, $V(P)$ is bisecting line
- For 3 points, creates a new "voronoi" point
- And for many points, $V(p_i)$ is intersection of halfplanes, so a convex polyhedron

- And nonempty of course.
- but might be infinite
- Given VD, can find nearest neighbor view planar point location:
- $O(\log n)$ using persistent trees

Space complexity:

- VD is a **planar graph**: no two voronoi edges cross (if count voronoi points)
- add one point at infinity to make it a proper graph with ends
- Euler's formula: $n_v - n_e + n_f = 2$
- (n_v is voronoi points, not original ones)
- But $n_f = n$
- Also, every voronoi point has degree at least 3 while every edge has two endpoints.
- Thus, $2n_e \geq 3(n_v + 1)$
- rewrite $2(n + n_v - 2) \geq 3(n_v + 1)$
- So $n - 2 \geq (n_v + 3)/2$, ie $n_v \leq 2n - 7$
- Gives $n_e \leq 3n - 6$

Summary: $V(P)$ has linear space and $O(\log n)$ query time.

2.1 Construction

VD is dual of projection of lower CH of lifting of points to parabola in 3D.
And 3D CH can be done in $O(n \log n)$
Can build each vornoi cell in $O(n \log n)$, so $O(n^2 \log n)$.

2.2 Plane Sweep

Basic idea:

- Build portion of Vor behind sweep line.
- problem: not fully determined! may be about to hit a new site.
- What is determined? Stuff closer to a point than to line
- boundary is a parabola
- boundary of know space is pieces of parabolas: "beach line"

- as sweep line descends, parabolas descend too.
- We need to maintain beach line as "events" change it

Descent of one parabola:

- sweep line (horizontal) y coord is t
- Equation $(x - x_f)^2 + (y - y_f)^2 = (y - t)^2$.
- Fix x, find dy/dt
- $2(y - y_f)dy/dt = 2(y - t)(dy/dt - 1)$
- So $dy/dt = -(y - t)/(y - y_f)$
- Thus, the higher y_f (farther from sweep line) the slower parabola descends.

Site event:

- Sweep line hits site
- creates new degenerate parabola (vertical line)
- widens to normal parabola
- adds arc piece to peach line.

Claim: no other create events.

- case 1: suppose one parabola passes through other
  - At crossover, two parabolas are tangent.
  - then "inner" parabola has higher focus then outer
  - so descends slower
  - so outer one stays ahead, no crossover.
- case 2: new parabola descends through intersection point of two previous parabolas.
  - At crossover, all 3 parabolas intersect
  - thus, all 3 foci and sweep line on boundary of circle with intersection at center.
  - called **circle event**
  - "appearing" parabola has highest focus
  - so it is slower: won't cross over
  - In fact, this is how parabola's **disappear** from beach line
  - outer parabolas catch up with, cross inner parabola.

Summary:

- only **site events** add to beach line
- only **circle events** remove from beach line.
- n site events
- so only n circle events
- as insert/remove events, only need to check for events in newly adjacent parabolas
- so $O(n \log n)$ time