Voronoi Diagram

Plane Sweep

- Implementation

04-M-14

Junhui DENG

deng@tsinghua.edu.cn

Beach Line

- The BL can be represented as a **dictionary**
- An important fact for the construction is that

  we do not need to **explicitly** store the parabolic arcs along a BL
- Actually for each parabolic arc along the current BL,

  we store the **site** that gives rise to this arc
- Note that a site may appear multiple (up to $O(n)$) times on the BL
- But as we mentioned earlier,

  the total length of the BL will never exceed $2n - 1 = O(n)$

Breakpoints

- There is a breakpoint between each consecutive pair of sites p_i and p_j
- Although breakpoints move as a function of the sweepline, observe that we can compute the **exact** location of the breakpoint as a **linear** function of
  - p_i, p_j, and
  - the position (y-coordinate) of the current sweepline
- Thus as with the BL, we can store all breakpoints **implicitly**
- Rather, they are computed only when **needed** //a constant time for each

Operations

- The important operations that we need to support on the BL are:
  - given a fixed location of the sweepline (at a given moment),
    determine the arc of the BL that intersects a given vertical ray
  - compute **predecessors** and **successors** on the BL
  - **insert** an new arc p_i with a given arc p_j, thus
    splitting the arc for p_j into two and
    creating 3 arcs for p_j, p_i and p_j
  - **delete** an arc from the BL
- Using dictionary structures again,
  each of the above operations can be done in **$O(\log n)$** time

Event Queue

❖ The event queue is a PQ with the ability both to insert and delete events

- The event with the largest y-coordinate can be extracted
- For each site we store its y-coordinate in the queue
- For each consecutive tripe (p_i, p_j, p_k) on the BL,
  compute their circumcircle
- If the lower endpoint of the circle lies below the sweepline,
  then a circle event, whose y-coordinate
  is the y-coordinate of the bottom point of the circumcircle,
  will be created and inserted into the queue, and
- When an arc vanishes, all circle events involving it will be deleted

Event Queue

- To do this, we assign
  - for each circle event,
    a cross link back to the triple of sites that generated it, and
  - for each consecutive triple of sites
    a cross link to the event that it generated in the PQ
- Observe that
  - each event involves $O(1)$ processing time
    plus a constant number of accesses to the various data structures
  - each of these accesses takes $O(\log n)$ time, and
    the data structure are always of size $O(n)$

Complexity

- Now we can conclude that ...
- Fortune's algorithm constructs the VD for n sites in the plane

  in $O(n \log n)$ time

  using $O(n)$ space