Algorithmics

Sebastian Iwanowski
FH Wedel

6. Fundamentals of Algorithmic Geometry
6.2 Sweep Techniques

Algorithmics 6

6.2 Sweep Techniques

Transformation static d-dimensional → dynamic (d-1)-dimensional

d = 1: Line Sweep

1) Maximum search among n numbers **O(n)**

2) Closest Pair: Among n numbers, search the two that are closest together. **O(n log n)**

Preprocessing: Sort all numbers O(n log n)

Sweep: Scan from left to right and keep the closest pair respectively O(n)

References:

Klein, Kap. 2.2 (in German)

Algorithmics 6

6.2 Sweep Techniques

Transformation static d-dimensional → dynamic (d-1)-dimensional

d = 2: Plane Sweep

3) Closest Pair: Among n points, search the two that are closest together. **O(n log n)**

Preprocessing: Sort all points by x-coordinate
Sweep: Scan from left to right with 2 vertical lines *left* and *right*:

Invariants:

Horizontal distance between *left* and *right* is the minimum distance of the closest pair left of *left*.
Line content maintains all points between *left* and *right* sorted by y-coordinate.

Events and actions:

left passes point p: p is deleted O(log n)
right passes point p: p is inserted into *line content* and its distance is computed O(log n)
to all other points of *line content* of which the y coordinate differs from p at most by the minimum distance between points found so far.

only constant number!

References: Klein, Kap. 2.3.1 (in German)

Algorithmics 6

6.2 Sweep Techniques

Transformation static d-dimensional → dynamic (d-1)-dimensional

Characteristic properties of sweep techniques:

- Scan over selected and sorted x-coordinates (events) from left to right:
  The events lie in an EPS (Event point schedule)
- Sweep status structure (SSS) with invariants (SLS: Sweep Line Status)
- Events are computed statically during preprocessing (i.e. original reference points) and dynamically during updating the SSS.
- Sleeping objects: right of SSS, will yet be considered
  Active objects: within SSS, are currently relevant for updating the SSS
  dead objects: left of SSS, need never be considered again

References:

Klein, Kap. 2.3.1 (in German)

Algorithmics 6

Application: Computation of Voronoi diagrams by plane sweep

Objects of SSS:

- Right vertical line L (current x of EPS)
- Left beach line consisting of:
  - Parabolic segments of Bisector (p,L), given by the adjacent spikes (see below)
  - Spikes: Bisectors B(p,q) for two adjacent reference points p and q. Each beach line segment has got two adjacent spikes (except for the first and the last).

All segments have got a horizontal axis, because L is vertical.
This makes the intersection points of the beach line monotonic in y coordinate.

The parabolic segments are ordered in SSS by y coordinate of their intersection points
(Note: The intersection coordinates need not be computed explicitely in order to obtain their order, see next slides)

Lemma: The overall size of the beach line and hence of SSS is of order O(n)

References:

Klein, Kap. 6.3 (in German), de Berg et al., ch. 7.2

Algorithmics 6

Application: Computation of Voronoi diagrams by plane sweep

Objects of EPS:

- Point events: x coordinate of a reference point p

  How do we insert the corresponding parabola at the correct position into the SSS?

  Perform a binary search in the SSS and check with parabolic segments consecutively:
  If the y coordinate of both adjacent bisectors is higher (lower) than p_y, search above (below).
  If p_y is in between the y coordinate of the adjacent bisectors, this is the corrsponding parabolic segment hit by the new parabola.

- Spike events: x coordinate of the sweep line at which a beach line segment vanishes.

  How do we compute the x coordinate of a spike event?

  Let (x_0,y_0) be the intersection point of two adjacent spikes.
  Let $p_i = (x_i,y_i)$ be one of the reference points contributing to one of the two spikes.
  Then $x := x_0 + | (x_i,y_i) - (x_0,y_0) |$.

References:

Klein, Kap. 6.3 (in German), de Berg et al., ch. 7.2

Algorithmics 6

Application: Computation of Voronoi diagrams by plane sweep

Events and actions during sweep:

- Spike event: Intersection of adjacent spikes: Associated beach line segment vanishes.
  This requires an update of the beach line in the SSS:
  The reference point between the two intersecting spikes is not relevant anymore.
  This requires the computation of a new spike and the intersection with its upper and lower neighbor (if on the right hand) as new spike events.
  These spike events have to be inserted into the EPS.
- Point event: New point is passed: Generation of new beach line segment.
  Analogously to the above, this requires the computation of new spikes (new adjacencies, update of SSS) and spike events (update of EPS).

Run time: Update of each event in O(log n)

O(n) events ➔ **Total time complexity**: O(n log n)

crucial! *This is optimal!*

References:

Klein, Kap. 6.3 (in German), de Berg et al., ch. 7.2