ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Algorithms for Radio Networks

Voronoi-Diagramme

University of Freiburg
Technical Faculty
Computer Networks and Telematics
Prof. Christian Schindelhauer

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

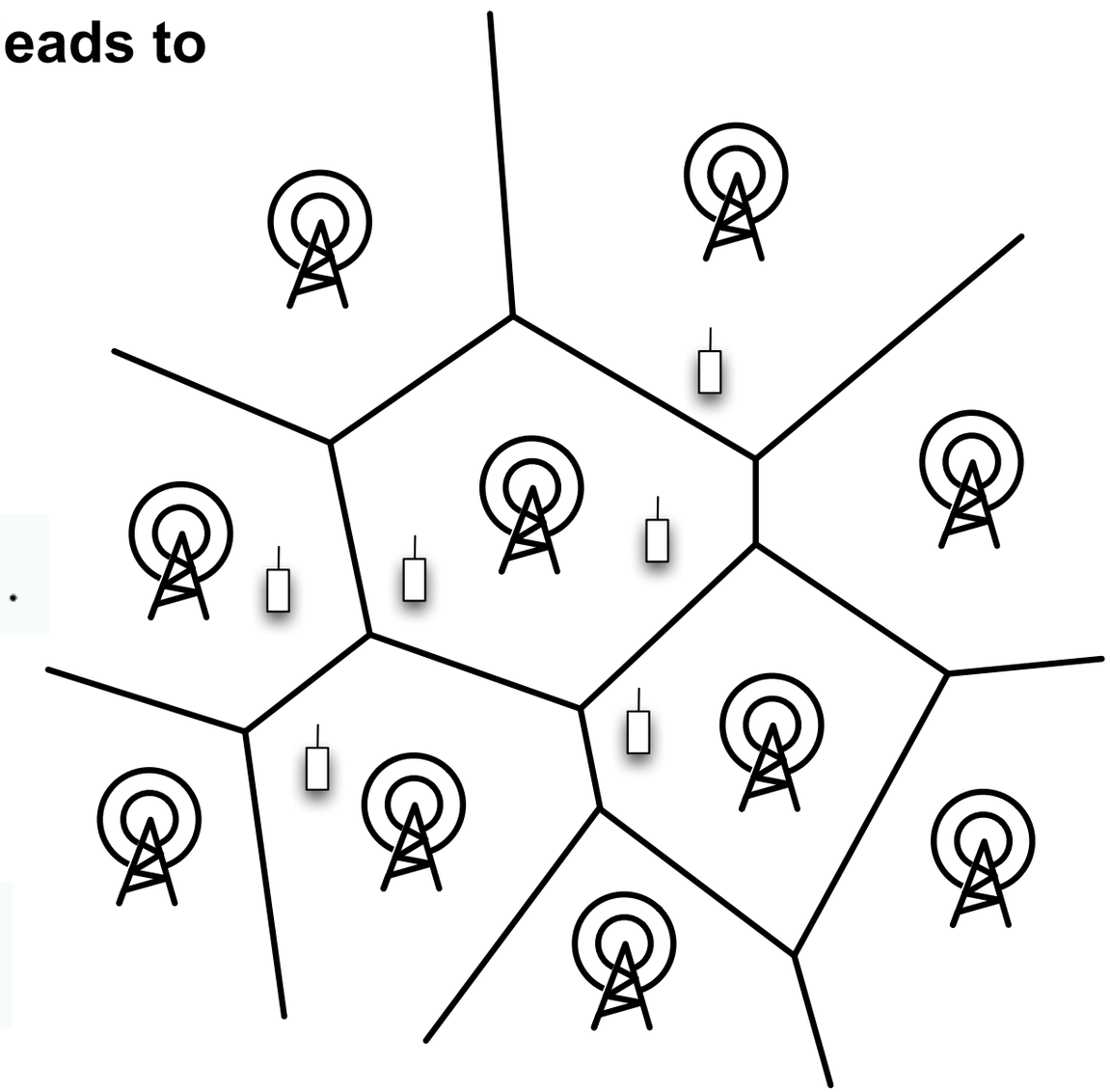
Voronoi-Diagrams Definition (I)

- **Best-Station-Problem w.r.t to path loss leads to Voronoi Diagrams**
- **Distance Measure:**
  - Euclidean Norm
  - = L_2-Norm
  - $p=(p_1,p_2)$, $q=(q_1,q_2) \in \mathbf{R}^2$

$$|p,q| := ||p,q||_2 := \sqrt{(p_1-q_1)^2+(p_2-q_2)^2}\ .$$

  - **Bisector B(p,q)**

$$B(p,q) := \{x \in \mathbb{R}^2 \mid |p,x| = |q,x|\}\ .$$



Computation of the Voronoi Diagram

- **Naive algorithm**
  - Compute for each point the intersection of all half-planes
  - Runtime: $O(n^2)$
- **Can it be done in linear time?**
- **Counter example**
  - Sorting of n number has minimum time $\Omega(n \log n)$
  - if only comparison may be used

Time Complexity (lower bound)

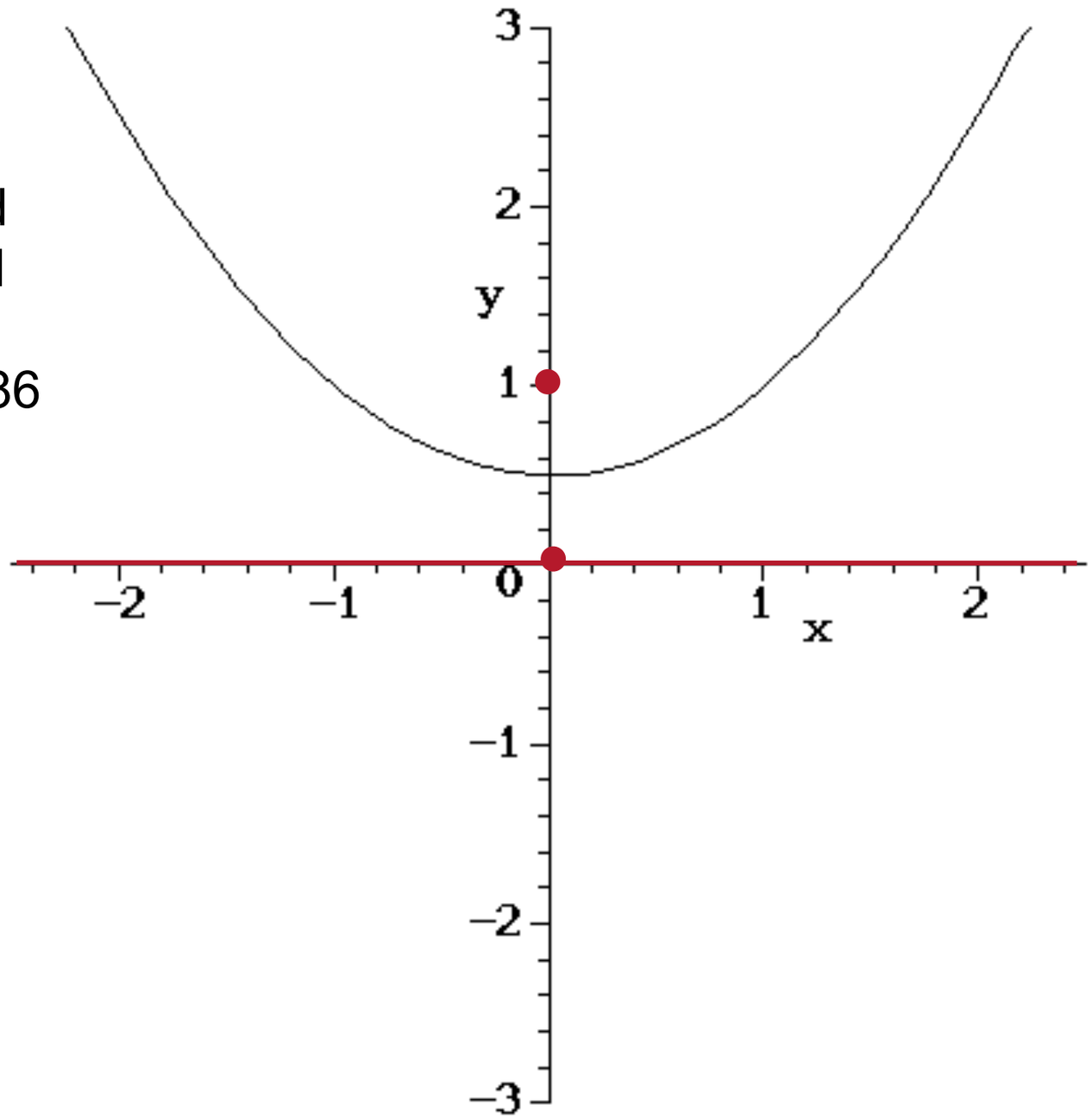
- **Theorem**
  - Sorting of number has time complexity of $\Omega(n \log n)$.
- **Theorem**
  - The computation of the convex hull of a given point set has time complexity $\Omega(n \log n)$
- **Proof**
  - Consider point set (x_1,x_1^2), (x_2, x_2^2), ..., (x_n, x_n^2)
  - Each point is on the convex hull
  - Sorted list is given by a round tour

Time Complexity (lower bound)

- **Theorem**
  - The computation of the convex hull of a given point set has time complexity $\Omega(n \log n)$
- **Theorem**
  - The computation of the Voronoi diagram of a given point set has time complexity $\Omega(n \log n)$
- **Proof**
  - The convex hull can be constructed from the Voronoi-diagram in linear time.
  - Therefore, every algorithm solving the Voronoi-diagram can be used to construct the convex hull

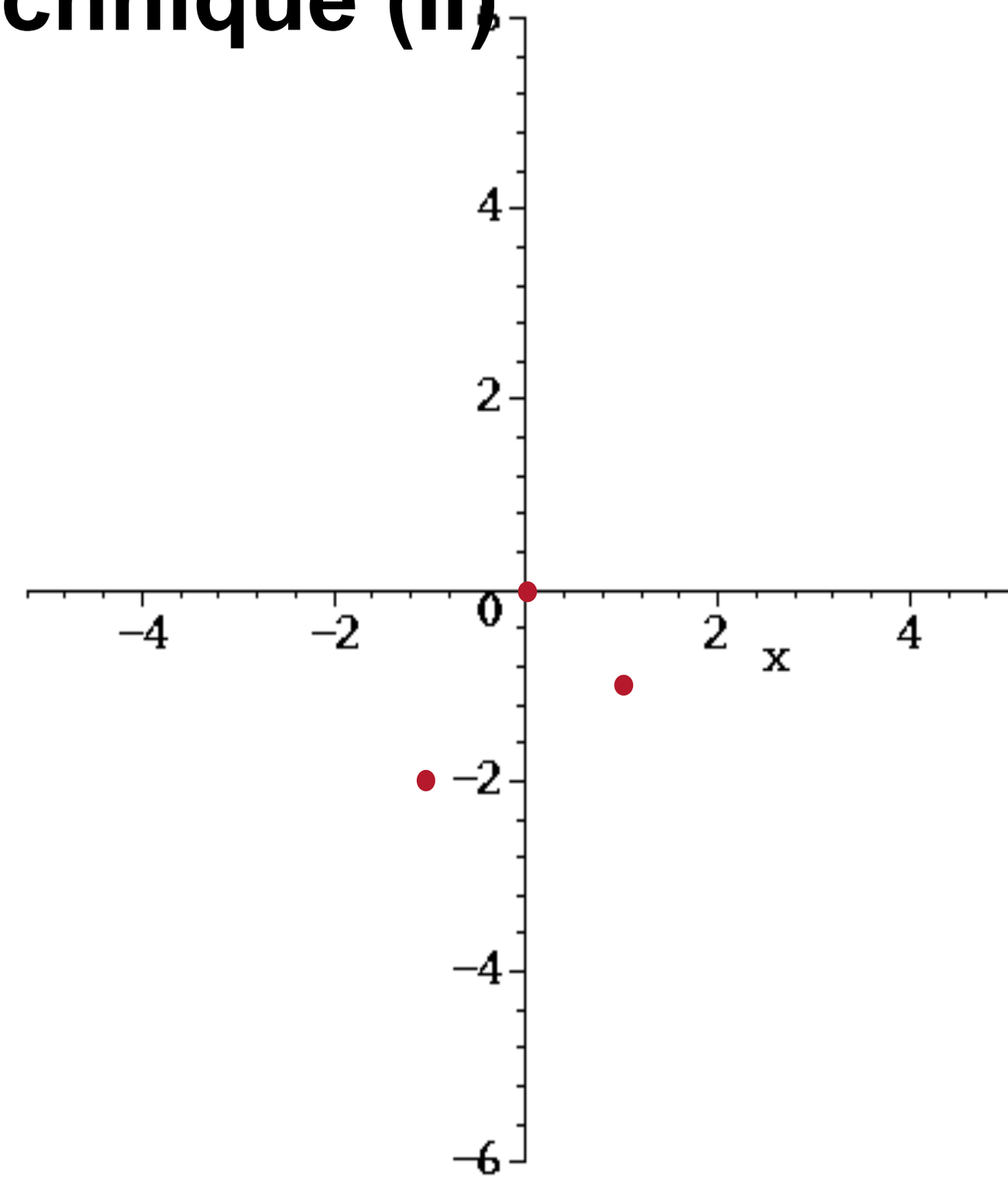
Sweep-Technique (I)

- **Steven Fortune.**
  - A sweepline algorithm for Voronoi diagrams. Proceedings of the second annual symposium on Computational geometry. Yorktown Heights, New York, United States, pp.313–322. 1986
- **Consider Bisector of point (0,a) und line y=0**
- **Parabola**
  - $x^2 + (y-a)^2 = y^2$
  - $y = x^2/(2a) + a/2$
- **Move the line vertically**



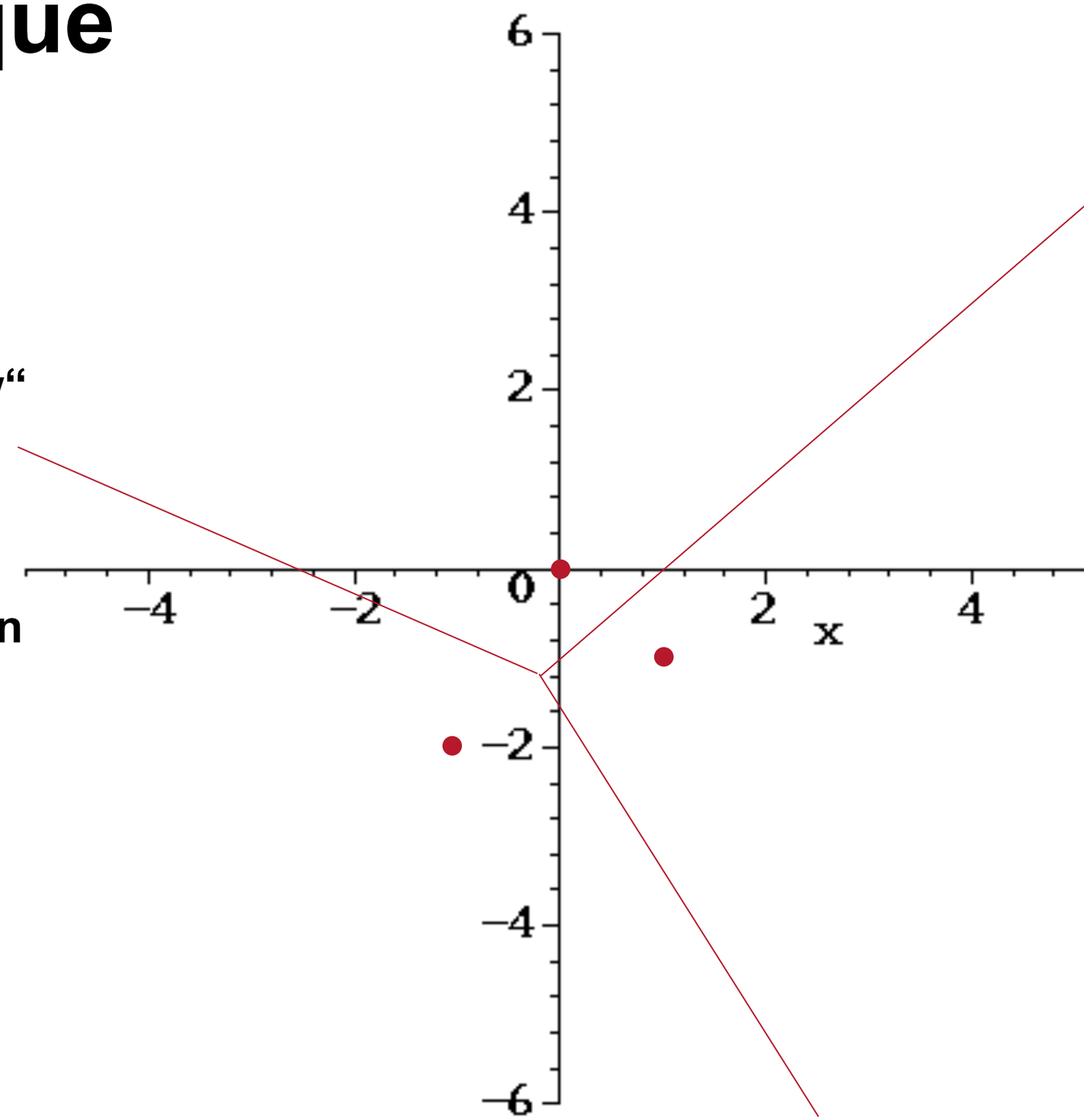
Sweep-Technique (II)

- **Multiple Points**
  - Consider a frontal wave front
  - of pieces of parabolas
- **Events**
  - When a piece of a parabola appears
  - or disappears
    - we get a point of the Voronoi-diagram



Sweep-Technique (III)

- Intersections of the parabola front drive form the Voronoi diagram
- Edges of the Voronoi diagram „grow“ when the sweep-line moves
- when two neighbored spikes meet, then a Voronoi point is created
- The number of the parabola pieces in the front line is linear



Event A: Inserting a Parabola

- **Sweep line encounters a point p $\in$ V**
  - Parabola begins as a half-line
  - It divides an existing parabola
    (if the sweep line is in general position)
  - Since no other operation generates parabolas, we have at most 2n-1 parabolic pieces
- **The two intersections of the old wave front define a point of a Voronoi edge**
  - The extension of the edge of the wavefront is also known as **spike**
  - The intersections of the parabolas move on these spikes until they meet a Voronoi point
  - Meeting potential spikes indicate possible Voronoi points

Event B: Deleting a Parabola

- **Two neighbored parabola pieces may move towards each other**
  - Parabolas between newer breakpoints disappear
  - Voronoi region of the disappeared is now behind the sweep line
  - Since this region is convex Voronoi region is processed
- **The meeting point of the break points must be a "known" spike**
- **Calculate earliest "date" at which deletes the sweep line is a parabola**
  - This can happen only between neighbored pieces of parabolas

Necessary Data Structures

- **Wave front**
  - Data structure of points
    - sorted w.r.t. x-coordinats
    - Search, insert, needs time O(log n)
- **Even priority queue**
  - priority is y-coordinate
  - update queue on each event

Event-Queue

- **Priority is the y-coordinate of the sweep-line**
  - Event A: parabola appears
    - priority = y-coordinate of the points of V
  - Event B: parabola disappears
    - priority = y-coordinate of the sweep line touching the perimeter circle of the points of any three neighbored parabola pieces
- **Update queue on every event**
- **Run-time O(log n) (amortized) for**
  - Inserting into queue
  - Erasing of elements of the queue

Sweep-Line-Algorithm for Voronoi-Diagrams

Algorithm Sweep-Line-VD(V)

```
Given V ⊂ R²
Init Q with events of type A
while Q ≠ ∅ do
    Let q event of Q with maximum priority
    Delete q from Q
    if q is of type A (Insert Parabola) then
        Find position in wave front and erase possible spike intersection of
            parabolas of Q
        Insert new parabola and divided parabolas into wave front
        Compute new events of type B and insert them into Q
    else (type B - Parabola disappears)
        Insert new Voronoi-point and edge into Voronoi-diagram
        Erase parabola from wavefront
        Erase events (of type B) from Q
        Compute new events (of type B) for the new Voronoi edgde
    fi
od
```

Analysis

- **Theorem**
  - The Voronoi-Diagram can be constructed in time O(n log n) for n points
- **Proof sketch**
  - The outer While-loop will be used n times
  - Each elementary operation in W and Q needs time O(log n)

Voronoi Diagrams
Features

- **Voronoi regions are konvex**
- **Voronoi diagrams of n point in the plane**
  - consists of segements, lines, rays and points
  - is a geometric graph
  - has $O(n)$ nodes and edges
  - cannot be computed in time $o(n \log n)$
  - can be computed in time O(n log n)

ALBERT-LUDWIGS-
UNIVERSITÄT FREIBURG

Algorithmen für drahtlose Netzwerke

University of Freiburg
Technical Faculty
Computer Networks and Telematics
Prof. Christian Schindelhauer

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG