ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG

Algorithmen für drahtlose Netzwerke

Voronoi-Diagramme

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG

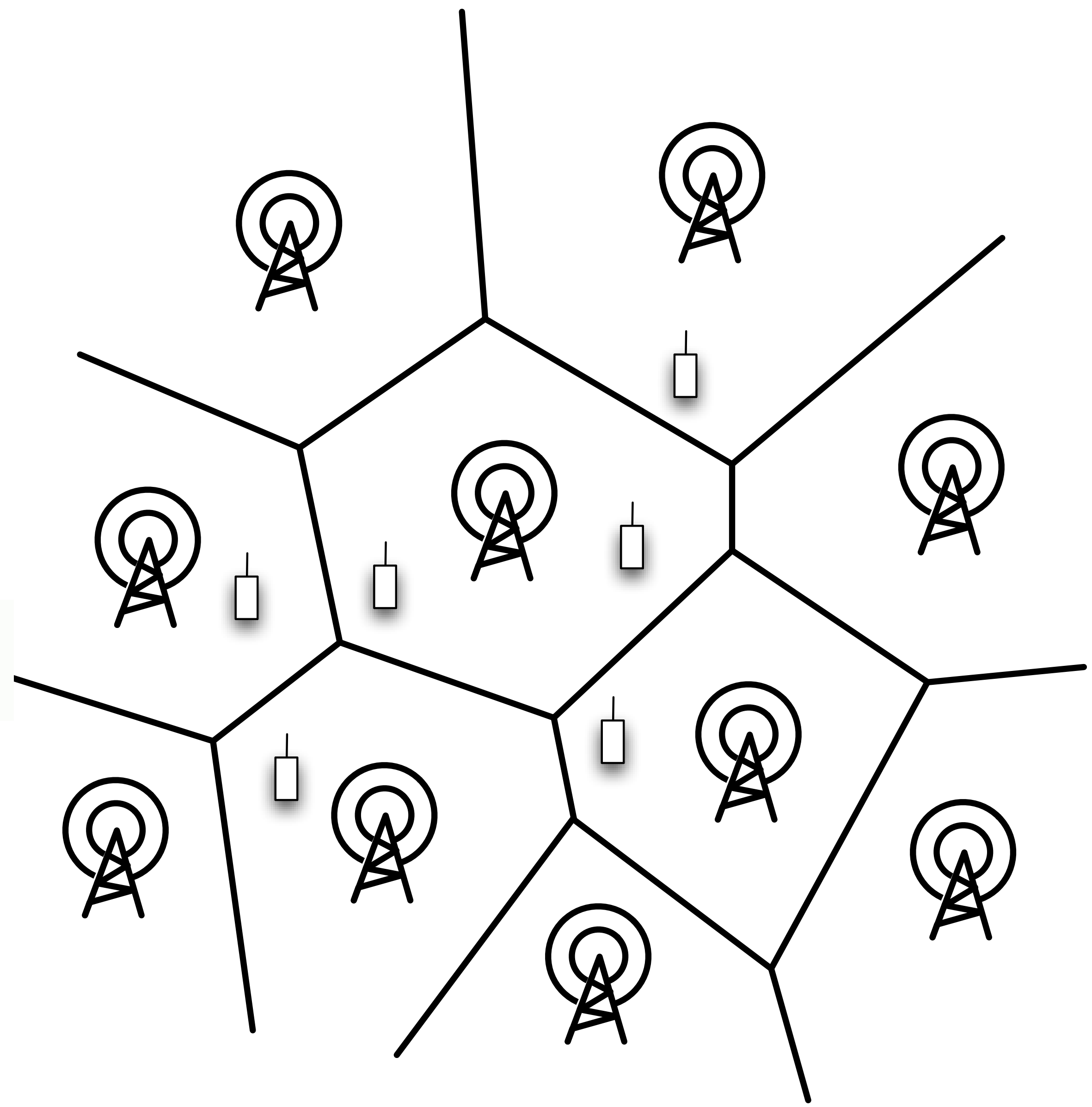
Voronoi-Diagramme Definition (I)

- **Best-Station-Problem nur unter Berücksichtung von Path Loss führt zu Voronoi-Diagrammen**
- **Abstandsmaß:**
  - Euklidischer Abstand
  - = L_2-Norm
  - $p=(p_1,p_2)$, $q=(q_1,q_2) \in \mathbf{R}^2$

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

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

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



Berechnung des Voronoi-Diagramms

- **Naiver Algorithmus:**
  - Berechne für jeden Punkt den Schnitt aller Halbebenen
  - Laufzeit: $O(n^2)$
- **Kann man es in Linearzeit berechnen?**
- **Gegenbeispiel**
  - Das Sortierproblem von n Zahlen hat eine Zeitkomplexität von $\Omega(n \log n)$
  - wenn man nur Vergleiche verwenden darf

Zeitkomplexität (untere Schranke)

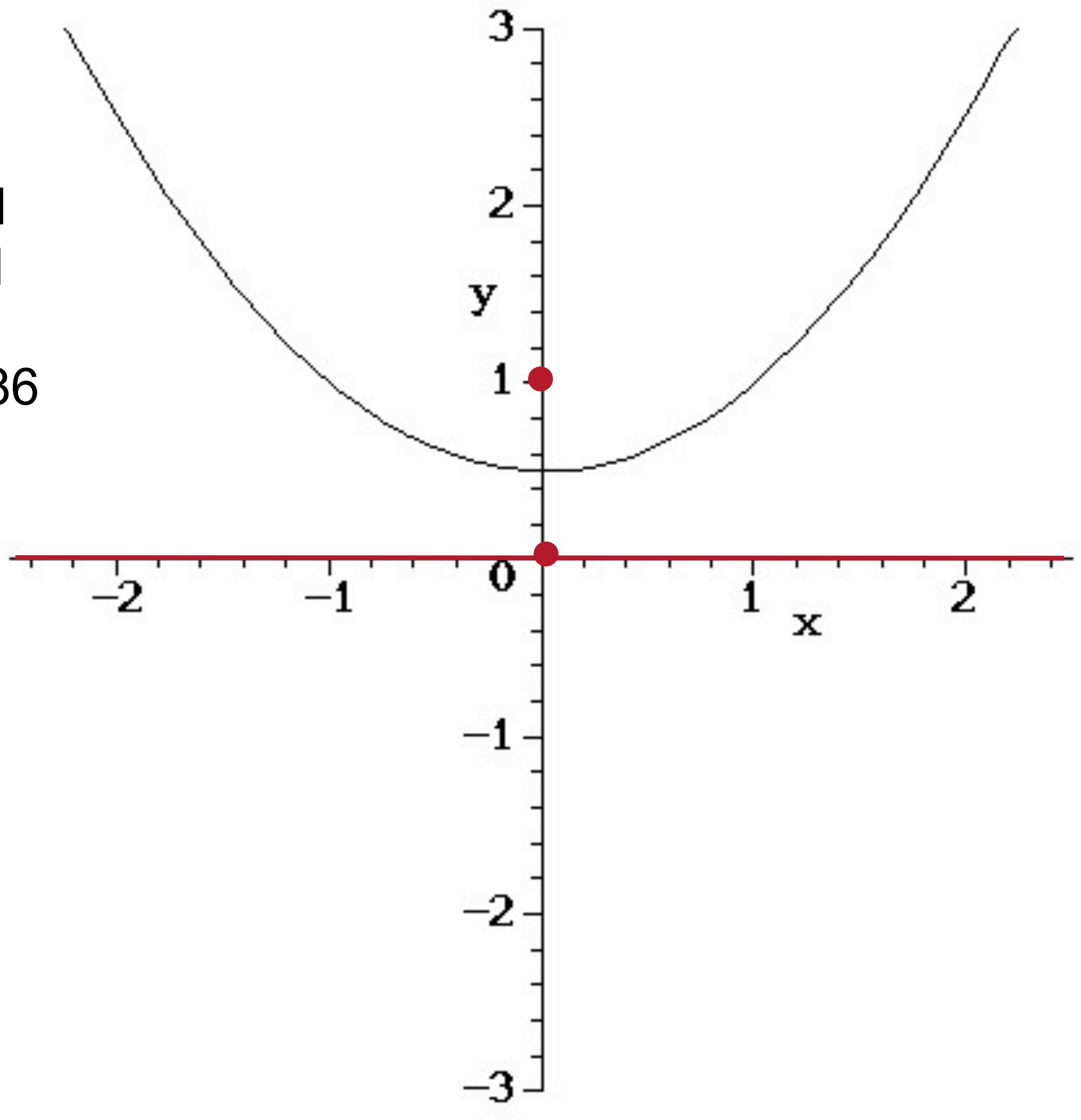
- **Theorem**
  - Das Sortierproblem von n Zahlen hat eine Zeitkomplexität von $\Omega(n \log n)$.
- **Theorem**
  - Die Berechnung der konvexen Hülle einer gegebenen Punktmenge hat mindestens Zeitkomplexität $\Omega(n \log n)$
- **Beweis**
  - Betrachte Punktmenge $(x_1, x_1^2), (x_2, x_2^2), \ldots, (x_n, x_n^2)$
  - Jeder Punkt ist auf der konvexen Hülle
  - Abwandern liefert eine sortierte Liste

Zeitkomplexität II (untere Schranke)

- **Theorem**
  - Die Berechnung der konvexen Hülle einer gegebenen Punktmenge hat mindestens Zeitkomplexität $\Omega(n \log n)$
- **Theorem**
  - Die Berechnung des Voronoi-Diagramms hat mindestens die Zeitkomplexität $\Omega(n \log n)$
- **Beweis**
  - Die offenen Voronoi-Gebiete beschreiben die konvexe Hülle der Punktmenge
  - Damit kann man mit Voronoi-Diagrammen in linearer Zeit die konvexe Hülle berechnen

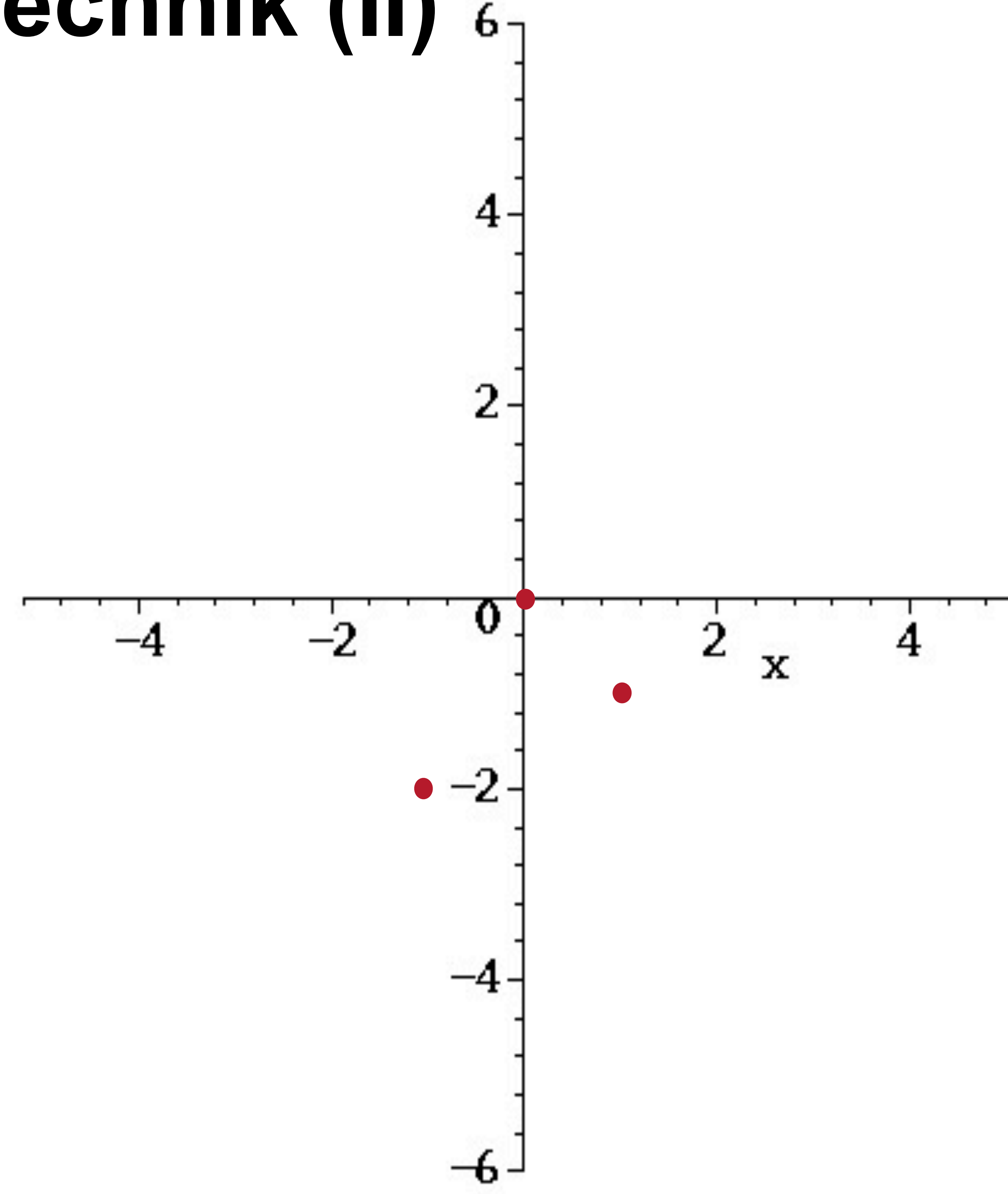
Sweep-Technik (I)

- **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
- **Betrachte Bisektor zwischen Punkt (0,a) und Linie y=0**
- **Ergibt Parabel mit**
  - $x^2 + (y-a)^2 = y^2$
  - $y = x^2/(2a) + a/2$
- **Nun bewegt sich die Sweep-Linie vertikal**



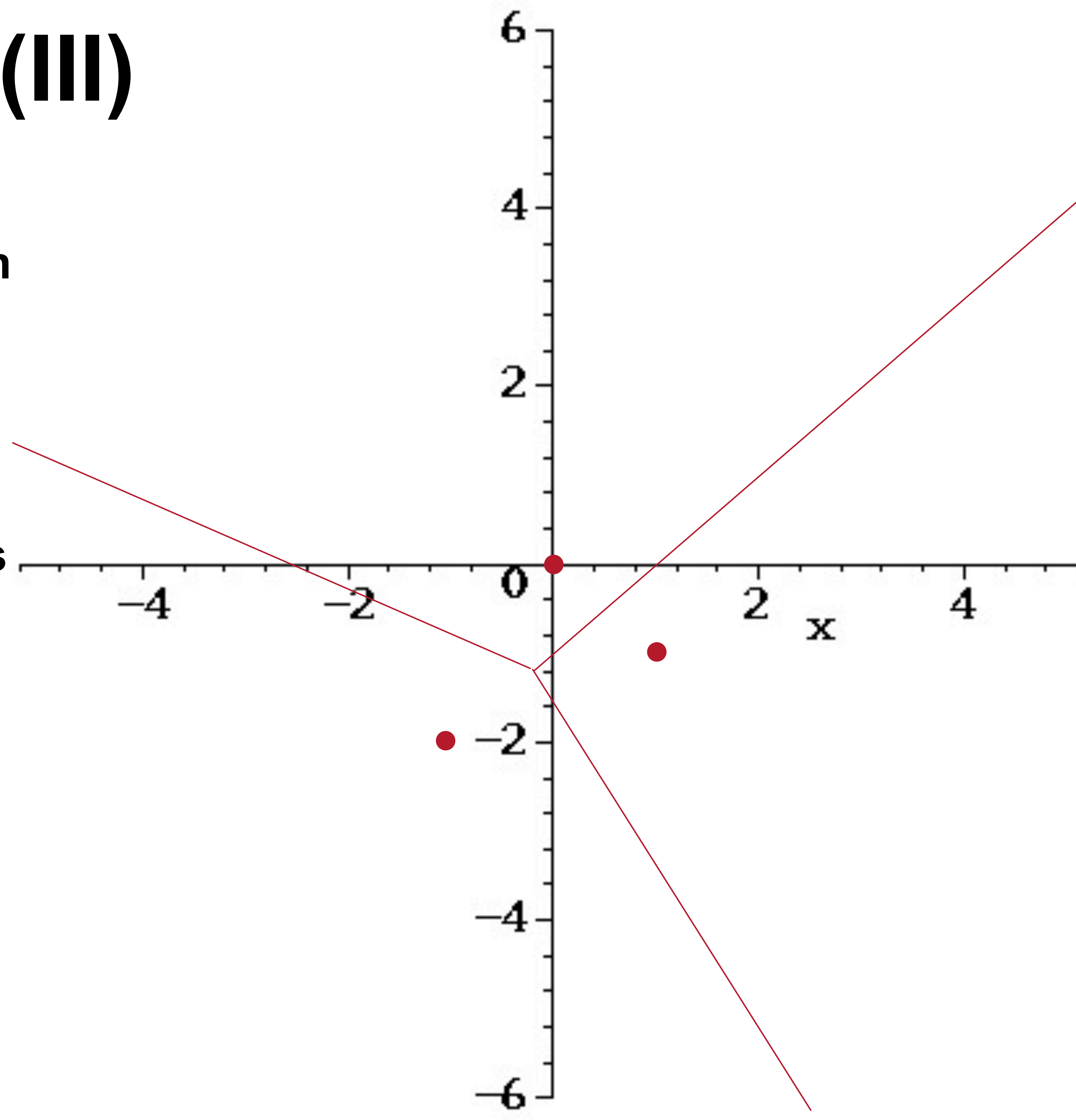
Sweep-Technik (II)

- **Mehrere Punkte**
  - Nur vordere Wellenfront ist von Interesse
  - Unterteilung der Wellenfront in Parabelstücke
- **Ereignisse**
  - Entstehen eines Parabelstücks
    - Nur wenn die Wellenfront einen Punkt streift
  - Verschwinden eines Parabelstücks
    - Dadurch entsteht Voronoi-Punkt



Sweep-Technik (III)

- Schnittpunkte der Wellenfront fahren das Voronoi-Diagramm ab
- Treffen sich zwei Parabeln an der Wellenfront, verschwindet ein Parabelstück
- Nur benachbarte Spikes kommen als nächster Treffpunkt in Frage
- Die Anzahl der Parabelstücke in der Frontlinie ist linear



Ereignis A: Einfügen neuer Parabel

- **Sweep-Line trifft auf Punkt p∈V**
- **Parabel beginnt als Halbgerade**
  - Halbiert bestehendes Parabelstück (falls Sweep-Line in allgemeiner Position)
  - Da keine andere Operation Parabeln erzeugt, höchstens 2n-1 Parabelstücke
- **Die beiden Schnittpunkte mit der alten Wellenfront definieren eine beginnende Voronoi-Kante**
  - Die Verlängerung der Kante über die Wellenfront hinaus, wird Spike genannt
  - Die Schnittpunkte der Parabeln bewegen sich auf diesen Spikes bis sie einen Voronoi-Punkt treffen
  - Treffpunkte möglicher Spikes zeigen mögliche Voronoi-Punkte an

Ereignis B: Löschen einer Parabel

- **Bewegung der Sweep-Line läßt Bruchpunkte aufeinanderlaufen**
  - Parabel zwischen Bruchpunkten verschwindet
  - Voronoi-Region der verschwundenen nun hinter Sweep-Linie
  - Da Region konvex ist diese Voronoi-Region abgearbeitet
- **Der Treffpunkt der Bruchpunkte muß ein “bekannter” Spike sein**
- **Berechne frühesten “Zeitpunkt”, an dem die Sweep-Linie eine Parabel löscht**
  - Hierfür müssen nur benachbarte Parabelstücke berücksichtigt werden
  - Nach Löschen, aktualisiere mögliche Treffpunkte

Notwendige Datenstrukturen

- **Wellenfront**
  - Datenstruktur mit Punkten aus V, die relevante Parabeln beschreiben
    - sortiert nach x-Koordinaten
    - Jede Suche, Einfüge-Operation in Zeit O(log n)
- **Ereignis-Warteschlange als Priority-Queue**
  - Priorität ergibt sich durch y-Koordinate der Sweep-Linie L
  - Aktualisiere Warteschlange bei jedem Ereignis

Ereignis-Warteschlange

- **Priorität ergibt sich durch y-Koordinate der Sweep-Linie L**
  - Erzeugen Parabel:
    - Priorität = y-Koordinate der Punkte aus V
  - Löschen Parabel:
    - Priorität = y-Koordinate der Sweep-Linie, die den Kreis mit Mittelpunkt des Spike-Schnittpunkts s und Radius |s,u| von unten berührt, wobei u beteiligter Knoten eines Spikes ist.
- **Aktualisiere Warteschlange bei jedem Ereignis**
- **Zeitaufwand O(log n) (amortisiert) für**
  - Einfügen in Warteschlange
  - Löschen eines Elements der Warteschlange

Sweep-Line-Algorithmus für Voronoi-Diagramme

Algorithmus Sweep-Line-VD(V)

Gegeben $V \subset R2$

Initialisiere Q mit Ereignissen A

while Q nicht leer do

Sei q aus Q mit maximaler Priorität

Entferne q aus Q

if q von Typ A (Einfügen Parabel) then

Suche Platz in Wellenfront und lösche evtl. Spike-Schnittpunkt der geschnitttenen Parabel mit Nachbarn aus Q

Füge neue Parabel und geschnittene Parabel in Wellenfront ein

Berechne neue Spike-Schnittpunkte mit Nachbarn, füge diese in Q ein

else (Lösche Parabel)

Füge neuen Voronoi-Punkt und Kanten in Voronoi-Diagramm ein

Lösche Parabel aus Wellenfront

Lösche Spike-Schnittpunkte dieser Parabel mit Nachbarn

Berechne neue Spike-Schnittpunkte für neu entstandende Voronoi-Kante

fi

od

end

Analyse

- **Theorem**
  - Für jede Menge V von n Punkten in der Ebene kann das Voronoi-Diagramm in Zeit O(n log n) konstruiert werden.
- **Beweisskizze:**
  - Die äußere While-Schleife wird O(n) mal durchlaufen
  - Der Gesamtspeicherverbrauch in W, Q und QEDS von VD(V) ist O(n)
  - Jede elementare Operation in W und Q benötigt bei geeigneter Implementation Zeit O(log n)

Voronoi-Diagramme
Eigenschaften

- **Voronoi-Regionen sind konvex**
- **Voronoi-Diagramm von *n* Punkten in der Ebene**
  - besteht aus Strecken, Geraden, Halbgeraden und Punkten
  - ist ein geometrischer Graph
  - hat $O(n)$ viele Knoten und Kanten
  - kann in Zeit $O(n^3)$ berechnet werden (naiver Algorithmus)
  - kann nicht in Zeit $o(n \log n)$ berechnet werden
  - kann in Zeit O(n log n) berechnet werden

ALBERT-LUDWIGS-UNIVERSITÄT FREIBURG

Algorithmen für drahtlose Netzwerke

Albert-Ludwigs-Universität Freiburg
Institut für Informatik
Rechnernetze und Telematik
Prof. Dr. Christian Schindelhauer

CoNe Freiburg

IIF INSTITUT FÜR INFORMATIK FREIBURG