Untere Schranke und Sweep für VD

Elmar Langetepe
University of Bonn

Untere Schranke mit Vorsortieren

- Sortieren nach X-Koordinate, dann konvexe Hülle in $O(n)$
- Beim Voronoi Diagramm nützt das nichts

Theorem 6.1 Angenommen die Punkte aus S mit $|S| = n$ sind bereits nach X-Koordinaten vorsortiert. Dann erfordert die Konstruktion des Voronoi Diagrammes immer noch Aufwand $\Omega(n \log n)$.

- Beweis: ϵ-closeness $\leq_n$ Voronoi Diagramm mit X Sortierung
- Korollar 1.6: ϵ-closeness Laufzeitkompl. $\Omega(n \log n)$

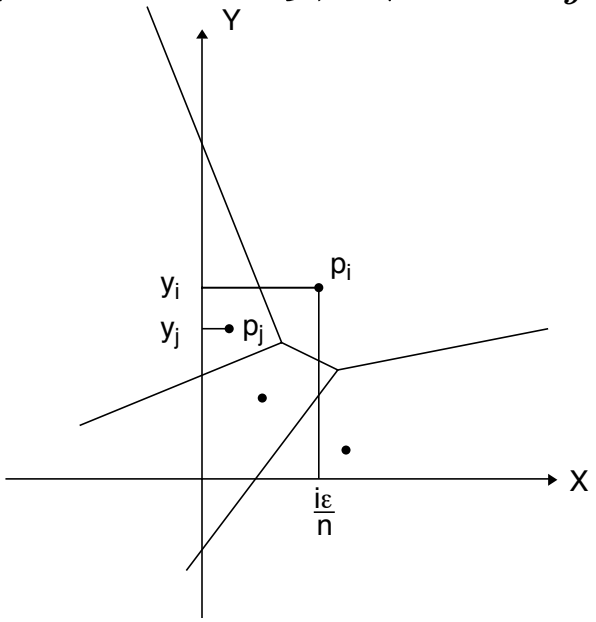
Beweis: Schranke mit Vorsortieren

Theorem 6.1 Angenommen die Punkte aus S mit $|S| = n$ sind bereits nach X-Koordinaten vorsortiert. Dann erfordert die Konstruktion des Voronoi Diagrammes immer noch Aufwand $\Omega(n \log n)$.

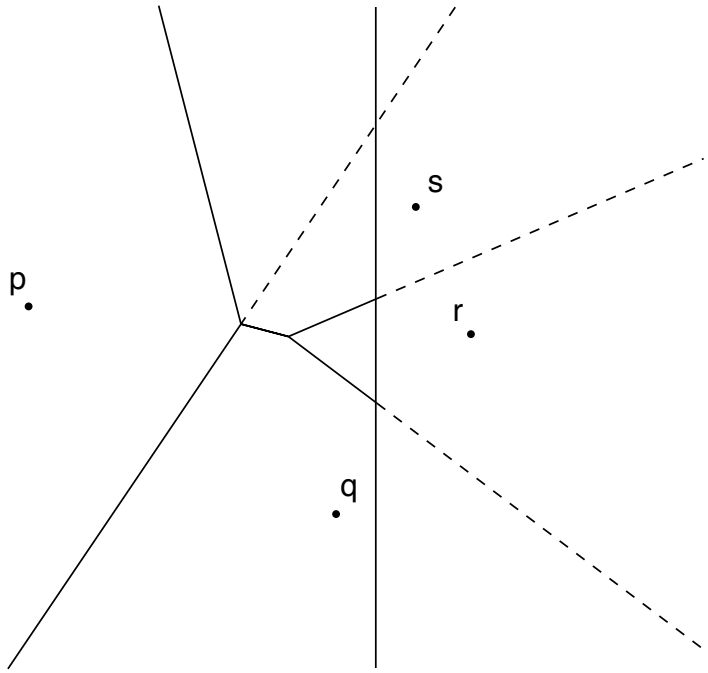
Beweis: Schranke mit Vorsortieren

Beweis: Schranke mit Vorsortieren

- Folge $(y_1, y_2, \ldots, y_n)$, Punkte $p_i = \left(\frac{i \cdot \epsilon}{n}, y_i\right)$, X-sortiert!
- Durchlaufen der Regionen Y-Achse: $O(n)$
- Alle $(0, y_i) \in \mathrm{VR}(p_i, \{p_1, \ldots, p_n\})$, sortiert Folge y_j
- Ein $(0, y_i) \in \mathrm{VR}(p_j, \{p_1, \ldots, p_n\})$, $|y_i - y_j| < \epsilon$

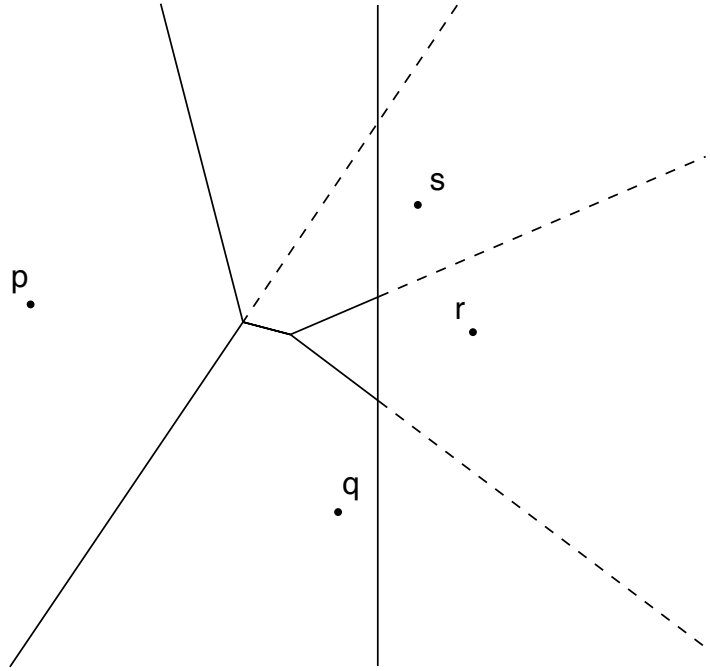


Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3



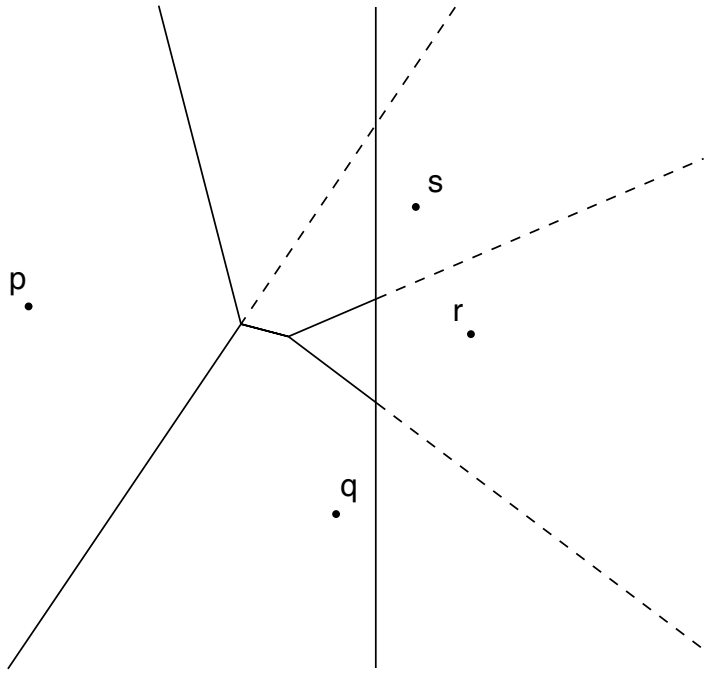
Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$



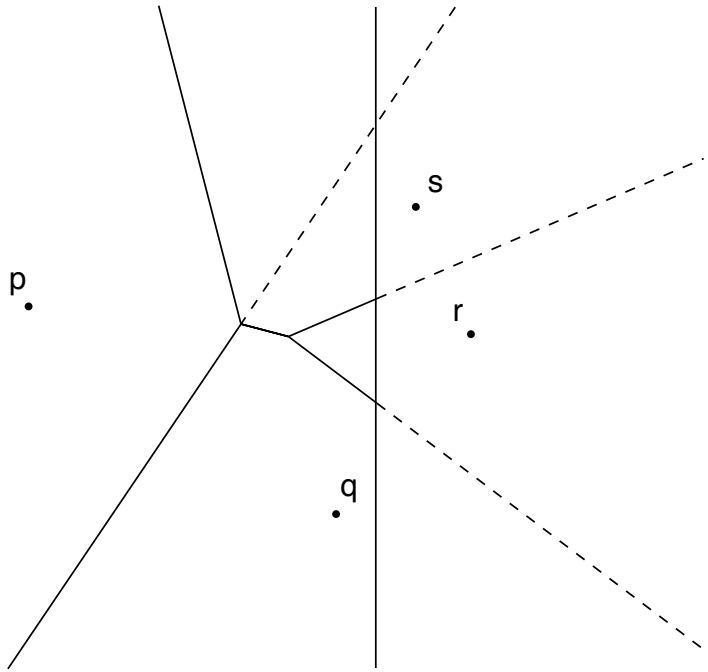
Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$
- Sweepl. über Punkte, X-sortiert



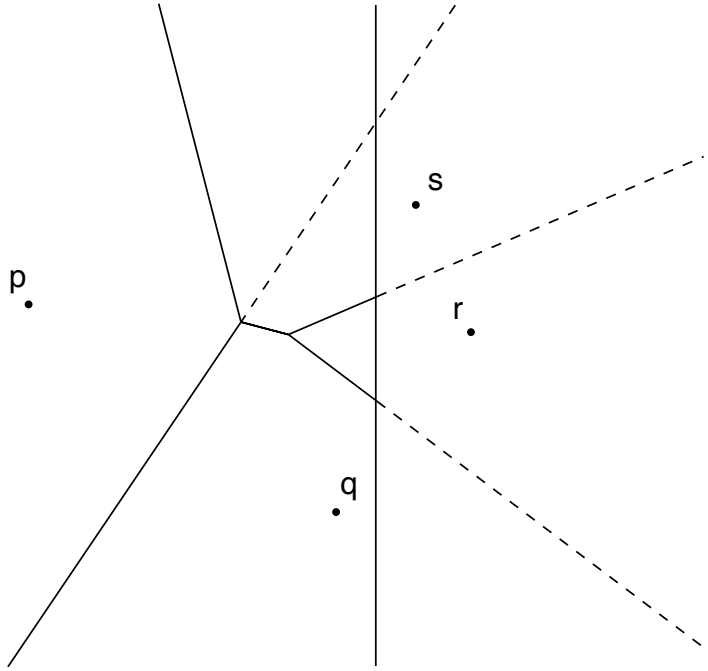
Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$
- Sweepl. über Punkte, X-sortiert
- Links Sweepline entsteht Voronoi-Diagramm



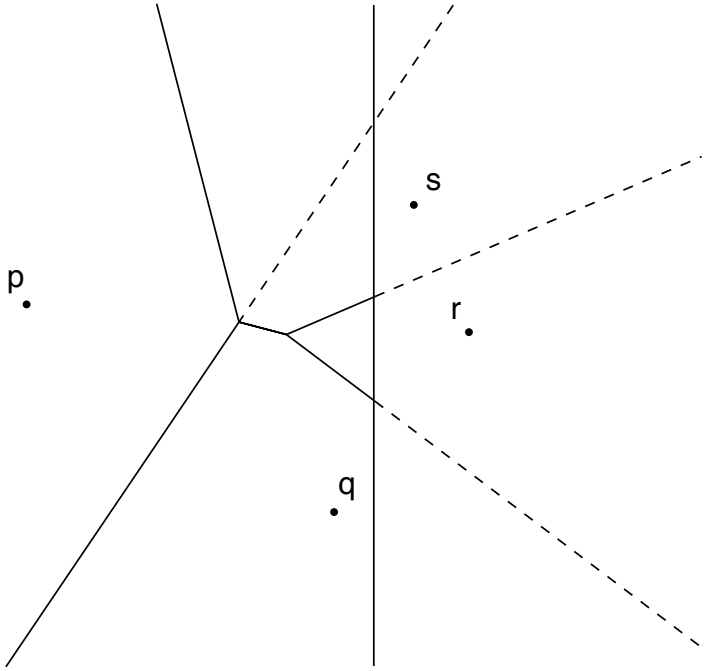
Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$
- Sweepl. über Punkte, X-sortiert
- Links Sweepline entsteht Voronoi-Diagramm
- Problem: Sweepline kennt nur Punkte links der Linie



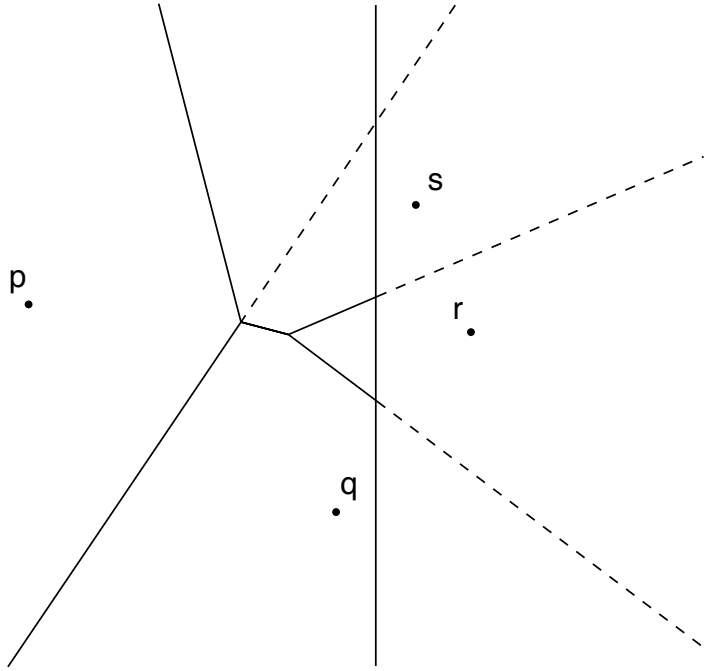
Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$
- Sweepl. über Punkte, X-sortiert
- Links Sweepline entsteht Voronoi-Diagramm
- Problem: Sweepline kennt nur Punkte links der Linie
- Punkte rechts können weit hineinragen

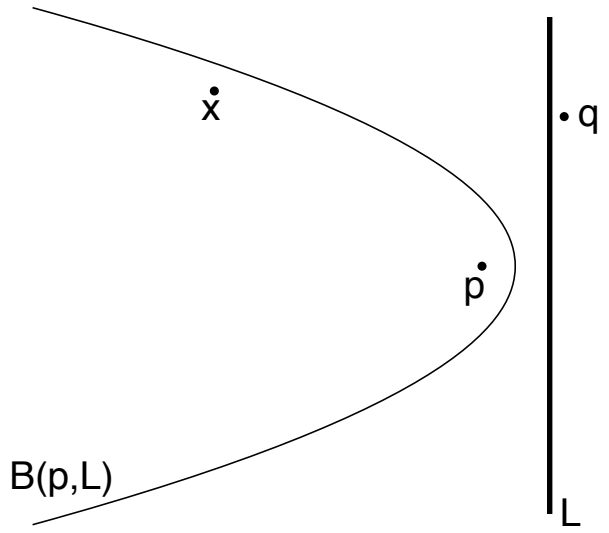


Voronoi Diagramm, Sweep, $O(n \log n)$, 6.3

- Deterministisch in $O(n \log n)$
- Sweepl. über Punkte, X-sortiert
- Links Sweepline entsteht Voronoi-Diagramm
- Problem: Sweepline kennt nur Punkte links der Linie
- Punkte rechts können weit hineinragen
- Abbhilfe: Sweepline selbst ist auch ein Objekt, simuliert mögliche Punkte weiter rechts

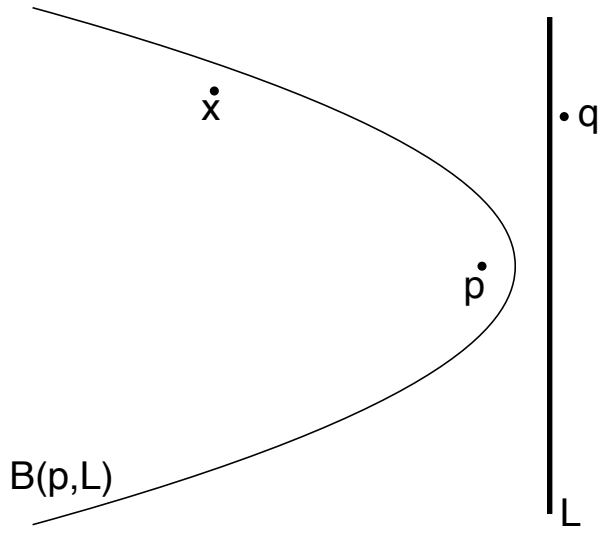


Voronoi Diagramm, Sweepline als Objekt



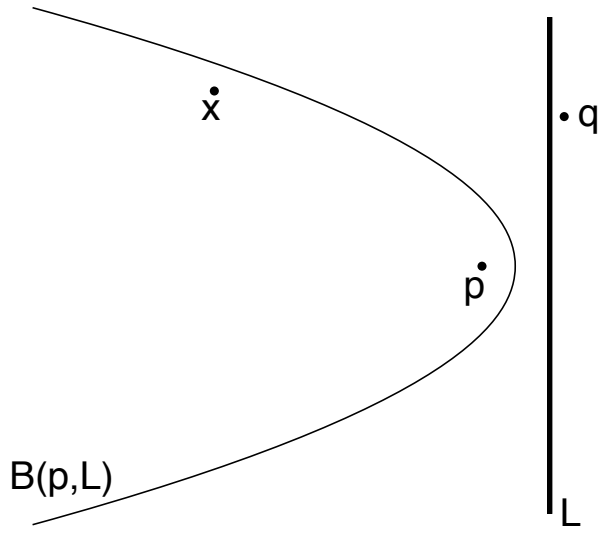
Voronoi Diagramm, Sweepline als Objekt

- Bisektor $B(p, L)$ zwischen L und p: Parabel



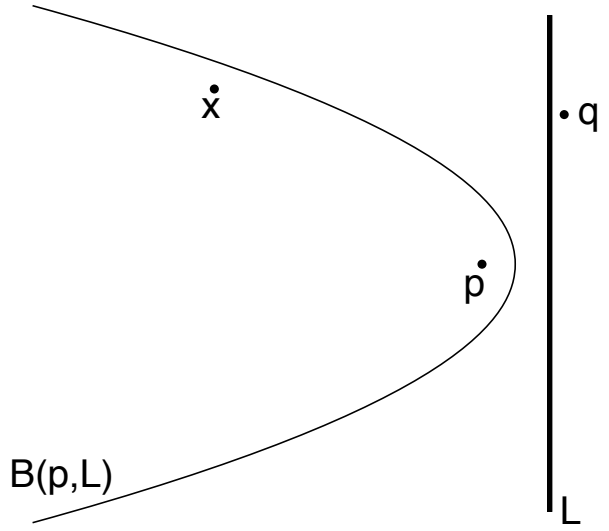
Voronoi Diagramm, Sweepline als Objekt

- Bisektor $B(p, L)$ zwischen L und p: Parabel
- Punkt q rechts kann nicht mehr *Besitzer* von x werden

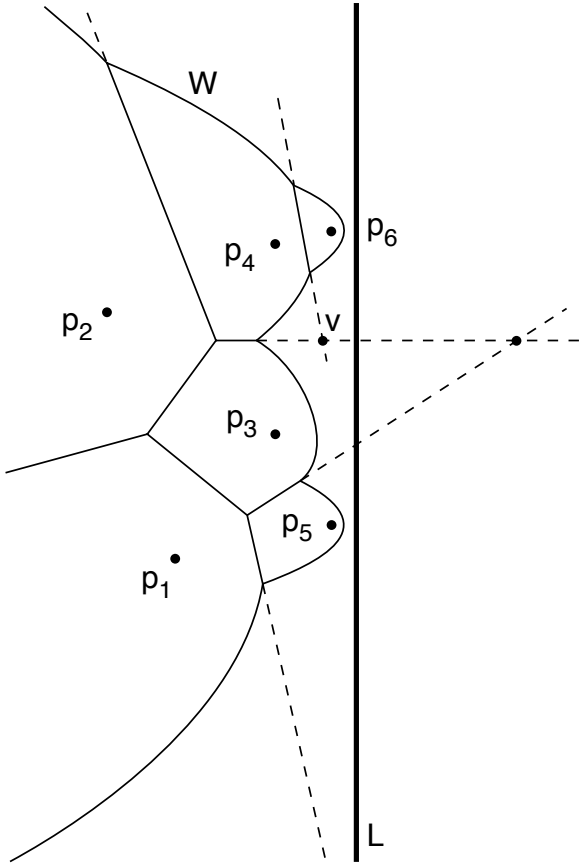


Voronoi Diagramm, Sweepline als Objekt

- Bisektor $B(p, L)$ zwischen L und p: Parabel
- Punkt q rechts kann nicht mehr *Besitzer* von x werden
- Parabel ist bereits korrekt bezüglich Zuteilung links

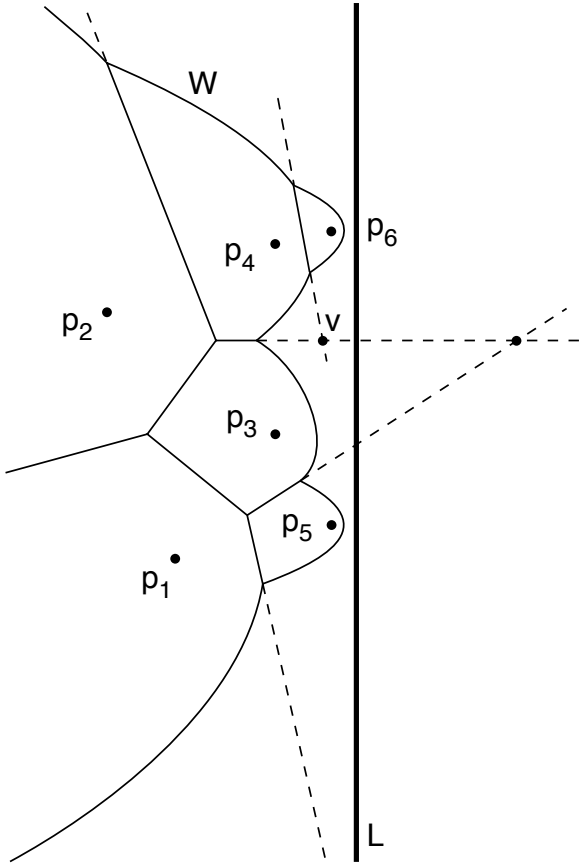


SSS: Wellenfront mit Verlängerungen



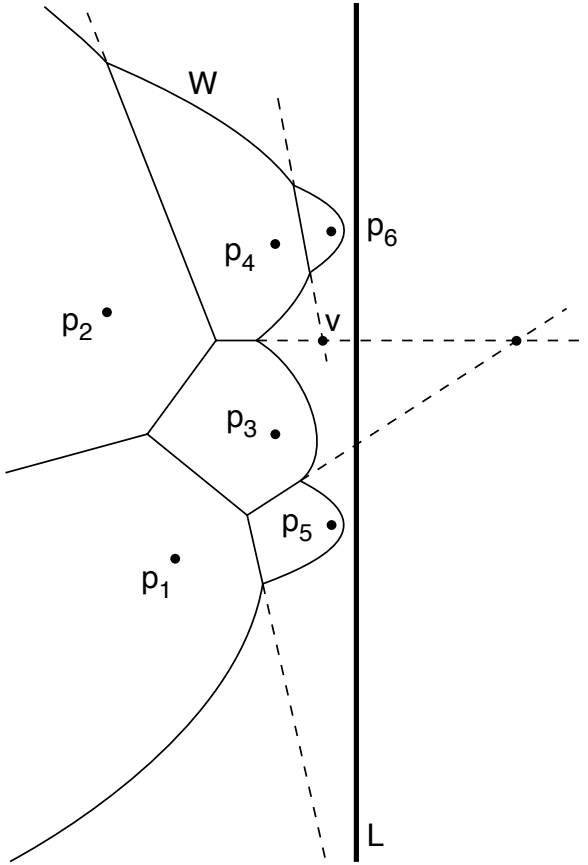
SSS: Wellenfront mit Verlängerungen

- Wellenfront: Mehrere Stücke $B(p_j, L)$ aneinandergereiht



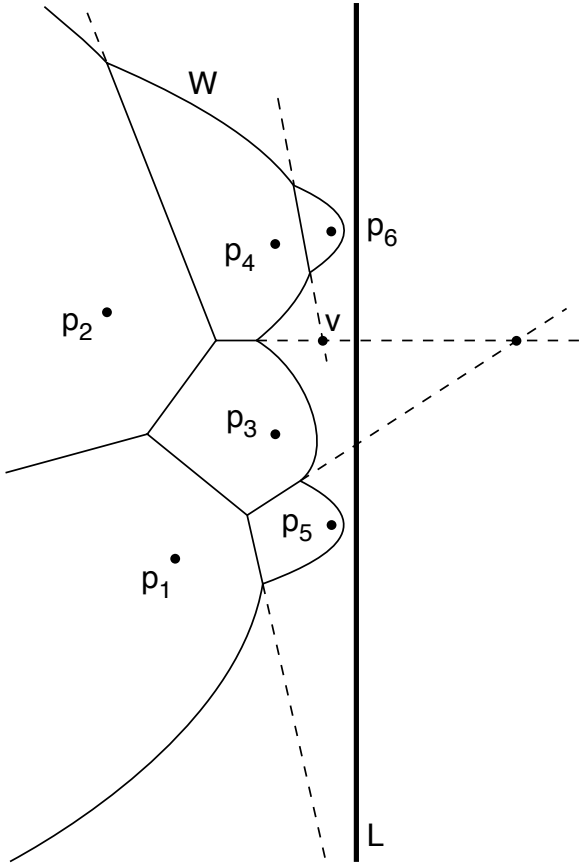
SSS: Wellenfront mit Verlängerungen

- Wellenfront: Mehrere Stücke $B(p_j, L)$ aneinandergereiht
- Links der Wellenfront: $VR(\{p_1, \ldots, p_{i-1})$ fertig



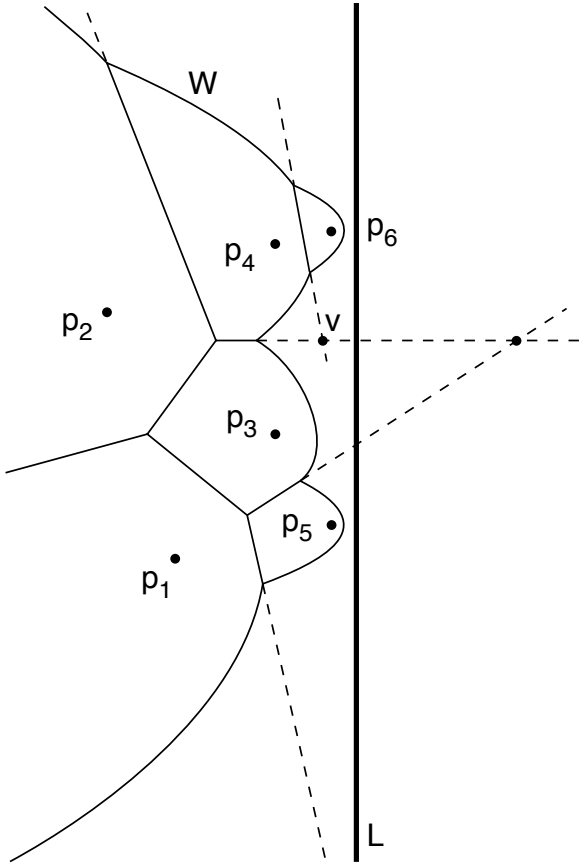
SSS: Wellenfront mit Verlängerungen

- Wellenfront: Mehrere Stücke $B(p_j, L)$ aneinandergereiht
- Links der Wellenfront: $VR(\{p_1, \ldots, p_{i-1})$ fertig
- Gebiet rechts der Wellenfront gehört L



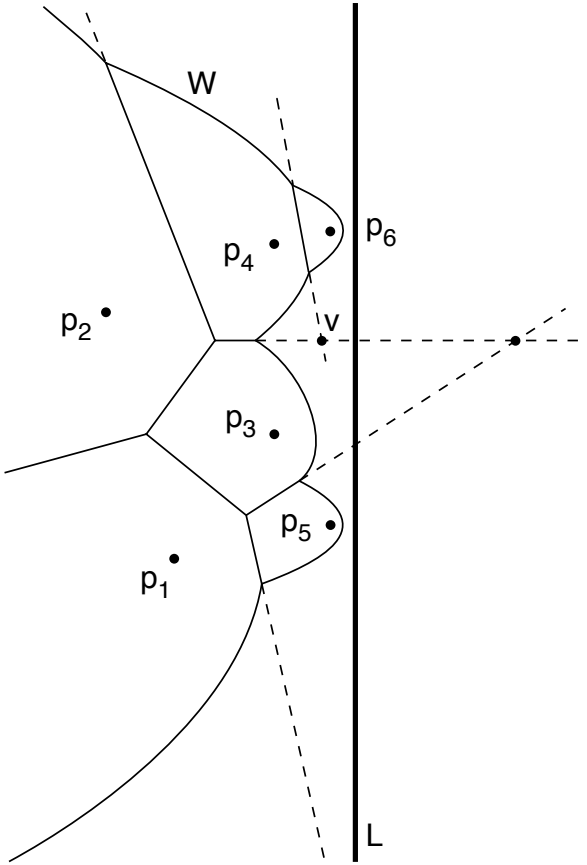
SSS: Wellenfront mit Verlängerungen

- Wellenfront: Mehrere Stücke $B(p_j, L)$ aneinandergereiht
- Links der Wellenfront: $VR(\{p_1, \ldots, p_{i-1})$ fertig
- Gebiet rechts der Wellenfront gehört L
- Zwischen den Parabeln: Spikes, Bisektoren zwischen Punkten

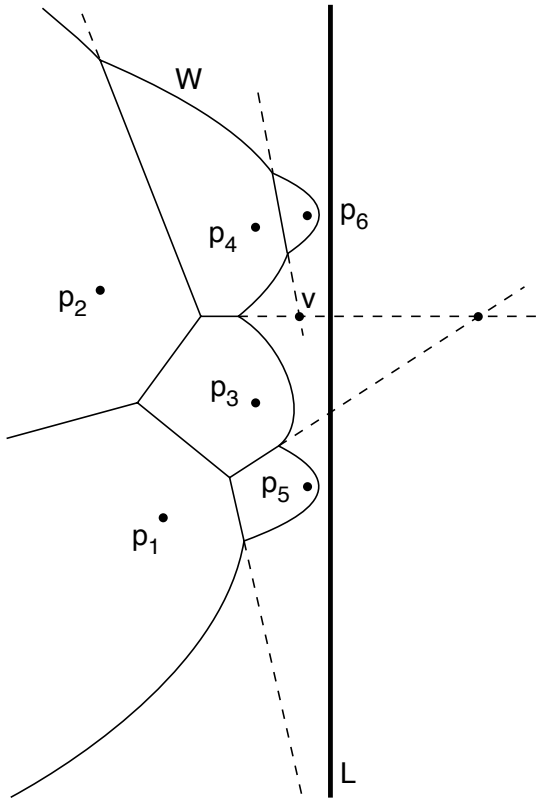


SSS: Wellenfront mit Verlängerungen

- Wellenfront: Mehrere Stücke $B(p_j, L)$ aneinandergereiht
- Links der Wellenfront: $VR(\{p_1, \ldots, p_{i-1})$ fertig
- Gebiet rechts der Wellenfront gehört L
- Zwischen den Parabeln: Spikes, Bisektoren zwischen Punkten
- Im weiteren Verlauf: Voronoi Knoten

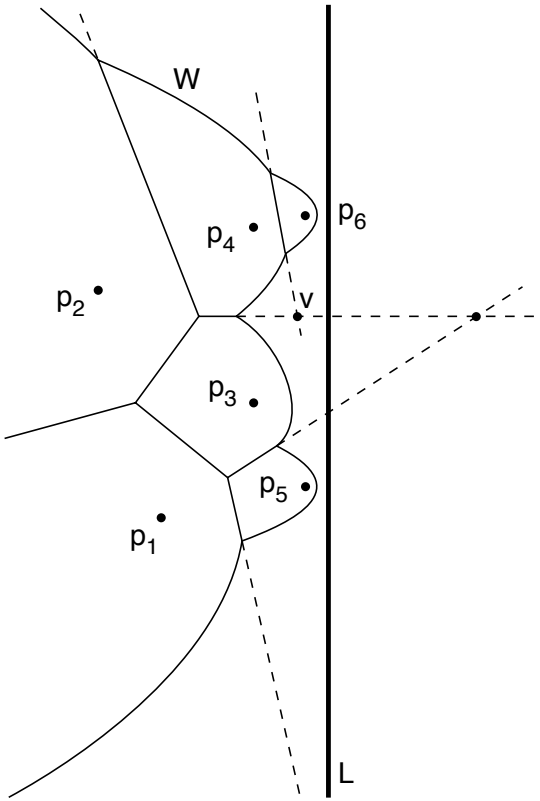


Formale Beschreibung SSS Wellenfront



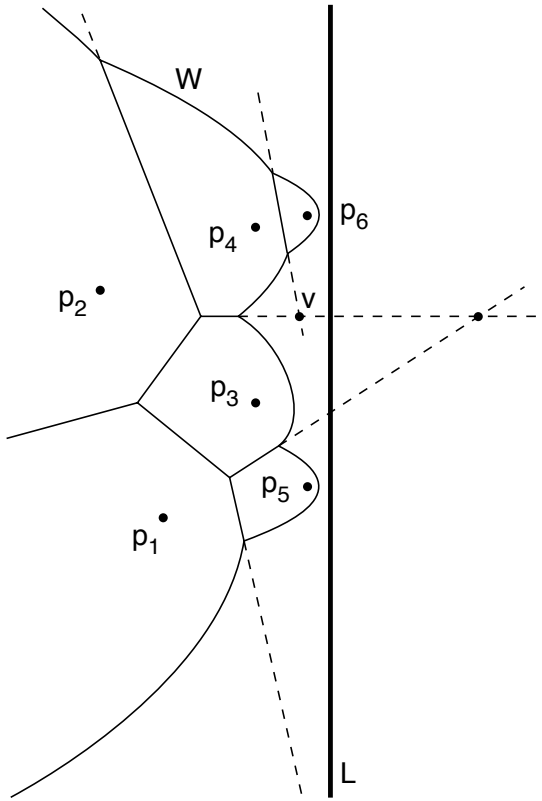
Formale Beschreibung SSS Wellenfront

- Parabelstücke geordnet nach Y-Richtung



Formale Beschreibung SSS Wellenfront

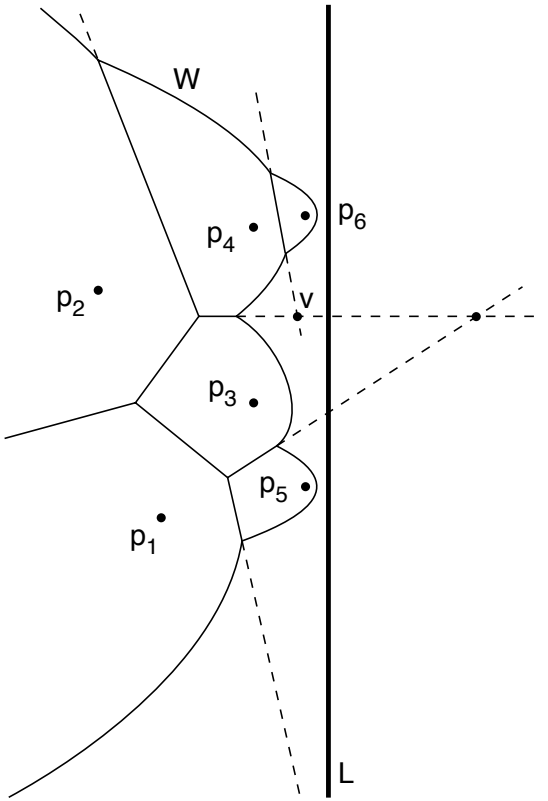
- Parabelstücke geordnet nach Y-Richtung
- Dazwischen Spikes geordnet nach Y-Richtung



Formale Beschreibung SSS Wellenfront

- Parabelstücke geordnet nach Y-Richtung
- Dazwischen Spikes geordnet nach Y-Richtung
- Geordnet nach Punkten:

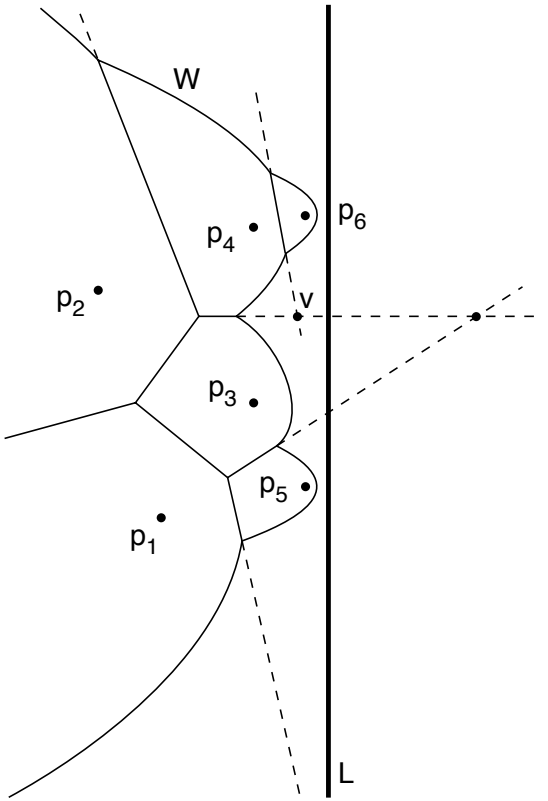
  $p_1, p_5, p_3, p_4, p_6, p_4, p_2$



Formale Beschreibung SSS Wellenfront

- Parabelstücke geordnet nach Y-Richtung
- Dazwischen Spikes geordnet nach Y-Richtung
- Geordnet nach Punkten:

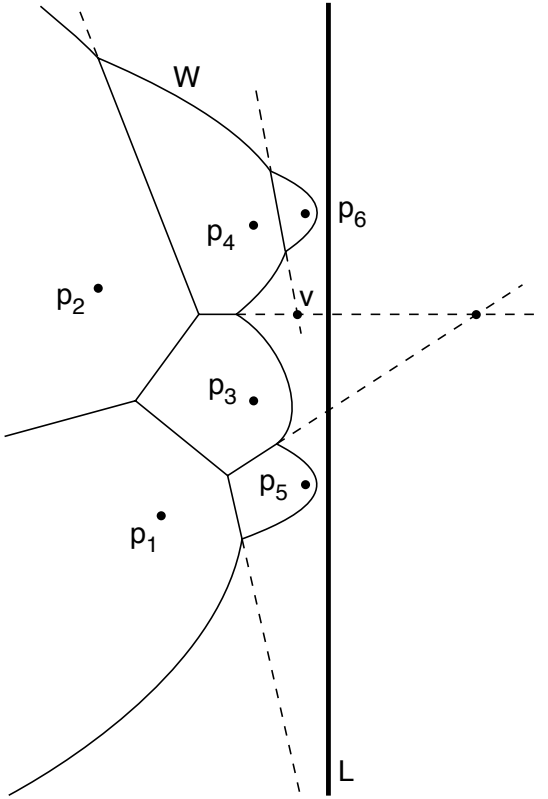
  $p_1, p_5, p_3, p_4, p_6, p_4, p_2$
- Abgespeichert in einem Baum



Formale Beschreibung SSS Wellenfront

- Parabelstücke geordnet nach Y-Richtung
- Dazwischen Spikes geordnet nach Y-Richtung
- Geordnet nach Punkten:

  $p_1, p_5, p_3, p_4, p_6, p_4, p_2$
- Abgespeichert in einem Baum
- Schlüssel (Parabeln, Spikes) Y-Koordinate



Wellenfront ist zusammenhängend!

Wellenfront ist zusammenhängend!

Lemma 6.10 Die Wellenfront ist zusammenhängend und Y-monoton.

Wellenfront ist zusammenhängend!

Lemma 6.10 Die Wellenfront ist zusammenhängend und Y-monoton.

Beweis: Eigenschaften der Parabeln!

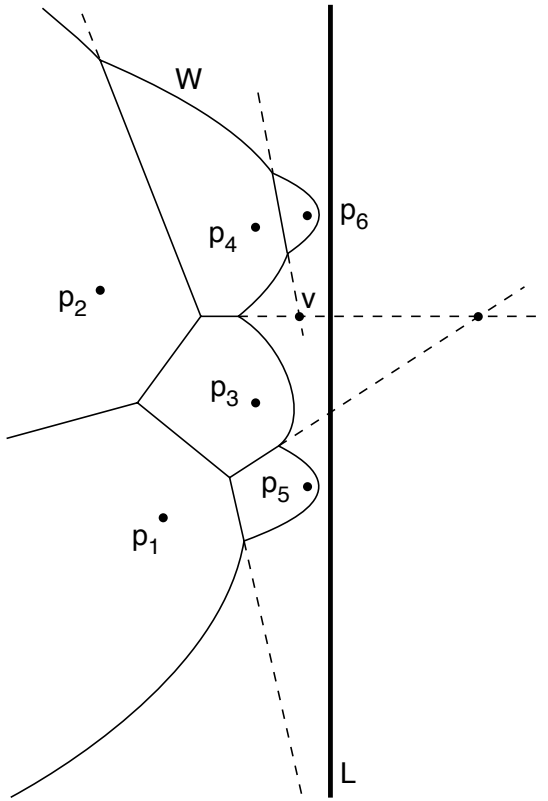
Wellenfront ist zusammenhängend!

Lemma 6.10 Die Wellenfront ist zusammenhängend und Y-monoton.

Beweis: Eigenschaften der Parabeln!

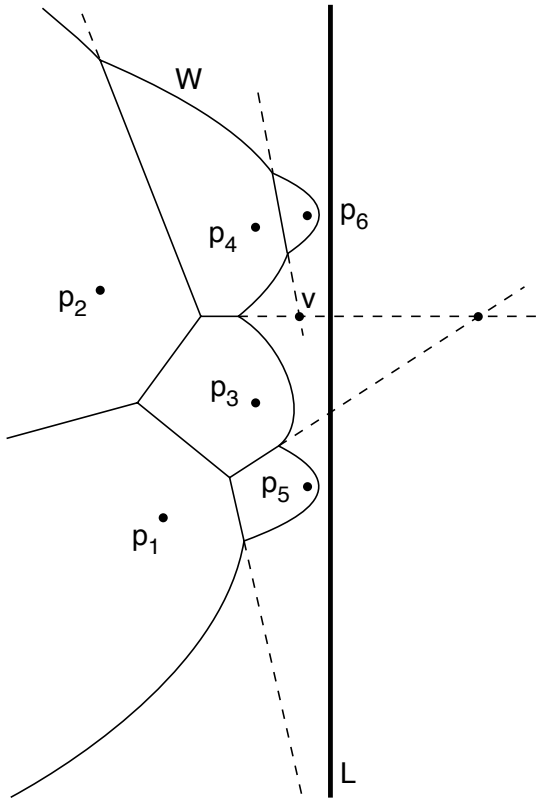
Datenstruktur beachten!

Ereignisse des Sweeps



Ereignisse des Sweeps

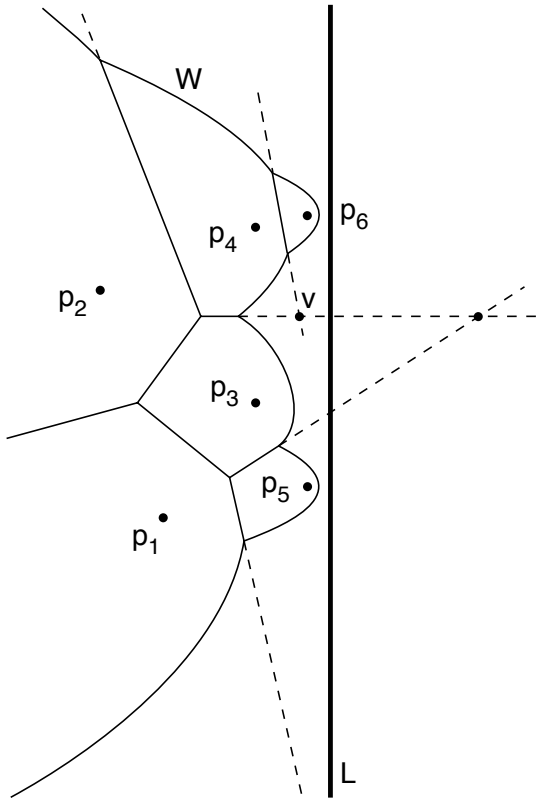
Wann verändert sich die Wellenfront?
Nur diskrete Zeitpunkte!



Ereignisse des Sweeps

Wann verändert sich die Wellenfront?
Nur diskrete Zeitpunkte!

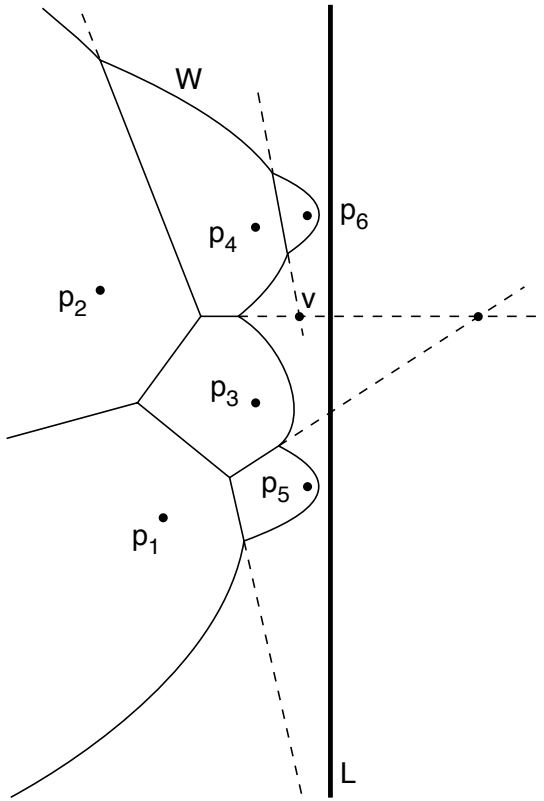
1. Parabelstück kommt hinzu: Ein neuer Punkt wird von L getroffen!
   Punkt-Ereignis,



Ereignisse des Sweeps

Wann verändert sich die Wellenfront?
Nur diskrete Zeitpunkte!

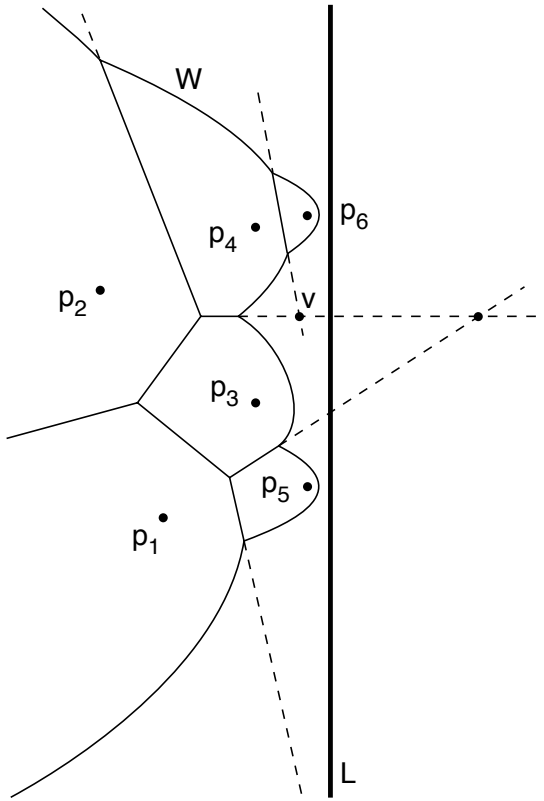
1. Parabelstück kommt hinzu: Ein neuer Punkt wird von L getroffen!
   Punkt-Ereignis, neue Parabel entsteht



Ereignisse des Sweeps

Wann verändert sich die Wellenfront?
Nur diskrete Zeitpunkte!

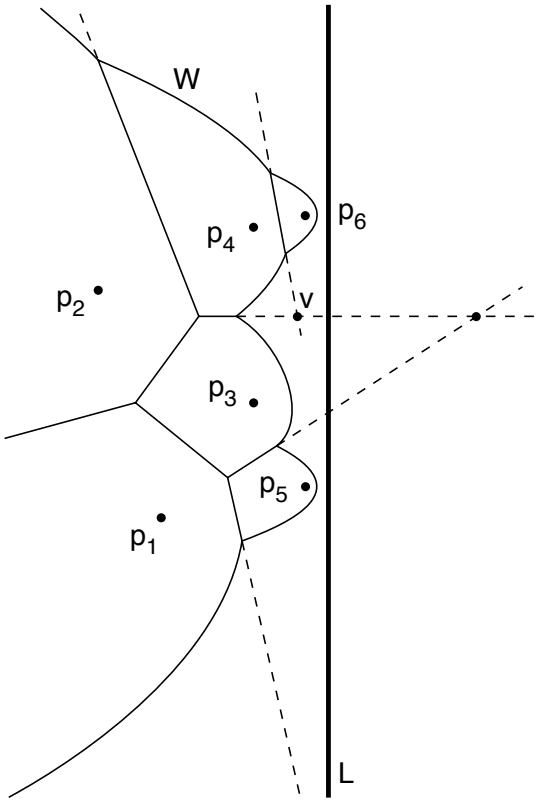
1. Parabelstück kommt hinzu: Ein neuer Punkt wird von L getroffen!
   Punkt-Ereignis, neue Parabel entsteht
2. Parabelstück verschwindet: Die *Welle* erreicht den Schnittpunkt zweier Spikes, Spike-Ereignis,



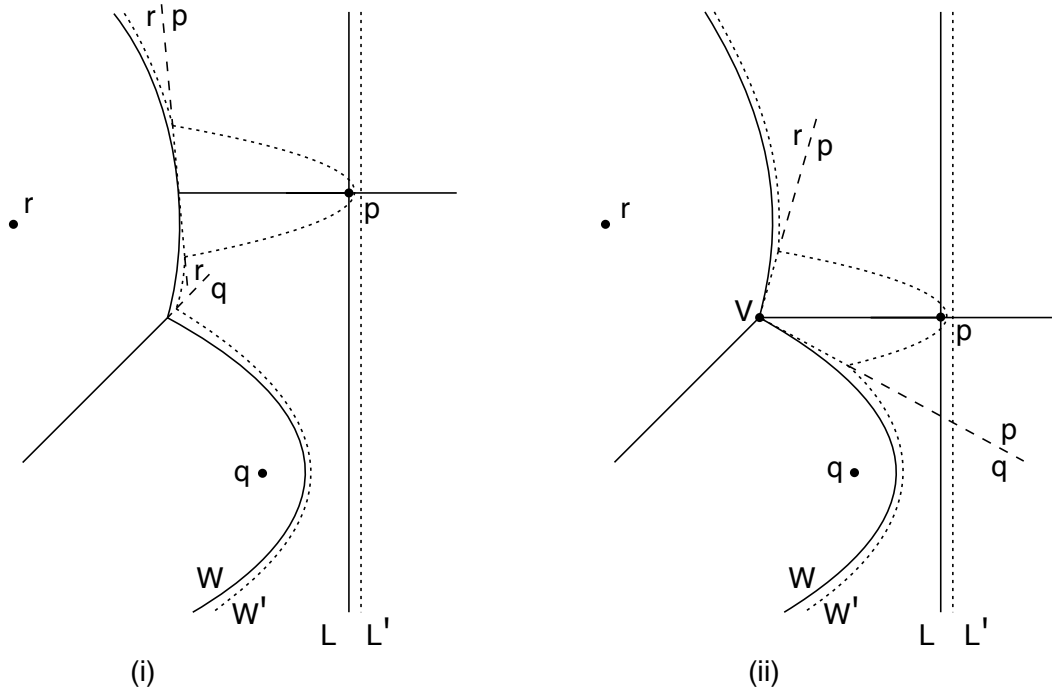
Ereignisse des Sweeps

Wann verändert sich die Wellenfront?
Nur diskrete Zeitpunkte!

1. Parabelstück kommt hinzu: Ein neuer Punkt wird von L getroffen!
   Punkt-Ereignis, neue Parabel entsteht
2. Parabelstück verschwindet: Die *Welle* erreicht den Schnittpunkt zweier Spikes, Spike-Ereignis, Voronoi-Knoten entsteht

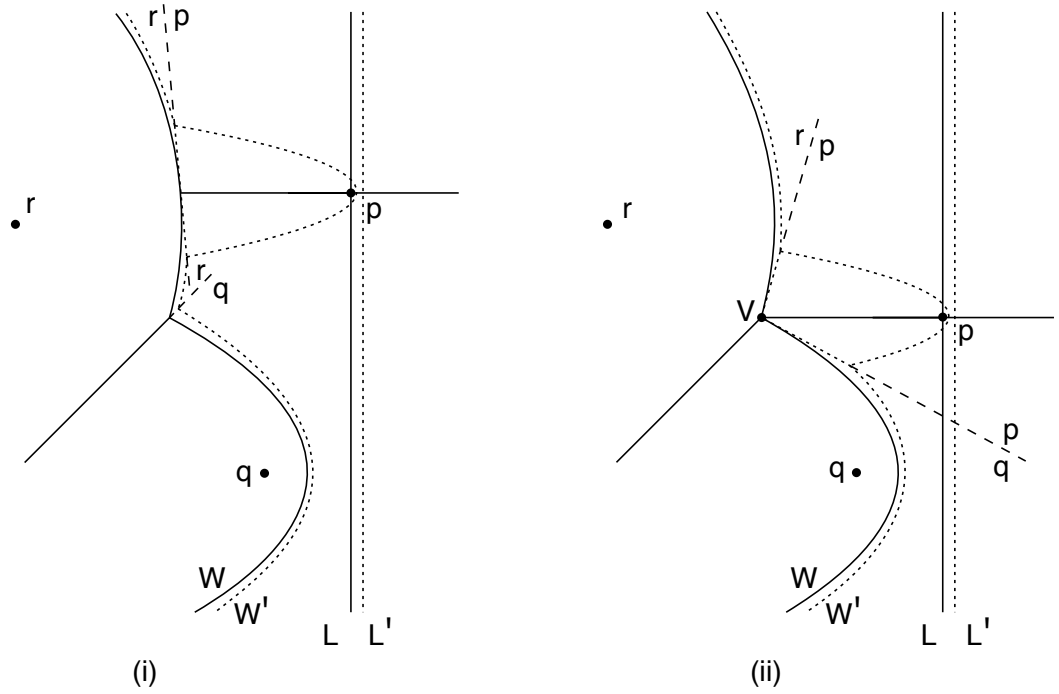


Behandlung: Punkt-Ereignis, SSS akt.



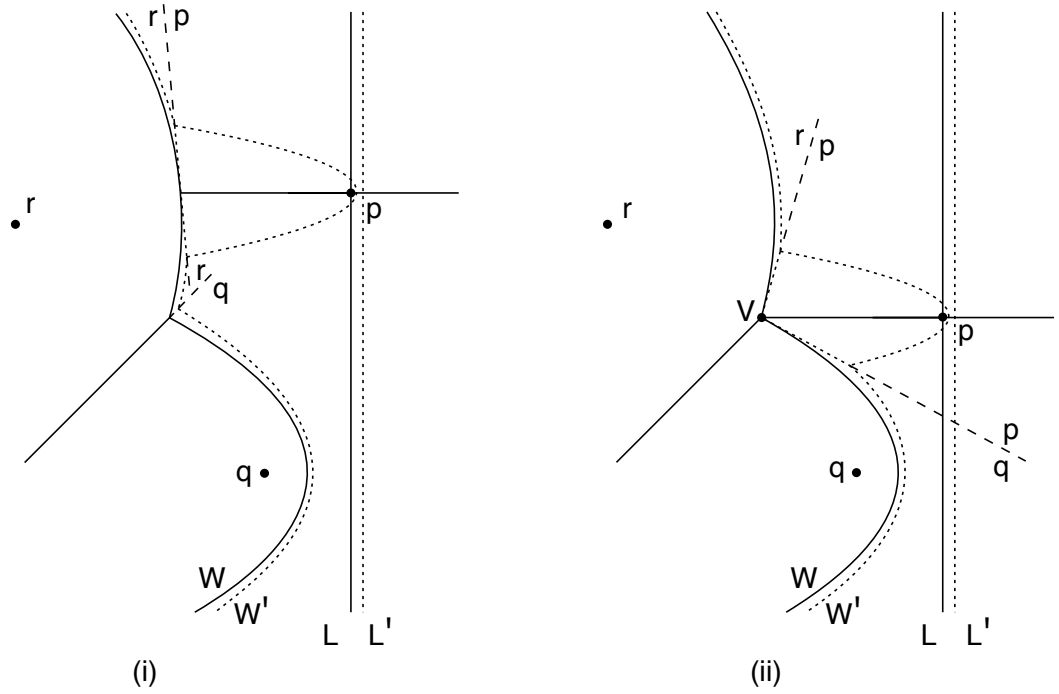
Behandlung: Punkt-Ereignis, SSS akt.

- Neue Parabel einfügen,



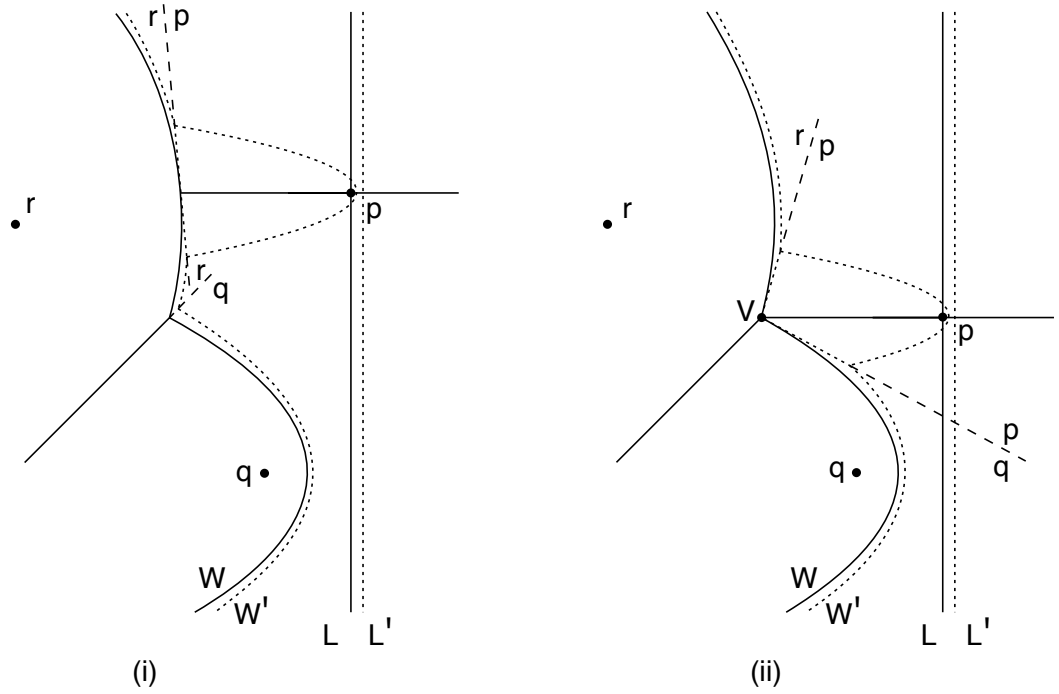
Behandlung: Punkt-Ereignis, SSS akt.

- Neue Parabel einfügen, neue Spikes entstehen



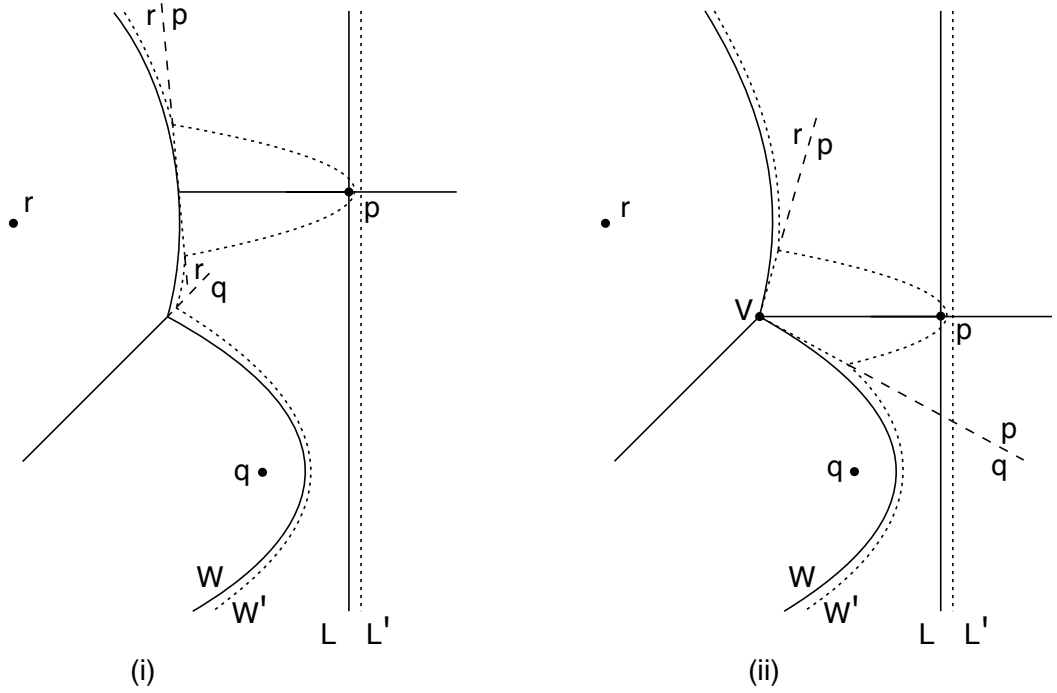
Behandlung: Punkt-Ereignis, SSS akt.

- Neue Parabel einfügen, neue Spikes entstehen
- Zwei Fälle: Innerhalb Parabelstück, Schnittpunkt von Spikes

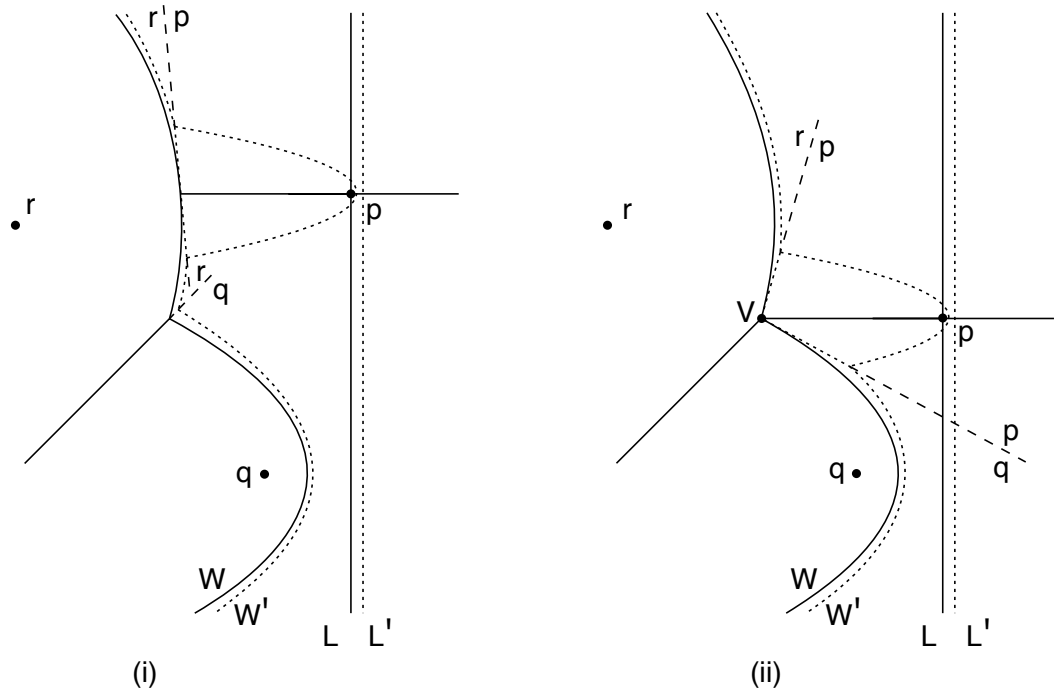


Behandlung: Punkt-Ereignis, SSS akt.

- Neue Parabel einfügen, neue Spikes entstehen
- Zwei Fälle: Innerhalb Parabelstück, Schnittpunkt von Spikes
- Zwei neue Spikes, ggf. neuer Voronoi Knoten

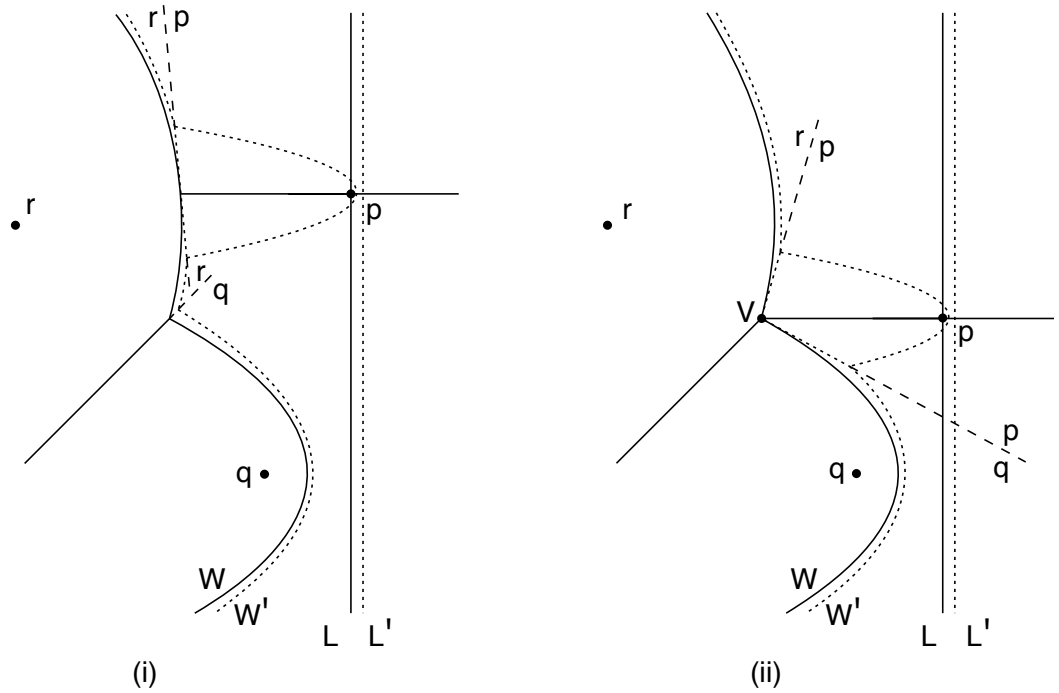


Behandlung: Punkt-Ereignis, SSS akt.



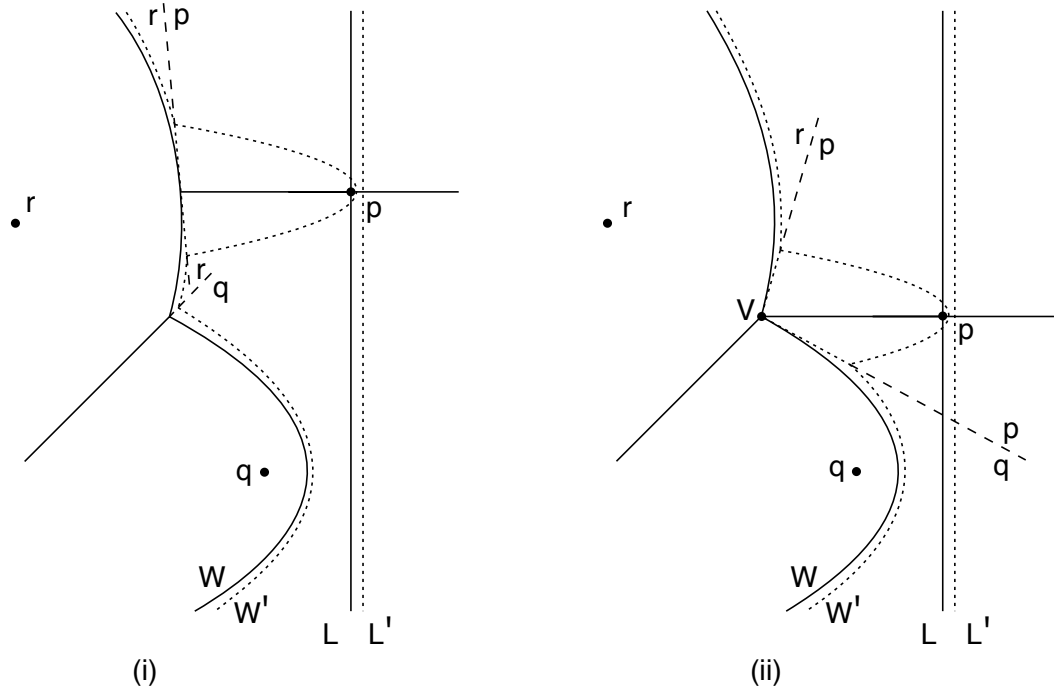
Behandlung: Punkt-Ereignis, SSS akt.

- Zwei neue Spikes, ggf. neuer Voronoi Knoten



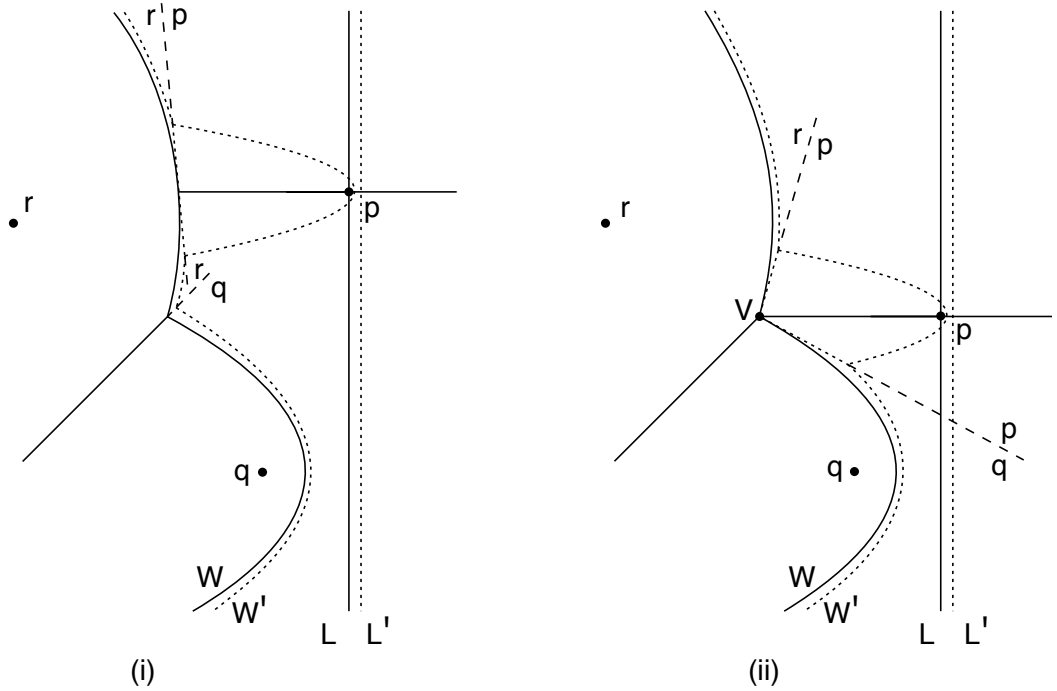
Behandlung: Punkt-Ereignis, SSS akt.

- Zwei neue Spikes, ggf. neuer Voronoi Knoten
- Parabelstücke (Spikestart) sortiert im Baum:
  Schlüssel $B(L, p_j)$, Y-Koordinate Segmente, Dyn.



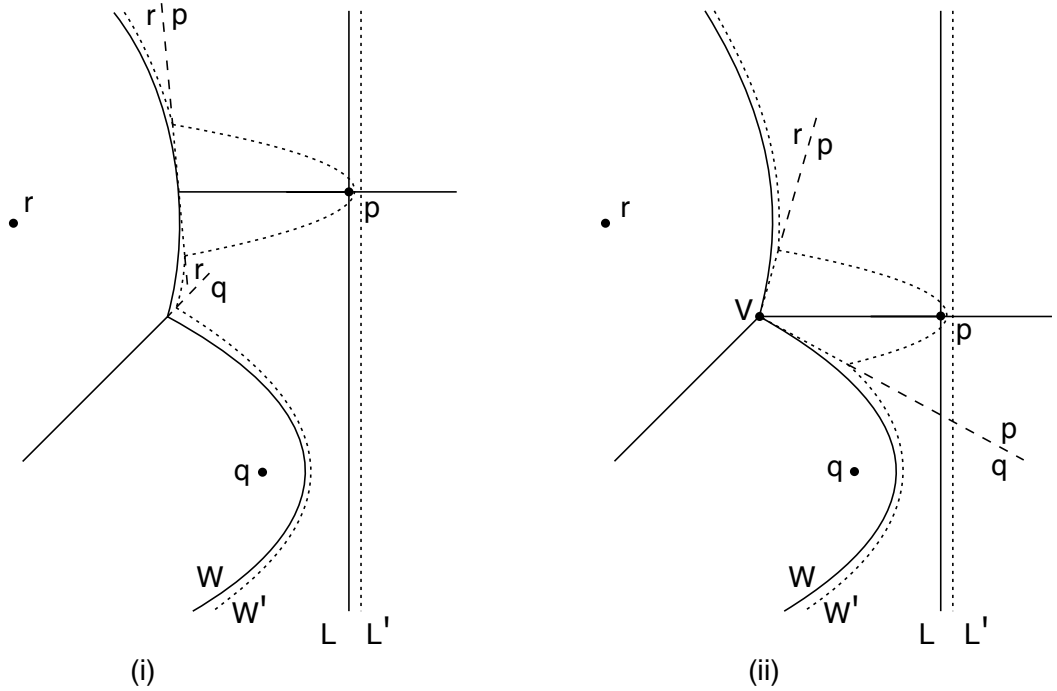
Behandlung: Punkt-Ereignis, SSS akt.

- Zwei neue Spikes, ggf. neuer Voronoi Knoten
- Parabelstücke (Spikestart) sortiert im Baum:
  Schlüssel $B(L, p_j)$, Y-Koordinate Segmente, Dyn.
- Zugriff $O(\log m)$, m Anzahl der Parabeln/Spikes,



Behandlung: Punkt-Ereignis, SSS akt.

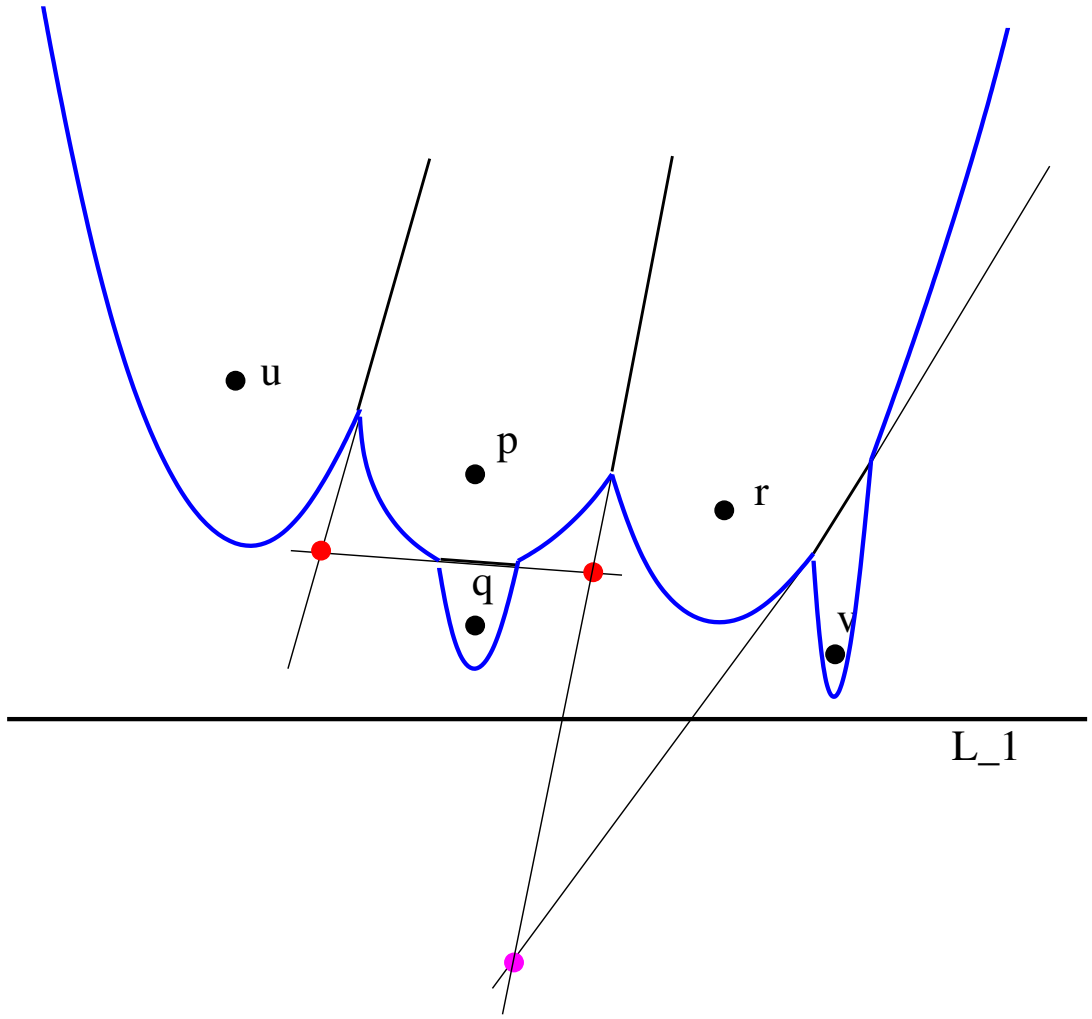
- Zwei neue Spikes, ggf. neuer Voronoi Knoten
- Parabelstücke (Spikestart) sortiert im Baum:
  Schlüssel $B(L, p_j)$, Y-Koordinate Segmente, Dyn.
- Zugriff $O(\log m)$, m Anzahl der Parabeln/Spikes, Akt.: $O(\log m)$



Behandlung: Spike-Ereignis, SSS-akt

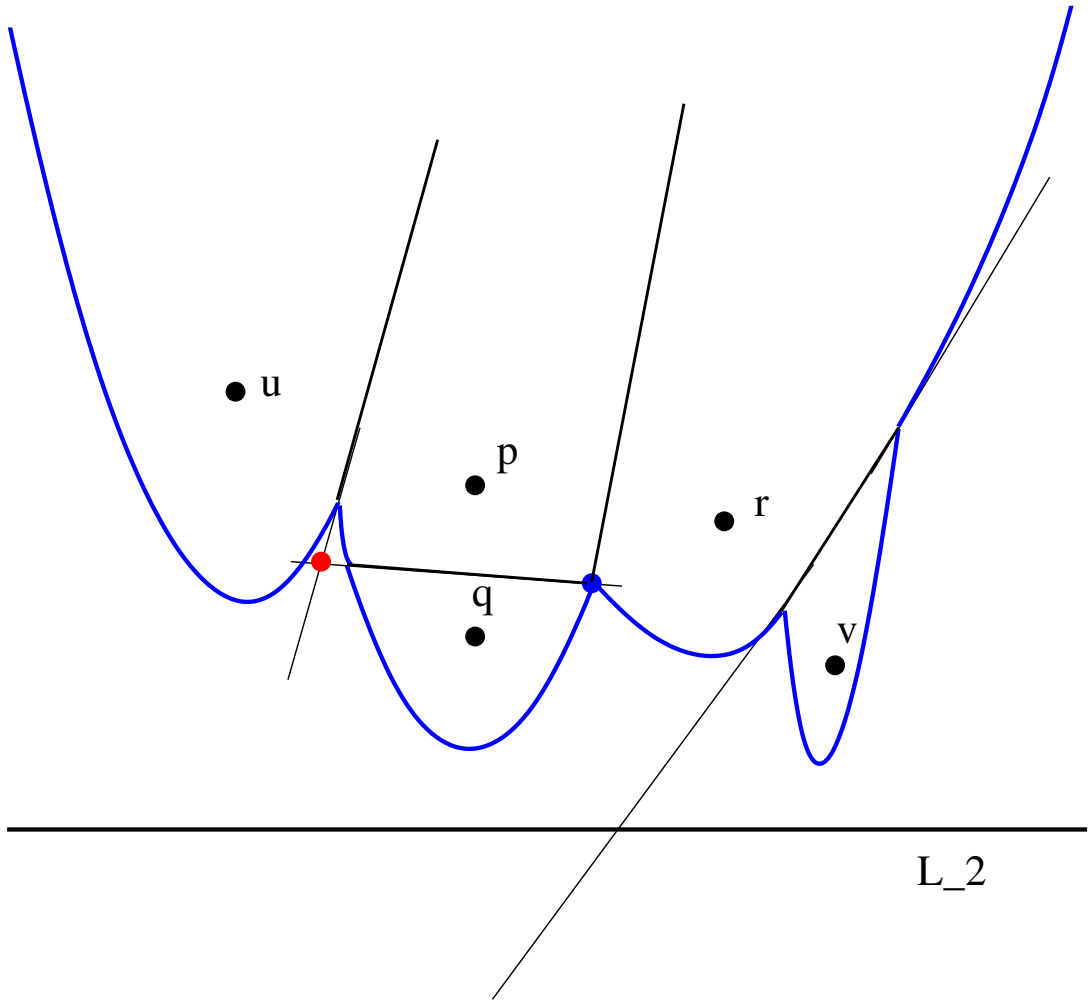
Behandlung: Spike-Ereignis, SSS-akt

- Parabelst. verschwindet, Spikes schneiden sich, Voronoi-Knoten $O(n)$ viele



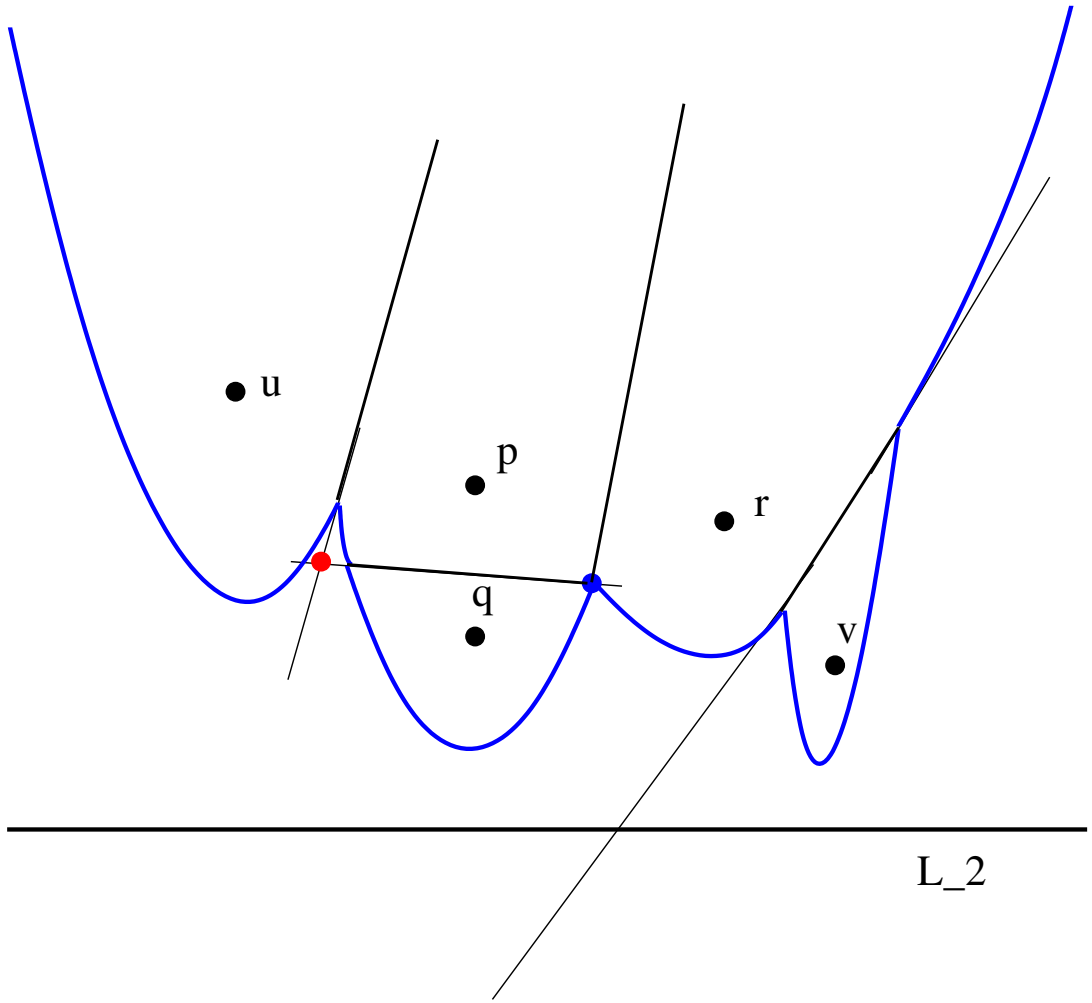
Behandlung: Spike-Ereignis, SSS-akt

- Parabelst. verschwindet, Spikes schneiden sich, Voronoi-Knoten $O(n)$ viele
- Zeitpunkt des Ereignisses (Position L_2) weiter zurück



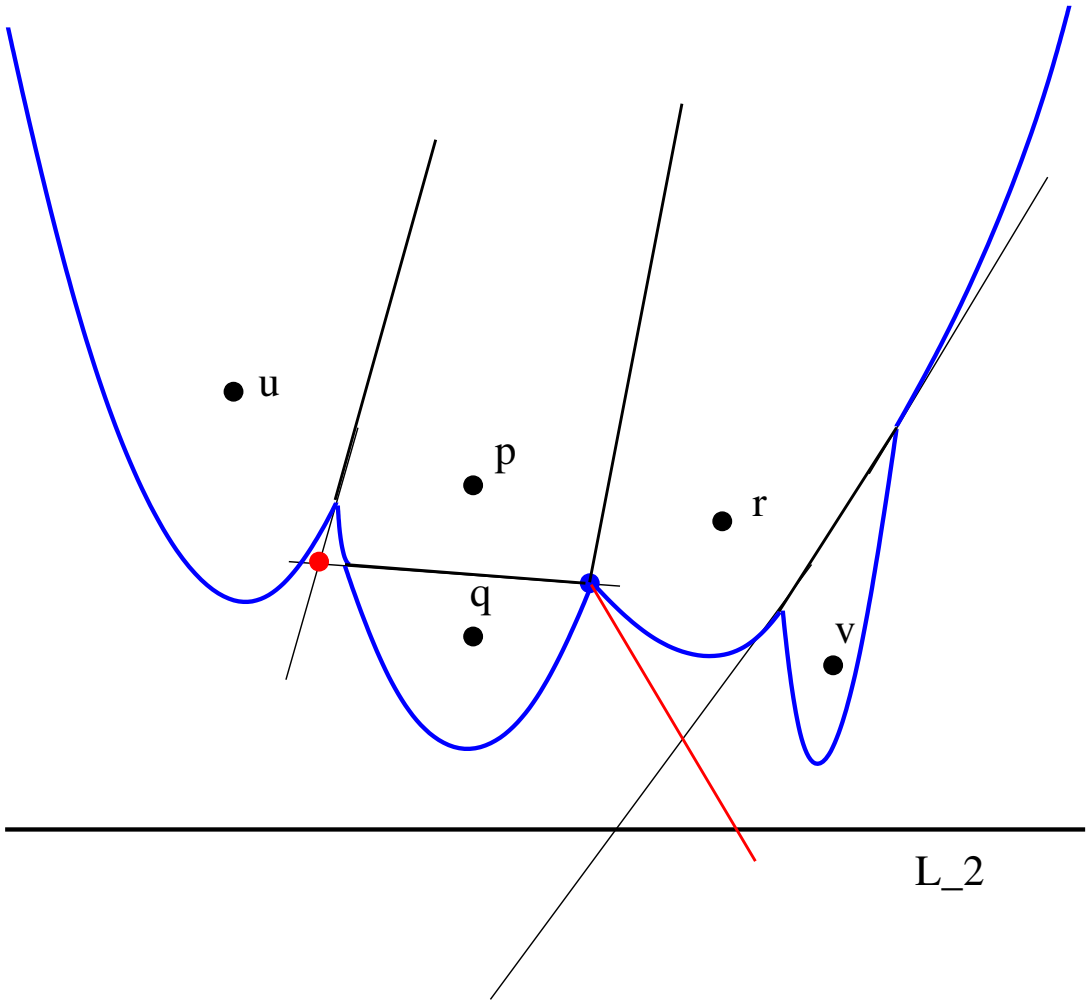
Behandlung: Spike-Ereignis, SSS-akt

- Parabelst. verschwindet, Spikes schneiden sich, Voronoi-Knoten $O(n)$ viele
- Zeitpunkt des Ereignisses (Position L_2) weiter zurück
- Parabel $B(L, p_j)$ entfernen: $O(\log m)$



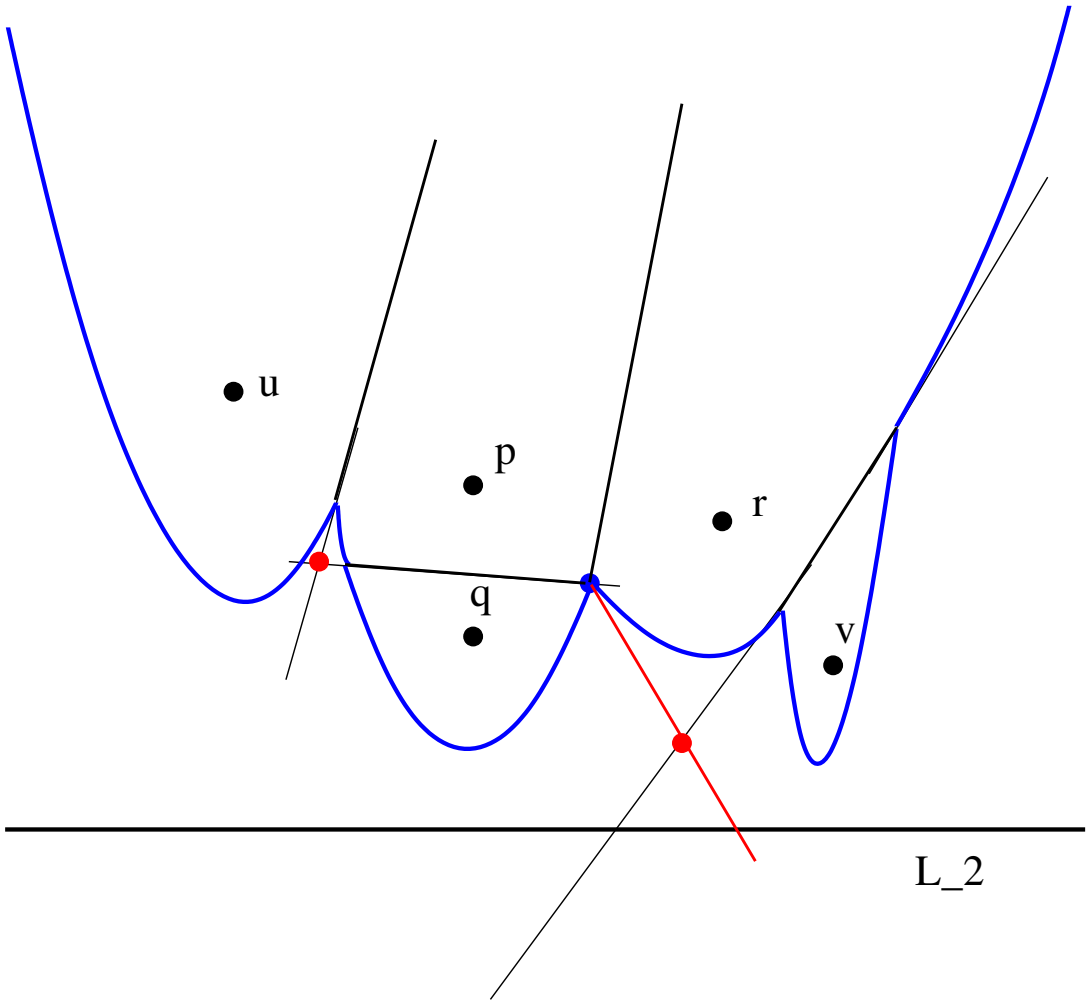
Behandlung: Spike-Ereignis, SSS-akt

- Parabelst. verschwindet, Spikes schneiden sich, Voronoi-Knoten $O(n)$ viele
- Zeitpunkt des Ereignisses (Position L_2) weiter zurück
- Parabel $B(L, p_j)$ entfernen: $O(\log m)$
- Ein neuer Spike zwischen neuen Nachbarn: $O(1)$

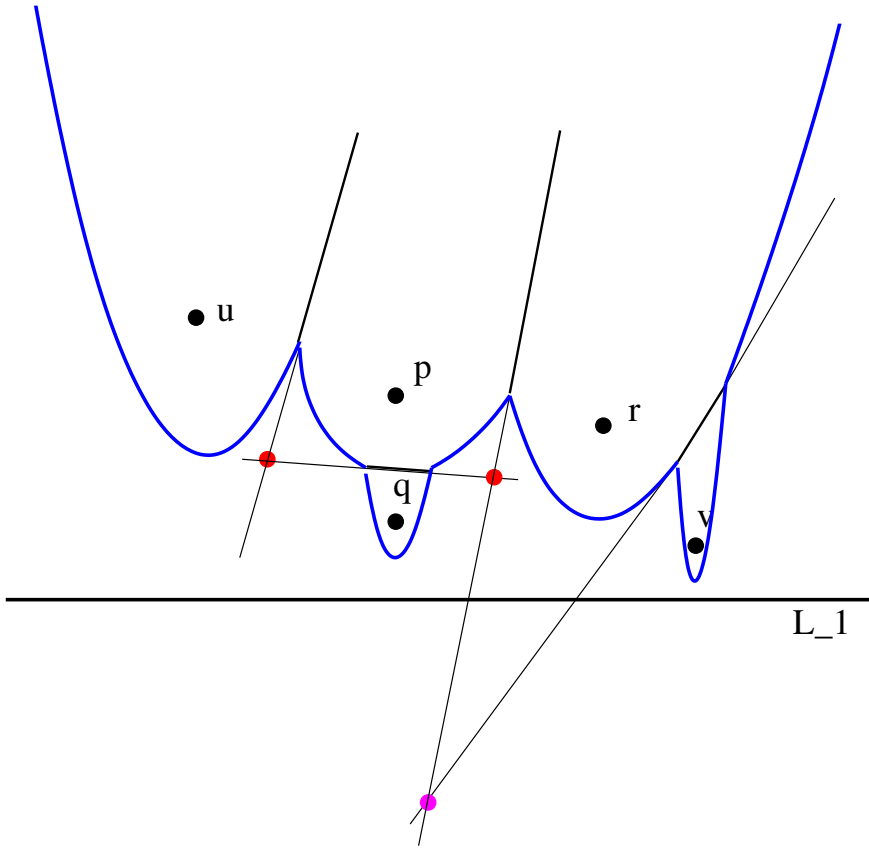


Behandlung: Spike-Ereignis, SSS-akt

- Parabelst. verschwindet, Spikes schneiden sich, Voronoi-Knoten $O(n)$ viele
- Zeitpunkt des Ereignisses (Position L_2) weiter zurück
- Parabel $B(L, p_j)$ entfernen: $O(\log m)$
- Ein neuer Spike zwischen neuen Nachbarn: $O(1)$
- Vorderster Schnittpunkt, nur Nachbarn

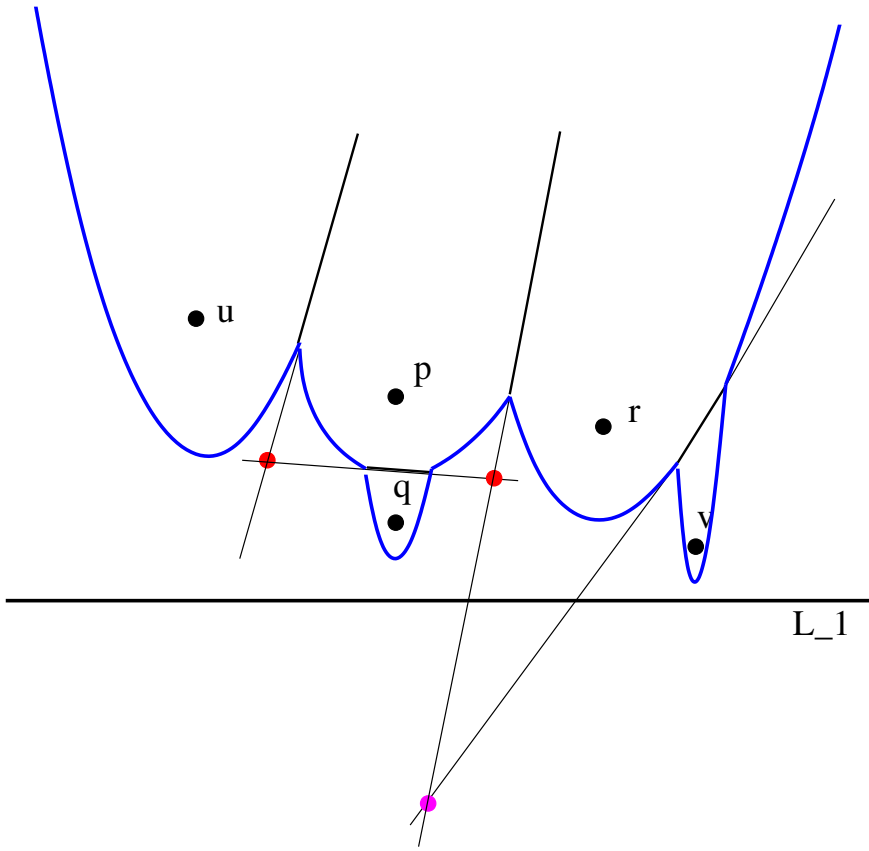


Ereignisstruktur akualisieren: m Parabeln



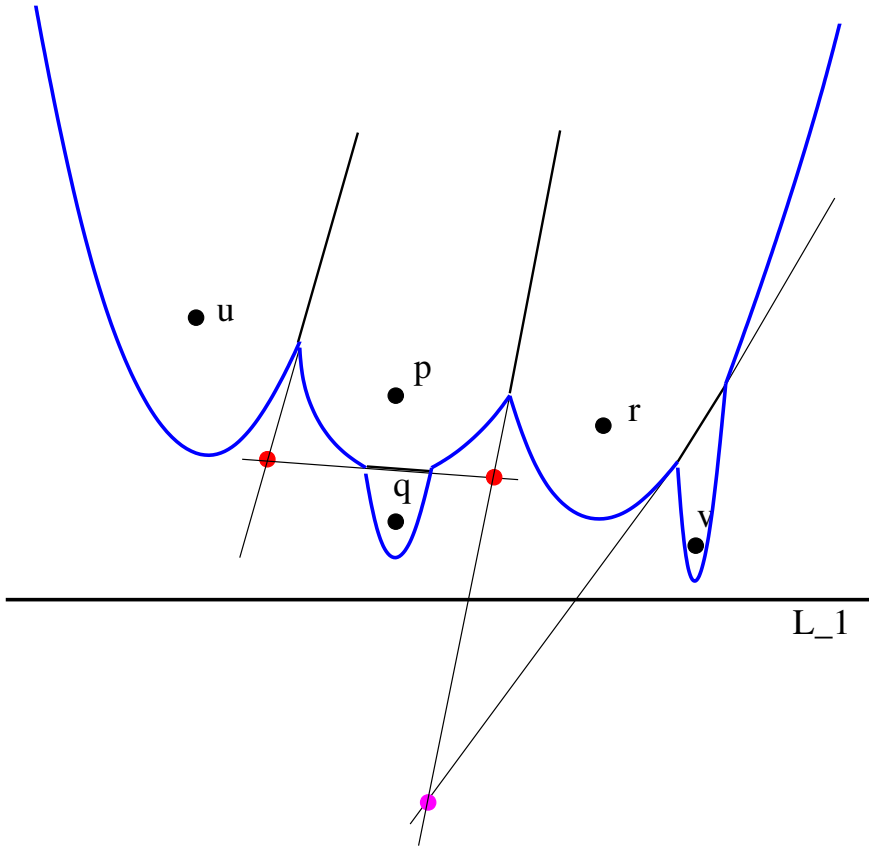
Ereignisstruktur akualisieren: m Parabeln

- Priority Queue,
  $O(n)$ Punktereignisse,
  sortieren: $O(n \log n)$



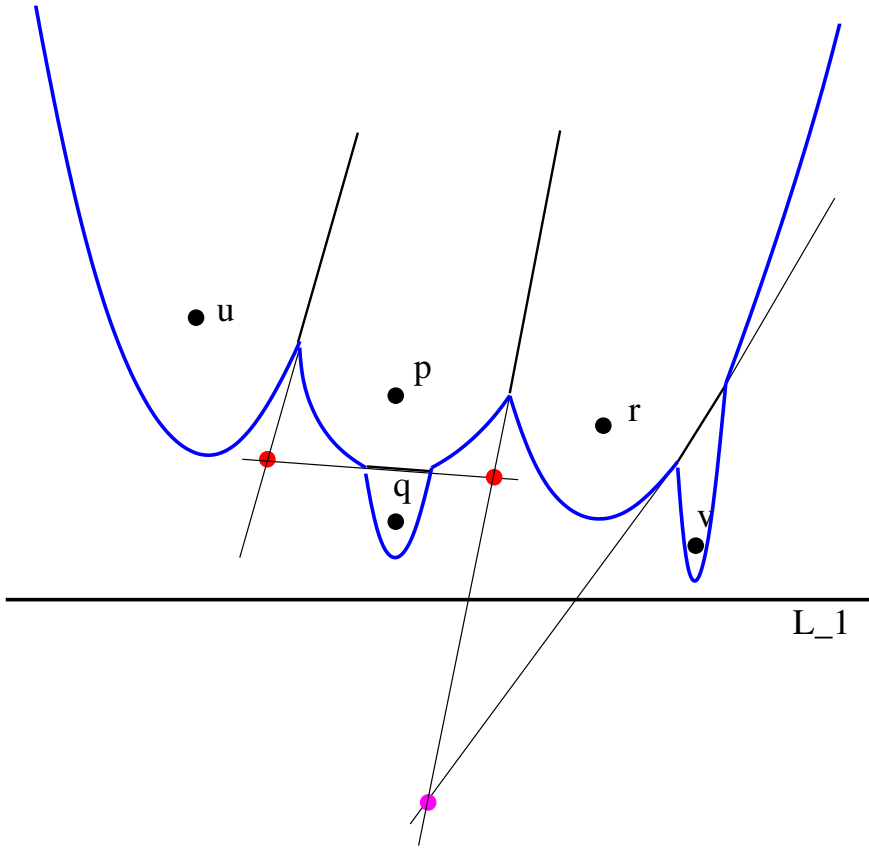
Ereignisstruktur akualisieren: m Parabeln

- Priority Queue,
  $O(n)$ Punktereignisse,
  sortieren: $O(n \log n)$
- Spike-Ereignisse (Voro-Knoten):
  Schnittpunkte der Spikes,
  jeweils die vordersten!



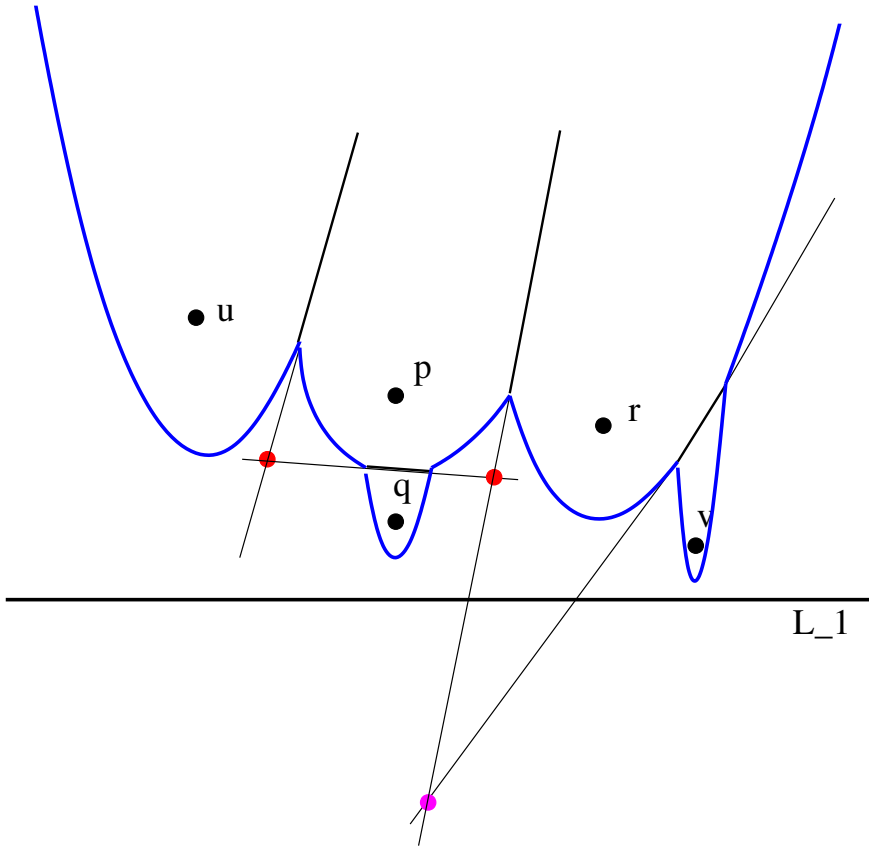
Ereignisstruktur akualisieren: m Parabeln

- Priority Queue,
  $O(n)$ Punktereignisse,
  sortieren: $O(n \log n)$
- Spike-Ereignisse (Voro-Knoten):
  Schnittpunkte der Spikes,
  jeweils die vordersten!
- Wie Segment-Schnitt-Sweep:
  Schnitte nur mit Nachbarn,
  nachgelagerte Events



Ereignisstruktur akualisieren: m Parabeln

- Priority Queue,
  $O(n)$ Punktereignisse,
  sortieren: $O(n \log n)$
- Spike-Ereignisse (Voro-Knoten):
  Schnittpunkte der Spikes,
  jeweils die vordersten!
- Wie Segment-Schnitt-Sweep:
  Schnitte nur mit Nachbarn,
  nachgelagerte Events
- Stets nur ersten Schnitt
  einfügen, andere entfernen,
  Anzahl: $O(m)$, Einf./Entf. $O(\log m)$



Gesamtlaufzeit/Gesamtkomplexität

Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS

Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES

Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES
- Maximal $O(n)$ Ereignisse finden statt: Punktereign./Voronoi-Knoten

Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES
- Maximal $O(n)$ Ereignisse finden statt:
  Punktereign./Voronoi-Knoten
- Einfügen/Entfernen Events, Einfügen/Entfernen Parabeln

Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES
- Maximal $O(n)$ Ereignisse finden statt:
  Punktereign./Voronoi-Knoten
- Einfügen/Entfernen Events, Einfügen/Entfernen Parabeln
- n-mal $O(\log(n+m))$ Laufzeit!

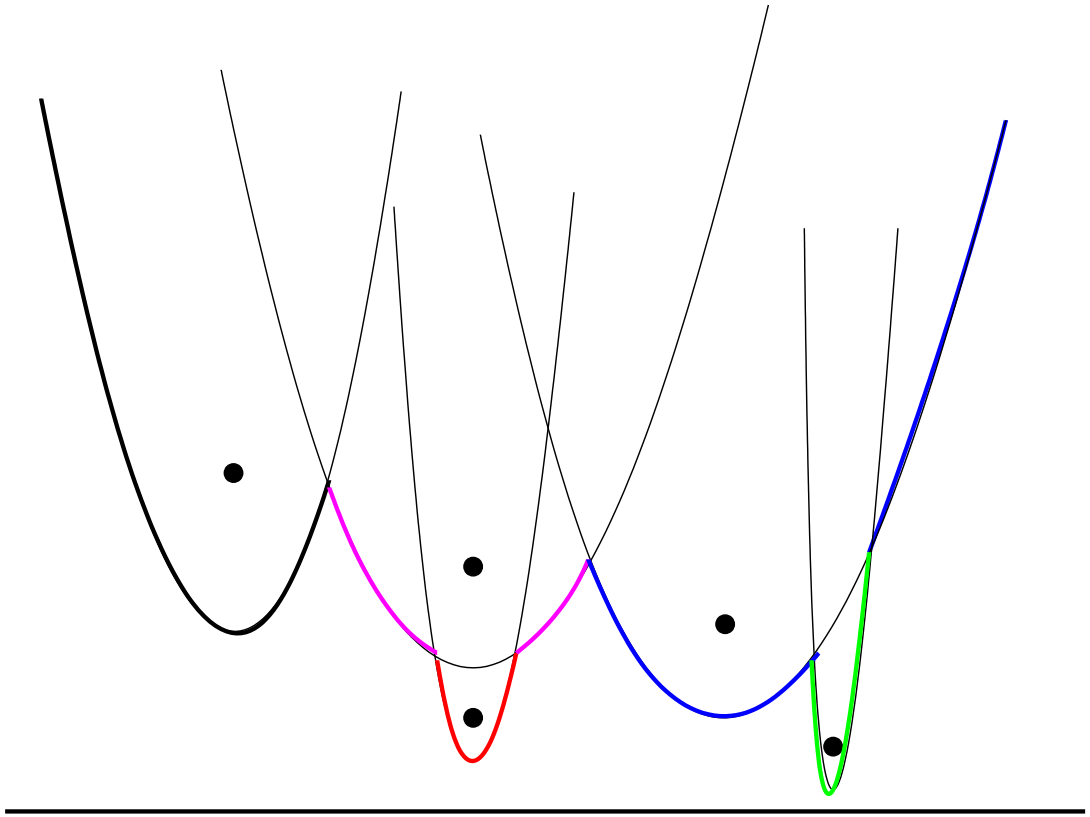
Gesamtlaufzeit/Gesamtkomplexität

- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES
- Maximal $O(n)$ Ereignisse finden statt: Punktereign./Voronoi-Knoten
- Einfügen/Entfernen Events, Einfügen/Entfernen Parabeln
- n-mal $O(\log(n+m))$ Laufzeit!
- Komplexität der Wellenfront: $m \in O(n)$

Gesamtlaufzeit/Gesamtkomplexität

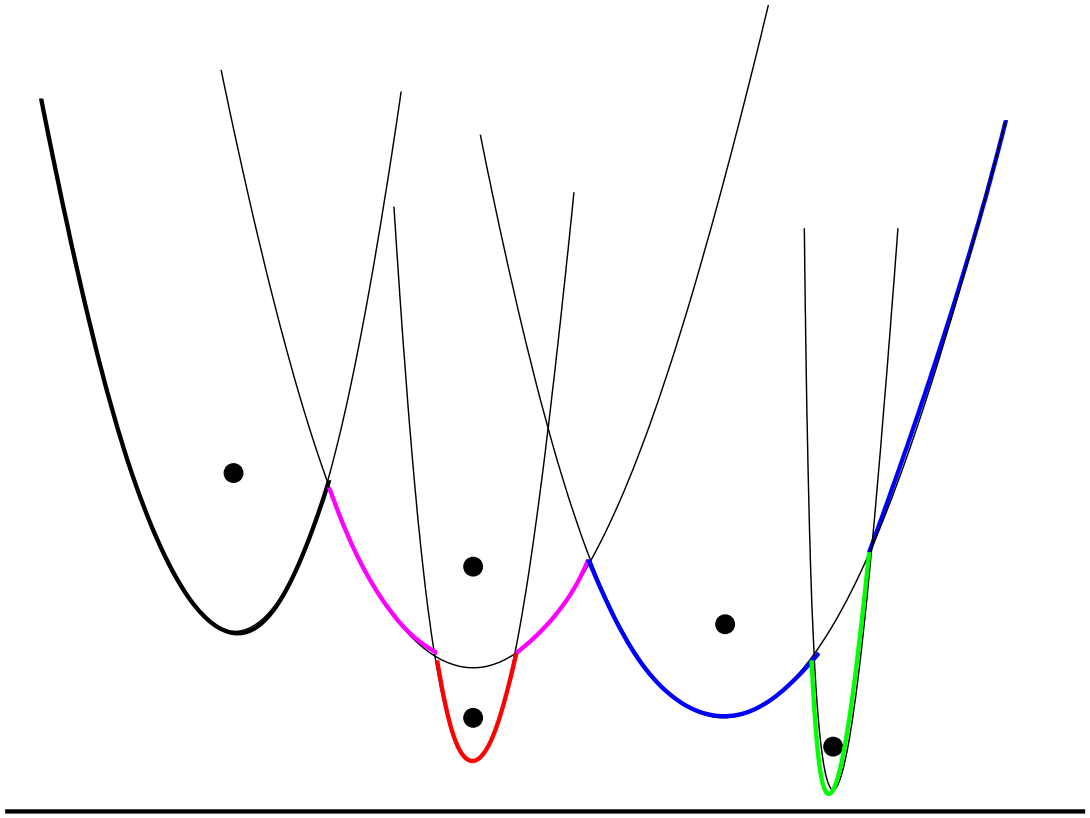
- Max. m Parabeln in SSS
- Max. $O(n+m)$ Ereignisse in ES
- Maximal $O(n)$ Ereignisse finden statt: Punktereign./Voronoi-Knoten
- Einfügen/Entfernen Events, Einfügen/Entfernen Parabeln
- n-mal $O(\log(n+m))$ Laufzeit!
- Komplexität der Wellenfront: $m \in O(n)$
- Begründung!

Komplexität der Wellenfront



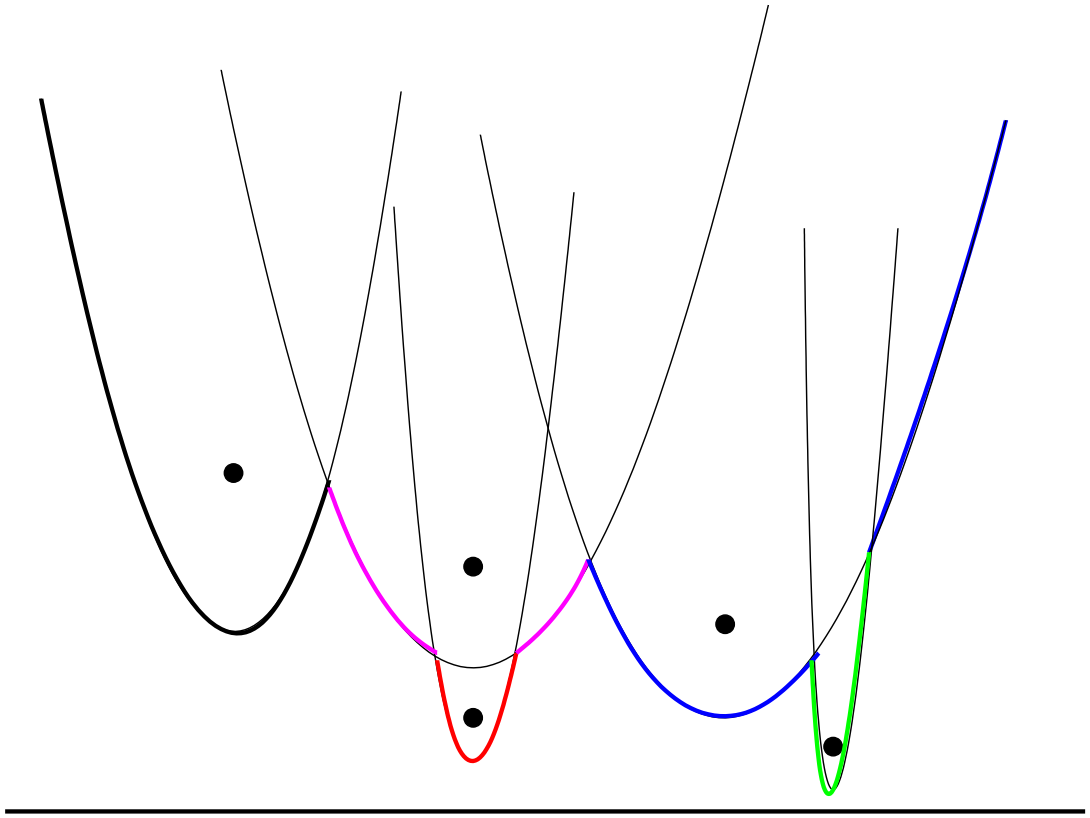
Komplexität der Wellenfront

- n Parabeln, von denen sich zwei maximal zweimal schneiden können



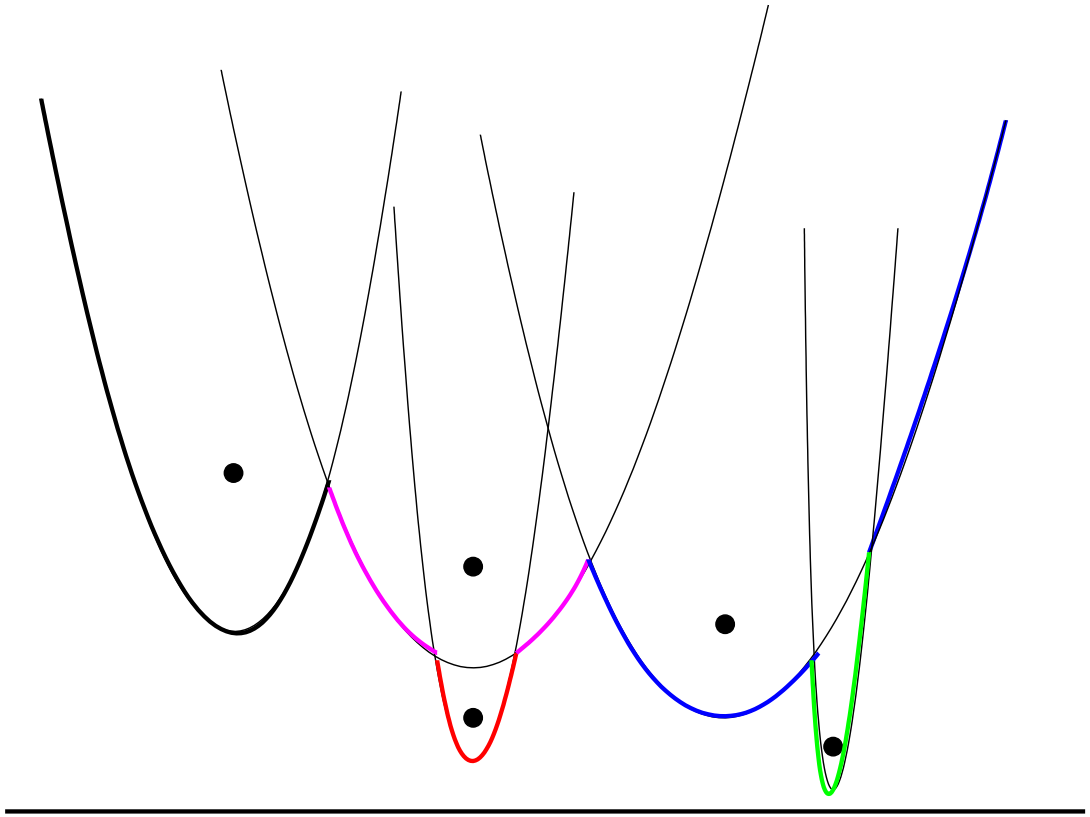
Komplexität der Wellenfront

- n Parabeln, von denen sich zwei maximal zweimal schneiden können
- Definiert auf dem gesamten Intervall



Komplexität der Wellenfront

- n Parabeln, von denen sich zwei maximal zweimal schneiden können
- Definiert auf dem gesamten Intervall
- DSS von n Buchstaben der Ordnung 2: $\lambda_2(n) \in O(n)$



Ergebnis

Ergebnis

Theorem 6.11 Das Voronoi Diagramm von n Punkten läßt sich mit dem Sweep Algorithmus in Zeit $O(n \log n)$ und mit Platz $O(n)$ berechnen, das ist optimal.

Ergebnis

Theorem 6.11 Das Voronoi Diagramm von n Punkten läßt sich mit dem Sweep Algorithmus in Zeit $O(n \log n)$ und mit Platz $O(n)$ berechnen, das ist optimal.

Beweis: $O(n)$ Events in je $O(\log n)$ Zeit

Ergebnis

Theorem 6.11 Das Voronoi Diagramm von n Punkten läßt sich mit dem Sweep Algorithmus in Zeit $O(n \log n)$ und mit Platz $O(n)$ berechnen, das ist optimal.

Beweis: $O(n)$ Events in je $O(\log n)$ Zeit

Korrektheit: Algorithmus schreibt Voronoi-Diagramm links der Wellenfront *in den Sand*

Voronoi Diagramm von Liniensegmenten

Voronoi Diagramm von Liniensegmenten

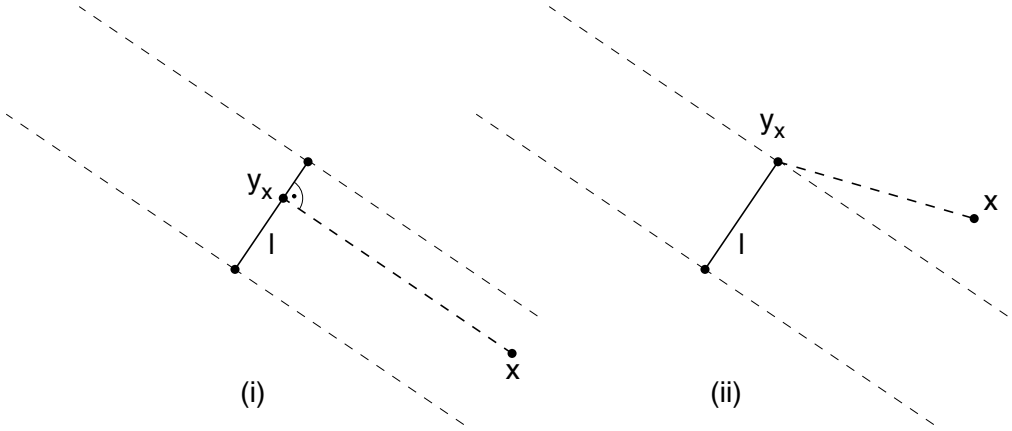
- Jetzt alle Objekte Punkte oder Liniensegmente

Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment

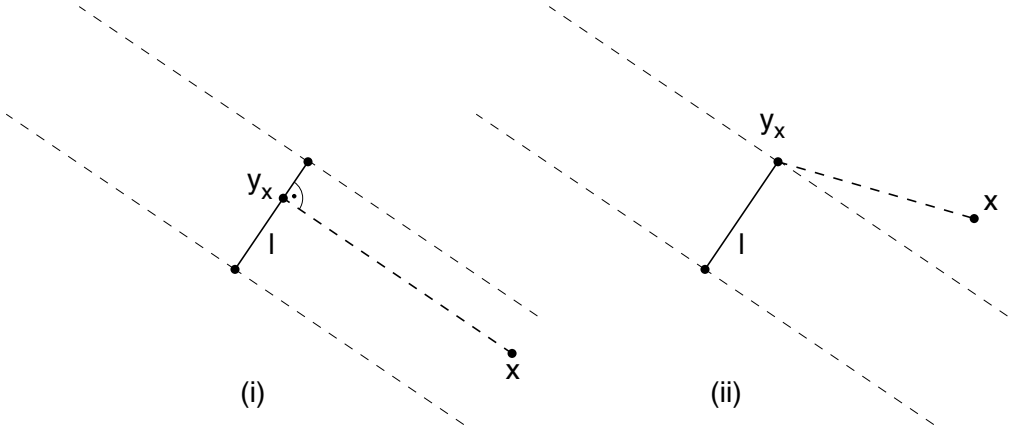
Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l, Streifen



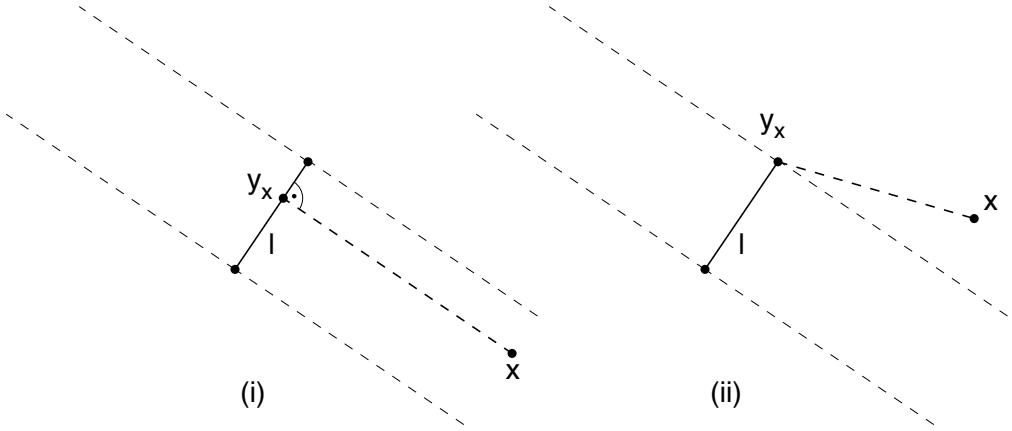
Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l, Streifen
- Bisektor zwischen zwei Segmenten l_1 und l_2



Voronoi Diagramm von Liniensegmenten

- Jetzt alle Objekte Punkte oder Liniensegmente
- Bisektor zwischen Punkt und Segment
- Abstand eines Punktes x zu Segment l, Streifen
- Bisektor zwischen zwei Segmenten l_1 und l_2
- $B(l_1, l_2) = \{x \in \mathbb{R}^2; |xl_1| = xl_2|\}$



Bisektor von Segmenten

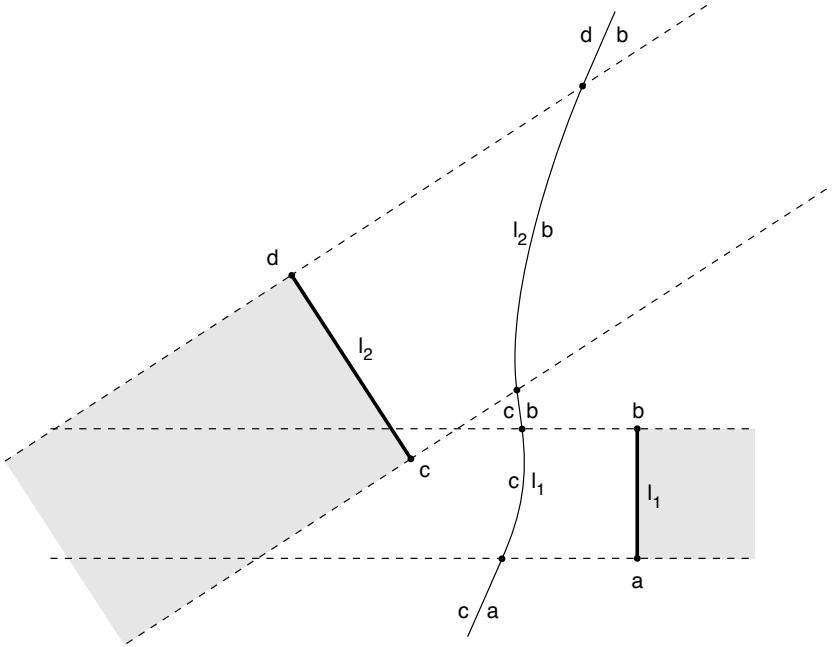
Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

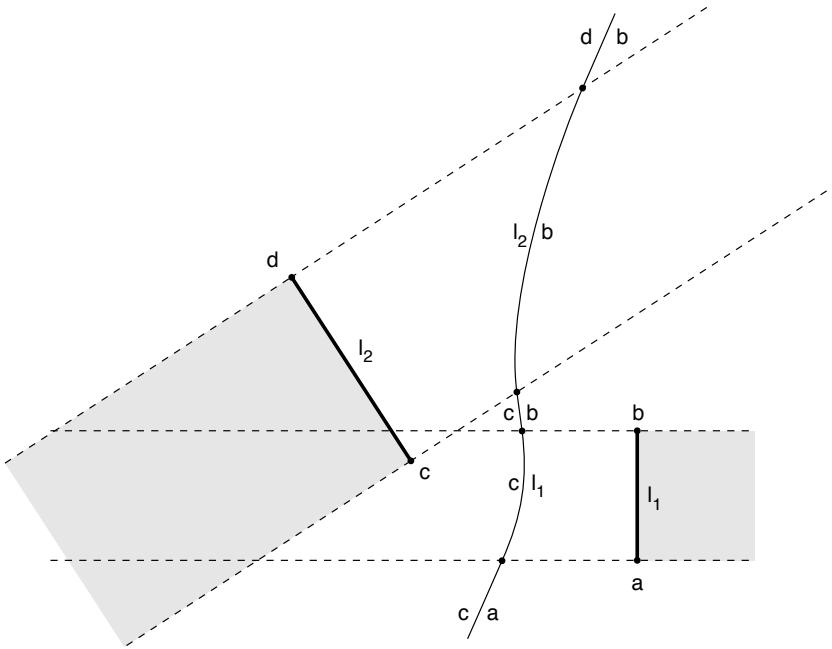


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt:
   Bisektorstück Gerade

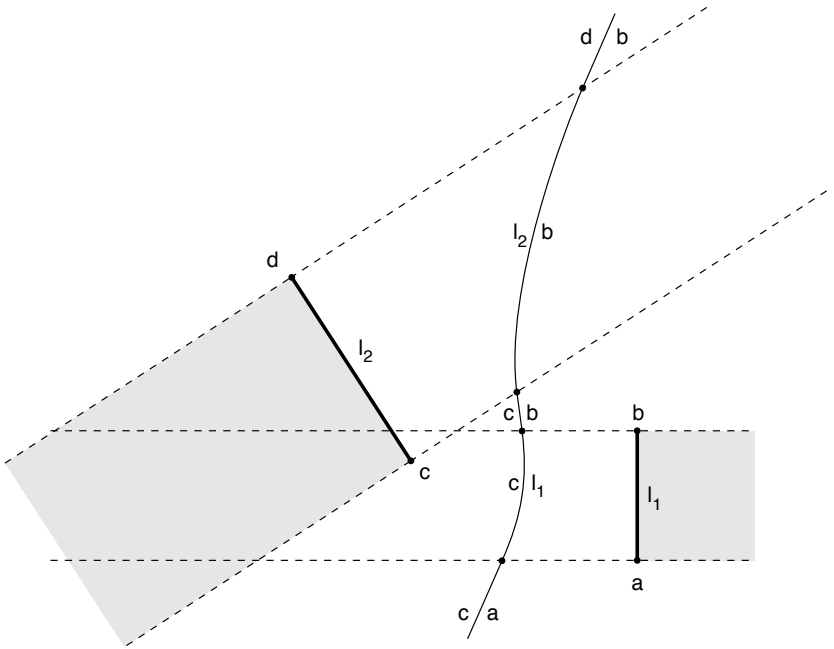


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt: Bisektorstück Gerade
2. l_1 Segment, l_2 Punkt: Bisektorstück Parabel

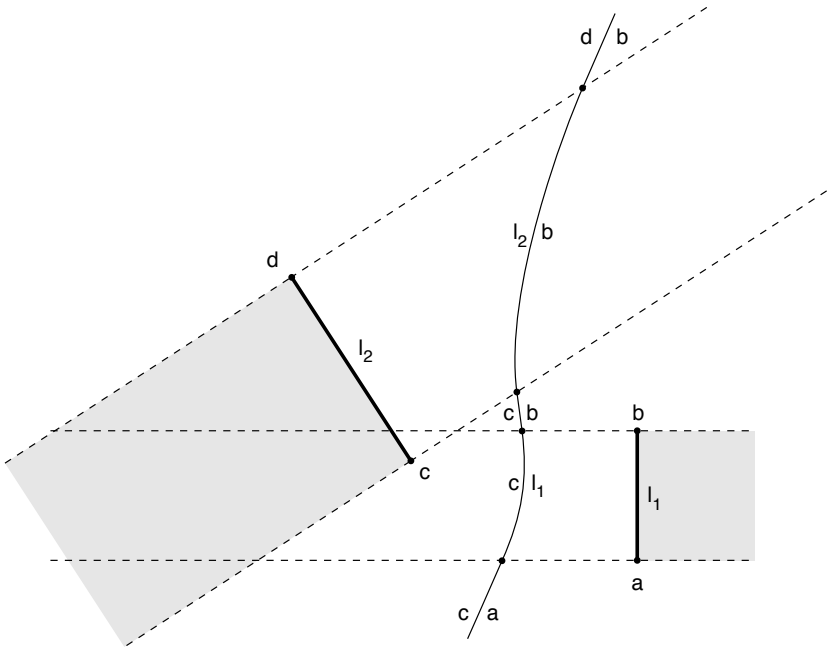


Bisektor von Segmenten

Lemma 5.24 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden.

Verantwortungsbereiche der Streifen, Lage zueinander

1. l_1 Punkt, l_2 Punkt:
   Bisektorstück Gerade
2. l_1 Segment, l_2 Punkt:
   Bisektorstück Parabel
3. l_1 Segment, l_2 Segment:
   Bisektorstück Gerade



Regionen sind sternförmig

Regionen sind sternförmig

Menge von Liniensegmenten S, Voronoi-Region von einem Liniensegment VR(l, S)!

Regionen sind sternförmig

Menge von Liniensegmenten S, Voronoi-Region von einem Liniensegment VR(l, S)!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region VR(l, S) gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in VR(l, S).

Regionen sind sternförmig

Menge von Liniensegmenten S, Voronoi-Region von einem Liniensegment VR(l, S)!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region VR(l, S) gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in VR(l, S).

Beweis:

Regionen sind sternförmig

Menge von Liniensegmenten S, Voronoi-Region von einem Liniensegment VR(l, S)!

Lemma 5.25 Sei S eine Menge von Liniensegmenten und $l \in S$. Für jeden Punkt x in der Voronoi-Region VR(l, S) gilt: Das Liniensegment xy_x zwischen x und dem Punkt $y_x \in l$ der am nächsten zu x liegt, liegt in VR(l, S).

Beweis: Widerspruch!

Regionen zusammenhängend

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Beweis:

Regionen zusammenhängend

Korollar 5.26 Die Voronoi-Regionen von Liniensegmenten sind zusammenhängend.

Beweis: Alle Punkte der Region haben *Blickkontakt* zu l!

Bisektor: Maximal 7 Stücke!

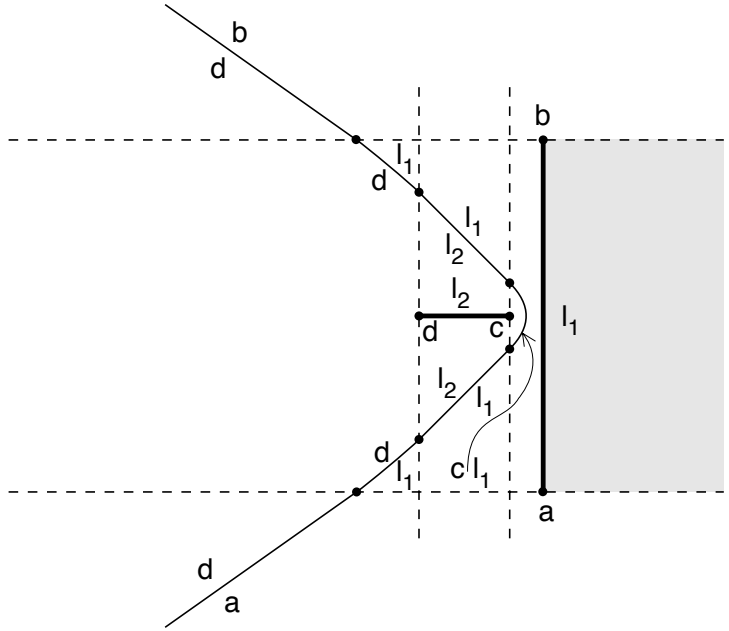
Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

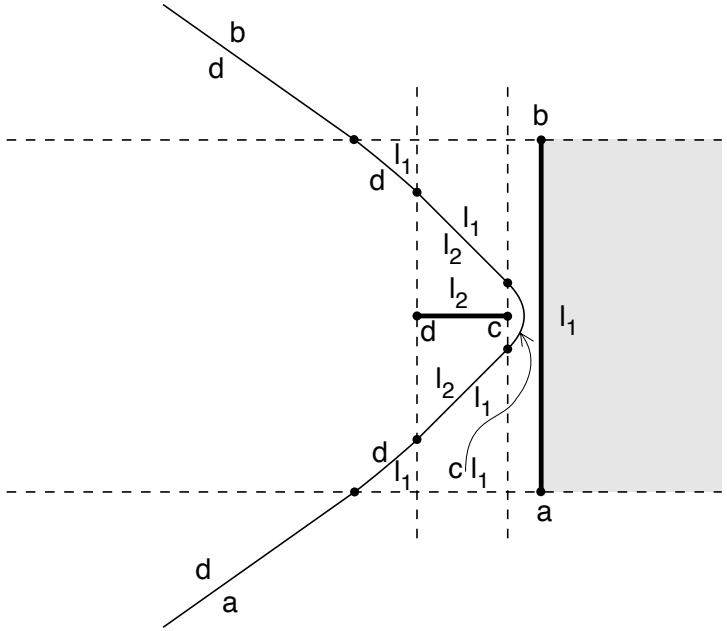
- Halbstreifen, insgesamt 8



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

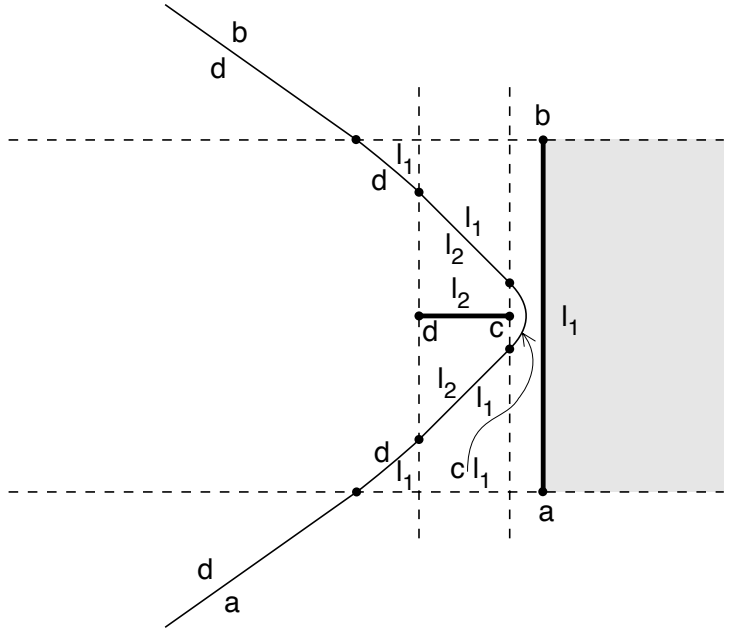
- Halbstreifen, insgesamt 8
- Einmal betreten,
  einmal verlassen, monoton



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

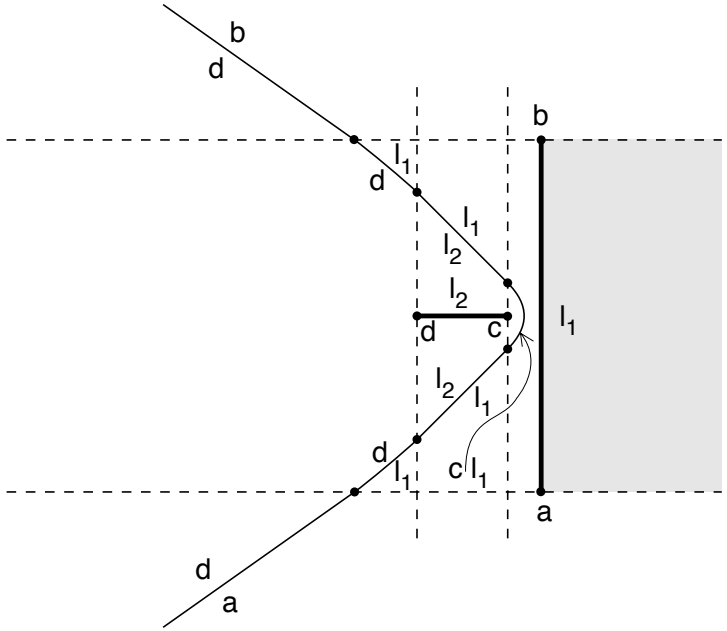
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

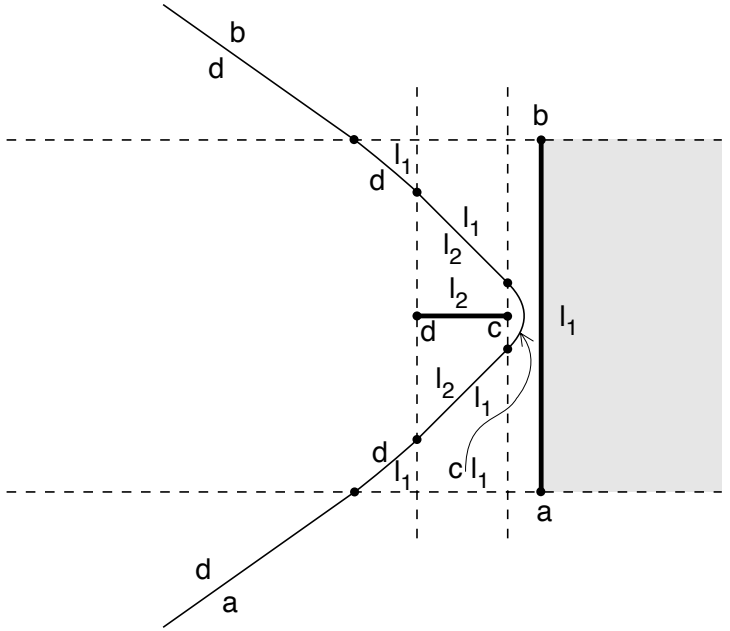
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle
- Bisektor betritt sukzessive 3 Streifen von l_1



Bisektor: Maximal 7 Stücke!

Lemma 5.27 Der Bisektor von zwei disjunkten Liniensegmenten l_1 und l_2 ist eine Kurve aus Parabelstücken, Liniensegmenten und zwei Halbgeraden und besteht aus maximal 7 Stücken.

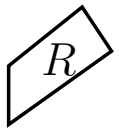
- Halbstreifen, insgesamt 8
- Einmal betreten, einmal verlassen, monoton
- Mind. Segment l_1 liegt auf konvex. Hülle
- Bisektor betritt sukzessive 3 Streifen von l_1
- max. 6 Kanten überqueren



Anwendung Bahnplanung: Kreisförmiger Agent

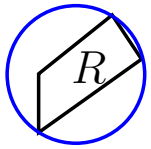
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter



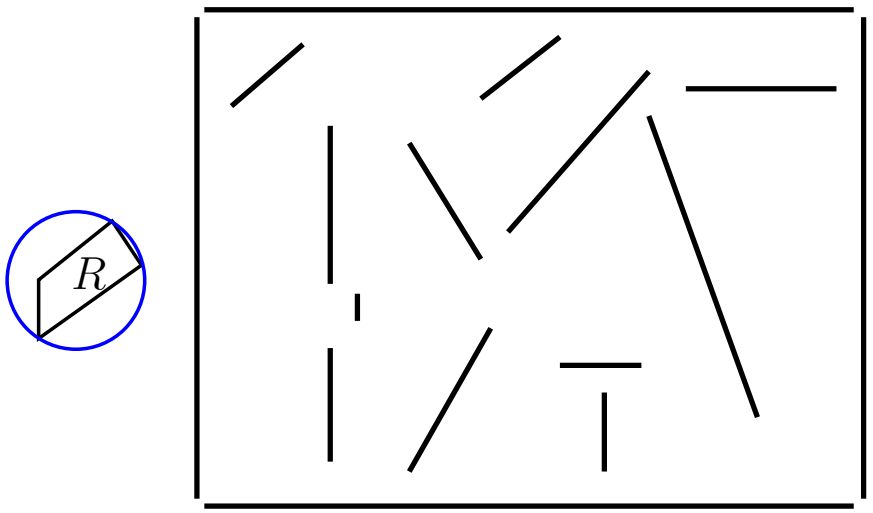
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse



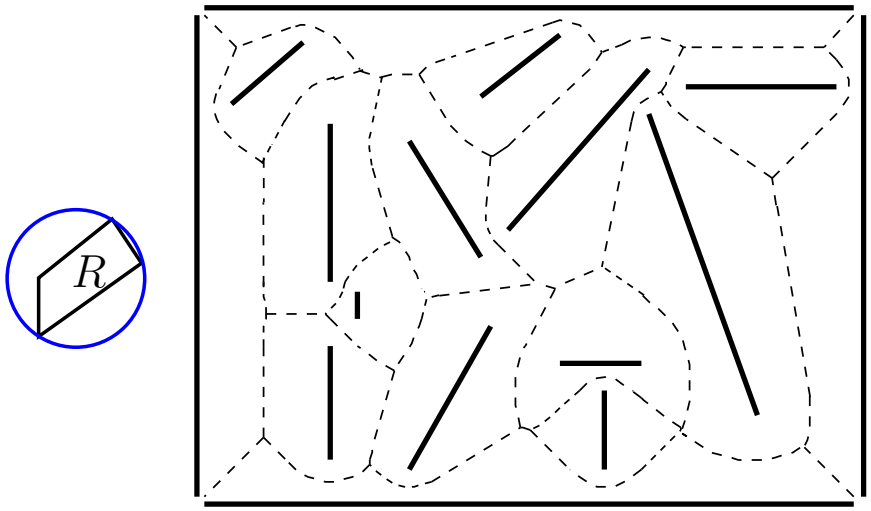
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:



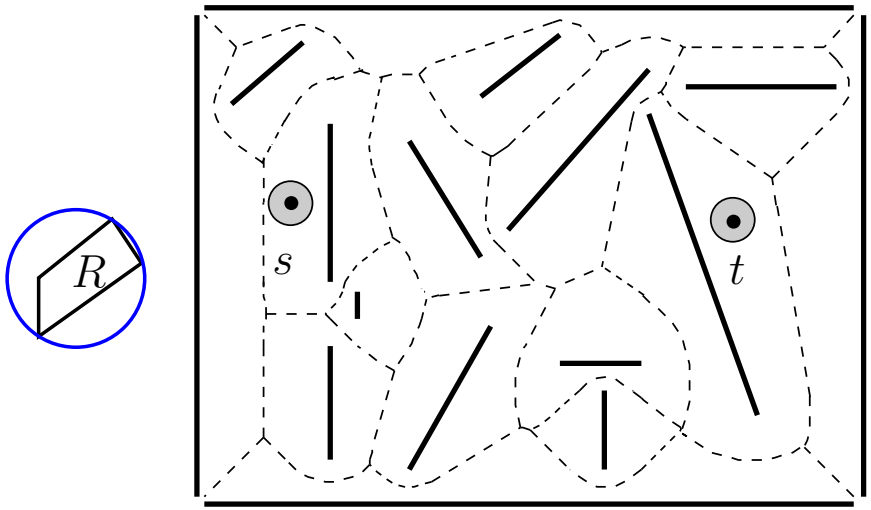
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:



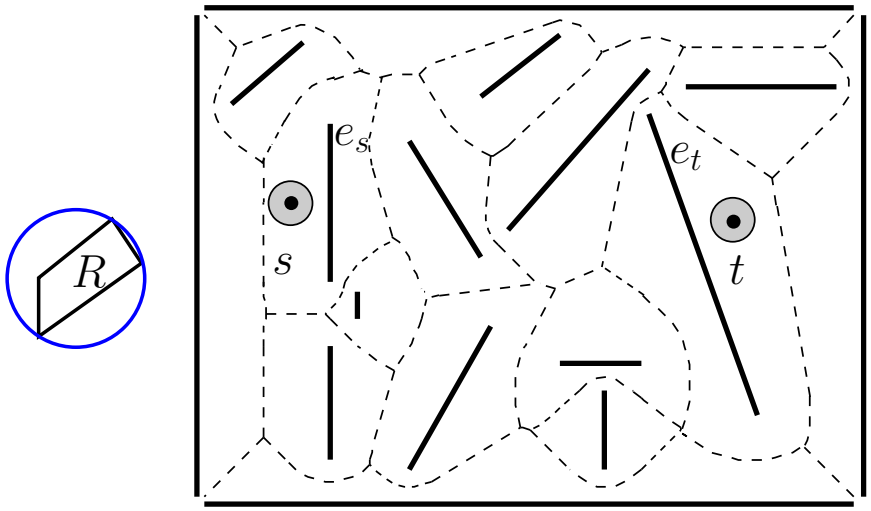
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:



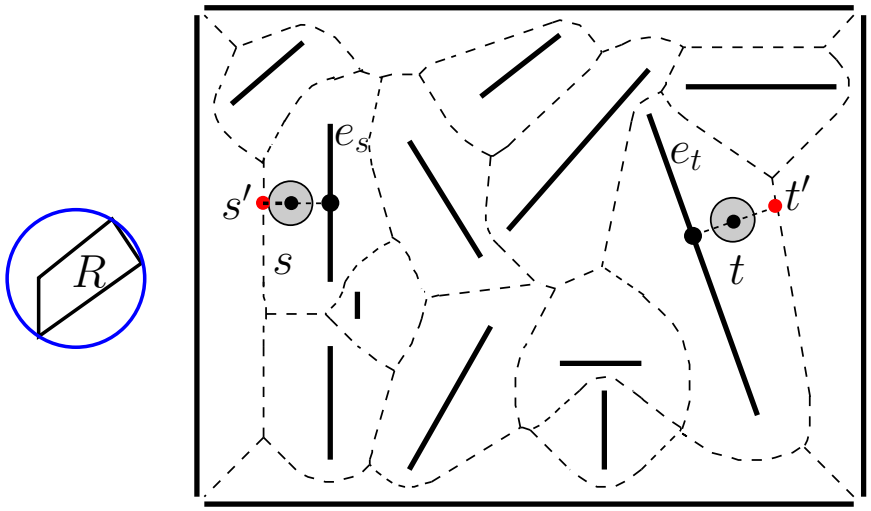
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:



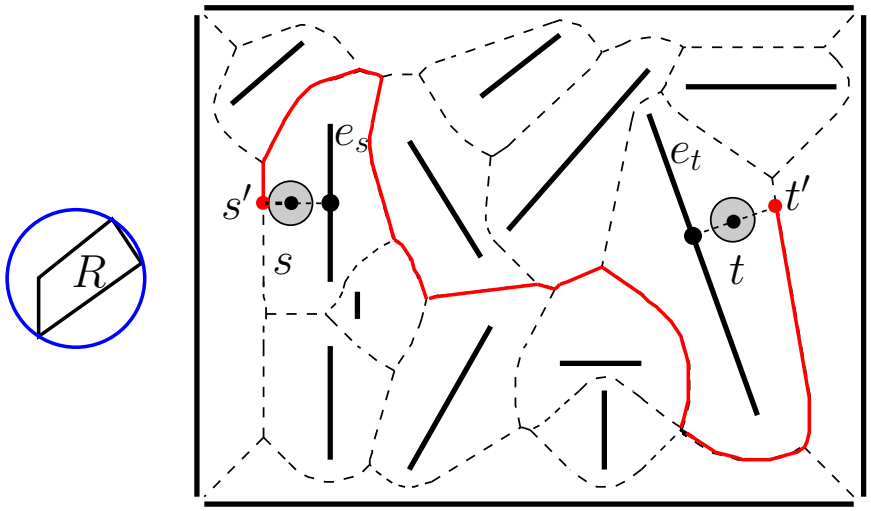
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:



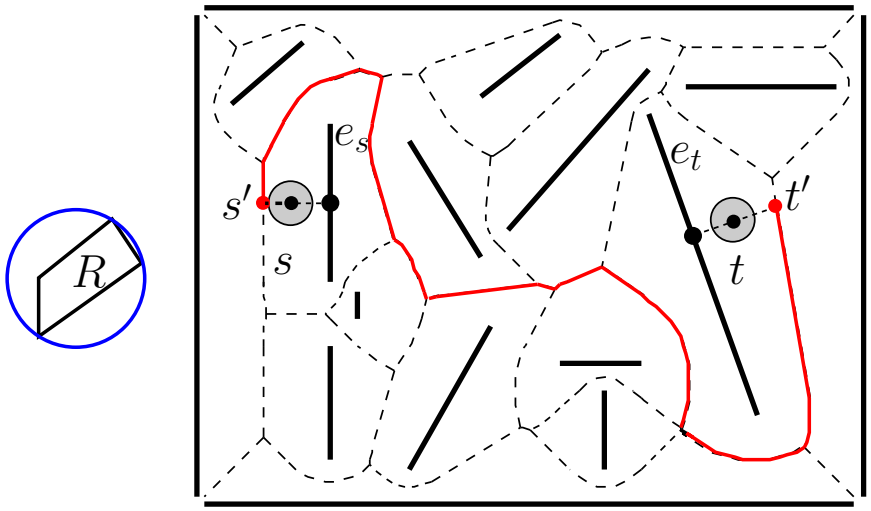
Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren:

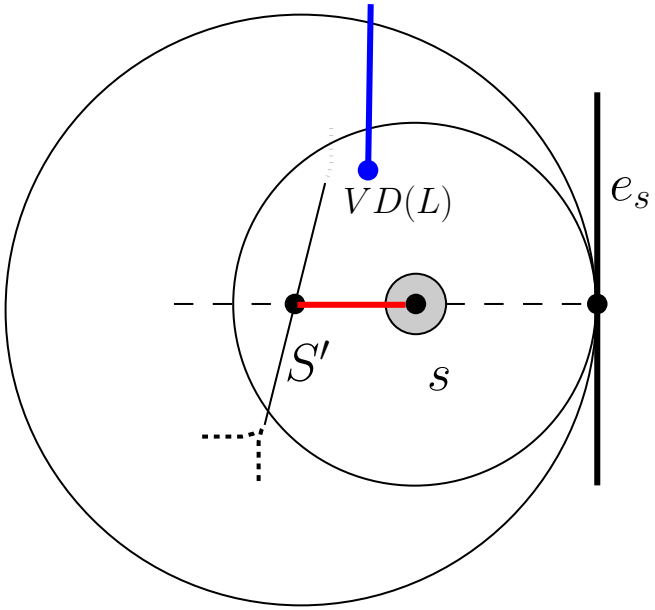


Anwendung Bahnplanung: Kreisförmiger Agent

- Verwende kleinsten Kreis um Roboter
- Voronoi Diagramm der Segmente der Hindernisse
- Weg auf Bisektoren: Möglichst großer Abstand

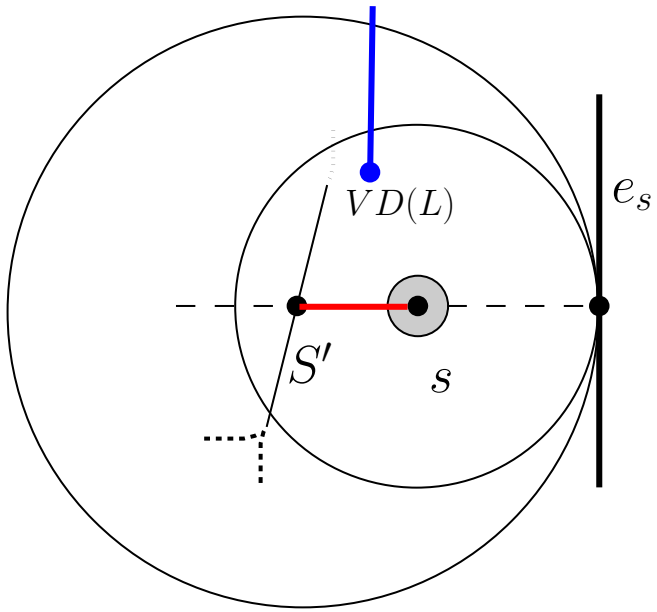


Start s' kann stets angelaufen werden



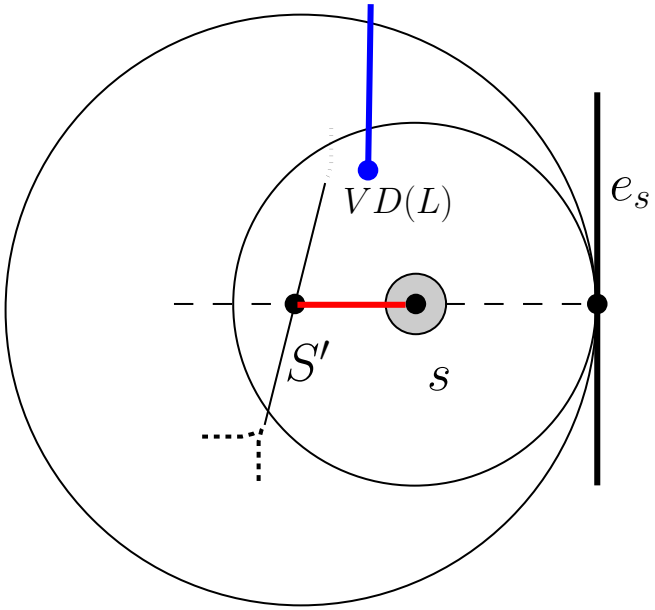
Start s' kann stets angelaufen werden

- s in Region von e_s,



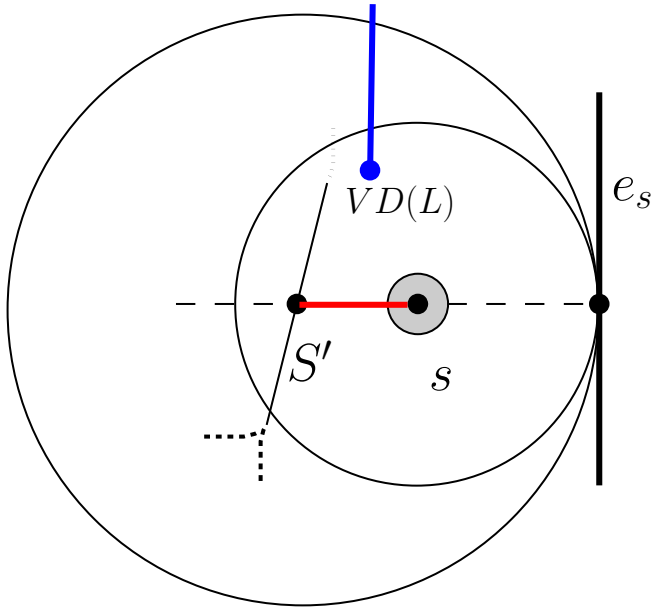
Start s' kann stets angelaufen werden

- s in Region von e_s, Kreis frei



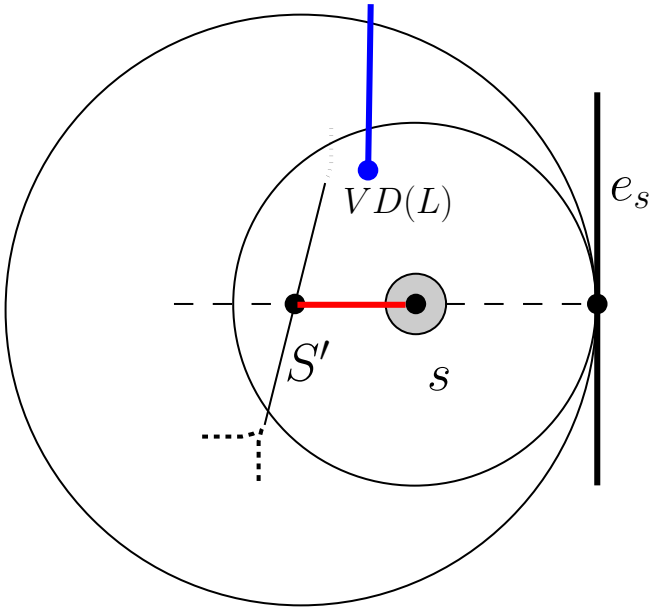
Start s' kann stets angelaufen werden

- s in Region von e_s, Kreis frei
- Kürzester Weg zu e_s, Strahl Richtung Bisektor



Start s' kann stets angelaufen werden

- s in Region von e_s, Kreis frei
- Kürzester Weg zu e_s, Strahl Richtung Bisektor
- Trifft Bisektor bei S',



Start s' kann stets angelaufen werden

- s in Region von e_s, Kreis frei
- Kürzester Weg zu e_s, Strahl Richtung Bisektor
- Trifft Bisektor bei S', Kreis/Weg ist frei!!