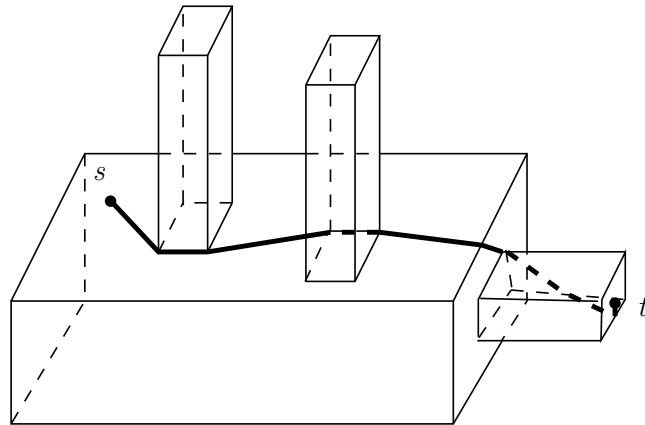


Offline Bewegungsplanung: Polyeder

Elmar Langetepe
University of Bonn

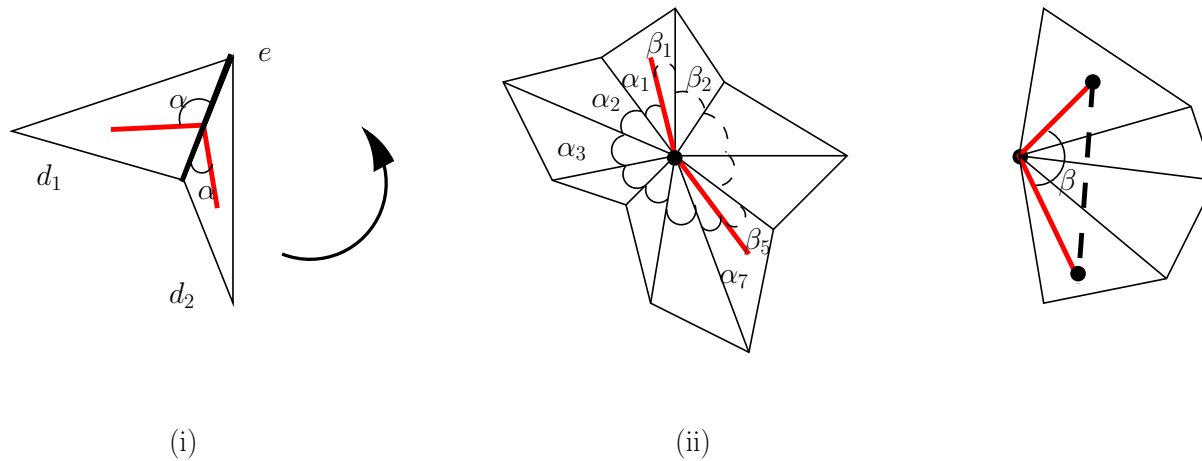
Downgrading: Oberfläche Polyeder!

- Natürliche Erweiterung der Polygone auf 3D
- Rand besteht aus Polygonen
- Keine dünnen Stellen: ϵ -Kugeln
- Datenstruktur QEDS: Triangulation Oberflächen, Navigation!



Lokale Eigenschaften (geodätisch)! **Lem. 1.39**

- Lokale Eigenschaften (geodätisch) kürzester Wege
- Alle Kanten: Eingangswinkel=Ausgangswinkel
- Knoten: Nur *nicht-konvexe* Ecken
- Konvexe Ecke: Ebenenschnitt



Globale Eigenschaften (Kürzeste)! **Lem. 1.40**

π_i, π_j kürzeste von s nach b_i resp. b_j ■

(i) π_i hat keine Selbstschnitte■

(ii) π_i schneidet jede Fläche max. einmal■

(iii) π_i, π_j kreuzen sich nicht im Innern einer Fläche■

Beweis!!! Lokal verkürzbar! (Tafel)■

Berechnungsansatz: Ideen

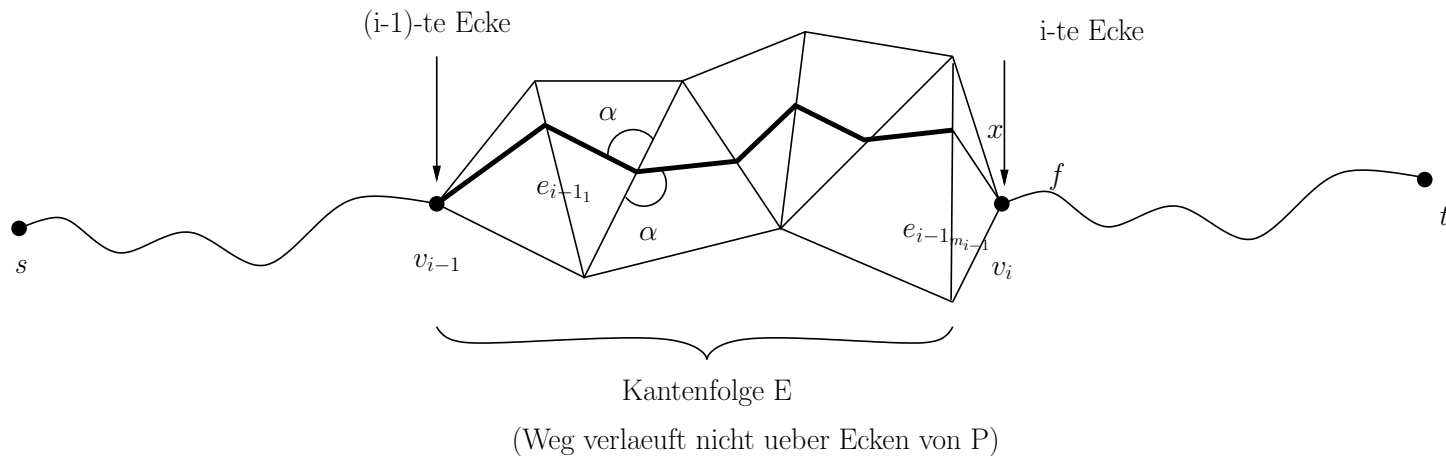
- Locus approach: Startpunkt s fest
- Nur letzte Schritte des Weges konkret
- Kombinatorisch gleiche zusammenfassen
- Zunächst nur Intervalle auf Kanten betrachten
- Sukzessive erweitern: Continuous Dijkstra
- In die Dreiecke fortpflanzen
- Query Struktur für alle Punkte auf P

Kombinatorik: Gesamtweg analysieren!

Folge von Kanten und Knoten! ■

$$\pi = \underbrace{v_0}_{=s}, \underbrace{e_{1,1}, \dots, e_{1,m_1}}_{\text{Kanten}}, \underbrace{v_1}_{\text{2. Ecke}},$$

$$\underbrace{e_{2,1}, \dots, e_{2,m_2}}_{\text{Kanten}}, v_2, \dots, e_{k-1,1}, \dots, e_{k-1,m_{k-1}}, \underbrace{v_k}_{=t}$$

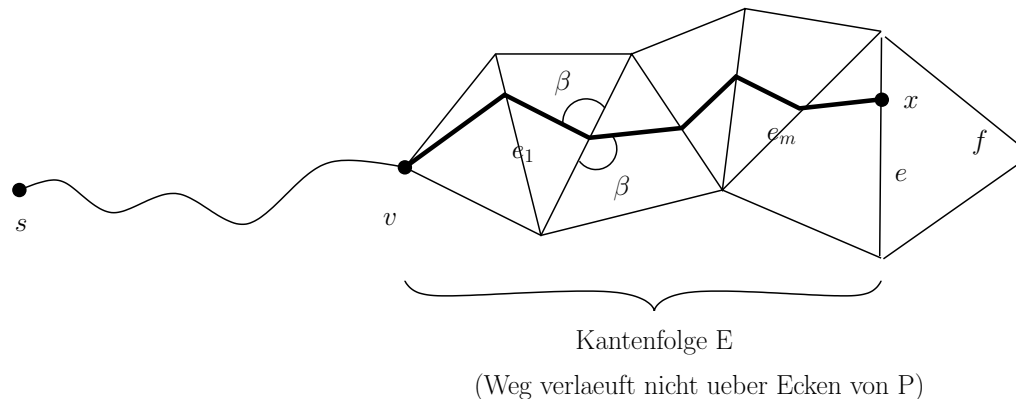


Letzte Schritte: Für beliebige Kanten!

- Kante e , letzter Knoten v , Kantenfolge E
- Kombinatorisch gleiche zusammenfassen, Dreiecke nicht beachten
- Optimalitätsintervall: Def. 1.42

$I(v, E) := \{x \in e \mid \exists \text{ Kürzeste } \delta \text{ von } s \text{ nach } x \text{ mit}$

- $\delta \cap \text{INT}(f) = \emptyset$
- $\delta \text{ endet mit } v, e_1, \dots, e_m, x \}$.



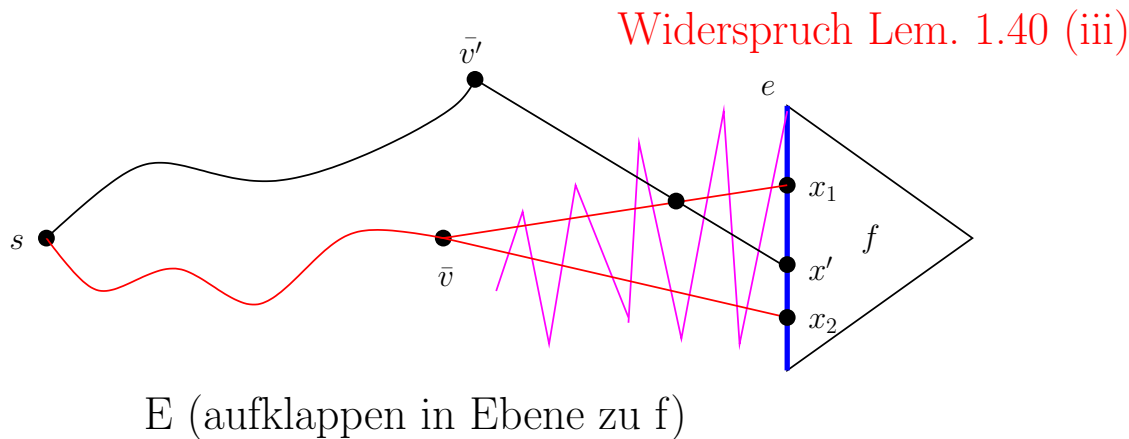
Lem. 1.43 Eigenschaften: $I(v, \epsilon)$

- (i) Jede solche Menge $I(v, E)$ ist Intervall auf e (evtl. leer).
- (ii) Zwei verschiedene Intervalle können sich nicht überlappen.
- (iii) e wird von Intervallen $I(v, E)$ ganz überdeckt.

Beweis!!! Gegeben: Kante e , letzter Knoten v , Kantenfolge E

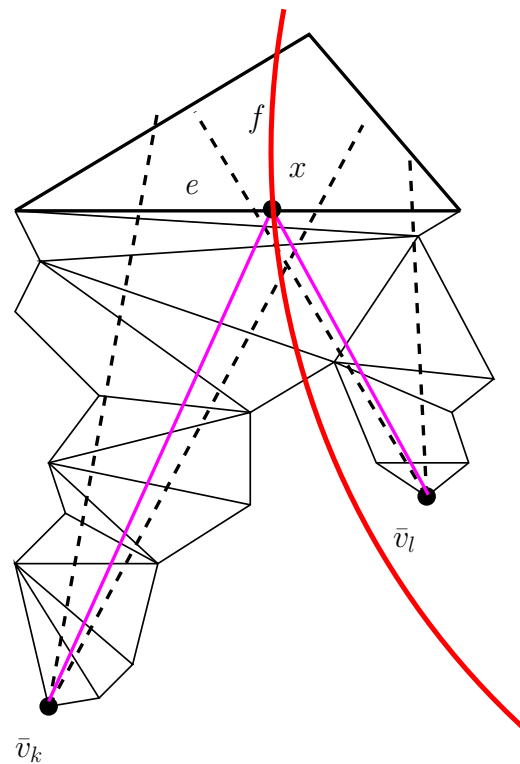
$I(v, \mathbf{E})$ ist Intervall auf e

- $I(v, \mathbf{E})$ leer \Rightarrow fertig! ■
- Annahme: Es gibt zwei Punkte x_1 und x_2 in $I(v, \mathbf{E})$ ■
- Zu zeigen: Alle Punkte dazwischen gehören zu $I(v, \mathbf{E})$ ■



Intervalle $I(v, \mathbf{E})$ überlappen sich nicht

- Gleiche Argumentation geht auch!!!
- ● Schönere Argumentation!!!



$$|\bar{v}_k - x| + d(v_k, s) = |\bar{v}_l - x| + d(v_l, s)$$

Hyperbel

Genau ein $x!!!$



e wird von Intervallen $I(v, E)$ überdeckt

Klar! ■ Jeder Punkt $x \in e$ wird von einem Kürzesten Weg besucht!

■
■
Insgesamt: **Lem. 1.43:** ■

- (i) Jede solche Menge $I(v, E)$ ist Intervall auf e (evtl. leer).
- (ii) Zwei verschiedene Intervalle können sich nicht überlappen.
- (iii) e wird von Intervallen $I(v, E)$ ganz überdeckt.

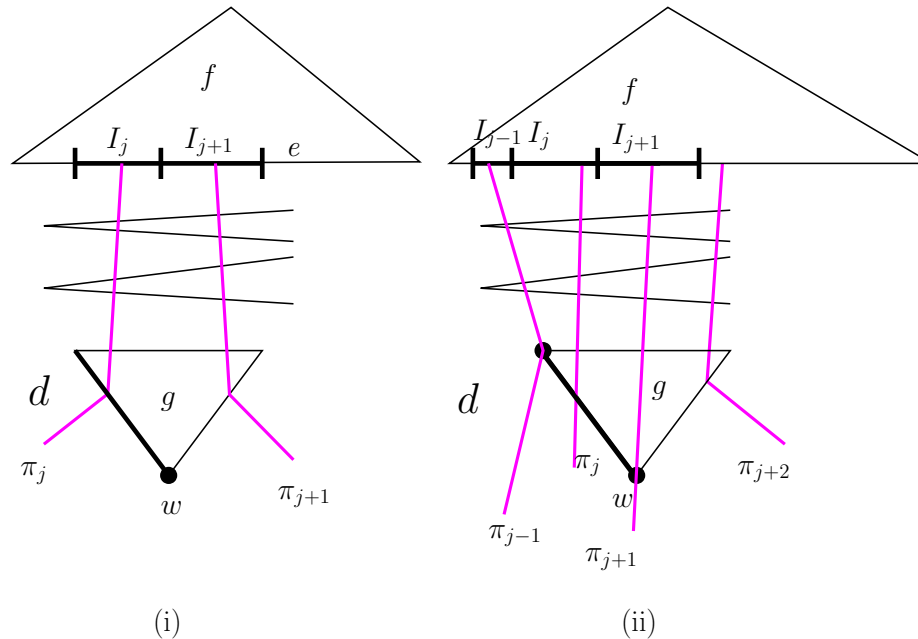
■

Alle $I(v, \mathbf{E})$ berechnen!

- Wie viele? ■
- Was machen wir mit dem Inneren der Dreiecke? ■
- **Lem. 1.44**: Kante e , $O(n)$ Intervalle $I(v, \epsilon)$ ■
- Zählargument: Klassisch!!! ■

Lem. 1.44: $O(n)$ Intervalle $I(v, \mathbf{E})$

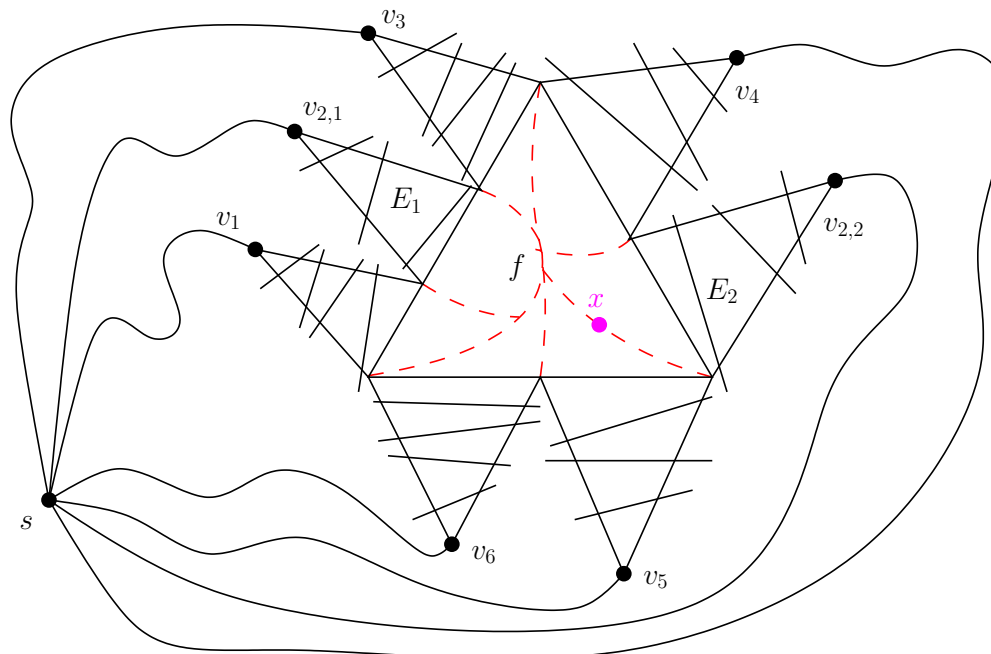
Benachbarte Intervalle trennen sich!!



- Kante d kann max zweimal als Trenner vorkommen!! ■
- Wegen Schnitteigenschaft!! Ausklappen ohne Überlappungen!■

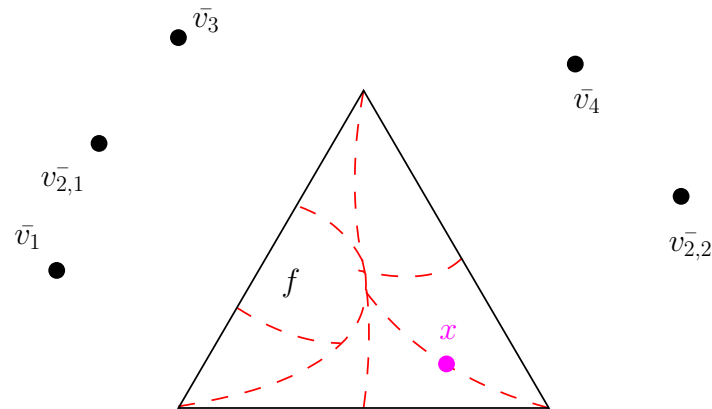
Das Innere der Dreiecke füllen

Annahme: Intervalle $I(v, E)$ berechnet! Situation für jedes Dreieck!
Mehrere Kantenfolgen!!



$$|x - v_{2,2}| + d(s, v_{2,2}) = |x - v_5| + d(s, v_5)$$

Aufteilen der Dreiecke!



$$|x - v_{2,2}| + d(s, v_{2,2}) = |x - v_5| + d(s, v_5)$$



- Gewichte: $d(s, v_i)$, **Regionen bezüglich Orte \bar{v}_i**
- Voronoi Diagramm mit additiven Gewichten $d(s, v_i)$
voroAdd.html
- Lokalisationsmöglichkeit! (Separators/Seidel)

Alg. 1.13

- Input: Triang. Polyder P , n Ecken, s ■
- Output: Kürzester Weg für beliebigen Anfragepunkt $t \in f$ ■
- Preprocessing: ■
 - Berechne Intervalle $I(v, E)$, $O(n^2 \log n)$
 - Für alle f **add. gew. VD**, $O(n^2 \log n)$ ■
 - **Lokalisationstechnik!(Separators/Seidel)**, $O(n^2)$ ■
- Query: ■
 - **Voronoi Region von t in $VD(f)$** $O(\log n)$
 - Kürzesten Weg aus $I(v, E)$: Über E und in v abgosp. Kürz. Weg zu s , k Segmente, $O(k)$ ■
 - Nur Länge, $O(1)$ ■

Ergebnis: Theorem 1.45

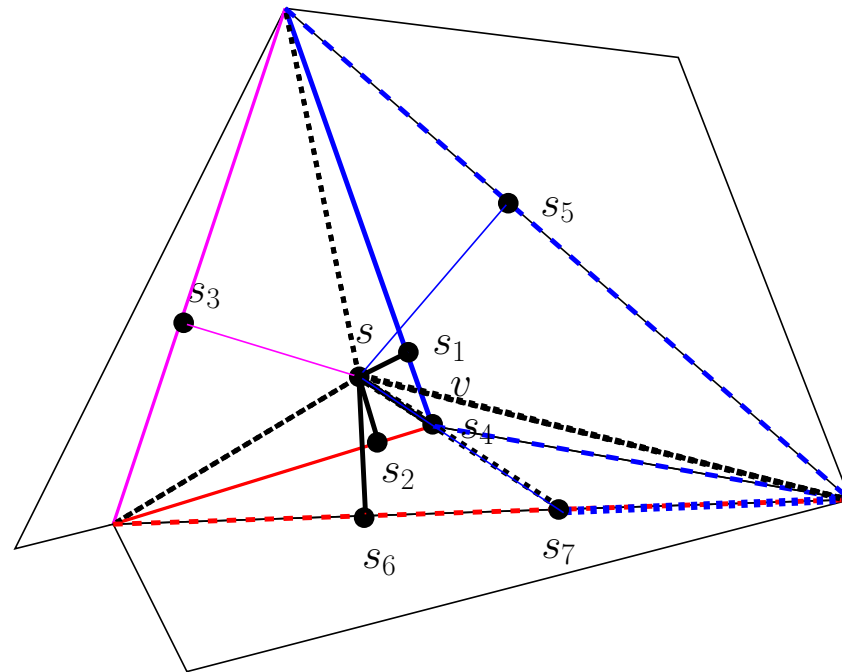
Sei s auf P fest, gegeben b auf P . Die Entfernung bzw. die Kürzeste von s nach b auf P läßt sich nach Vorbereitungszeit $O(n^2 \log n)$ mit Platz $O(n^2)$ in Zeit $O(\log n)$ bzw. $O(\log n + k)$ berechnen.■

(Mount, Mitchell, Papdimitriou, 1986)■

Berechnung aller $I(v, \mathbf{E})$

Continuous Dijkstra: Sukzessive (Teil)Intervalle festlegen!

■ Garantiepunkt s_i auf dem Intervall, nächster Punkt zu s . ■ Kürzeste Wege zu Knoten merken! ■ Listen von Intervallen aufbauen! ■



Alg. 1.12 Continuous Dijkstra

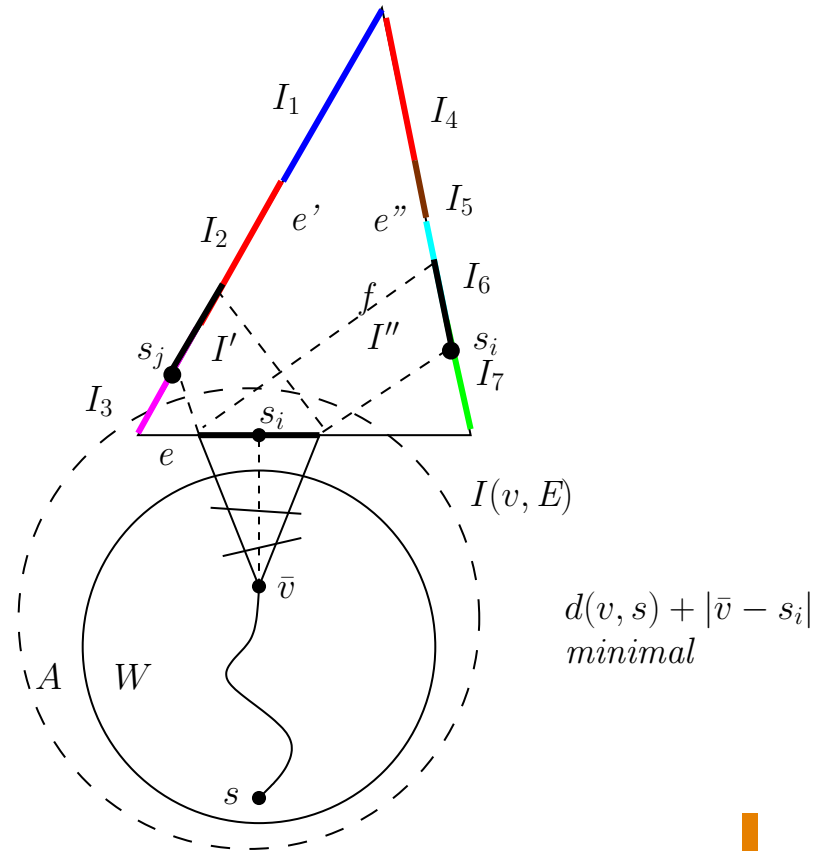
- DS:

- Priority Queue W : Intervalle nach s_i -Abstände.
- Intervalllisten auf den Kanten in balancierten Baum vorhalten.

- Iterationsschritt:

- Wähle Intervall $I(v, E)$ mit kürzestem Abstand.
 - $d(s, v) + |\bar{v} - s_i|$ ist am geringsten! ■ Wird stets Intervall bleiben! (Eventuell später gekürzt). ■
- Führe dieses Intervall auf die Kanten des Dreiecks fort, auf dem s_i lag und das jenseits der Folge E liegt.■
- Sortiere zwei neue Intervalle in die Liste der Intervalle der beiden Kanten ein. Bestimme jeweilige s_j und füge in W ein.■

Schritt des Einsortierens!



Einsortieren in Baum der Intervalle! $O(\log n)$!