Übung 2

EAG 2.1

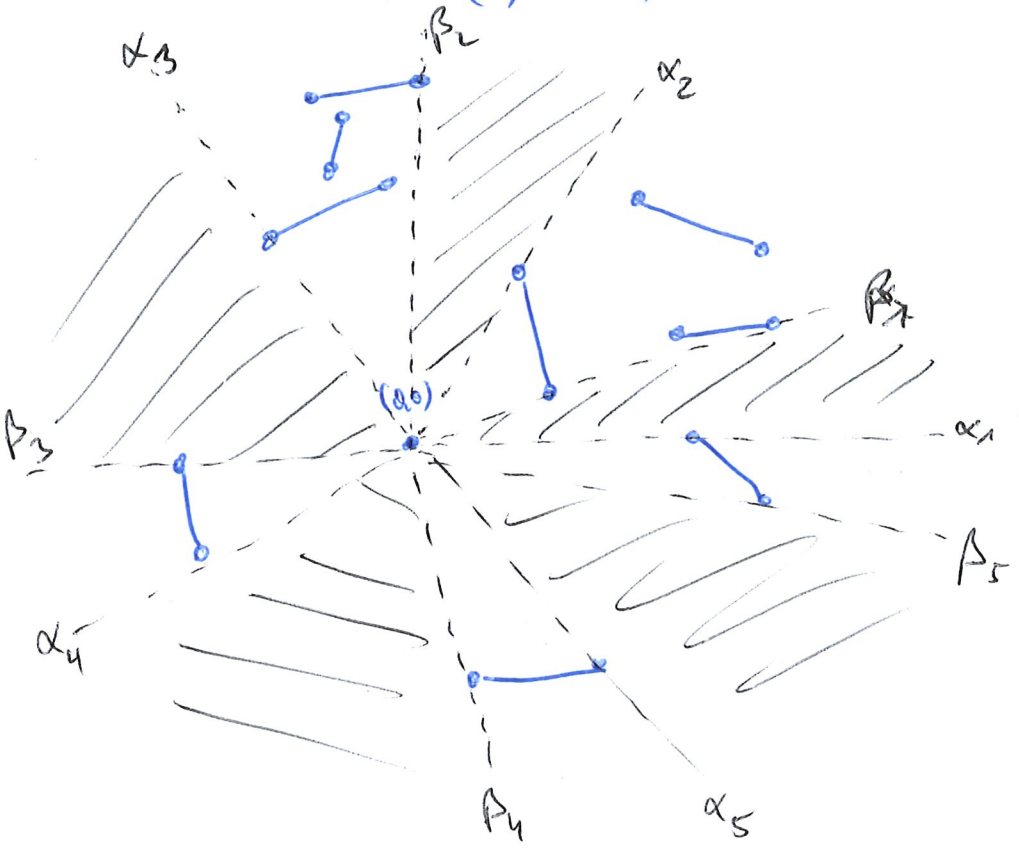
Aufwärmproblem

Gegeben: n Segmente im $\mathbb{R}^2$ $S_1, \dots, S_n =: S$

Gesucht: Intervalle $[\alpha_i, \beta_i]$ mit

$I = \bigcup [\alpha_i, \beta_i]$ maximal

und $K(I) \cap S = \emptyset$



S_i sind hier als offene Mengen zu betrachten, d.h. Endpunkte dürfen auf dem Rand eines Intervals liegen.

$K(I) \triangleq$

Kreiskegel aller Intervalle mit Ursprung $(0,0)$

Wie löst man das mit einem Sweep-Line-Algorithmus?

- Plane-Sweep (also eine gerade Linie von links nach rechts) scheint nicht sinnvoll.
- Betrachte Angular-Sweep: Ein Strahl, der sich um einen Punkt dreht.

Idee: → Eventpoints

- Sortiere Endpunkte der Segmente bzgl. ihres ccw-Winkels zur x-Achse

- Für jeden Eventpoint:
  - zähle einen Zähler hoch, wenn ein neues Segment beginnt.
  - zähle einen Zähler herunter, wenn ein Segment endet

- Gebe ein Intervall aus, wenn Zähler auf 0 fällt bis er wieder auf 1 geht

Laufzeit

- Sortieren $O(n \log n)$
- $2n$ Event points, je $O(1)$ Bearbeitungszeit
  $\rightarrow O(n)$

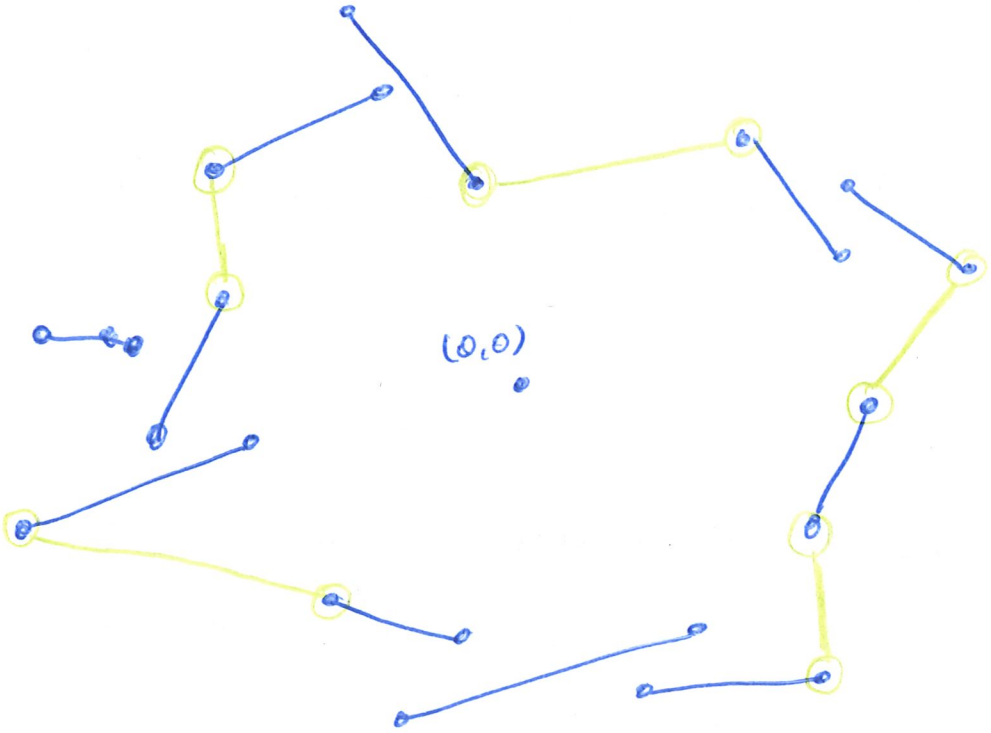
$\Rightarrow O(n \log n)$

Zu beachten:

Wie codieren wir das Intervall?

Problem: Die Winkel sind ggf. nicht exakt darstellbar (irrational), auch wenn die Inputdaten/Segmentenden rationalen Zahlen entsprechen.

Wie umgeht man das Problem?



Anstatt die Winkel zuspeichern, beschreibe die Intervalle über Segmente.

Problem 3

Gegeben: Punktmenge $P = \{p_1, \dots, p_n\} \subsetneq \mathbb{R}^2$

Frage: Enthält P kollineare Punkte

Naiv: Teste Kollinearität für jedes Tripel aus $\binom{P}{3}$.
Laufzeit $O(n^3)$

Wichtig: Liegen s, p, q auf einer Linie, so ist die Linie $l_{s,p}$ identisch mit der Linie $l_{s,q}$

~~Linie~~ Gerade durch s und p

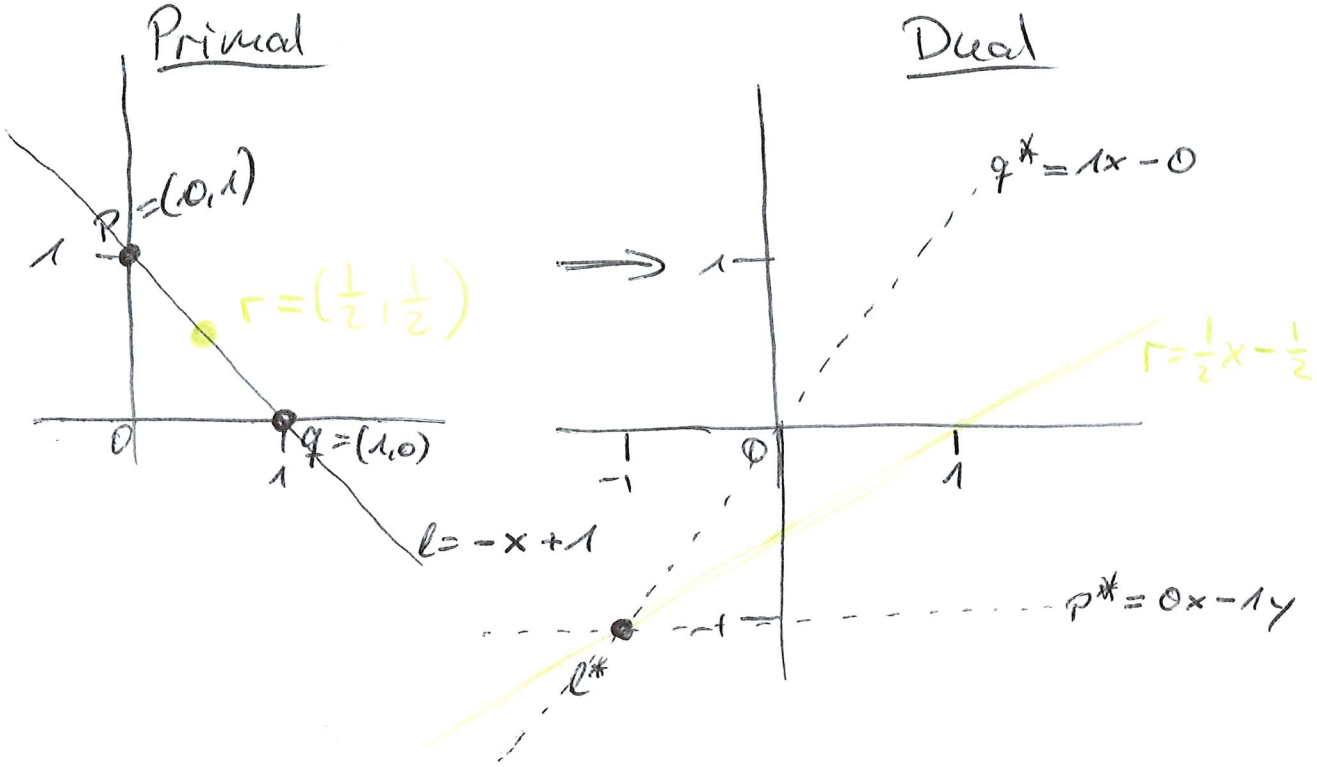
Da beide Linien durch s gehen, müssen wir nur die Steigung $\frac{y_q - y_s}{x_q - x_s}$ mit $\frac{y_p - y_s}{x_p - x_s}$ vergleichen.

$\Rightarrow$ Prüfe für jeden ~~Knoten~~ Punkten $s \in P$, ob unter allen ~~Knoten~~ Punkten $p \in P \setminus \{s\}$ der Wert $\frac{y_p - y_s}{x_p - y_s}$ mind. 2x vorkommt.

Mit einem balancierten ~~Daten~~ Suchbaum dauert das $O(n \log n)$ für jedes $s \in P \Rightarrow O(n^2 \log n)$

Mit Hashmaps geht das (im Erwartungsfall) sogar in $O(n^2)$.

Die Idee des Algorithmus lässt sich auch visualisieren:



Transformiere jeden Punkt $p=(p_x,p_y)$ in eine Gerade p^* mit $y=p_x x-p_y$ und jede Gerade l mit $y=mx+b$ in einen Punkt $l^*=(m,-b)$

Man kann zeigen:

Sind p,q,r kollineare Punkte, dann haben p^*,q^*,r^* einen gemeinsamen Schnittpunkt.

(Für parallele Geraden definieren wir den Schnittpunkt bei $x=\infty$)

Diese Dualität bietet nette Eigenschaften

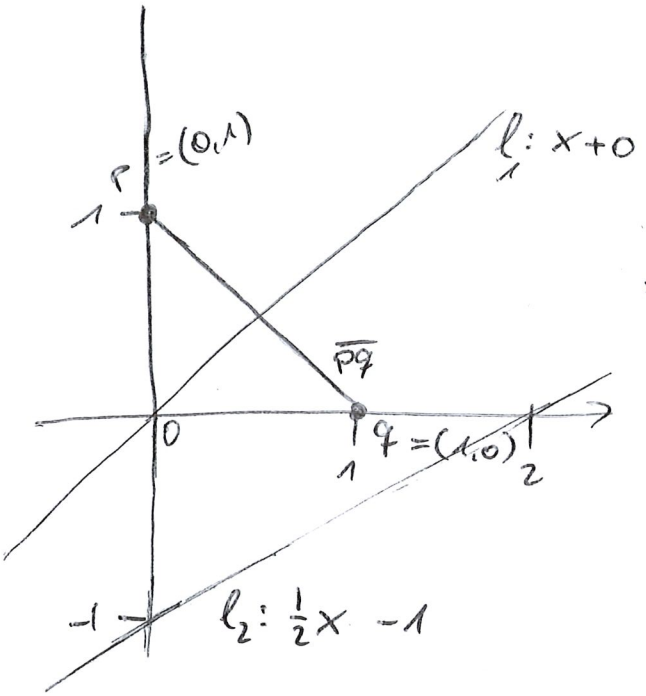
p liegt "oberhalb" von $\ell \Leftrightarrow \ell^*$ liegt oberhalb von p^*

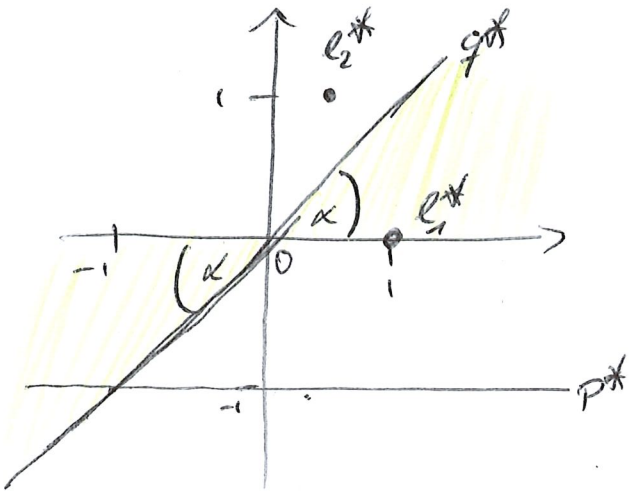
p liegt auf $\ell \Leftrightarrow \ell^*$ liegt auf p^*

p liegt unter $\ell \Leftrightarrow \ell^*$ liegt unter p^*

"Incidence Preserving"

Wie ist das mit Segmenten?





Jeder Punkt im α-Kegel entspricht einer Geraden im Primalen, welche $\overline{pq}$ schneidet!