1.3.6. Satz: Sei S Menge von n Pkten im $\mathbb{R}^2$.

a). $CH(S)$ kann in Zeit $O(n \log n)$ berechnet werden (worst case)

b). Falls S lexik. nach xy-Koord. sortiert ist, dann kann $CH(S)$ in Zeit $O(n)$ berechnet werden.

Bew. siehe oben.

1.3.7. Bemerkung:

1) Alg. optimal, da CH-Problem (1.a.) genauso schwierig ist wie sortieren.
Bew. Übung.

2). obere bzw. untere Hülle sind auch für sich alleine wichtig (für bestimmte Probleme)

3). Optimalität gilt für worst case; es gibt Situationen, in denen Alg. I. besser ist (wenn # Ecken von $CH(S) < \log n$).

1.3.8. Varianten von Graham's Scan:

a) siehe Übung

b). untere u. obere Hülle gleichzeitig berechnen

c). Berechnung der gesamten Hülle in einer Phase.

Idee: 1. Sortiere $S = p_1 .. p_n$.

2. Allgemeiner Schritt:

Bearbeite p_i, $i = 4..n$.

Situation: $C_{i-1} := CH(\{p_1 .. p_{i-1}\})$ ist berechnet.

p_{i-1} ist maximal in $\{p_1 .. p_{i-1}\}$

$\Rightarrow p_{i-1}$ ist die rechteste Ecke von C_{i-1}.

$\Rightarrow \overline{p_{i-1} p_i} \cap C_{i-1} = \{p_{i-1}\}$.

Der eigentliche Schritt:

Berechne Berührpkte t und b der beiden Tangenten von p_i an C_{i-1}.

Genauer: $t \leftarrow p_{i-1}$ d.h. orientation$(p_i, t, succ(t)) \leq 0$

while $(p_i, t, succ(t))$ nicht lefturn do

$t \leftarrow succ(t)$

od

$b \leftarrow p_{i-1}$ d.h. orientation$(p_i, b, pred(b)) \geq 0$

while $(p_i, b, pred(b))$ nicht rightturn do

$b \leftarrow pred(b)$

od

Bea: Bie ähnliche Argumentation:
die # unterwegs besuchten Pkte kann in $\Omega(n)$ liegen,
aber jeder von ihnen wird anschließend aus
der konvexen Hülle entfernt und kann danach
nie wieder als Eckpkt in Erscheinung treten
Also wird er auch nie wieder besucht
$\Rightarrow$ Es kann insgesamt höchstens n Besuche geben
$\Rightarrow$ $O(n \log n)$ Laufzeit

Nachdem wir nun b und t berechnet haben:

- Entferne alle Pkte zwischen b und t
- Füge p_i nach C_i ein.

Weitere Implementierungsdetails und Laufzeitanalyse siehe 3. Übung.

Bsp. zu c) siehe Ilmordner.

1.4. Algorithmus III: Devide & Conquer. Triangulierung schauen

1.4.1. Spezialfall: konvexe Hülle von zwei konvexen Polygonen P und Q.

$P = p_1 .. p_m$, $Q = q_1 .. q_e$, Ecken gg. Uhrzeigersinn sortiert.

Aufgabe: Berechne $CH(P \cup Q)$

Triviale Lsg: Graham's Scan auf die Menge aller Ecken. $O(n \log n)$

Bessere Lsg: Ausnutzen der Polygonstruktur. $O(n)$.

1) Finde Sortierung aller Ecken in Zeit $O(n)$

2) Berechne die Hülle in Zeit $O(n)$ (siehe S. 1.3.6. b).) $\Big\}$ $O(n)$

zu 1): Betrachte P:

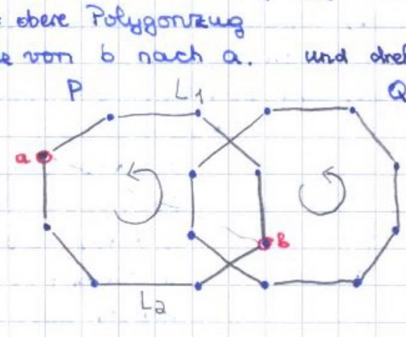
(i) Finde extreme Ecken a und b in Zeit $O(m)$

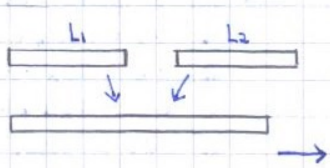
$L_2 :=$ untere Polygonzug

starte bei a und laufe gg. Uhrzeigersinn über P bis b erreicht ist.

$L_1 :=$ obere Polygonzug

laufe von b nach a. und drehe um.





(ii) Mische L_1 und L_2 zu einer sortierten Gesamtliste L_P zusammen in Zeit $O(m)$ (→ siehe Mergesort)
Analog: Sortiere Folge L_Q der Ecken von Q in Zeit $O(\ell)$.
(iii) Mische L_P und L_Q in Zeit $O(m+\ell) = O(n)$ (wobei $n := m+\ell$) zu einer sortierten Gesamtfolge zusammen.
→ dann Graham's Scan.

1.4.2. Allgemein: Sei $S \subset \mathbb{R}^2$, $|S| = n$.

```
CONVEX_HULL(S).
if |S| = 1 then
    output S
else
    teile S in zwei möglichst gleich große Teile S1 und S2   } DEVIDE
    (zB. |S1| = ⌈|S|/2⌉ und |S2| = ⌊|S|/2⌋
    P ← CONVEX_HULL(S1)                                       } CONQUER
    Q ← CONVEX_HULL(S2)
    berechne CH(P ∪ Q) wie in letzter Vorlesung gezeigt.      } MISCHSCHRITT
fi.
```

↑ hier wird die eigentliche Rechnung durchgeführt.

1.4.3. Laufzeit:
Teilen: $O(n)$
Mischen: $O(n)$
Merge auf Listen
Graham's Scan (ohne Sortieren).

$$\Rightarrow T(n) = \begin{cases} C_0, & n = 1 \\ C_1 \cdot n + 2 \cdot T(n/2), & n > 1 \end{cases}$$

Das ist also dieselbe Rekursion wie für Mergesort (aber mit anderen Konstanten natürlich).
⇒ Gesamtlaufzeit des Verfahrens ist $O(n \log n)$ (siehe Analyse von Mergesort).

1.5. Eine Anwendung von CONVEX_HULL

1.5.1. Problem: Geg. sind n Halbebenen $H_1 \dots H_n$ von $\mathbb{R}^2$
Berechne $P = \bigcap_{i=1}^{n} H_i$

1.5.2. Def: Eine abgeschlossene Halbebene = {alle Pkte auf gleicher Seite einer Geraden ℓ} ∪ ℓ.

1.5.3 Anm: Schnitt P ist konvexes (möglicherweise unbeschränktes) Polygon, da der Schnitt konvexer Mengen wieder konvex ist.

1.5.4. Ziel und Lösungsansatz:
Ziel: Berechne die Folge der Ecken von P gg. den Uhrzeigersinn.
Lösung: Zurückführung auf konvexe Hülle einer geeigneten Pktmenge.
Dazu transformieren wir die definierenden Geraden in "duale" Pkte.

1.5.5. Definition: (Geometr. Transformation)
a). Sei $\ell = \{y : \underbrace{y = ax + b}_{\text{Geradengleichung von } \ell}, x \text{ bel.}, a,b \text{ fest}\}$ eine nicht vertikale Gerade.
Der Punkt $D(\ell) := (a,b)$ heißt der duale Punkt zu ℓ.
b). Sei $p = (a,b) \in \mathbb{R}^2$ ein Pkt. Die Gerade $D(P) = \{y : y = -ax + b, x \text{ Beliebig}\}$ heißt die duale Gerade zu p.
Abbildung D erlaubt die relative Lage von Objekten.

1.5.5 Lemma: Pkt p liegt auf (oberhalb / unterhalb) einer Geraden ℓ
⇔ Gerade $D(P)$ liegt auf (oberhalb / unterhalb) Pkt $D(\ell)$.

Beweis: Sei $p = (p_x, p_y)$, $\ell = \{y \in \mathbb{R} : y = ax + b, x \in \mathbb{R}\}$ (a,b fest).
$\Rightarrow D(\ell) = (a,b)$ und $D(P) = \{y : y = -p_x x + p_y, x \in \mathbb{R}\}$

- sei p gelegen auf ℓ
$\Leftrightarrow p_y = a \cdot p_x + b$
$\Leftrightarrow -b = ap_x - p_y$
$\Leftrightarrow b = -p_x a + p_y$
$\Leftrightarrow D(P)$ liegt auf $D(\ell)$.

- sei p gelegen oberhalb ℓ (unterhalb analog).
$\Leftrightarrow ax + b < p_y \ \forall x \in \mathbb{R}$
Bzw. sogar $ap_x + b < p_y$.
$\Leftrightarrow -ap_x - b > -p_y$
$\Leftrightarrow -ap_x + p_y > b$
$\Leftrightarrow D(P)$ liegt oberhalb $D(\ell)$.

1.5.7 Folgerung: Wenn $p = \ell_1 \cap \ell_2 \Rightarrow D(\ell_1), D(\ell_2) \in D(p)$.
eigentlich "$\Leftrightarrow$"
Bew. siehe Kursordner.

1.5.8 Betrachte folgendes Problem:
Seien $\ell_1 .. \ell_n$ n nicht vertikale Geraden im $\mathbb{R}^2$.
$\ell_i^+ :=$ Halbraum oberhalb von ℓ_i $(i \in \{1..n\})$
$\ell_i^- :=$ Halbraum unterhalb von ℓ_i $(i \in \{1..n\})$.
Sei weiter $m \leq n$.
Nun berechne: $S := \ell_1^+ \cap \ldots \cap \ell_m^+ \cap \ell_{m+1}^- \cap \ldots \cap \ell_n^-$.

$\mapsto$ **1.5.8.1 Beobachtung:** Unser ursprüngliches Problem kann in dieser Weise formuliert werden.

Sei $S^+ := \bigcap_{i=1}^{m} \ell_i^+$ und $S^- := \bigcap_{j=m+1}^{n} \ell_j^-$

$\Rightarrow S = S^+ \cap S^-$
Wir berechnen S^+ und S^- getrennt.
Zunächst S^+ (S^- analog):
S^+ ist ein nach oben unbeschränktes konvexes Polygon.

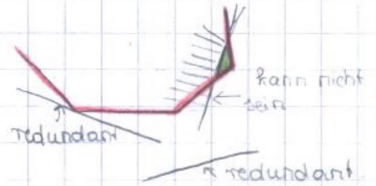
$\mapsto$ **1.5.8.2 Def:** Eine Gerade ℓ_i, $1 \leq i \leq m$ heißt redundant, falls sie nicht zum Rand von S^+ beiträgt, d.h. es gibt keine Kante von S^+ auf ℓ_i.

Beobachtung: Redundante Geraden können ignoriert werden.
Frage: Wie findet man sie? $\rightarrow$ Dualität.

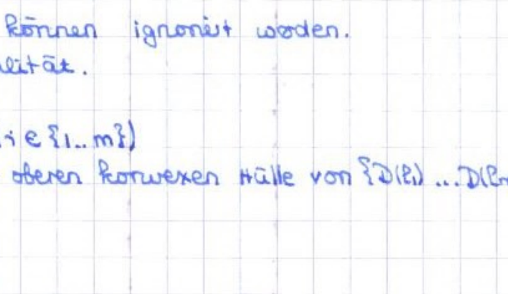
$\mapsto$ **1.5.8.3 Lemma:** ℓ_i ist redundant $(i \in \{1..m\})$
$\Leftrightarrow$ Pkt $D(\ell_i)$ ist keine Ecke der oberen konvexen Hülle von $\{D(\ell_1) \ldots D(\ell_m)\}$.

Beweis:
Vorbemerkung:
Sei ℓ_i redundant
$\Rightarrow S^+ \cap \ell_i = \emptyset$ oder Ecke von S^+
Sei v die Ecke von S^+, die ℓ_i am nächsten liegt.

Bea: Zwei Ecken kann es im Schnitt nicht geben, denn:



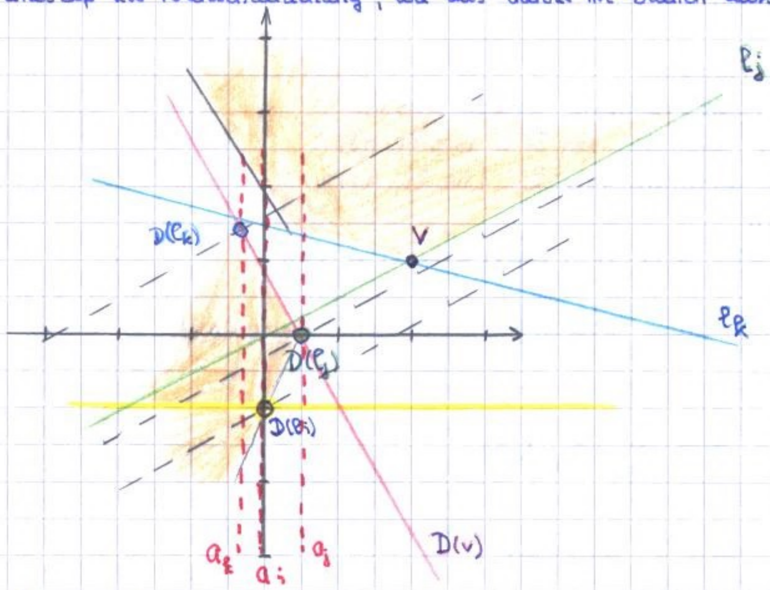
Beobachtung:
1). $\exists$ zwei nicht redundante Geraden ℓ_j und ℓ_k mit $v = \ell_j \cap \ell_k$.
Skizze:
2). Steigung von ℓ_i liegt zw. Steigungen von ℓ_j und ℓ_k, da sonst entweder ℓ_i nicht redundant oder eine andere Ecke als v näher zu ℓ_i liegt.



3). Pkt $D(\ell_i)$ liegt auf oder unterhalb der Geraden $D(v)$
Bea: v ist auf oder oberhalb ℓ_i
$\Leftrightarrow D(v)$ auf oder oberhalb $D(\ell_i)$
$\Leftrightarrow D(\ell_i)$ auf oder unterhalb $D(\ell_i)$
4). $D(\ell_j)$ und $D(\ell_k)$ liegen auf der Geraden $D(v)$
Bea: v liegt auf ℓ_j und ℓ_k
$\Leftrightarrow D(v)$ liegt auf $D(\ell_j)$ und $D(\ell_k)$

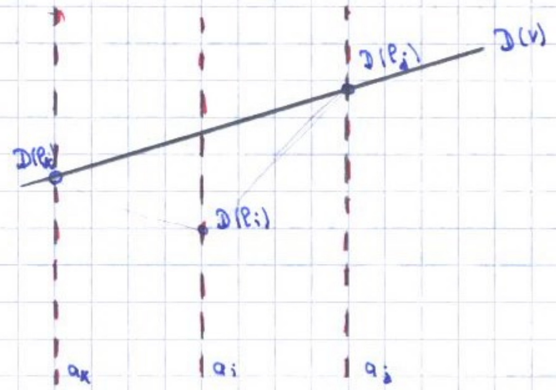
$\rightarrow$

Kleines Bsp zur Veranschaulichung, wie das Ganze im Dualen aussieht:



$\ell_j = \{y : y = \frac{1}{2}x\}$
$\Rightarrow D(\ell_j) = (\frac{1}{2}, 0)$
$\ell_k = \{y : y = -\frac{1}{4}x + \frac{3}{2}\}$
$\Rightarrow D(\ell_k) = (-\frac{1}{4}, \frac{3}{2})$.
$v = (2,1)$
$\Rightarrow D(v) = \{y : y = -2x + 1\}$
$\ell_i = \{y : y = -1\}$
$\Rightarrow D(\ell_i) = (0, -1)$

Dh. also: Situation im Dualen:



Der eigentliche Beweis:

„⇒" sei ℓ_i redundant

z.z. $D(\ell_i)$ ist keine Ecke der oberen $CH(\{D(\ell_1) .. D(\ell_m)\})$

Setze: $D(\ell_i) := (a_i, b_i)$
$D(\ell_j) := (a_j, b_j)$
$D(\ell_k) := (a_k, b_k)$

dh. a_i, a_j und a_k sind Steigungen von ℓ_i, ℓ_j und ℓ_k

Beob. 2) $\Rightarrow a_k \leq a_i \leq a_j$ $\Rightarrow D(\ell_j)$ ist weiter rechts als $D(\ell_i) \Rightarrow D(\ell_i)$ kann nicht mehr die rechteste Ecke der obe. Hülle sein!

Beob. 3) $\Rightarrow D(\ell_i)$ liegt auf oder unterhalb $D(v)$

↳ $D(\ell_k)$ ist weiter links $\Rightarrow D(\ell_i)$ nicht die linkeste Ecke

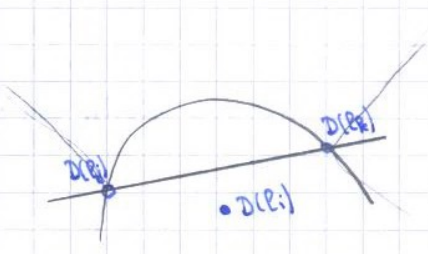
↳ $D(\ell_i)$ auf oder unterhalb $D(v) \Rightarrow D(\ell_i)$ ist nicht Ecke der oberen Hülle.

„⇐" (rückwärts lesen von „⇒")

Sei $D(\ell_i)$ nicht Ecke der oberen Hülle.

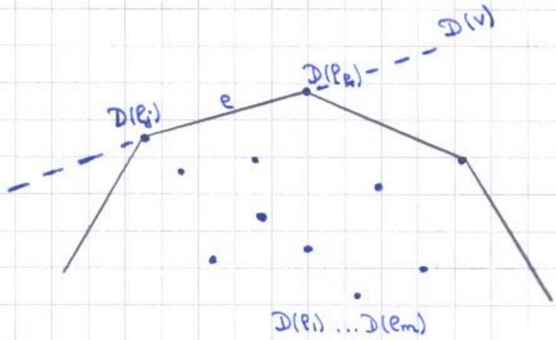
z.z. ℓ_i redundant

$D(\ell_i)$ nicht Ecke der oberen Hülle $\Rightarrow \exists\ D(\ell_j)$ und $D(\ell_k)$ so, dass $D(\ell_i)$ auf oder unterhalb der Geraden durch $D(\ell_j)$ und $D(\ell_k)$ liegt.



1.5.8.4. Lemma: $v \in \mathbb{R}^2$ ist Ecke von S^+

$\Leftrightarrow$ Die Gerade $D(v)$ enthält eine Kante der oberen Hülle von $\{D(\ell_1) \dots D(\ell_m)\}$.



Beweis:

„$\Leftarrow$": $D(v)$ enthält eine Kante e der oberen Hülle von $\{D(\ell_1) \dots D(\ell_m)\}$.

z.z. v ist Ecke von S^+

Sei $e = (D(\ell_j), D(\ell_k))$

Dualität: $v \in \ell_j \cap \ell_k$

obere Hülle $\Rightarrow$ alle Pkte $D(\ell_i)$, $i \in \{1..m\}$ liegen unterhalb oder auf $D(v)$

Dualität $\Rightarrow$ Gerade ℓ_i $(1 \le i \le m)$ liegt unterhalb oder auf v

d.h. v liegt über oder auf ℓ_i.

$\Rightarrow v \in S^+$

v = Schnittpkt zweier Geraden $\Rightarrow$ v ist Ecke von S^+

(Bem: ℓ_j und ℓ_k sind nicht redundant

denn wenn sie es wären $\Rightarrow$ $D(\ell_j)$ und $D(\ell_k)$ wären nicht Ecken von $CH(D(\ell_1) .. D(\ell_m))$ nach L.1.5.8.3 $\Rightarrow$ ↯)

„$\Rightarrow$" v ist Ecke von S^+

$\Rightarrow \quad v = \ell_j \cap \ell_k$

$$\left.\begin{array}{l} 1.5.7 \Rightarrow D(\ell_j), D(\ell_k) \in D(v) \\ \Rightarrow D(v) \text{ enthält also die Kante } e := (D(\ell_j), D(\ell_k)) \end{array}\right\} (*)$$

bleibt z.z: diese Kante ist die Kante der konvexen Hülle.

Es gilt: $v \in S^+$

$\Rightarrow$ v liegt über oder auf ℓ_i $\forall\, i \in \{1..m\}$

$\Rightarrow$ $D(\ell_i)$ liegt unter oder auf $D(v)$ $\forall\, i \in \{1..m\}$

$\Rightarrow$ $D(v)$ enthält eine Kante der oberen Hülle oder $D(v)$ ist redundant

Wegen (*) kann $D(v)$ nicht redundant sein

$\Rightarrow$ $D(v)$ enthält eine Kante der oberen Hülle.

Dieses Lemma liefert einen Algorithmus zur Berechnung von $S^+ = \bigcap_{i=1}^{m} \ell_i^+$

1.5.8.5. Algorithmus:

1. Berechne die dualen Pkte $p_i = D(\ell_i)$, $i = 1..m$ — $O(m)$
2. Berechne die obere konvexe Hülle H von $\{p_1 .. p_m\}$ — $O(m \log m)$

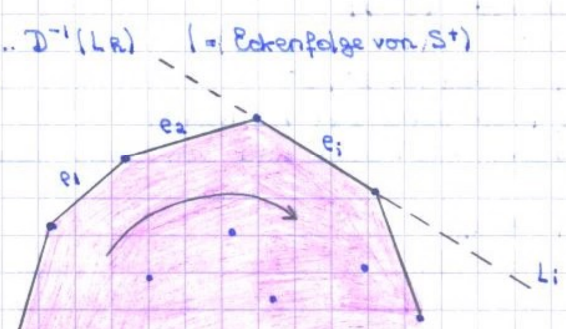
   Seien $e_1 .. e_k$ die Kanten von H von links nach rechts
3. Berechne die Folge der Geraden $L_1 .. L_k$ so, dass $e_i \subset L_i$ — $O(m)$
4. Ausgabe:

   $D^{-1}(L_1) \dots D^{-1}(L_k)$ (= Eckenfolge von S^+) — $O(m)$

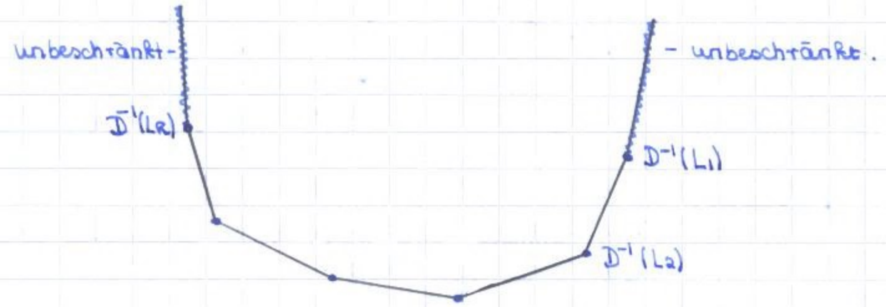
Achtung: $D^{-1} \neq D$.

L_i: $y = ax + b$

$\Rightarrow D^{-1}(L_i) = (-a, b)$



$e_1 \dots e_k$ Kanten von H von links nach rechts.
$\Rightarrow L_1 \dots L_k$ nach Steigungen absteigend sortiert. (siehe Skizze $\Rightarrow$ klar!)
$\Rightarrow$ Ecken von S^+ sind nach x-Koordinaten absteigend sortiert, d.h. von rechts nach links.



Frage: Wie finden wir die beiden unbeschränkten Kanten?
Antwort: Linke Gerade mit minimaler Steigung.
Rechte Gerade mit maximaler Steigung.

1.5.8.6. Zwischenresultat:

$S^+ = \bigcap_{i=1}^{m} l_i^+$ kann in Zeit $O(m \log m)$ berechnet werden.

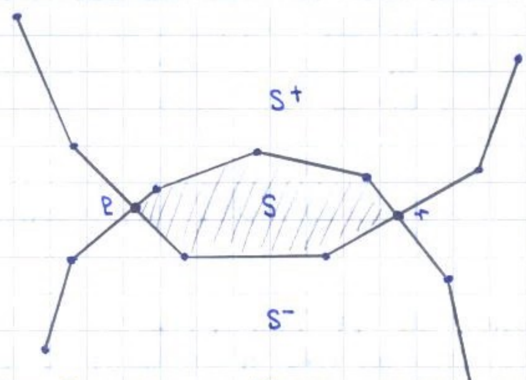
$\underbrace{O(m)}_{D, D^{-1} \dots} + \underbrace{O(m \log m)}_{\text{Graham's Scan.}}$

Falls $l_1 \dots l_m$ nach Steigung sortiert gegeben sind, dann Laufzeit $O(m)$ ($\rightsquigarrow$ Graham's Scan)

$S^- = \bigcap_{j=m+1}^{n} l_j^-$ kann genauso (symmetrisch) berechnet werden.

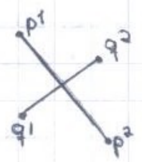
Es bleibt noch $S = S^+ \cap S^-$ auszurechnen.

1.5.8.7. Berechne S.



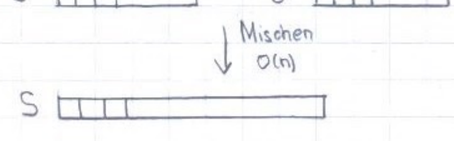
Finde Schnittpkte l und r der Ränder von S^+ und S^-
Achtung: Sonderfälle:

- ex. nicht ($S = \emptyset$)
- $l = r$ ($S = \{l\} = \{r\}$)
- l, r Ecken von S^+, S^-



Allgemein: Innere Schnittpkte von Kanten
Voraussetzung: Die Ränder sind von links nach rechts sortiert.

$\Rightarrow$ Laufzeit: $O(n)$.



und gleichzeitig jeweils das e_i, welches eingefügt wird, in int S^+ und S^- schreiben und vergleichen.
Sobald Änderung $\Rightarrow$ Kante gefunden. $\Rightarrow$ Schnittpkt. berechnen
$\Rightarrow$ Gesamtlaufzeit: $O(n)$.