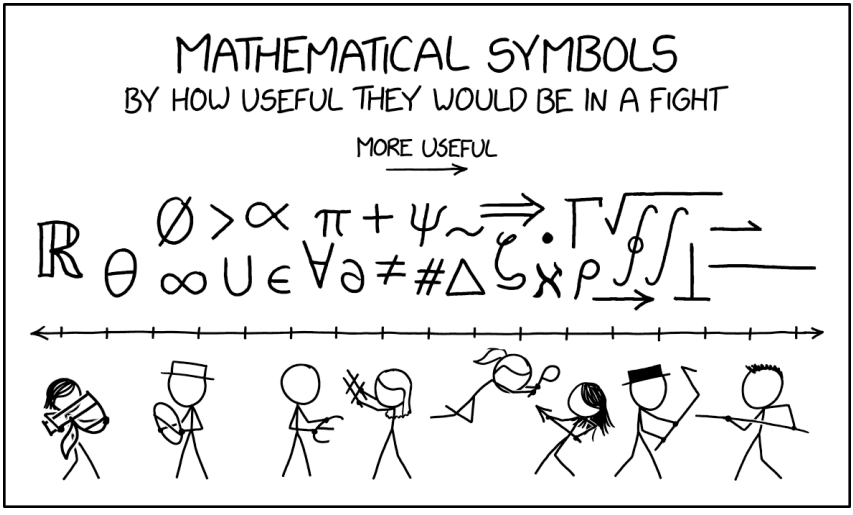
ETH zürich

D&A - Übungsstunde 7 (Osterferien)

Diese Folien wurden nicht oder kaum in einer Übungsstunde behandelt

Comic der Woche



xkcd

Übersicht

Heutiges Programm

Intro
Geometrische Eigenschaften
Konvexe Hülle
Sweepline
Geometrisches Divide & Conquer: Dichtestes Punktepaar
Outro

n.ethz.ch/~agavranovic

- Link zum Material für die Übungsstunden
- Webseite des Assistenten
- Mail an Assistenten

1. Intro

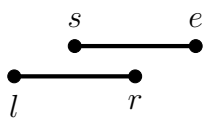
Intro

Osterferien

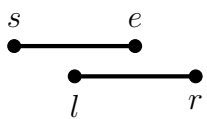
2. Geometrische Eigenschaften

Überlappung von zwei Intervallen

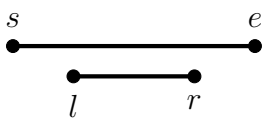
Zwei Intervalle (l, r) und (s, e) überlappen, falls



$l \leq s < r$



$l < e \leq r$



oder $s \leq l \leq r \leq e$

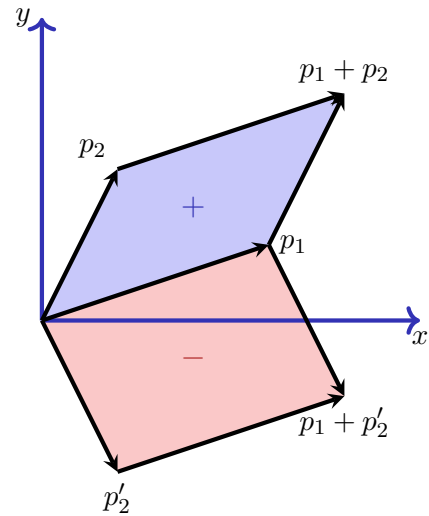
$\Rightarrow$ Ob sich 2 Intervalle überschneiden, kann in konstanter Zeit festgestellt werden.

Eigenschaften von Strecken

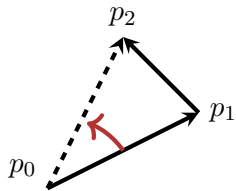
Kreuzprodukt zweier Vektoren $p_1 = (x_1, y_1)$, $p_2 = (x_2, y_2)$ in der Ebene

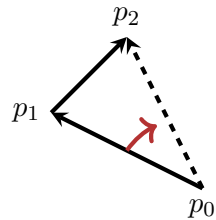
$$p_1 \times p_2 = \det \begin{bmatrix} x_1 & x_2 \\ y_1 & y_2 \end{bmatrix} = x_1 y_2 - x_2 y_1$$

Vorzeichenbehafteter Flächeninhalt des Parallelogramms



Abbiegerichtung



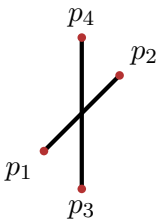


nach links:
$(p_1 - p_0) \times (p_2 - p_0) > 0$

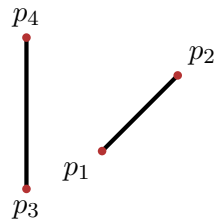
nach rechts:
$(p_1 - p_0) \times (p_2 - p_0) < 0$

Schnitt zweier Strecken

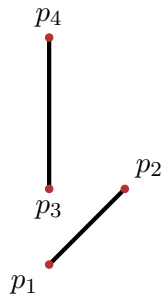
Feststellen, dass zwei Strecken sich schneiden, ohne Schnittpunkt zu berechnen (Division!)?



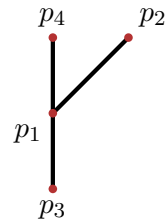
Schnitt: p_1 und p_2 gegenüber bzgl. $\overline{p_3p_4}$ und p_3, p_4 gegenüber bzgl. $\overline{p_1p_2}$



Kein Schnitt: p_1 und p_2 auf der gleichen Seite von $\overline{p_3p_4}$



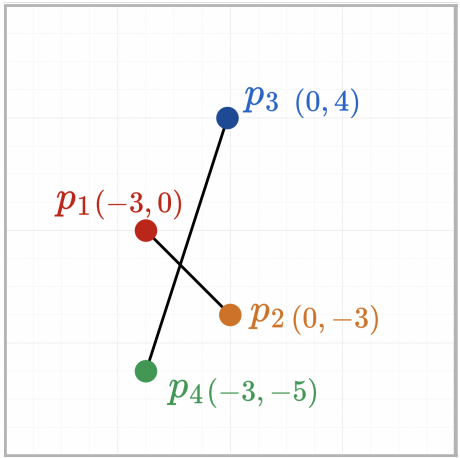
Kein Schnitt: p_3 und p_4 auf der gleichen Seite von $\overline{p_1p_2}$



Schnitt: p_1 auf $\overline{p_3p_4}$

Schnitt zweier Strecken

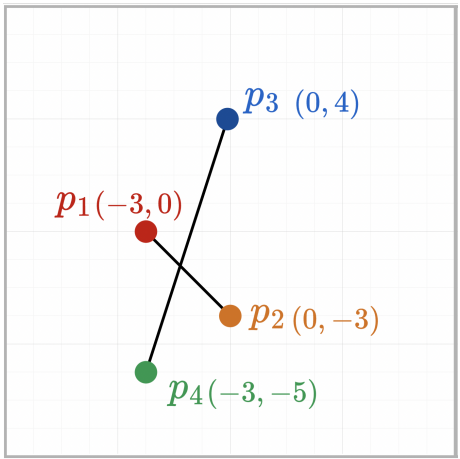
Teilaufgabe (a)



Schnitt oder kein Schnitt?

Schnitt zweier Strecken

Teilaufgabe (a)



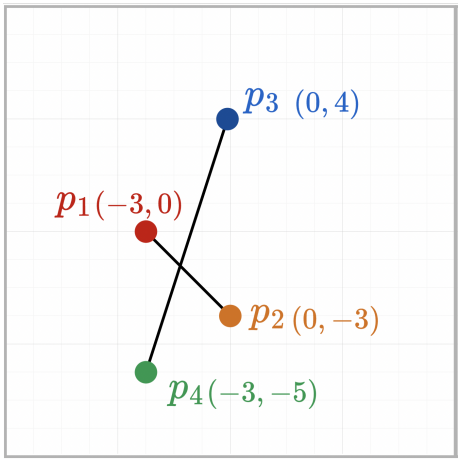
Schnitt oder kein Schnitt?

Schnitt

p_1, p_2 gegenüber bzgl. $\overline{p_4 p_3}$
und p_3, p_4 gegenüber bzgl. $\overline{p_1 p_2}$.

Schnitt zweier Strecken

Teilaufgabe (a)

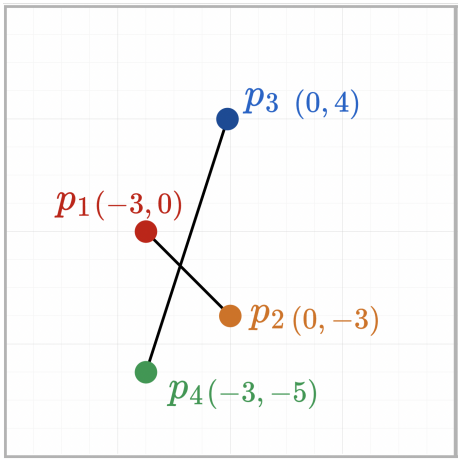


$$(p_3 - p_4) \times (p_1 - p_4) =$$
$$= ((0,4) - (-3,-5)) \times ((-3,0) - (-3,-5)) = (3,9) \times (0,5) = det \begin{bmatrix} 3 & 0 \\ 9 & 5 \end{bmatrix}$$
$$= (3)(5) - (0)(9) = \mathbf{15} > \mathbf{0}.$$

$$(p_3 - p_4) \times (p_2 - p_4) =$$
$$= ((0,4) - (-3,-5)) \times ((0,-3) - (-3,-5))$$
$$= (3,9) \times (3,2) = det \begin{bmatrix} 3 & 3 \\ 9 & 2 \end{bmatrix}$$
$$= (3)(2) - (3)(9) = \mathbf{-21} < \mathbf{0}.$$

Schnitt zweier Strecken

Teilaufgabe (a)

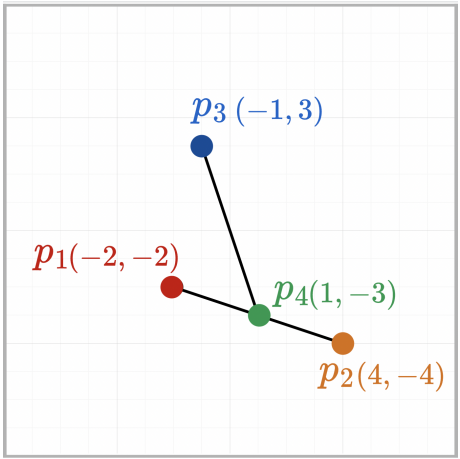


und $(p_2 - p_1) \times (p_3 - p_1) =$
$= ((0,-3) - (-3,0)) \times ((0,4) - (-3,0))$
$= (3,-3) \times (3,4) = det \begin{bmatrix} 3 & 3 \\ -3 & 4 \end{bmatrix}$
$= (3)(4) - (3)(-3) = \mathbf{21} > \mathbf{0}.$

$(p_2 - p_1) \times (p_4 - p_1) =$
$= ((0,-3) - (-3,0)) \times ((-3,-5) - (-3,0))$
$= (3,-3) \times (0,-5) = det \begin{bmatrix} 3 & 0 \\ -3 & -5 \end{bmatrix}$
$= (3)(-5) - (0)(-3) = \mathbf{-15} < \mathbf{0}.$

Schnitt zweier Strecken

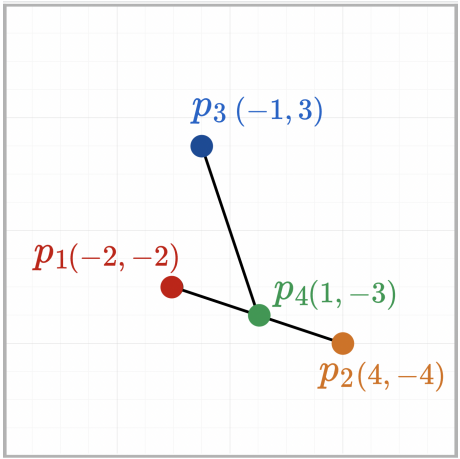
Teilaufgabe (b)



Schnitt oder kein Schnitt?

Schnitt zweier Strecken

Teilaufgabe (b)



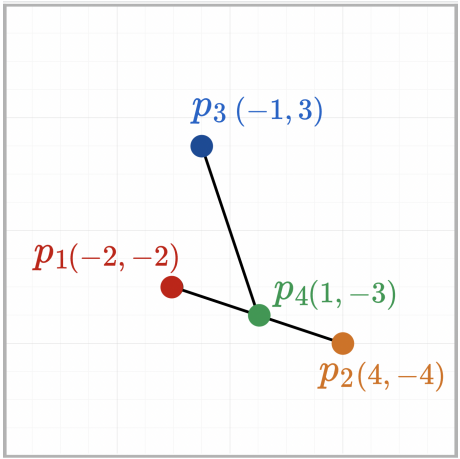
Schnitt oder kein Schnitt?

Schnitt

p_4 ist aus zwei Gründen auf $\overline{p_1 p_2}$.

Schnitt zweier Strecken

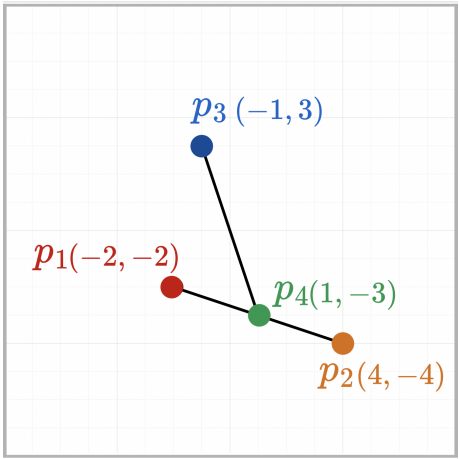
Teilaufgabe (b)



Erstens,
$(p_2 - p_1) \times (p_4 - p_1) =$
$= ((4,-4) - (-2,-2)) \times ((1,-3) - (-2,-2))$
$= (6,-2) \times (3,-1) = det \begin{bmatrix} 6 & 3 \\ -2 & -1 \end{bmatrix}$
$= (6)(-1) - (3)(-2) = \mathbf{0}.$

Schnitt zweier Strecken

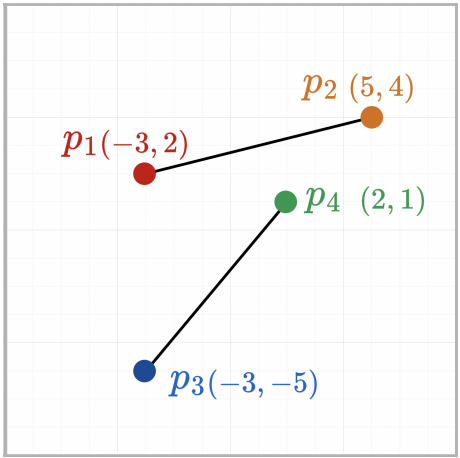
Teilaufgabe (b)



Dies zeigt jedoch nur, dass p_4 auf der von $\overline{p_1p_2}$ gebildeten Gerade liegt.
Für die Schlussfolgerung, dass p_4 in $\overline{p_1p_2}$ liegt, ist zu beachten, dass
$-2 = p_1[0] \leq 1 = p_4[0] \leq 4 = p_2[0]$
und
$-4 = p_2[1] \leq -3 = p_4[1] \leq -2 = p_1[1]$.

Schnitt zweier Strecken

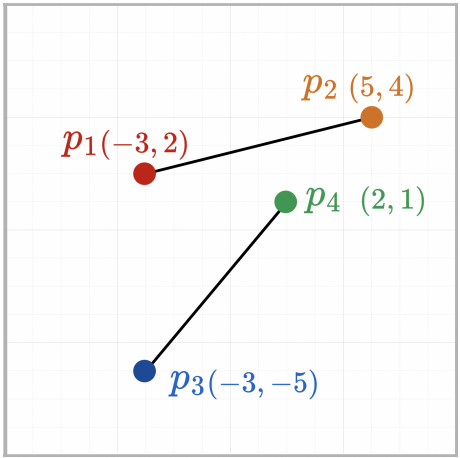
Teilaufgabe (c)



Schnitt oder kein Schnitt?

Schnitt zweier Strecken

Teilaufgabe (c)



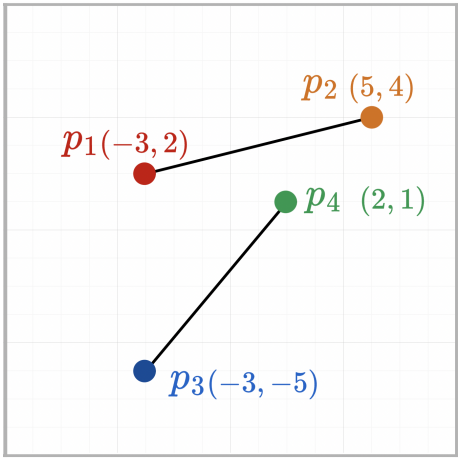
Schnitt oder kein Schnitt?

Kein Schnitt

p_3 und p_4 auf der gleichen Seite von $\overline{p_1p_2}$.

Schnitt zweier Strecken

Teilaufgabe (c)

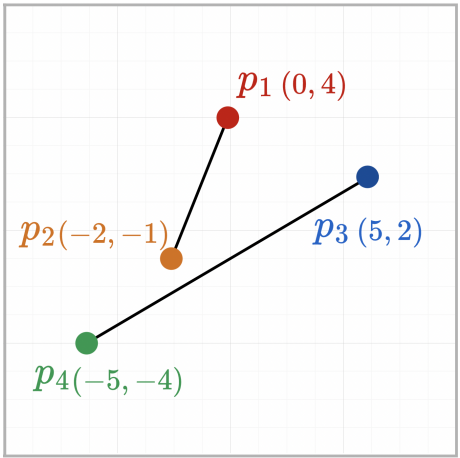


$$(p_2 - p_1) \times (p_3 - p_1) =$$
$$= ((5,4) - (-3,2)) \times ((-3,-5) - (-3,2))$$
$$= (8,2) \times (0,-7) = det \begin{bmatrix} 8 & 0 \\ 2 & -7 \end{bmatrix}$$
$$= (8)(-7) - (0)(2) = -\mathbf{56} < \mathbf{0}.$$

$$(p_2 - p_1) \times (p_4 - p_1) =$$
$$= ((5,4) - (-3,2)) \times ((2,1) - (-3,2))$$
$$= (8,2) \times (5,-1) = det \begin{bmatrix} 8 & 5 \\ 2 & -1 \end{bmatrix}$$
$$= (8)(-1) - (5)(2) = -\mathbf{18} < \mathbf{0}.$$

Schnitt zweier Strecken

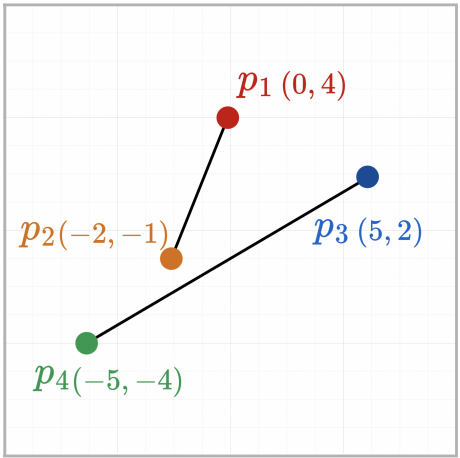
Teilaufgabe (d)



Schnitt oder kein Schnitt?

Schnitt zweier Strecken

Teilaufgabe (d)



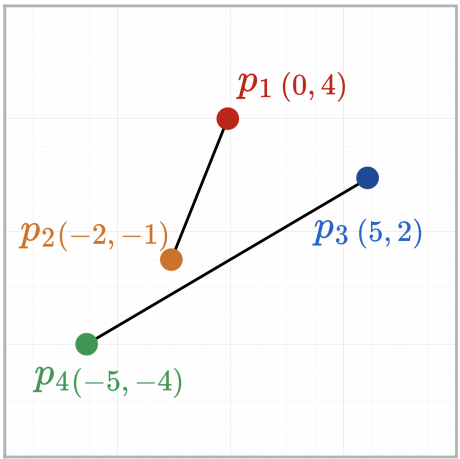
Schnitt oder kein Schnitt?

Kein Schnitt

p_1 und p_2 auf der gleichen Seite von $\overline{p_4p_3}$.

Schnitt zweier Strecken

Teilaufgabe (d)



$$(p_3 - p_4) \times (p_1 - p_4) =$$
$$= ((5,2) - (-5,-4)) \times ((0,4) - (-5,-4))$$
$$= (10,6) \times (5,8) = det \begin{bmatrix} 10 & 5 \\ 6 & 8 \end{bmatrix}$$
$$= (10)(8) - (5)(6) = \mathbf{60} > \mathbf{0}.$$

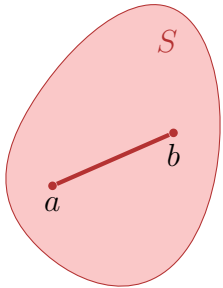
$$(p_3 - p_4) \times (p_2 - p_4) =$$
$$= ((5,2) - (-5,-4)) \times ((-2,-1) - (-5,-4))$$
$$= (10,6) \times (3,3) = det \begin{bmatrix} 10 & 3 \\ 6 & 3 \end{bmatrix}$$
$$= (10)(3) - (6)(3) = \mathbf{12} > \mathbf{0}.$$

3. Konvexe Hülle

Konvexe Hülle

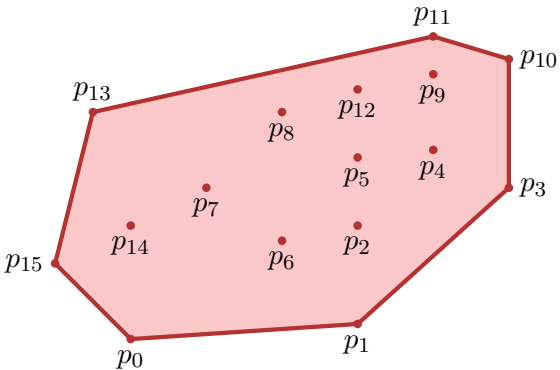
Teilmenge S eines reellen Vektorraums heisst **konvex**, wenn für alle $a, b \in S$ und jedes $\lambda \in [0, 1]$:

$$\lambda a + (1 - \lambda)b \in S$$



Konvexe Hülle

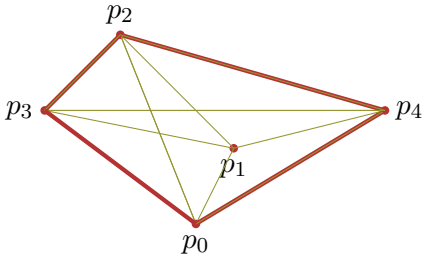
Konvexe Hülle $H(Q)$ einer Menge Q von Punkten: kleinstes konvexes Polygon P, so dass jeder Punkt von Q auf P oder im Inneren von P liegt.



Jarvis Marsch / Gift Wrapping Algorithmus

1. Starte mit Extrempunkt (z.B. dem untersten Punkt) $p = p_0$
2. Suche Punkt q, so dass $\overline{pq}$ am weitesten rechts liegende Gerade, d.h. jeder andere Punkt liegt links von der Geraden $\overline{pq}$ (oder auf der Geraden näher bei p).
3. Fahre mit $p \leftarrow q$ bei (2) weiter, bis $p = p_0$.

Illustration Jarvis



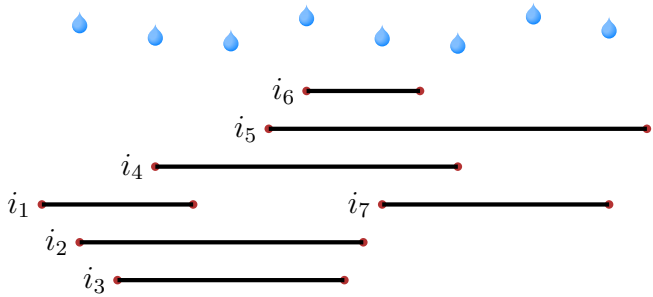
1. Setze $H \rightarrow \emptyset$.
2. Finde den untesten Punkt q.
3. Finde den am weitesten rechts liegenden Punkt p aus der Perspektive von q
4. Füge p zu H hinzu.
5. Setze $q \leftarrow p$ und wiederhole ab Schritt 3 bis q wieder der unteste Punkt ist
6. H ist die konvexe Hülle.

Graham Scan

- Graham Scan: Ein weiterer Algorithmus zur Berechnung der konvexen Hülle
- Siehe Implementierung in den Vorlesungsfolien
- Zeitkomplexität:
  - Jarvis March: $\mathcal{O}(h \cdot n)$, wobei h die Anzahl der Eckpunkte auf der konvexen Hülle ist
  - Graham Scan: $\mathcal{O}(n \log n)$
- **Frage**: Wann ist der Jarvis March ein besserer Algorithmus?
- **Antwort**: Jarvis March ist besser, wenn h im Vergleich zu n klein ist, da seine Zeitkomplexität von der Anzahl der Eckpunkte auf der konvexen Hülle abhängt.
- Kommentar: Chan's Algorithm verbessert beide, wird aber in diesem Kurs nicht gelehrt.

4. Sweepline

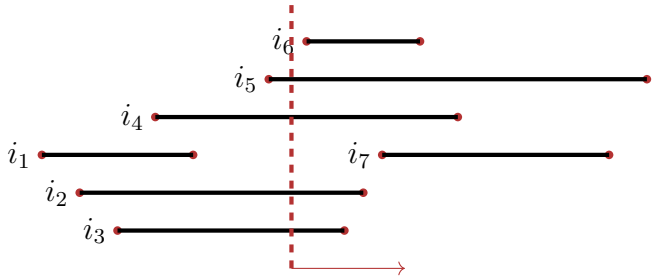
Vorbereitung: Überlappende Intervalle



Fragen:

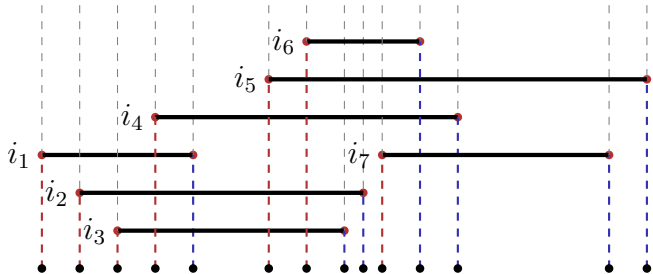
- Wie viele Intervalle überlappen maximal?
- Welche Intervalle werden (nicht) nass?
- Welche Intervalle liegen unmittelbar übereinander

Vorbereitung: Überlappende Intervalle



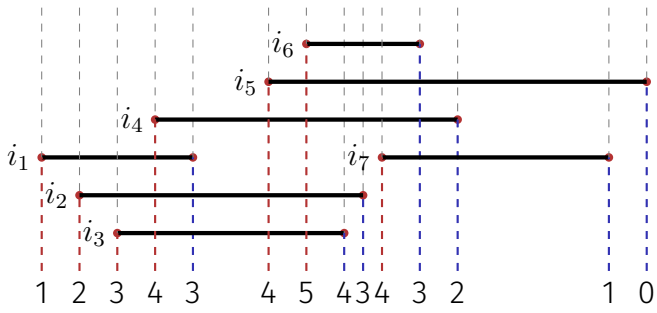
Idee Sweepline: Vertikale Linie, wandert in x-Richtung, beobachtet die geometrichen Objekte.

Vorbereitung: Überlappende Intervalle



Ereignisliste: Liste von Punkten, an denen sich der Zustand, den die Sweepline beobachtet, ändert.

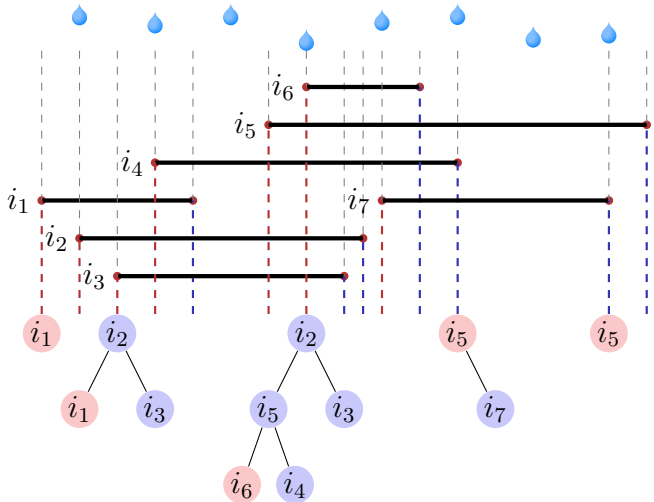
Vorbereitung: Überlappende Intervalle



F: Wie viele Intervalle überlappen maximal? Sweepline führt Zähler der am linken (rechten) Endpunkt eines Intervalls inkrementiert (dekrementiert) wird.

A: Maximaler Zählerstand

Vorbereitung: Überlappende Intervalle

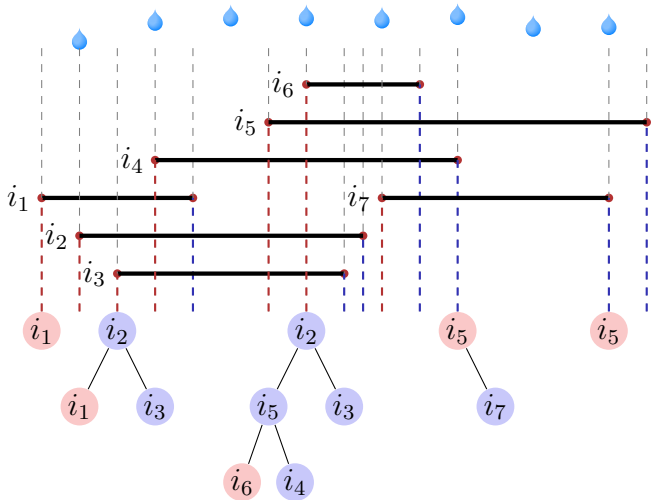


F: Welche Intervalle werden nass?

Sweepline führt einen Suchbaum der die Intervalle nach vertikaler Anordnung enthält.

A: Intervalle, die irgendwann im Baum ganz links stehen.

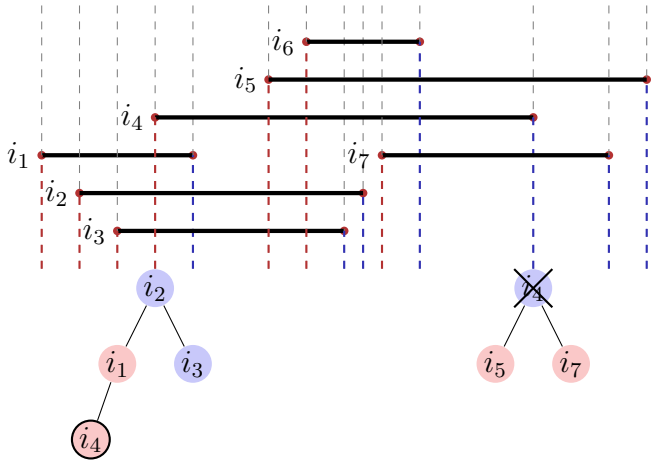
Vorbereitung: Überlappende Intervalle



F: Wieso verwenden wir nicht einen Max-Heap (statt eines BSTs)?

A: Löschen eines beliebigen Elements (nicht das Maximum) von einem Heap ist nicht einfach!

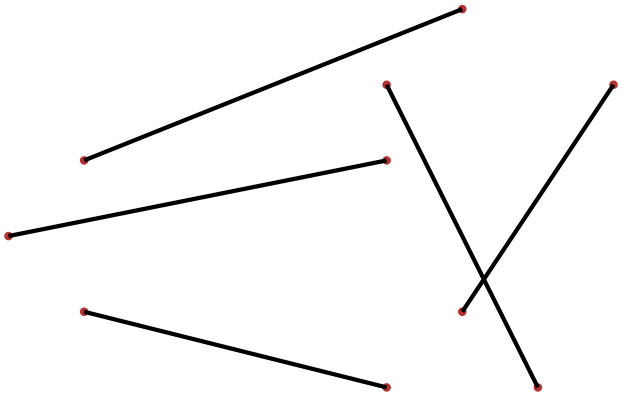
Vorbereitung: Überlappende Intervalle



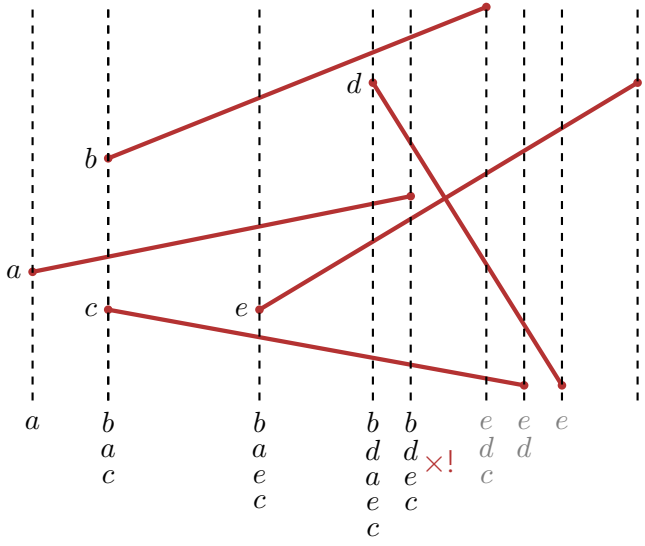
F: Welche Intervalle sind benachbart?

A: Intervalle, die beim Intervallstart nebeneinander liegen oder die beim Intervallende eines anderen Intervalls zu Nachbarn werden.

Schnittpunkt vieler Strecken



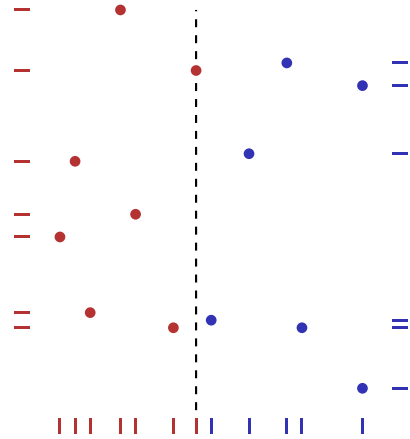
Schnittpunkt von Strecken



5. Geometrisches Divide & Conquer: Dichtestes Punktepaar

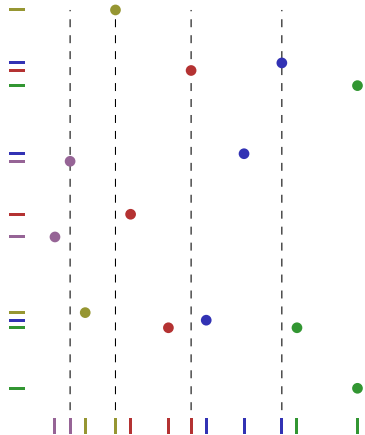
Divide And Conquer: Dichtestes Punktepaar

- Punktmenge P, zu Beginn $P \leftarrow Q$
- Arrays X und Y, welche die Punkte aus P enthalten, sortiert nach x- bzw. nach y-Koordinate.
- Teile Punktmenge ein in zwei (annähernd) gleich grosse Mengen P_L und P_R, getrennt durch vertikale Gerade durch einen Punkt von P.
- Teile Arrays X und Y entsprechend in X_L, X_R. Y_L und Y_R.



Divide And **Conquer**: Dichtestes Punktepaar

- Rekursiver Aufruf jeweils mit P_L, X_L, Y_L und P_R, X_R, Y_R. Erhalte minimale Abstände δ_L, δ_R.
- (Wenn nur noch $k \leq 2$ Punkte: berechne direkt minimalen Abstand)
- Nach rekursivem Aufruf $\delta = \min(\delta_L, \delta_R)$. Kombiniere und gib bestes Resultat zurück.



Implementierung

- Ziel: Rekursionsgleichung (Laufzeit) $T(n) = 2 \cdot T(\frac{n}{2}) + \mathcal{O}(n)$.
- Nichttrivial: nur Arrays Y und Y'
- Idee: Merge umgekehrt: durchlaufe (nach y-Koordinaten vorsortiertes) Y und hänge dem Auswahlkriterium der x-Koordinate folgend an Y_L und Y_R an. Genauso für Y'. Laufzeit $\mathcal{O}(|Y|)$.

Gesamtlaufzeit: $\mathcal{O}(n \log n)$.

Fragen

- Wie vergleicht sich der Algorithmus mit einem Brute-Force-Ansatz?
  - Divide and Conquer reduziert die Problemgröße in jedem Schritt und hat eine Zeitkomplexität von $\mathcal{O}(n)$, während ein Brute-Force-Ansatz eine Zeitkomplexität von $\mathcal{O}(n^2)$ hat.
- Warum vermeiden wir das Sortieren in jedem Schritt der Rekursion?
  - Sortieren hat eine Zeitkomplexität von $\mathcal{O}(n \log n)$, während die Zeitkomplexität des Conquer-Schritts linear sein sollte.

6. Outro

Bis zum nächsten Mal

Schöne Osterferien!