5. Konvexe Hüllen

Definitionen und Eigenschaften (Hinrichs 2001)

Eine Menge $S \subseteq \mathbb{R}^d$ heißt *konvex*, wenn mit $p \in S$ und $q \in S$ auch das Segment, das p und q verbindet, in S liegt, d.h. $\overline{pq} \subseteq S$. $\overline{pq}$ ist dabei die Menge aller Punkte der Form $\alpha p + \beta q$ mit $\alpha \geq 0$, $\beta \geq 0$ und $\alpha + \beta = 1$.
Eine *Konvexkombination* von Punkten $p_1, p_2, \ldots, p_k$ ist eine Summe der Form $\sum_{i=1}^{k} \alpha_i p_i$ mit $\alpha_i \geq 0$, $\sum_{i=1}^{k} \alpha_i = 1$.

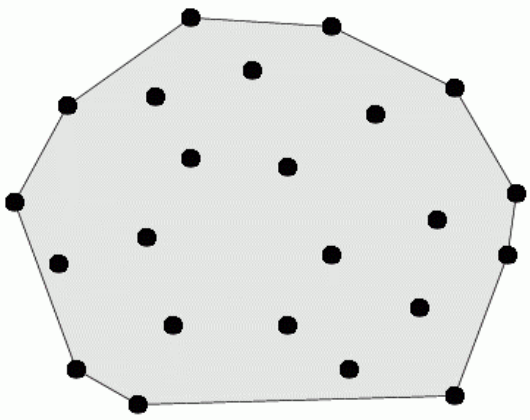
- Ein Liniensegment besteht somit aus allen Konvexkombinationen seiner Endpunkte, und ein Dreieck besteht aus allen Konvexkombinationen seiner 3 Eckpunkte. Ein Tetraeder besteht aus allen Konvexkombinationen seiner 4 Ecken.

Definition 3.1.2:

Die konvexe Hülle einer Menge $S \subseteq \mathbb{R}^d$ ist

$$ch(S) = conv(S) = \bigcap_{\substack{T \supset S \\ T \text{ konvex}}} T \quad .$$

- Durchschnitt konvexer Mengen konvex $\Rightarrow$ ch(S) konvex $\Rightarrow$ konvexe Hülle einer Menge S ist die kleinste konvexe Menge, die S umfaßt.



Analogie (?): Nägel und Gummiband

Charakterisierung der konvexen Hülle:

1. Die konvexe Hülle von S ist die Menge aller Konvexkombinationen von Punkten aus S.
2. Die konvexe Hülle von $S \subseteq \mathbb{R}^d$ ist die Menge aller Konvexkombinationen von d+1 (oder weniger) Punkten aus S.
3. Die konvexe Hülle von $S \subseteq \mathbb{R}^d$ ist der Schnitt aller Halbräume, die S umfassen. Ein Halbraum im 2-d ist eine Halbebene, d.h. die Menge aller Punkte, die auf oder auf einer bestimmten Seite einer Geraden liegen. Ein Halbraum im 3-d ist die Menge aller Punkte, die auf oder auf einer bestimmten Seite einer Ebene liegen. Die Verallgemeinerung des Begriffs Halbraum auf den $\mathbb{R}^d$ ist offensichtlich.
4. Die konvexe Hülle einer endlichen Menge von Punkten $S \subseteq \mathbb{R}^2$ ist das kleinste konvexe Polygon P, das S umfaßt, d.h. es gibt kein anderes konvexes Polygon P' mit $P \supset P' \supseteq S$, $P \neq P'$.
5. Die konvexe Hülle einer endlichen Menge von Punkten $S \subseteq \mathbb{R}^2$ ist das S umfassende konvexe Polygon P mit der kleinsten Fläche.
6. Die konvexe Hülle einer endlichen Menge von Punkten $S \subseteq \mathbb{R}^2$ ist das S umfassende konvexe Polygon P mit dem kleinsten Umfang.
7. Die konvexe Hülle einer Menge von Punkten $S \subseteq \mathbb{R}^2$ ist die Vereinigung aller Dreiecke, die durch Punkte aus S aufgespannt werden. Dies ist eine Reformulierung von 2) für d = 2.

wir beweisen einen Teil der Aussage 4:

Die konvexe Hülle *ch*(*S*) einer endlichen Menge *S* im R^2, |*S*| = *n*, ist ein konvexes Polygon, dessen Ecken zu *S* gehören.

Beweis: Induktion über n

$n \leq 2$: trivial

$n \geq 3$: Wähle $p \in S$, betrachte $S' = S \setminus \{p\}$

IV: $ch(S')$ ist konvexes Polygon, dessen Ecken zu S' gehören

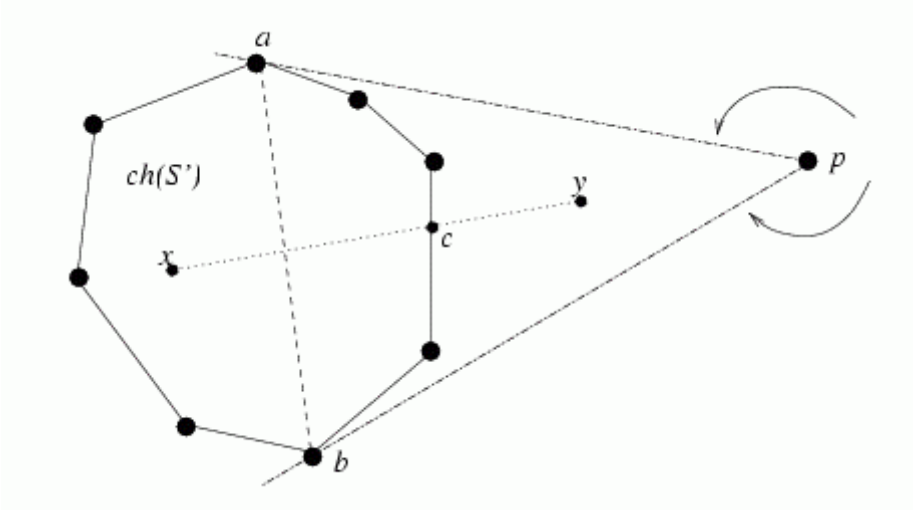
1. Fall: $p \in ch(S')$ $\surd$

2. Fall: $p \notin ch(S')$:

drehe Strahl ab p, der $ch(S')$ nicht schneidet,

- im GUZS, bis Berührpunkt a mit $ch(S')$ gefunden

- im UZS, bis Berührpunkt b mit $ch(S')$ gefunden



zu zeigen: $ch(S) = ch(S') \cup \triangle(a, b, p)$

$\supseteq$ $\surd$ nach Definition $ch(S)$

Konvexität von $ch(S') \cup \triangle(a, b, p)$:

sei $x \in ch(S')$, $y \in \triangle(a, b, p) \setminus ch(S')$

Sei nun *c* der Schnittpunkt der Strecke *xy* mit dem Rand von *ch*(*S*').

$c \in \triangle(a, b, p)$, da $\overline{ap}$ und $\overline{bp}$ Teile von Tangenten an $ch(S')$

$\rightarrow$ $\quad \overline{xc} \in ch(S')$, $\overline{cy} \in \triangle(a, b, p)$ $\square$

Verallgemeinerung der "Ecken" von Polygonen auf beliebige konvexe Mengen:

Definition:

Ein Punkt p einer konvexen Menge C heißt *Extremalpunkt* von C, wenn es keine zwei von p verschiedenen Punkte $q_1, q_2 \in C$ gibt, so daß p auf dem offenen Liniensegment $\overline{q_1 q_2}$ liegt.

- Die Menge ext(S) der Extremalpunkte der konvexen Hülle ch(S) einer endlichen Menge $S \subseteq \mathbb{R}^2$ ist die kleinste Teilmenge $T \subseteq S$ mit ch(T) = ch(S).
- Annahme: Die konvexe Hülle ch(S) einer endlichen Punktmenge $S \subseteq \mathbb{R}^2$ wird beschrieben durch die Eckpunkte ihres Randpolygons, wobei diese in der Reihenfolge angegeben werden müssen, in der man sie beim Durchlaufen des Randes entgegen dem Uhrzeigersinn antrifft.
- Sei $S \subseteq \mathbb{R}^2$, |S| = n.
  Problem CH: Bestimme ch(S).
  Problem EXT: Bestimme ext(S).
- Offensichtlich gilt: EXT $\rightarrow_{O(1)}$ CH.

Untere Schranke für die Berechnung der konvexen Hülle

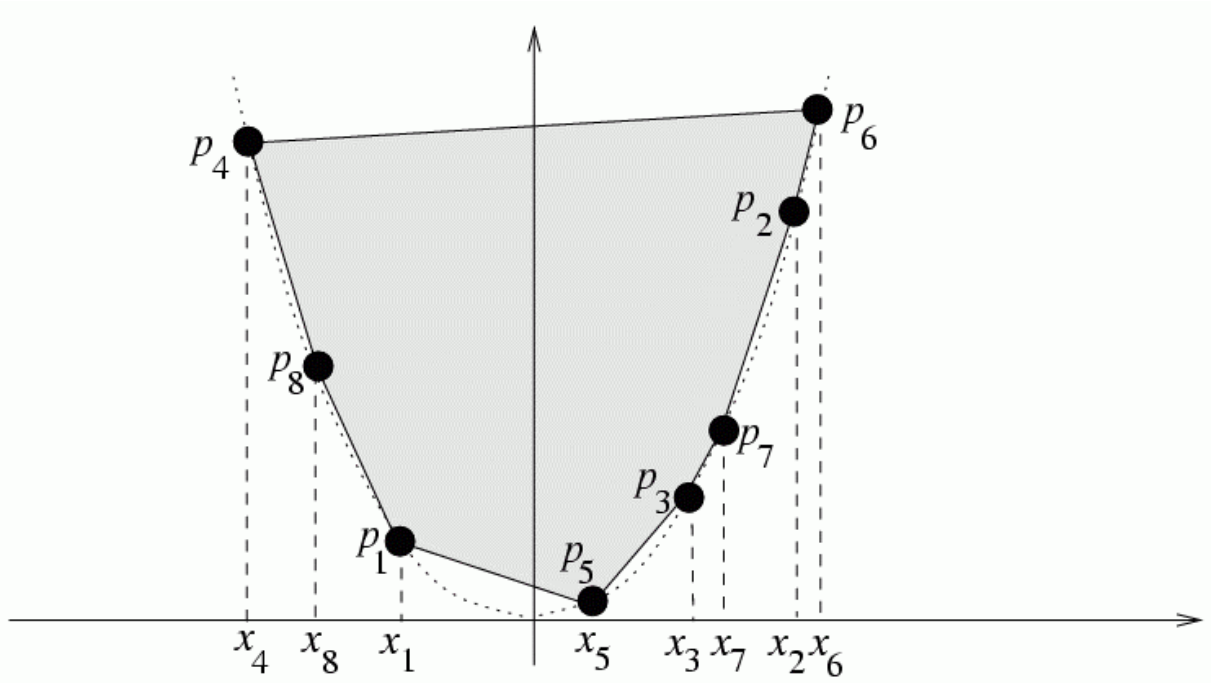
Satz: Die Konstruktion der konvexen Hülle einer endlichen Menge *S* mit |*S*| = *n* benötigt die Zeit $\Omega(n \log n)$.

Beweis: Reduktion auf Sortieren von n Zahlen

Sei A beliebiger Algorithmus zur Konstruktion der konvexen Hülle

Gegeben n reelle Zahlen $x_1, ..., x_n$

Setze $S = \{p_i := (x_i, x_i^2) : \ i = 1, ..., n\}$



Mit A konstruiere $ch(S)$: alle p_i als Ecken!

Lineares Durchlaufen der Ecken von $ch(S)$,

beginnend beim p_i mit kleinster x–Koordinate,

ergibt aufsteigende Sortierung der x_i in Linearzeit.

Wäre A schneller als $O(n \log n)$, könnte man entsprechend auch schneller sortieren $\rightarrow$ Widerspruch! $\square$

Somit:

- Jeder Algorithmus, der das "Konvexe Hülle" Problem löst, benötigt mindestens $\Omega(n \cdot \log n)$ Zeit.
- Man kann zeigen, daß im algebraischen Entscheidungsbaum-modell $\Omega(n \cdot \log n)$ auch eine untere Schranke für EXT ist, EXT also nicht einfacher zu lösen ist als CH.

- Sei $S \subseteq \mathbb{R}^2$, $|S| = n$.

Naive Algorithmen: Sammeln von Ideen

- Identifikation der nicht-extremalen Punkte reicht aus, um ext(S) zu bestimmen.

Satz:

Ein Punkt $p \in S$ ist nicht-extremal genau dann, wenn er innerhalb eines abgeschlossenen Dreiecks liegt, dessen Eckpunkte aus S und verschieden von p sind.

Algorithmus EXT-PUNKTE zur Bestimmung von ch(S):

1. ext(S) := S;
2. Für jedes der $\binom{n}{3}$ Dreiecke, die durch Punkte aus S gebildet werden, entferne aus ext(S) alle die Punkte, die im Innern des Dreiecks liegen.
3. Bestimme einen Punkt q im Innern von ext(S), z.B. den Centroid q von ext(S).
4. Sortiere alle Punkte aus ext(S) bzgl. ihres Polarwinkels um q.

- $O(n^3)$ Dreiecke und O(n) Punkte $\Rightarrow$ Schritt 2 benötigt $O(n^4)$ Zeit
- Schritt 3 benötigt O(n) Zeit
- Schritt 4 benötigt O(n log n) Zeit

$\Rightarrow$ Algorithmus EXT-PUNKTE bestimmt konvexe Hülle von n Punkten in $O(n^4)$ Zeit.

Definition:

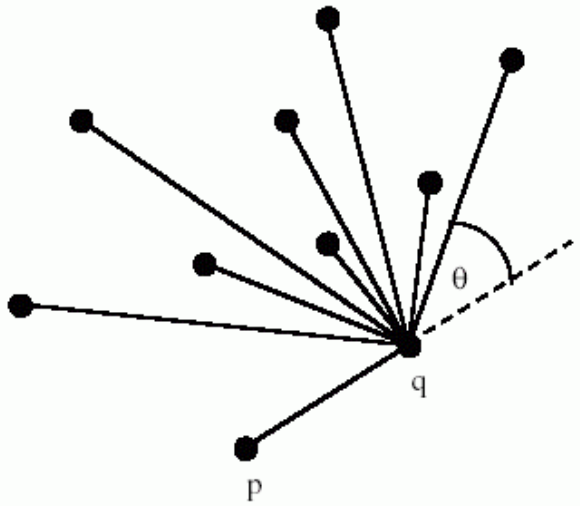
Eine Kante, die von zwei Punkten aus S begrenzt wird, heißt extremal, wenn alle Punkte von S auf der Kante oder auf einer Seite der durch die Kante festgelegten Geraden liegen.

- Jede extremale Kante ist Kante der konvexen Hülle ch(S).
- Algorithmus EXT-KANTEN bestimmt die konvexe Hülle ch(S) mittels der extremalen Kanten:
  1. Bestimme für jede Kante $\overline{p_i p_j}$, ob alle übrigen Punkte von S auf $\overline{p_i p_j}$ oder zur Linken der durch $\overline{p_i p_j}$ festgelegten Geraden liegen. Ist dies der Fall, so ist $\overline{p_i p_j}$ extremale Kante.
  2. Verkette die extremalen Kanten, um ch(S) zu erhalten.
- Für jede der $O(n^2)$ möglichen Kanten müssen alle n Punkte überprüft werden $\Rightarrow$ Schritt 1 benötigt $O(n^3)$ Zeit
- Schritt 2 benötigt O(n) Zeit

$\Rightarrow$ Algorithmus EXT-KANTEN bestimmt konvexe Hülle von n Punkten in $O(n^3)$ Zeit.

Jarvis's march

- Algorithmus von Jarvis (1973), auch bekannt als gift wrapping Methode
- Nutze nach Bestimmung einer Kante $\overline{pq}$ der konvexen Hülle aus, daß eine weitere Kante existieren muß, die q als einen ihrer Endpunkte besitzt. Diese Kante ist diejenige, die mit $\overline{pq}$ den kleinsten Winkel θ bildet:



vom aktuellen Endpunkt aus wird zu jedem der übrigen Punkte der Winkel bestimmt; der Punkt mit dem kleinsten Winkel bestimmt die nächste (Extrem-) Kante, die zur konvexen Hülle gehört.

Startpunkt: Punkt mit kleinster *y*-Koordinate

Anschauliche Vorstellung: ein Band wird um die Menge herumgelegt und, ausgehend vom Startpunkt, jeweils zum Punkt mit dem kleinsten Winkel "straffgezogen".

verallgemeinerungsfähig auf höhere Dim. $\Rightarrow$ dann nicht Band, sondern "Fläche" zum Einwickeln $\Rightarrow$ daher der Name "gift wrapping".

Pseudocode:

Algorithm: GIFT WRAPPING
Find the lowest point (smallest y coordinate).
Let i_0 be its index, and set $i \leftarrow i_0$.
repeat
    for each $j \neq i$ do
        Compute counterclockwise angle θ from previous hull edge.
    Let k be the index of the point with the smallest θ.
    Output (p_i, p_k) as a hull edge.
    $i \leftarrow k$
until $i = i_0$

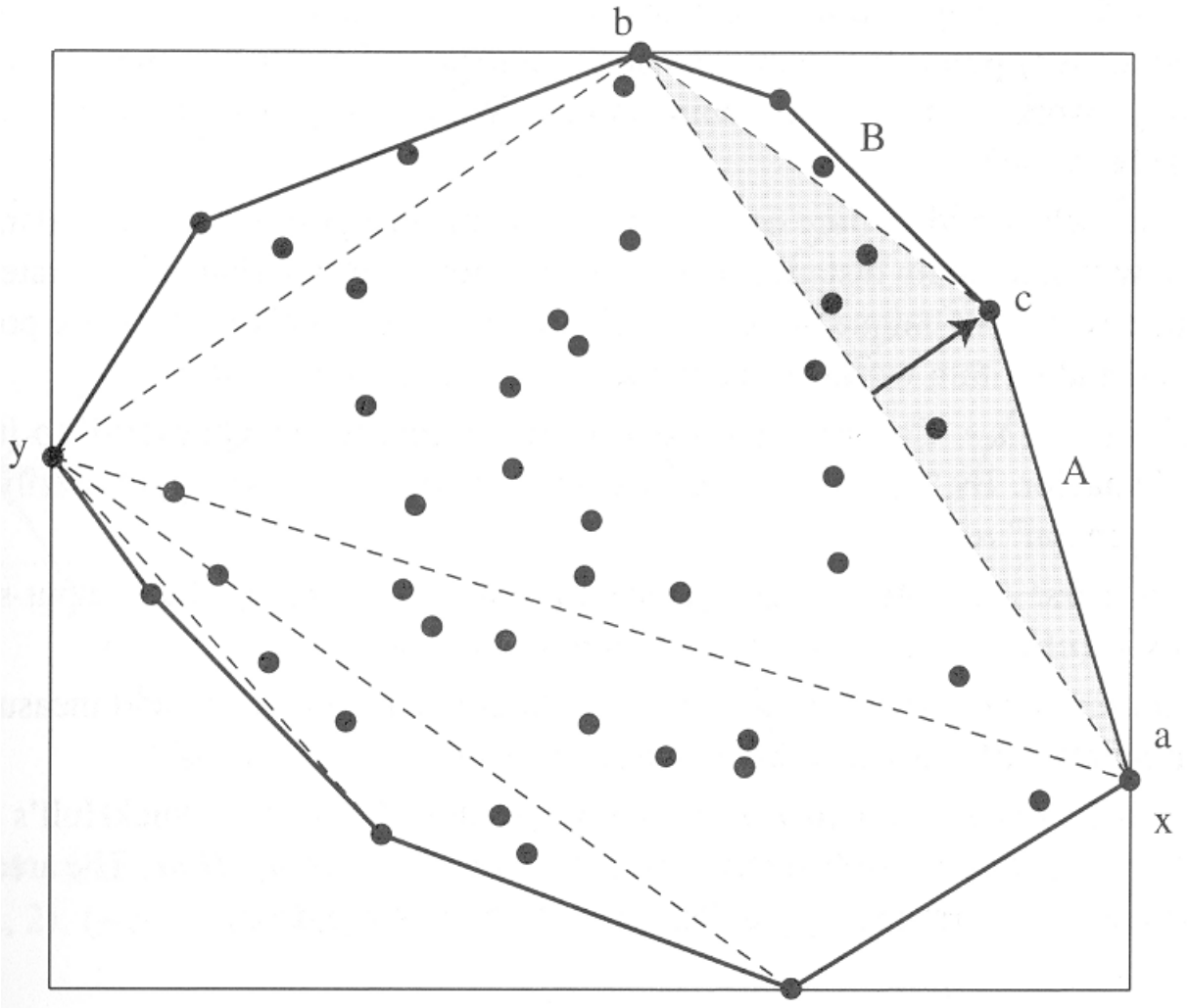
Zahl der Schleifendurchläufe:

- äußere Schleife: so oft, wie neue Kanten gezogen werden
- innere Schleife: n–1 Punkte werden zum Winkelvergleich herangezogen

$\Rightarrow$
Jarvis's March bestimmt die konvexe Hülle einer gegebenen Menge von n Punkten im R^2 in der Zeit $O(nh)$, wenn h die Zahl der Ecken der konvexen Hülle ist (outputsensitiver Algorithmus) und mit Speicherplatzbedarf $O(n)$.

Der QuickHull-Algorithmus

- Analogie zu Quicksort
- divide-and-conquer-Prinzip
- Grundidee: zunächst die "vielen" Punkte hinauswerfen, die klar im Inneren liegen, dann auf Bereiche näher am Rand konzentrieren und diese rekursiv genauso behandeln
- zunächst wird Strecke zwischen 2 Extrempunkten aus *S* gezogen (z.B. mit min. und max. *x*-Koordinate), dann wird oberhalb dieser Kante ein weiterer Extrempunkt (mit max. Abstand von der Kante) bestimmt und der Inhalt des Dreiecks eliminiert
- setze dies rekursiv für die verbleibenden Rest-Mengen fort



(aus O'Rourke 1998)

Pseudocode:

Algorithm: QUICKHULL

function $QuickHull(a, b, S)$

    if $S = \emptyset$ then return ()

    else

        $c \leftarrow$ index of point with max distance from ab.

        $A \leftarrow$ points strictly right of (a, c).

        $B \leftarrow$ points strictly right of (c, b).

        return $QuickHull(a, c, A) + (c) + QuickHull(c, b, B)$

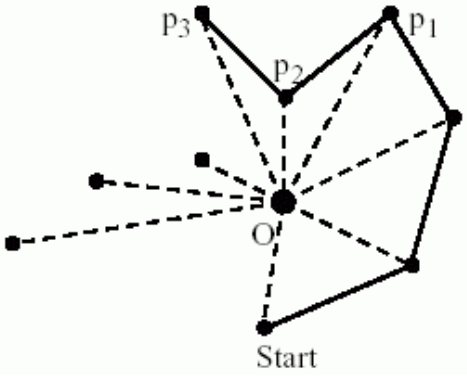
Komplexität:
QuickHull bestimmt die konvexe Hülle einer gegebenen Menge von *n* Punkten im "besten" Fall in der Zeit $O(n \log n)$, im schlimmsten Fall jedoch in der Zeit $O(n^2)$.
Speicherplatzbedarf $O(n)$. (Hinrichs 2001)

Graham's scan

- Ist es wirklich notwendig, wie in Algorithmus EXT-PUNKTE alle $O(n^3)$ Dreiecke zu betrachten, um die Extremalpunkte zu finden?
- Graham zeigte 1972, daß man die Extremalpunkte in linearer Zeit finden kann, wenn man im Gegensatz zum Algorithmus EXT-PUNKTE den Sortierschritt zuerst durchführt.
- Annahme: Ein Punkt im Innern der konvexen Hülle sei bekannt. OBdA sei dies der Ursprung des Koordinatensystems.

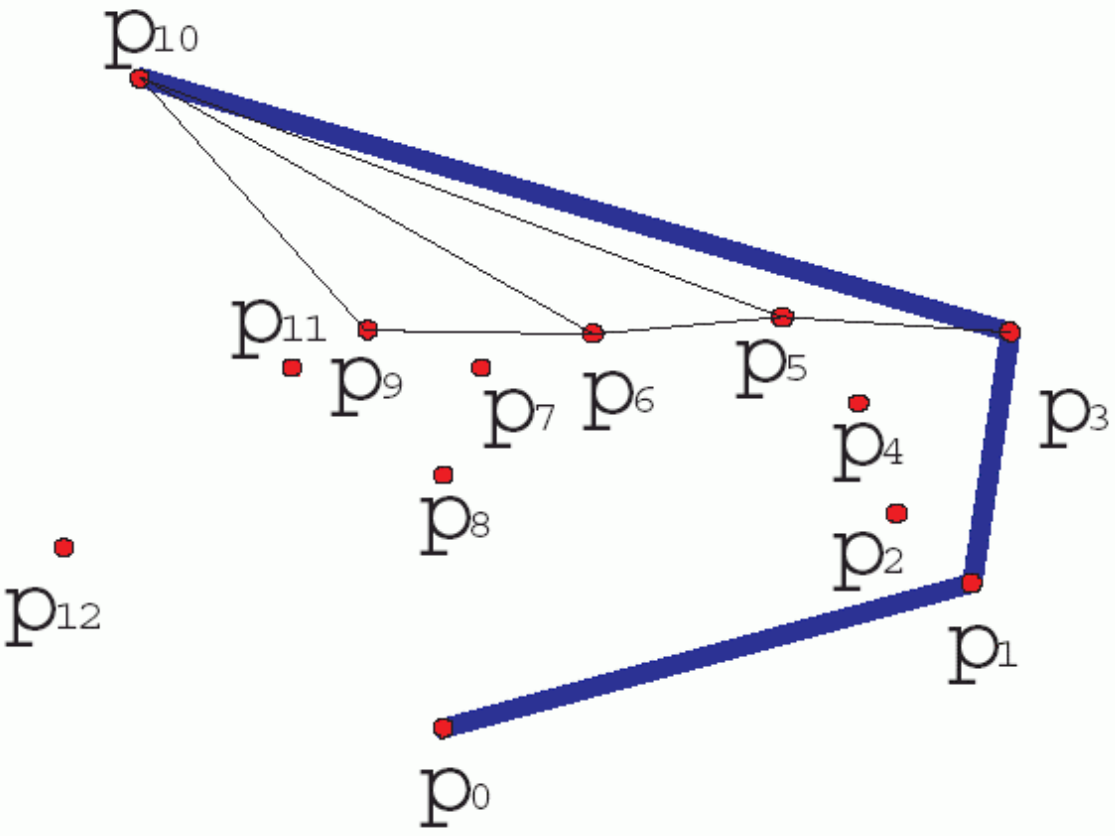
- Man sortiere die Punkte lexikografisch bzgl. des Polarwinkels (1. Kriterium) und der Entfernung zum Ursprung (Vergleich bzgl. Entfernung erfolgt nur, wenn die Punkte beide den gleichen Polarwinkel besitzen). Die sortierten Punkte werden in einer doppelt verketteten Liste abgelegt.

- Ist ein Punkt nicht Eckpunkt der konvexen Hülle, so liegt er in einem Dreieck (O, p, q), wobei p und q zwei aufeinander-folgende Eckpunkte der konvexen Hülle sind.
- Graham's Algorithmus durchläuft die geordnete Liste und eliminiert dabei die Punkte, die nicht Eckpunkte der konvexen Hülle sind. Nach dem Durchlauf bleiben die Eckpunkte der konvexen Hülle in der gewünschten Anordnung übrig.



(hier würde p_2 eliminiert)

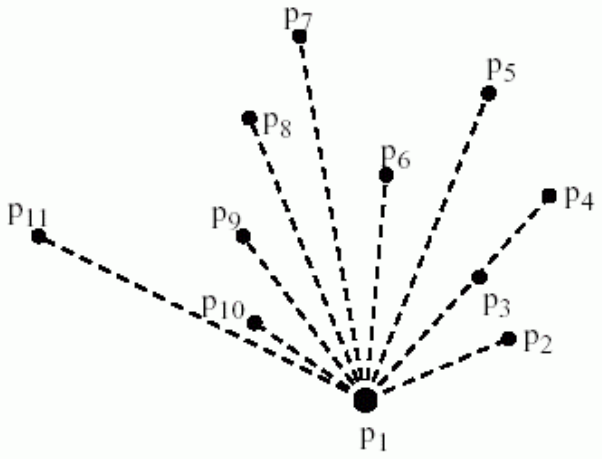
weiteres Beispiel:



genauere Beschreibung:

- Beginne Durchlauf an einem Punkt, der sicherlich Eckpunkt der konvexen Hülle ist, z.B. am kleinsten Punkt bzgl. $<_y$.
- Untersuche wir wiederholt Tripel von aufeinanderfolgenden Punkten, die um den Ursprung entgegen dem Uhrzeigersinn angeordnet sind, daraufhin, ob sie einen reflexiven Winkel, d.h. einen Winkel $\geq \pi$, bilden. Ist der innere Winkel $\angle(p_1, p_2, p_3)$ reflexiv, so bilden (p_1, p_2, p_3) eine Rechtsdrehung, ansonsten eine Linksdrehung.
- Konvexität $\Rightarrow$ beim Durchlauf der Eckpunkte der konvexen Hülle entgegen dem Uhrzeigersinn nur Linksdrehungen
- (p_1, p_2, p_3) Rechtsdrehung $\Rightarrow$ p_2 nicht Eckpunkt der konvexen Hülle
- Durchlauf der geordneten Punkteliste abhängig vom Winkeltest:
  1. Bilden (p_1, p_2, p_3) eine Rechtsdrehung, so wird p_2 aus der Liste eliminiert und als nächstes Tripel (p_0, p_1, p_3) betrachtet.
  2. Bilden (p_1, p_2, p_3) eine Linksdrehung, so schreitet man in der Liste weiter voran und betrachtet als nächstes das Tripel (p_2, p_3, p_4).
- Durchlauf terminiert, wenn der Ausgangspunkt wieder erreicht wird.
- Winkeltest benötigt O(1) Zeit.
- Nach jedem Test schreiten wir entweder voran (Schritt 2), oder wir eliminieren einen Punkt. Da es nur n Punkte gibt, kann man höchstens n Vorwärtsschritte durchführen, aber auch höchstens n Punkte eliminieren und daher höchstens n-mal zurückgehen $\Rightarrow$ Durchlauf benötigt O(n) Zeit.

- Modifikation des Algorithmus von Graham: Sortiere die gegebenen Punkte nicht bzgl. eines Punktes im Innern der konvexen Hülle, sondern bzgl. eines Eckpunktes der konvexen Hülle, z.B. des kleinsten Punktes bzgl. $<_y$.



Komplexität:

Graham's scan bestimmt die konvexe Hülle einer gegebenen Menge von n Punkten in optimaler Zeit O(n·log n) und Speicherplatz O(n).

Nachteil jedoch:
Erweiterung auf höhere Dimensionen nicht möglich, da Graham's scan grundlegend auf der Sortierung der Punkte nach ihrem Polarwinkel beruht, was sich nicht auf höhere Dim. übertragen lässt.

Inkrementeller Algorithmus:

- verwendet Idee aus dem obigen Beweis des Satzes, dass die konvexe Hülle einer endlichen Menge in der Ebene ein Polygon ist
- nimmt einen Punkt nach dem anderen her und konstruiert die konvexe Hülle inkrementell durch Vereinigen mit einem Dreieck (oder mit der leeren Menge)

gegeben: Menge S von n Punkten $p_i = (x_i, y_i) \in \mathbb{R}^2$, $i = 1, ..., n$

Prinzip:

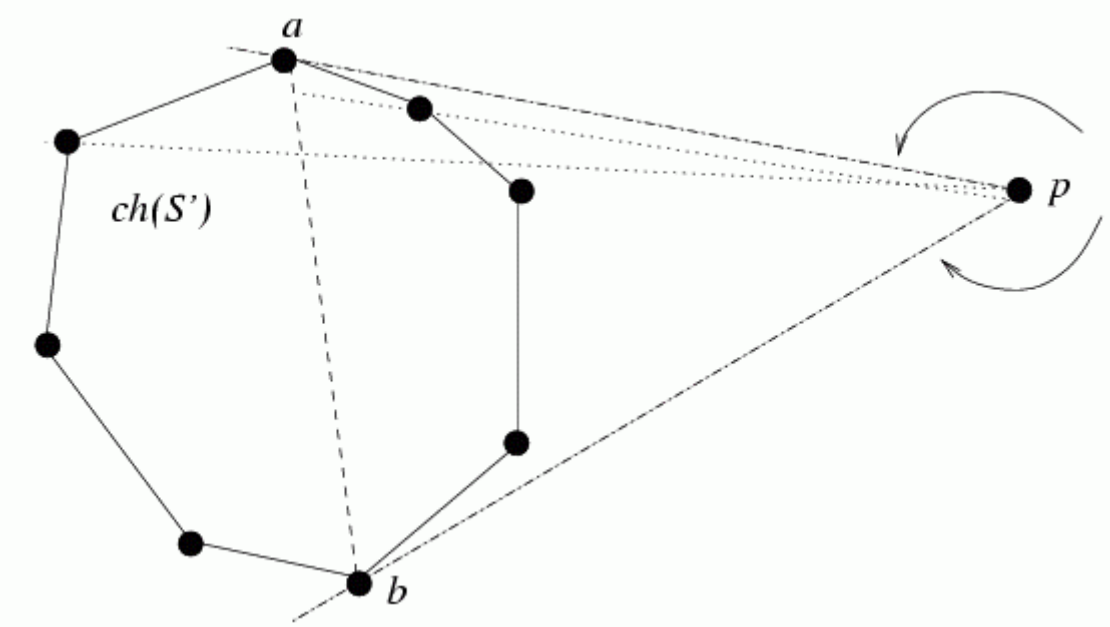
Starte mit der konvexen Hülle von drei Punkten $S' = \{p_1, p_2, p_3\} \subseteq S$

nimm sukzessive weitere Punkte $p \in S$ zu S' hinzu:

falls $p \in ch(S')$: nichts zu tun, $S' := S' \cup \{p\}$

sonst: bestimme Tangenten von p an $ch(S') \Rightarrow a,\ b$

füge $\triangle(a, b, p)$ zu $ch(S')$ hinzu, $S' := S' \cup \{p\}$



verbleibende Probleme: Für konvexes Polygon $C = ch(S')$:

(P1) Test, ob p innerhalb C liegt

(P2) Bestimmung der Tangenten von $p \notin C$ an Rand von C

Bem.: Anbau von $\triangle(a, b, p)$ geht in Zeit $O(1)$

Lösung von (P1) (Enthaltensein in C): naiv mit Strahltest:

Schnittest mit allen Kanten von C

$\rightarrow$ Zeit $O(n)$ pro einzufügendem Punkt

Lösung von (P2) (Tangentenberechnung): naiv mit Strahltest:

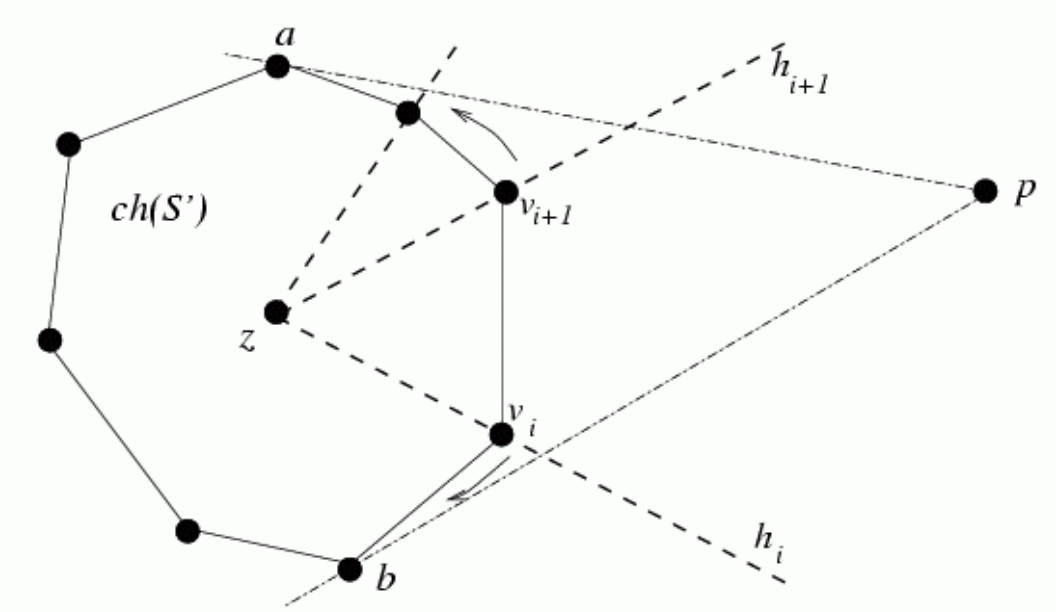
Strahltest für alle Ecken v von C: $\vec{pv} \cap (C - \partial C) \neq \emptyset$?

$\rightarrow$ Zeit $O(n)$ pro einzufügendem Punkt

$\rightarrow$ Laufzeit $O(n^2)$, nicht optimal!

Idee zur Beschleunigung von (P1) und (P2):

Annahme: Punkt z ("Zentrum") im Innern von C bekannt,
z.B. Schwerpunkt von $\triangle(p_1, p_2, p_3) \rightarrow z \in ch(S')\forall S' \subset S$
Halbgeraden h_i von z durch Ecke v_i von C, i=1,2,...,
zerlegen die Ebene in Sektoren mit Spitzen in z



für gegebenen Punkt $p \neq z$ Bestimmung des Sektors, in dem p liegt:
Halbgeraden h_i um z sind nach Steigung geordnet
$\rightarrow$ Binärsuche $\rightarrow$ Suchzeit $O(\log n)$
dynamisch: mit balanciertem Binärbaum für die h_i
$\rightarrow$ auch Einfügen/Streichen von h_i's in Zeit $O(\log n)$

Für (P2) (Tangenten): jetzt ist Kante $v_i v_{i+1}$ bekannt
von v_{i+1} laufe bis a gefunden, von v_i bis b gefunden
alle dabei durchlaufenen Kanten von C werden entfernt
$\rightarrow$ alle (P2)–Instanzen zusammen in Zeit $O(n)$

Satz: Die Konvexe Hülle $ch(S)$ mit $|S| = n$ kann inkrementell in Zeit $O(n \log n)$ berechnet werden.

Nachteil: Verwaltungsaufwand für balancierten Binärbaum

Verbesserung des mittleren Laufzeitverhaltens des inkrementellen Algorithmus möglich durch Verwendung einer zufälligen Einfüge-Reihenfolge ("*randomisierter Algorithmus*"; s. Klein 1997, Keßler 1998).

Optimale ausgabesensitive Algorithmen

- Asymptotische Zeitkomplexität O(n log n) kann als Funktion von n nicht weiter verbessert werden, da Ω(n log n) eine untere Schranke für das "Konvexe Hülle" Problem darstellt.
- Zeitkomplexität O(n·h) von Jarvis's march auch von der Anzahl h von Punkten auf der konvexen Hülle abhängig
- Genaue Untersuchung der unteren Schranke für das "Konvexe Hülle" Problem: bei Berücksichtigung des weiteren Parameters h kann man eine untere Schranke von Ω(n log h) zeigen.
- Gibt es Algorithmen, die diese untere Schranke auch erreichen?
- Kirkpatrick und Seidel 1986, Chan 1995 $\Rightarrow$ JA!

Algorithmus von Kirkpatrick und Seidel

- Neuartige Variation des Divide & Conquer Paradigmas: "marriage before conquest"
- Algorithmus konstruiert obere und untere konvexe Hülle separat.
- Statt jeweils 2 Teillösungen nach Durchführung der Rekursion zusammenzufügen, findet der Algorithmus eine obere Tangente, genannt *Brücke*, der beiden Mengen L und R, bevor die oberen Hüllen für L und R rekursiv konstruiert werden.
- Brücke bekannt $\Rightarrow$ alle Punkte, die unterhalb der Brücke zwischen ihren beiden Endpunkten liegen, können in der weiteren Rekursion ignoriert werden
  $\Rightarrow$ niedrigere asymptotische Zeitkomplexität:

Satz:

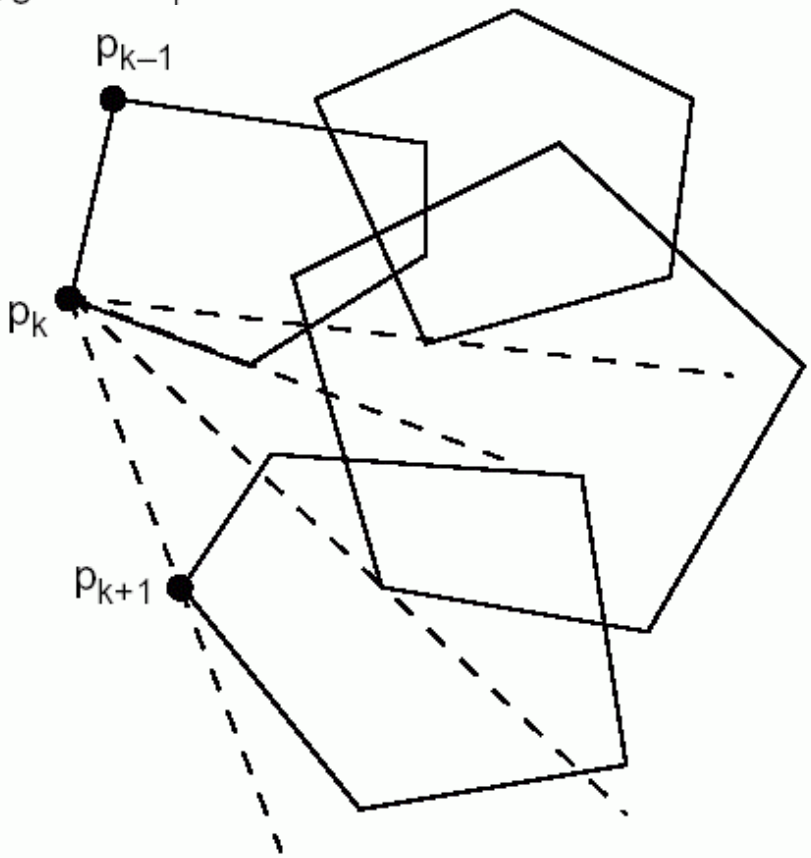
Der Algorithmus von Kirkpatrick und Seidel bestimmt die konvexe Hülle einer gegebenen Menge von n Punkten in optimaler Zeit $O(n \cdot \log h)$ und Speicherplatz $O(n)$. ■

- Kirkpatrick und Seidel benutzen Median-Find Algorithmus, der Median einer Menge von n Elementen in O(n) Zeit bestimmt. Konstante vor dem n jedoch sehr groß ⇒ Algorithmus nicht praktikabel ⇒ Algorithmus von Kirkpatrick und Seidel hauptsächlich von theoretischem Interesse
- D. G. Kirkpatrick, R. Seidel:
  The ultimate planar convex hull algorithm?
  SIAM Journal on Computing, Vol. 15, p. 287 - 299, 1986.

Algorithmus von Chan

- Praktikabler als der Algorithmus von Kirkpatrick und Seidel ist der Algorithmus von Chan, der die gleiche optimale Zeitkomplexität $O(n \cdot \log h)$ und Speicherplatzkomplexität O(n) besitzt und sich auf den 3-dimensionalen Fall verallgemeinern läßt.
- Idee: Kombination von
  Jarvis's march (gift wrapping) und Graham's scan
- Jarvis's march berechnet konvexe Hülle durch eine Folge von h Einwickelschritten, in denen jeweils in O(n) Zeit eine Kante der Hülle bestimmt wird ⇒ Zeitkomplexität $O(n \cdot h)$

- Beschleunigung des Einwickelschrittes durch Preprocessing:
  - Wähle $1 \leq m \leq n$ und partitioniere die Menge S der n Punkte in $\lceil n/m \rceil$ Teilmengen S_i der Größe $\leq m$.
  - Berechne mit Graham's scan die konvexe Hülle CH_i jeder Teilmenge S_i in jeweils $O(m \cdot \log m)$ Zeit $\Rightarrow \lceil n/m \rceil$ möglicherweise überlappende konvexe Polygone CH_i mit jeweils höchstens m Eckpunkten, die in $O((n/m) \cdot m \cdot \log m) = O(n \cdot \log m)$ Zeit bestimmt werden.
  - Einwickelschritt: Bestimme nächste Kante der konvexen Hülle CH von S aus den $\lceil n/m \rceil$ Tangenten an die konvexen Polygone CH_i :



Bestimmung einer Tangente an ein konvexes Polygon mit m Eckpunkten durch binäre Suche in $O(\log m)$ Zeit $\Rightarrow$ Kosten eines Einwickelschrittes sind $O((n/m) \cdot \log m)$.

T. M. Chan:
Optimal output-sensitive convex hull algorithms in two and three dimensions,
Discrete & Comput. Geometry, Vol. 16, p. 361 - 368, 1996.

Konvexe Hüllen in höheren Dimensionen (Hinrichs 2001)

- Konvexe Hülle einer endlichen Menge S von Punkten im $\mathbb{R}^d$ ist ein konvexes Polyeder.
- Umgekehrt ist ein konvexes Polyeder im $\mathbb{R}^d$ die konvexe Hülle einer endlichen Menge von Punkten.
- (d–1)-dimensionale Randflächen bezeichnet man als *Facetten*.
- Struktur der konvexen Hülle in höherdimensionalen Räumen kompliziert $\Rightarrow$ Ausgabe der konvexen Hülle stellt eine unüberwindbare untere Schranke für die Berechnung der konvexen Hülle dar
- Klee hat 1966 gezeigt, daß die konvexe Hülle von n Punkten im $\mathbb{R}^d$ aus $\Omega(n^{\lfloor d/2 \rfloor})$ Facetten bestehen kann $\Rightarrow$ konvexe Hülle für d = 4 besitzt quadratische Komplexität $\Rightarrow$ für $d \geq 4$ ist O(n log n) Algorithmus unmöglich
- "gift wrapping" Methode von Chand und Kapur (1970): Wandere von einer Facette zu einer Nachbarfacette, ähnlich wie man ein durch ebene Flächen begrenztes Objekt in Papier einpackt (2-d: Jarvis's march).

Satz:

Der Zeitaufwand der "gift wrapping" Methode für die Bestimmung der konvexen Hülle einer Menge von n Punkten im $\mathbb{R}^d$ beträgt im schlimmsten Fall $O(n^{\lfloor d/2 \rfloor + 1}) + O(n^{\lfloor d/2 \rfloor} \cdot \log n)$. ■

- " beneath-beyond" Methode von Kallay (1981) besitzt on-line Eigenschaft und ist ähnlich zum inkrementellen Algorithmus zur Berechnung der konvexen Hülle im 2-d: Betrachte jeweils einen Punkt p aus der Menge S, und falls p außerhalb der gegenwärtigen konvexen Hülle liegt, so wird der "Berührungskegel" mit Spitze p an die konvexe Hülle konstruiert und der Teil der konvexen Hülle, der durch diesen Kegel verdeckt wird, durch diesen ersetzt.

Satz:

Der Zeitaufwand der "beneath-beyond" Methode für die Bestimmung der konvexen Hülle einer Menge von n Punkten im $\mathbb{R}^d$ beträgt $O(n^{\lfloor d/2 \rfloor + 1})$. ■

Speziell im dreidimensionalen Raum kann die konvexe Hülle mit einen divide-and-conquer-Algorithmus oder mit einer Verallgemeinerung des Algorithmus von Chan bestimmt werden:

Satz:

Die konvexe Hülle einer Menge von n Punkten im $\mathbb{R}^3$ kann mit einem Divide & Conquer Algorithmus in O(n log n) Zeit und mit dem Algorithmus von Chan in O(n log h) Zeit bestimmt werden.■

Ein anderes Problem, das mit dem der konvexen Hülle zusammenhängt:

Durchschnitt von unteren Halbebenen im R^2 (Klein 1997)

Gegeben seien n Geraden G_1, G_2, ..., G_n in der Ebene, keine davon verlaufe genau vertikal

$\Rightarrow$ jedes G_i lässt sich darstellen in der Form

$G_i = \{ (x, y) \in \mathrm{R}^2 \mid y = a_i x + b_i \}$

mit reellen Zahlen a_i , b_i .

Gesucht ist der Durchschnitt der *unteren Halbebenen*

$H_i = \{ (x, y) \in \mathrm{R}^2 \mid y \leq a_i x + b_i \}$.

D.h., das lineare Ungleichungssystem

$$\begin{aligned} a_1 x + b_1 &\geq y \\ &\vdots \\ a_n x + b_n &\geq y \end{aligned}$$

ist zu lösen.

"Lineares Programmieren":
Maximierung von Zielfunktionen $f(x, y)$ mit solchen Ungleichungen als Nebenbedingungen.

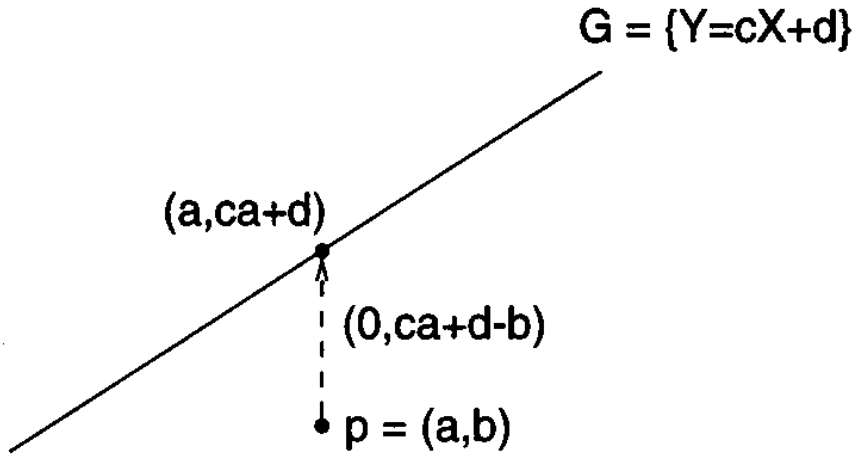
Zusammenhang zu konvexen Hüllen:
wir benötigen den Begriff der *Dualität* zwischen Punkten und Geraden.

Definiere Abbildungen

$$p = (a, b) \mapsto p^* := \{ (x, y) \mid y = ax + b \}$$
$$G = \{ (x, y) \mid y = ax + b \} \mapsto G^* := (-a, b)$$

(Beachte: $(a, b)^{**} = (-a, b)$; keine "perfekte" Dualität.)

"Gerichteter vertikaler Abstand" $gva(p, G)$ zwischen einem Punkt p und einer Geraden G:
y-Koordinate des senkrechten Vektors, der von p zu G führt (positiv, wenn p unterhalb von G liegt).



Hilfssatz:

Es gilt $gva(p, G) = - gva(G^*, p^*)$.

Insbesondere gilt:

p liegt oberhalb von $G \Leftrightarrow p^*$ verläuft oberhalb von G^*.

Beweis: nachrechnen (vgl. Klein 1997).

Hilfssatz:

Seien $p = (a, b)$, $q = (c, d)$, $a \neq c$. Dann gilt:

$$p^* \cap q^* = (\text{Gerade}(pq))^*.$$

Beweis:

$\text{Gerade}(pq) = \{ (x, y) \mid y = \frac{b-d}{a-c}x + \frac{ad-bc}{a-c} \}$.

Andererseits haben $p^* = \{ (x, y) \mid y = ax + b \}$ und $q^* = \{ (x, y) \mid y = cx + d \}$ den Punkt

$r = \left(\frac{d-b}{a-c}, \frac{ad-bc}{a-c} \right)$ gemeinsam

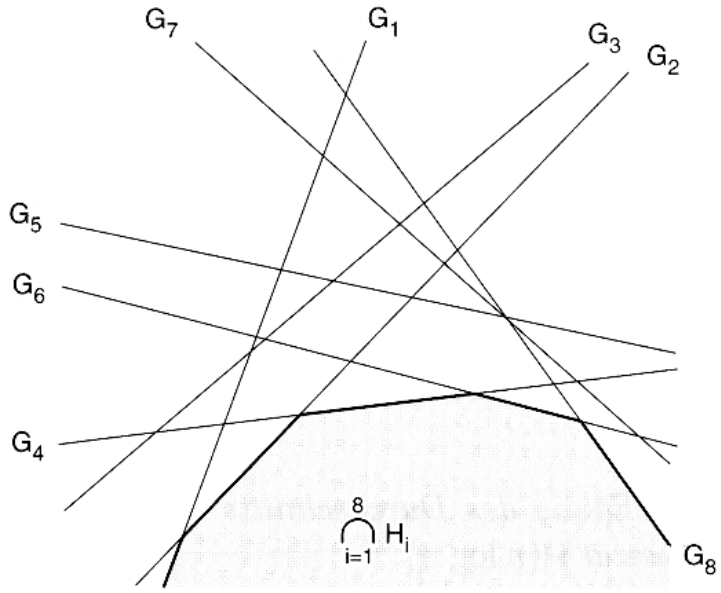
$\Rightarrow r = (\text{Gerade}(pq))^*$. □

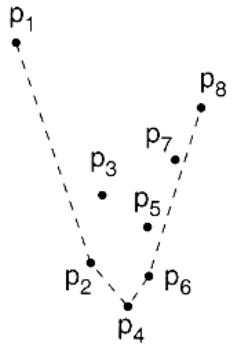
Die folgende Abbildung (aus Klein 1997) zeigt ein Arrangement von 8 Geraden und ihre dualen Punkte.

Beachte: je weiter links die dualen Punkte, desto steiler ihre zugehörigen Geraden.

Einige Geraden leisten keinen Beitrag zum Rand des Durchschnitts der unteren Halbebenen:

genau diejenigen, deren duale Punkte nicht auf dem unteren Rand der *konvexen Hülle* der dualen Punktmenge liegen!





Satz:
In einem Arrangement von endlich vielen, nicht senkrechten Geraden G_i ist ein Punkt $G_i \cap G_j$ genau dann ein Eckpunkt des Durchschnitts der unteren Halbebenen der G_i, wenn das Liniensegment $G_i^* G_j^*$ eine untere Kante der konvexen Hülle der dualen Punkte G_i^* ist.

Beweis:
Sei $p_i = G_i^*$. Es gilt:
$p_i p_j$ ist untere Kante der konvexen Hülle
$\Leftrightarrow$ alle übrigen p_k liegen oberhalb der Geraden $p_i p_j$
↳ 1. Hilfssatz
$\Leftrightarrow$ alle übrigen Geraden p_k^* verlaufen oberhalb von
(Gerade $p_i p_j)^* = p_i^* \cap p_j^*$
↑
2. Hilfssatz
$\Leftrightarrow$ alle übrigen G_k^{**} verlaufen oberhalb von
$G_i^{**} \cap G_j^{**} = (G_i \cap G_j)^{**}$
↳ 2 mal 1. Hilfssatz
$\Leftrightarrow$ alle übrigen G_k verlaufen oberhalb von $G_i \cap G_j$

$\Leftrightarrow$ $G_i \cap G_j$ ist Eckpunkt des Schnitts der unteren Halbebenen. □

Anwendung der Dualitätsabbildung erfordert $O(n)$ Zeit.
Reihenfolge von Kanten bzw. Punkten bleibt erhalten

$\Rightarrow$ Durchschnitt von n Halbebenen ist in linearer Zeit transformierbar auf die Konstruktion der konvexen Hülle von n Punkten, und umgekehrt.

Damit ist auch gezeigt:

Satz: Die Berechnung des Durchschnitts von n Halbebenen hat die Zeitkomplexität $\Omega(n \log n)$.