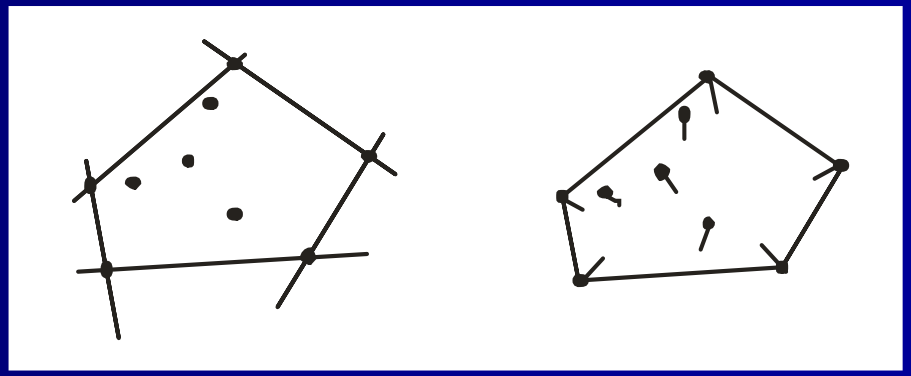
Konvexní obálky

I.Kolingerová

Obsah:
1. Formulace problému
2. Aplikace
3. Konvexní obálka v 2D
4. Konvexní obálka v 3D

Formulace problému

- Konvexní obálka množiny bodů *S*
  - průnik všech poloprostorů, které obsahují *S*



- Ozn.: *CH(S)*,
  conv(S)

- Nebo: Množina v R^d je konvexní, pokud pro lib. 2 body *p,q* této množiny leží úsečka *pq* zcela uvnitř množiny.
- Nebo: *CH(S)* množiny bodů *S* v R^d je hranice nejmenší konvexní množiny, která obsahuje *S*.

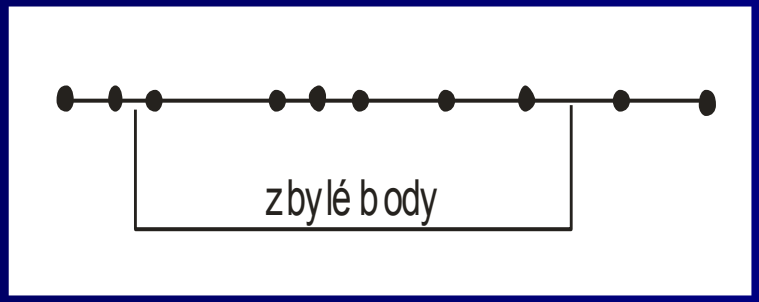
Aplikace

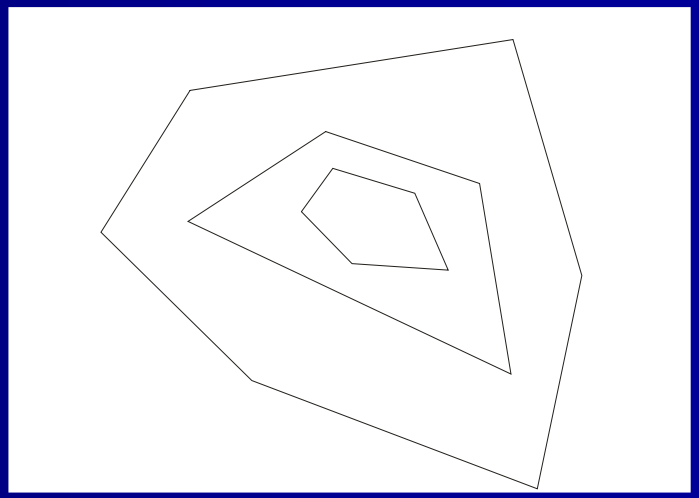
- Vyhýbání se kolizím
- Nalezení přímky, která vyhovuje datům
- Nejmenší box (2D, 3D)
- Tvarová analýza
- Počítačová grafika
- Optimalizace

Aplikace

- Statistika (např. [Pre85])

  a) Robustní odhad

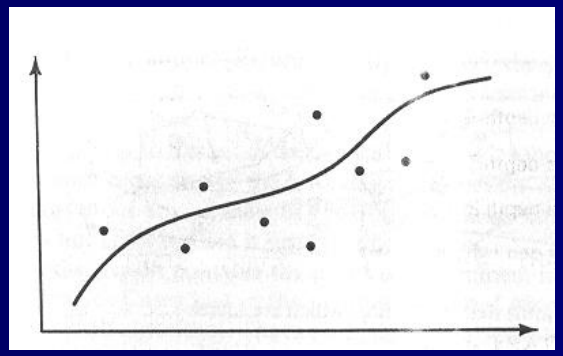




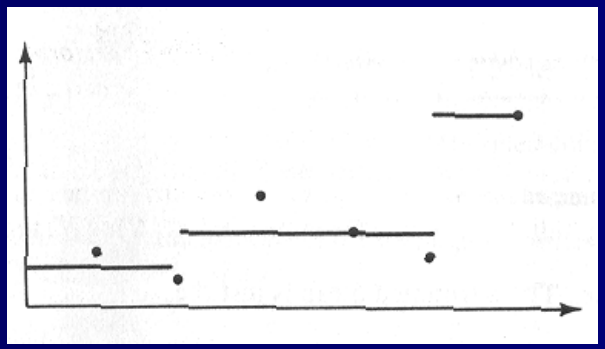
  b) Shlukování

  - míra „kompaktnosti" clusteru – poloměr množiny bodů = poloměr jeho konv. obálky

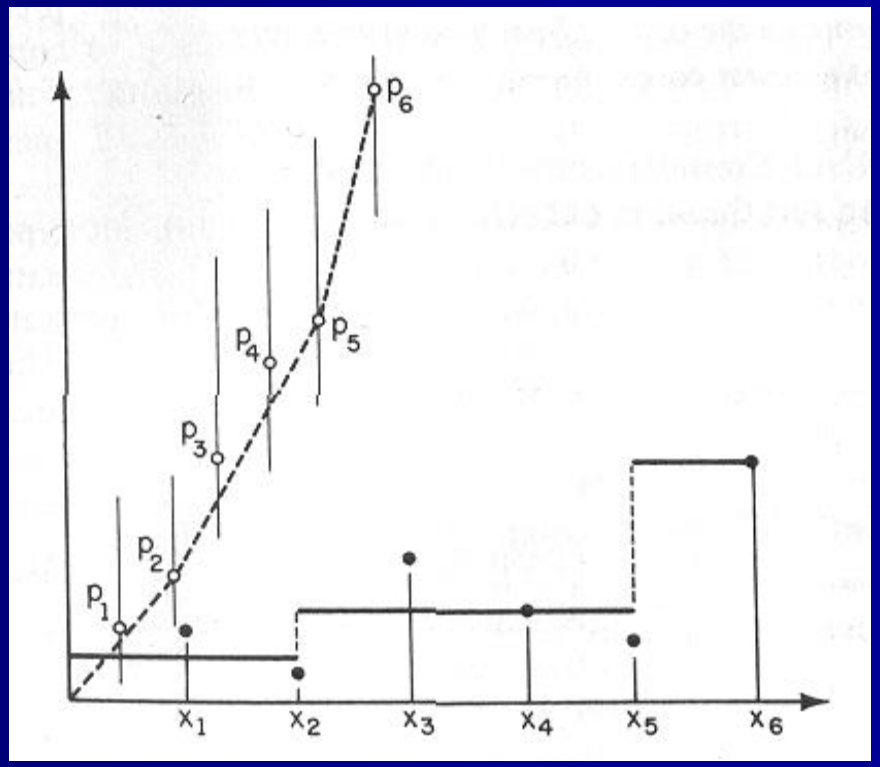
c) Izotonická regrese



Izotónní funkce (nikoliv nejlepší)
Aproximující body



Nejlepší izotónní aproximace (schod. funkce)



Dolní CH(S) diagramu částeč. součtů
s_j = sum y_i (i=1..j) definuje izotónické proložení

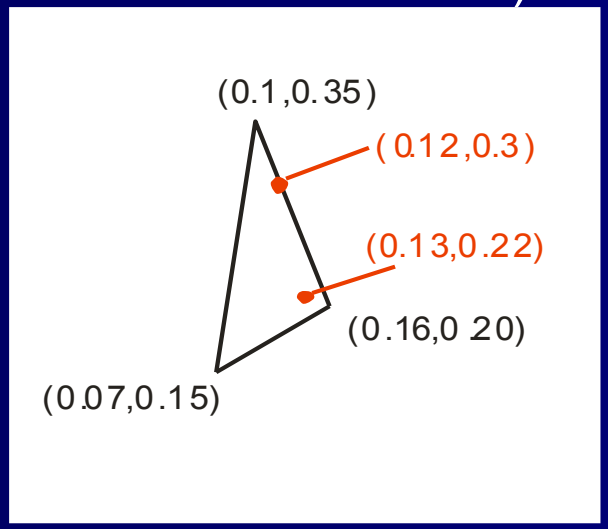
Aplikace - příklad [Ber97]

Komponenty A, B

Směsi S1, S2: S1=10%A, 35%B, S2=16%A, 20%B

Jde z toho namíchat a) 12%A, 30%B? Ano (2:1)

b) 13%A, 22%B? Ne, ale jde z S1, S2, S3, S3=7%A, 15%B (S1:S2:S3) = (1:3:1)



$l_1 : l_2 : \ldots : l_n$ poměr,

p_i = body, $\lambda_i = l_i/L$,

$\lambda_i >= 0$ pro všechna i,

$$L = \sum_{j=1}^{n} l_j$$

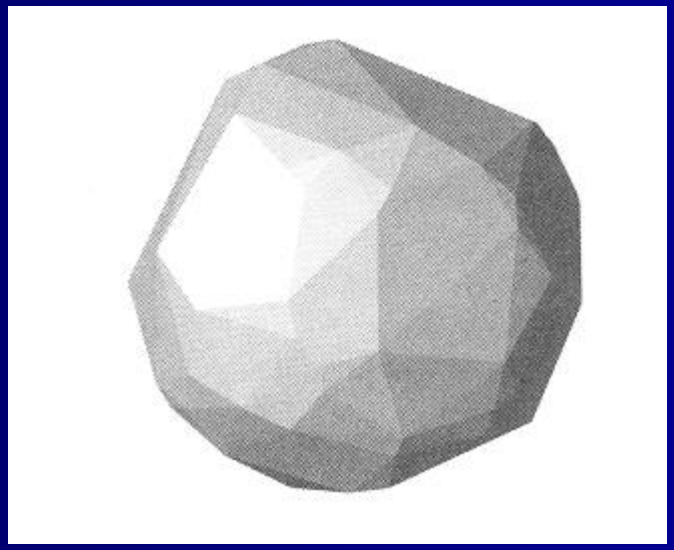
$$\sum_{i=1}^{n} \lambda_i = 1$$

Obecně pro n směsí, $n > 3$:

$$\sum_{i=1}^{n} \lambda_i p_i \quad \text{směs}$$

⇒Konvexní obálka = množina všech možných konvexních kombinací bodů

d=3:

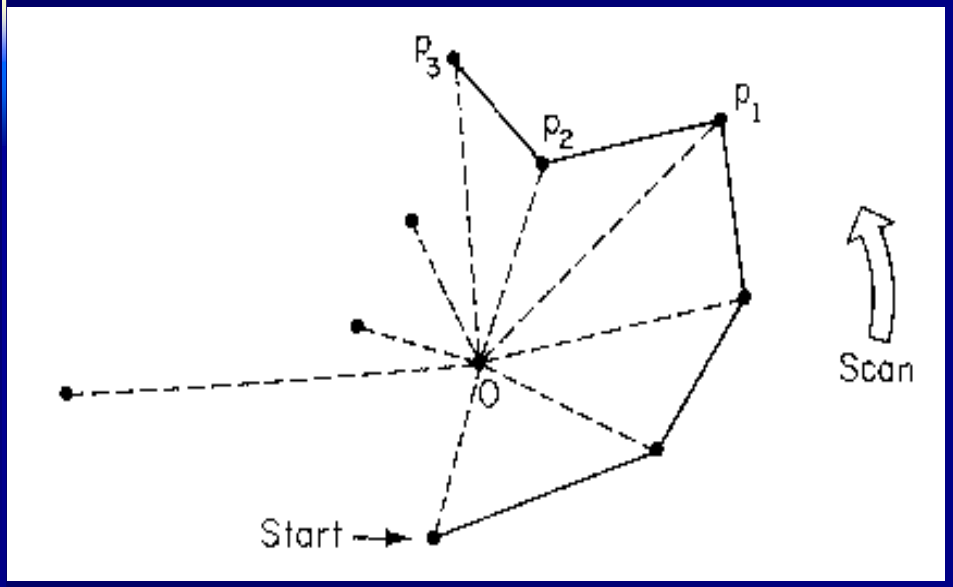


Dolní meze

- $O(N \log N)$ v E^2, E^3
- $O(N\ F)$ pro algoritmy „citlivé na výstup", F- počet stěn $CH(S)$
- $O(N)$ – pro seřazené body
- $O(\log N)$ - pro každý nový bod v „on-line" problému
- Některé algoritmy v oček. případě $O(N \log N)$ ale v nejhorším případě $O(N^2)$ – např. Quickhull (viz např. [Pre85])

1. Rovina

Grahamovo prohledávání [Pre85]

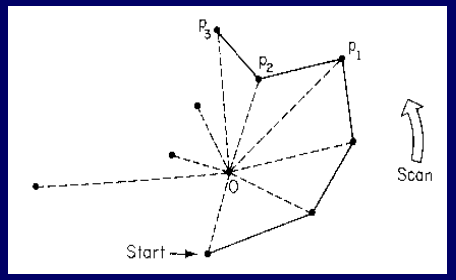


Start: extrémní bod (nejpravější bod s nejmenší souř.*y*)

1. *p*1*p*2*p*3 je pravotočivý:
   Eliminuj vrchol *p*2 a zkontroluj *p*0*p*1*p*3
2. *p*1*p*2*p*3 je levotočivý:
   Zkontroluj *p*2*p*3*p*4

Kritérium vyřazení bodu:
Úhel *p*1*p*2*p*3 ≥ π (pravotočivost)
- ale nepočítat úhly, užít orientační test

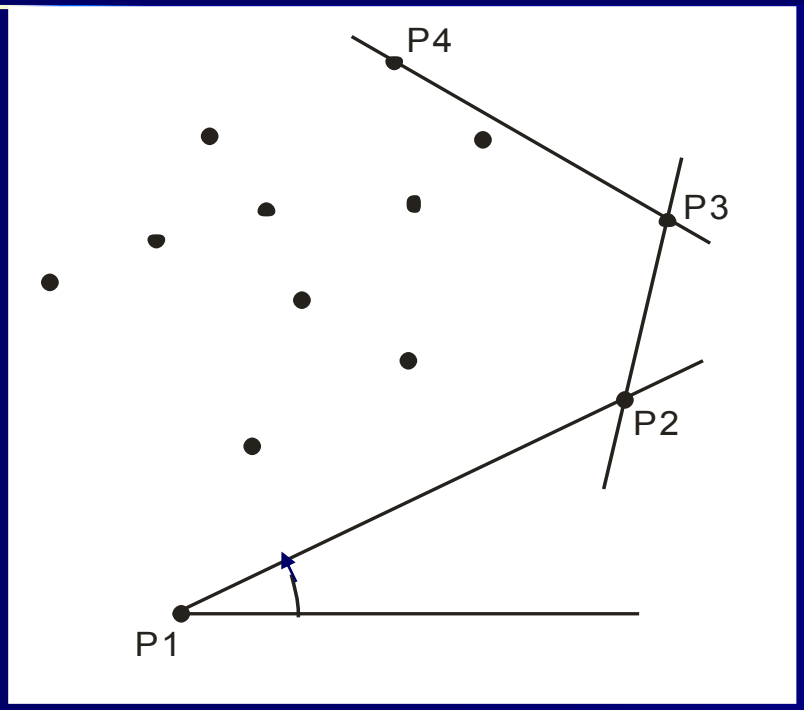
Grahamovo prohledávání



O(N log *N)* čas,
O(N) paměť

+ Optimální v nejhorším případě
- Nemusí být optimální v průměru
- Nejde zobecnit pro d>2
- Není on-line
- Sériový

Jarvisův pochod (balení dárku) [Pre85]

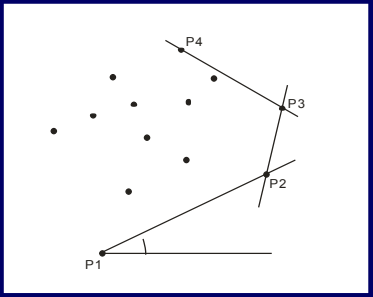


Start: extrémní bod, jde se po hranách

Vybírá se nejmenší úhel

$O(N^2)$ čas v nejhor. př.,
$O(K\ N)$ očekávaný
(pro K bodů v $CH(S)$)
$O(N)$ paměť

Jarvisův pochod

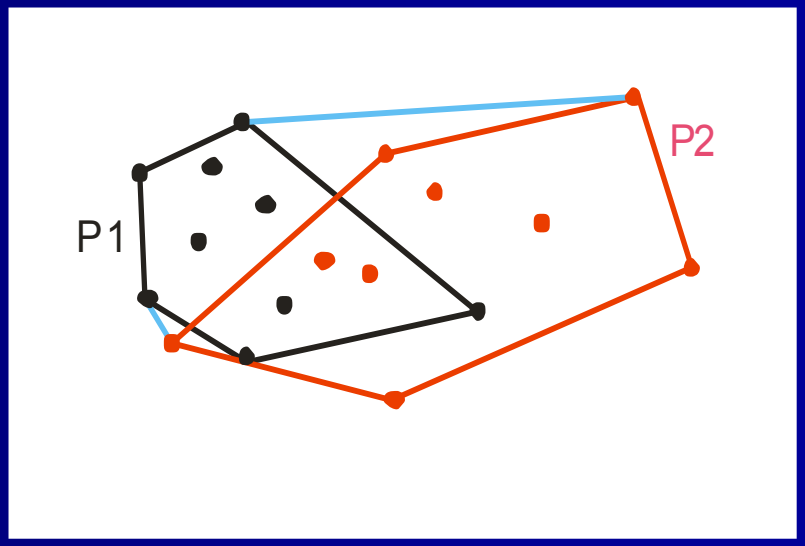


- Není optimální v nejhorším případě
+ Může být optimální v průměru (Je $K << N$?)
+ Jde zobecnit pro d=3 i více
+ Citlivý na výstup
- Není on-line
- Sériový

Rozděl a panuj

[Pre85]

1. Pro |*S*| malé přímo zkonstruuj *CH(S)*, konec
2. *S* -> *S*1, *S*2: |*S*1| ≈ |*S*2|
3. Rekurzivně najdi *CH(S*1*)*, *CH(S*2*)*,
4. Spoj *CH(S*1*)* a *CH(S*2*)* na *CH(S)* (obálka sjednocení 2 konvex. polygonů)



O(N log *N),*
pokud spojovací krok v *O(N)*

Rozděl a panuj

CH(*P*1 u *P*2):

1. Najdi bod *p* vnitřní v *P*1
2. Urči, zda *p* je vnitřní v *P*2. *O(N)*

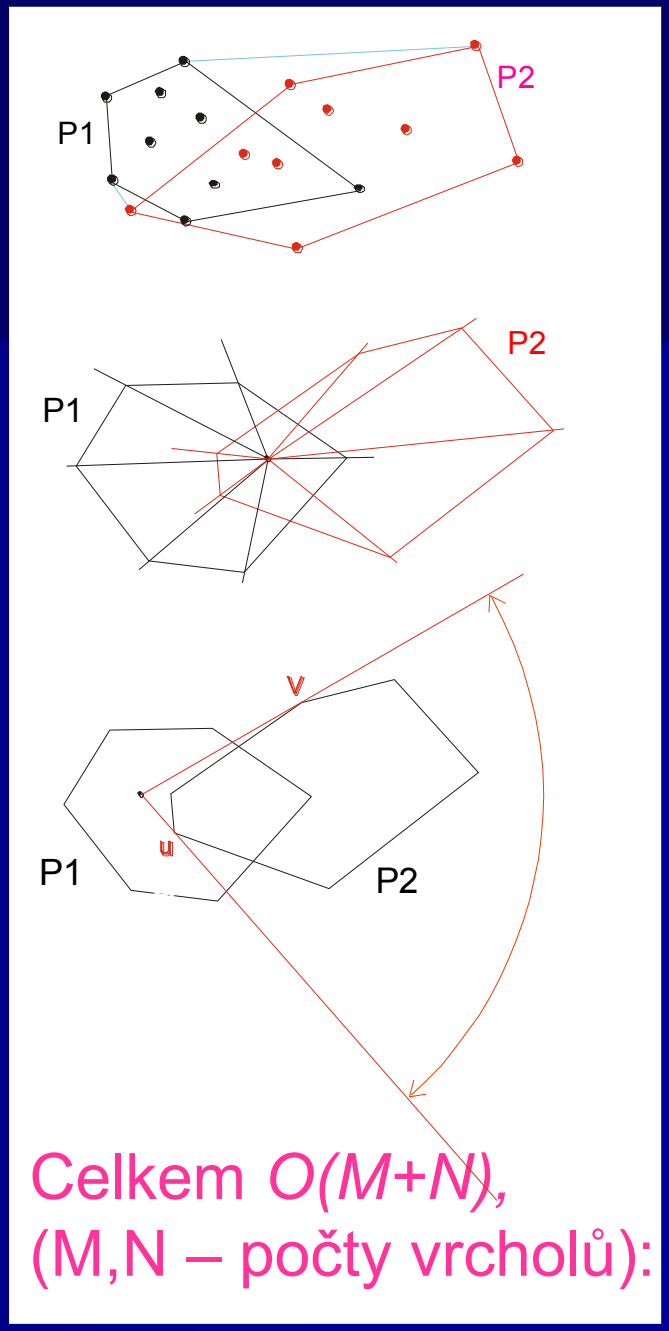
   Pokud ne, jdi na 4
3. // *p* je vnitřní v *P*2

   Spoj *P*1 a *P*2 do 1 seznamu, seřaz. vůči *p* (podle úhlu) *O(N)*

   jdi na 5
4. // *p* není vnitřní v *P*2

   *P*2 leží v klínu s úhlem <=π => 2 části *P*2, 1 lze vyřadit, 2. setřídit spolu s *P*1 *O(N)*
5. Graham. prohl. => *CH*(*P*1 u *P*2):

   zde O(N)



Celkem *O(M+N)*, (M,N – počty vrcholů):

Varianty problému

On-line – přicházejí nové body, algoritmus je dostane „na žádost“

Real-time – přicházejí nové body, rychlostnení řízena CH(S) algoritmem (ale obvykle se předp. rovnoměrnost v čase)

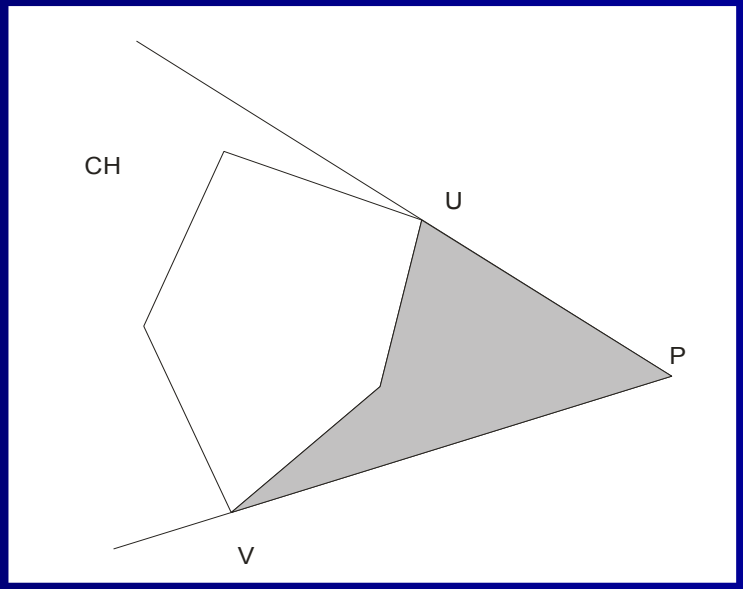
Dynamická údržba konv. obálky – podporovat i rušení bodů

Kinetická CH(S) – body se pohybují

Přibližná CH(S) – viz např. [Pre85]

Nástin řešení on-line [Oro94]

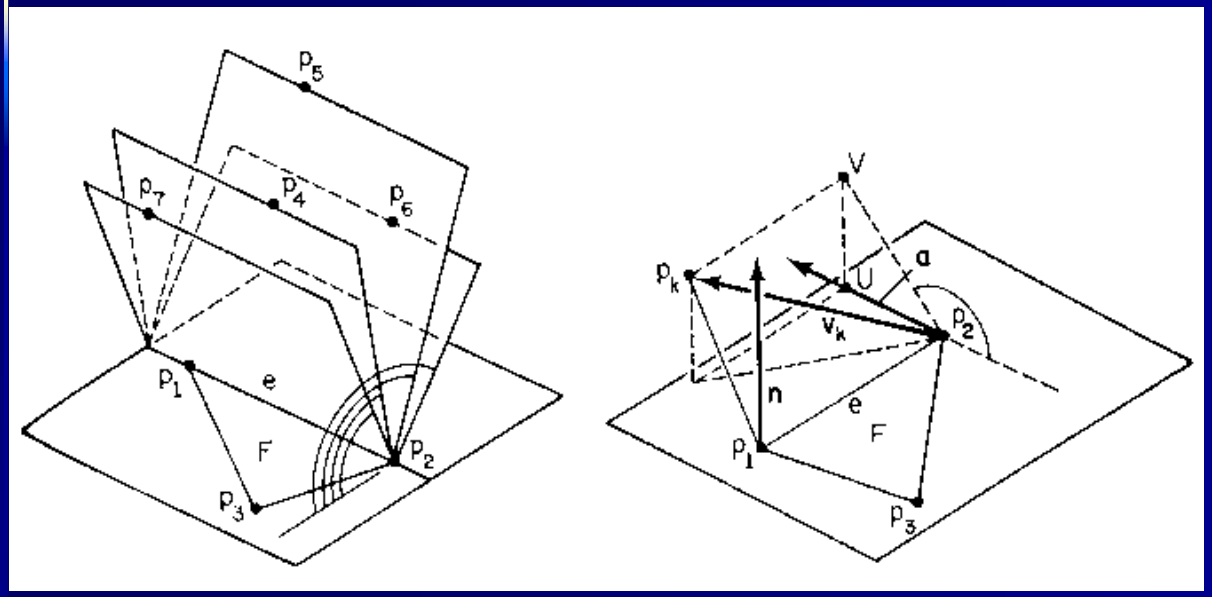
- Rozhodnout, zda p_i je vně aktuální CH(S) z i-1 bodů
- Pokud ano, najít *u,v* a odstranit část *CH* mezi nimi



$O(N^2)$, jde zobecnit pro větší dimenzi

2. Vyšší dimenze

Balení dárku [Pre85]



O(*N F*),

F - počet stěn obálky

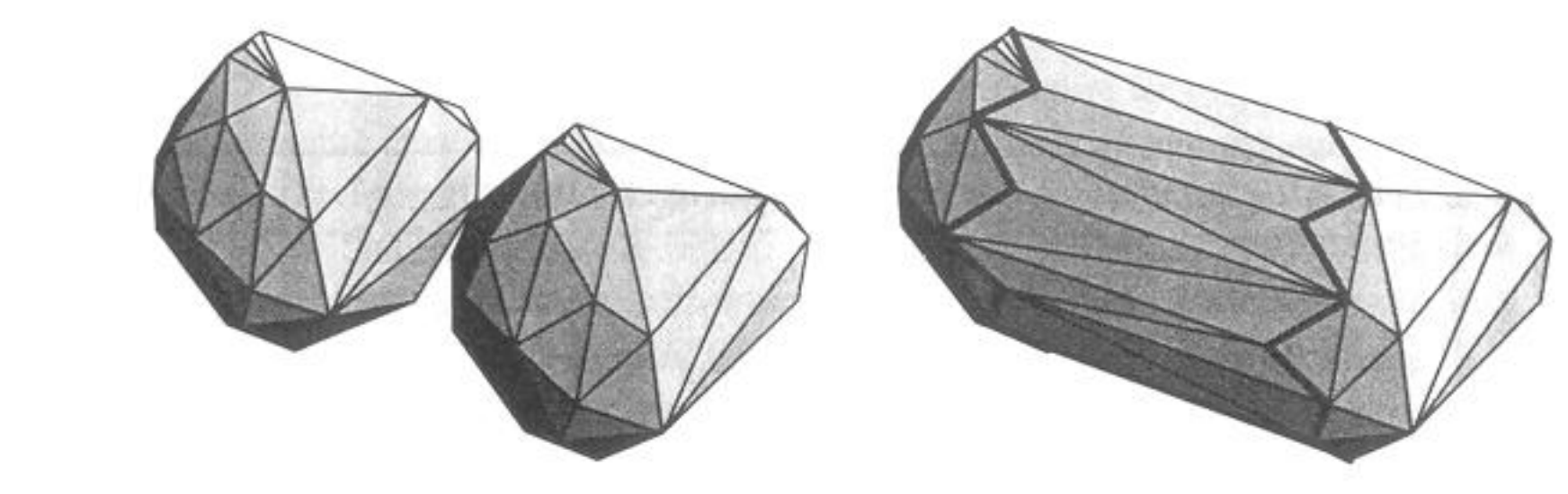
=>

citlivé na výstup

Chceme ze stěny *F* přejít přes hranu *e* na *F´*: hledáme polorovinu, která tvoří největší úhel < π s polorovinou obsahující *F* (zde polorovina obsahující *p*6)

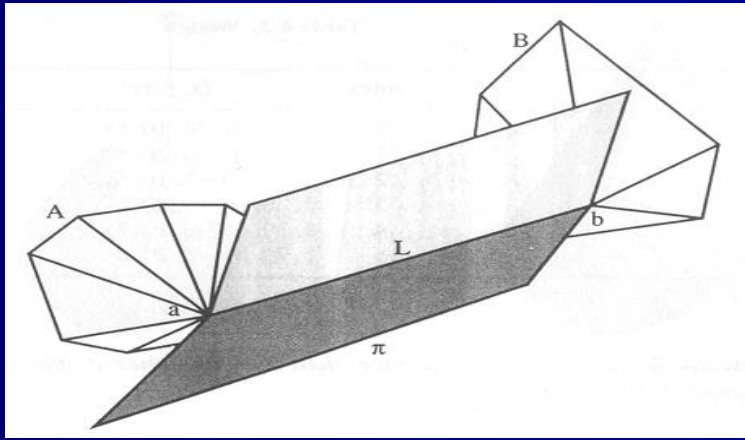
Rozděl a panuj

- Napřed body setřídit podle *x*, rozdělit na 2 skupiny, rekurzivně sestrojit obálky, spojit je
- Spojení: $O(N)$ => $O(N \log N)$ celkem



Rozděl a panuj

- Rovina rotována kolem L, nejprve narazí na *c* – sousední vrchol k *a* nebo k *b* => stěna
- Např. *c* prvkem *A* => pokračování rotací kolem *cb*
- Pak vyhodit zakryté části



Odkazy

[Ber97] de Berg, M., van Kreveld, M., Overmars, M., Schwarzkopf, O.: Computational Geometry. Algorithms and Applications, Springer-Verlag (1997)

[ORo94] O'Rourke J.: Computational Geometry in C, Cambridge University Press, 1994 (first edition), 1998 (second edition)

[Pre85] Preparata F.P., Shamos M.I.: Computational Geometry: An Introduction, Springer Verlag New York Berlin Heidelberg Tokyo, 1985