①

Základní literatura

de Berg, Cheong, Overmars, van Kreveld

Computational Geometry, Springer, 3. vydání

2007

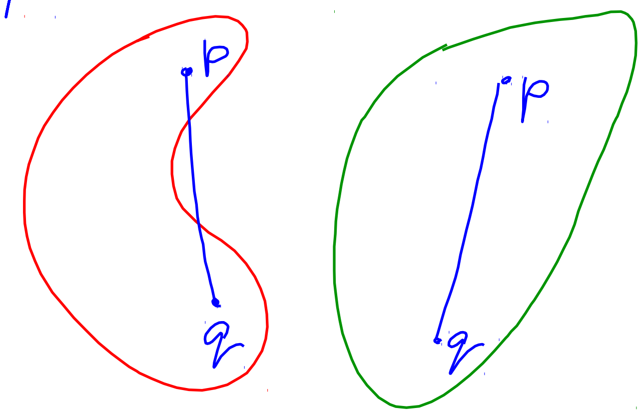
Konvexní obal v rovině

Množina K v rovině je konvexní, jestliže s každými body $p, q \in K$ obsahuje úsečku pq.

$p = (p_x, p_y)$, $q = (q_x, q_y)$

Každý bod úsečky je tvaru

$$\lambda p + (1-\lambda) q \qquad \lambda \in [0,1]$$



$$(\lambda p_x + (1-\lambda) q_x, \lambda p_y + (1-\lambda) q_y)$$

K je konvexní $\iff \forall p,q \in K \ \forall \lambda \in [0,1] \quad \lambda p + (1-\lambda) q \in K$

Konvexní obal množiny $M \subseteq \mathbb{R}^2$

- nejmenší konvexní množina obsahující množinu M

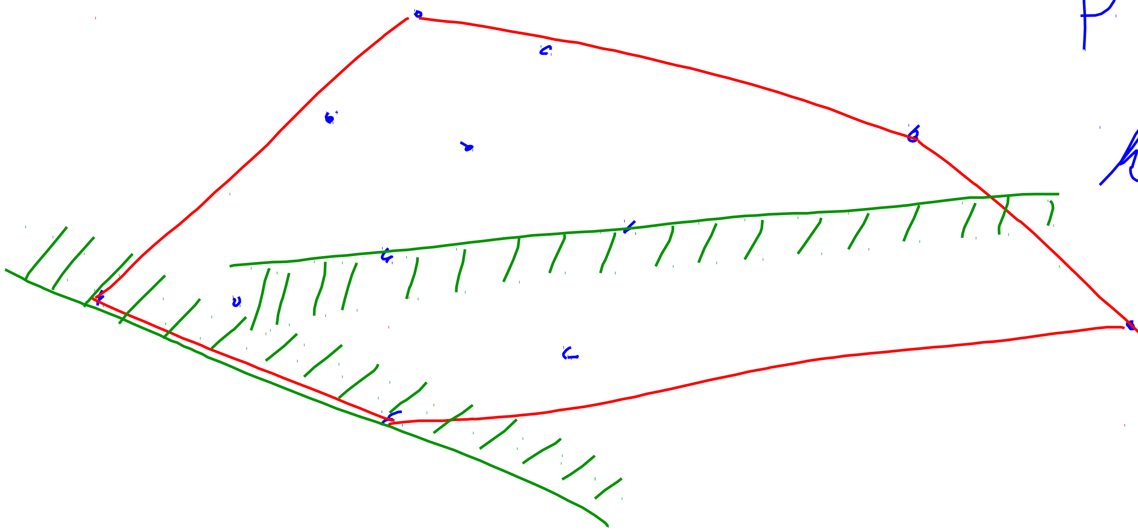
Tedy $\bigcap_{\substack{K \supseteq M \\ K \text{ konvexní}}} K$ je ona nejmenší konvexní množina.

- rovná se rovněž průniku polorovin obsahujících množinu M

$$\bigcap_{\substack{H \supseteq M \\ H \text{ polorovina}}} H$$

④

Konvexní obaly konečných množin bodů v rovině



P konečná množina bodů

konvexní obal

$$CH(P) = \bigcap H$$

H polorovina

míčena drojící bodů z P

takovou, že všechny body z P v ní

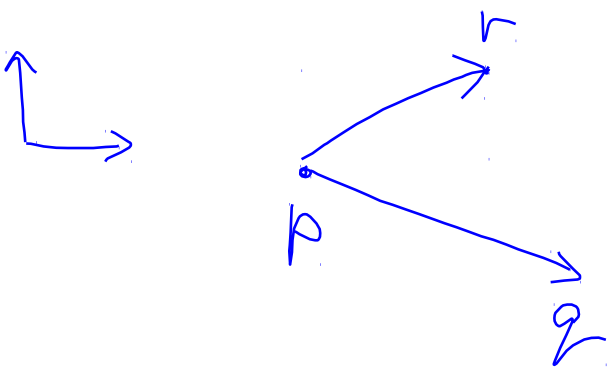
leží

Co chceme po algoritmu

Vstupem je konečná množina bodů v rovině $P = \{p_1, \dots, p_n\}$

Výstupem jsou vrcholy konvexního obalu seřazené ve směru hodinových ručiček.

Nechť přímka pq je orientovaná $\overrightarrow{pq}$. Pak bod r leží vlevo od této přímky (ne na přímce), právě když



$$\det\begin{pmatrix} q_x-p_x & r_x-p_x \\ q_y-p_y & r_y-p_y \end{pmatrix} > 0$$

6

$$\det \begin{pmatrix} q_x - p_x & r_x - p_x \\ q_y - p_y & r_y - p_y \end{pmatrix} = \det \begin{pmatrix} 1 & p_x & p_y \\ 1 & q_x & q_y \\ 1 & r_x & r_y \end{pmatrix}$$

Naivní algoritmus

- procházíme uspořádané dvojice bodů $p\,q$
- zjišťujeme, zda ostatní body z P leží vlevo od $\overrightarrow{pq}$ nebo na $\overrightarrow{pq}$
- pokud ano dáme pq do množiny E
- procházíme dvojice z E, abychom našli posloupnost vrcholů konvexního obalu

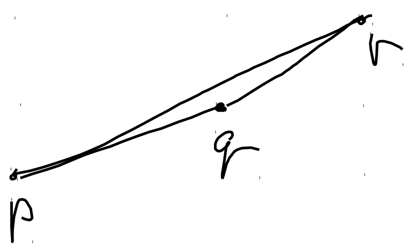
Nevýhody

- časově náročný $O(n^2) \cdot O(n) = O(n^3)$

- řazení vrcholů

P!!!!!

pq
pr
qr



chyba dá

$pr \in E$
$pq \in E$
$qr \notin E$

(8)

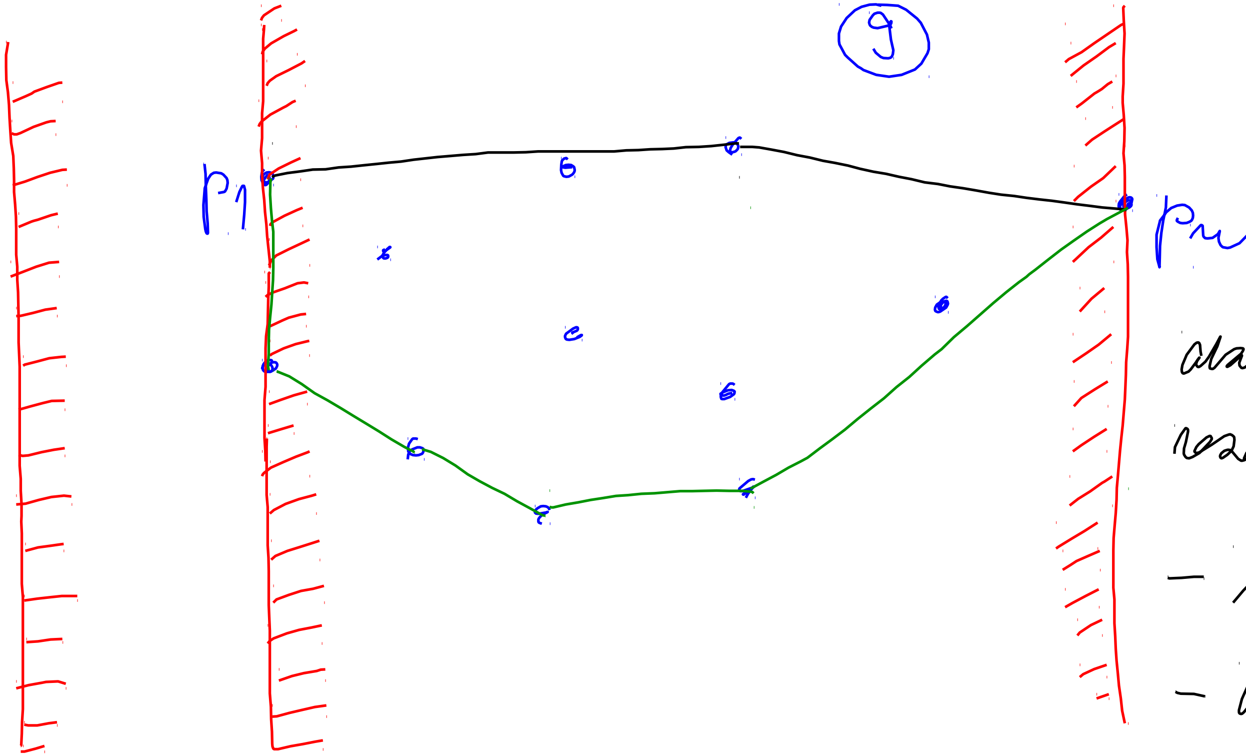
Lepší algoritmus

$P = \{p_1, \dots, p_n\}$

Označme p_1 bod nejvíce vlevo, pokud je jich více tak ten, který je nejnižší (minimální bod v lexikografickém uspořádání: $p < q \iff$ $p_x < q_x$ nebo $p_x = q_x$ a $p_y > q_y$)

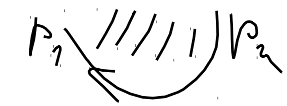
Označme p_n maximální bod v tomto uspořádání.

9



Hranice konvexního obalu je body p_1 a p_n rozdělena na 2 části

- horní konvexní obal 

- dolní konvexní obal 

Algoritmus bude hledat horní a dolní konvexní obal.

Základní myšlenka algoritmu spočívá v tom, že body množiny P uspořádáme v lexikografickém uspořádání

$$p < q \iff (p_x < q_x) \text{ nebo } (p_x = q_x \text{ a } p_y > q_y)$$

Zavedeme značení podle uspořádání

$$p_1 < p_2 < \ldots\ldots < p_n$$

Hledáme horní konvexní obal postupně pro množiny

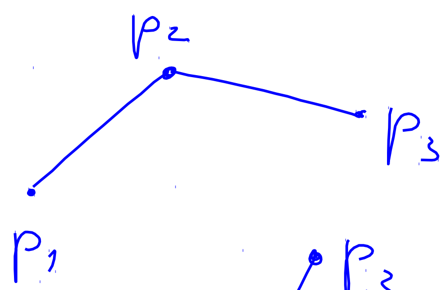
$$\{p_1, p_2, \ldots p_i\}$$

Do konvexního obalu vložíme první p_1, p_2.

Dále přidáme bod p_3

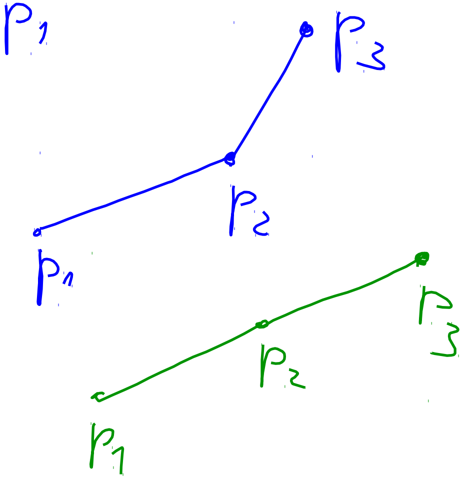
– dvě možnosti

1



Jestliže p_3 leží vpravo od $\overrightarrow{p_1 p_2}$, říkáme, že body $p_1 p_2 p_3$ tvoří zatáčku vpravo. V tomto případě nic neděláme.

2

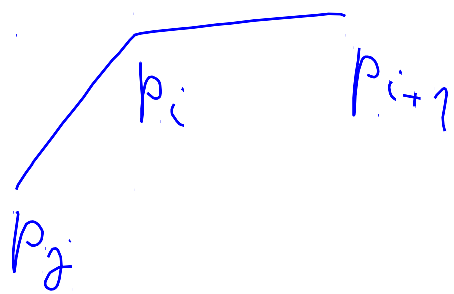


Body $p_1 p_2 p_3$ nedělají zatáčku vpravo. V tomto případě bod p_2 vyřadíme z konvexního konv. obalu.

⑫

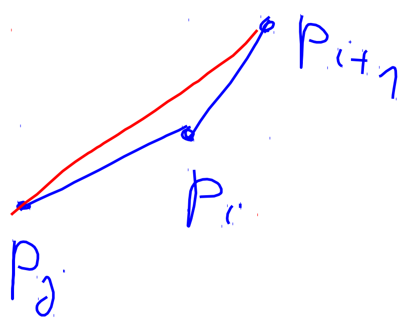
Mějme konv. obal pro $\{p_1, \dots p_i\}$

Přidáme bod p_{i+1} a zjišťujeme zda poslední 3 body v novém konv. obalu tvoří zatáčku vpravo.

① 

nový konvexní obal pro $\{p_1, \dots p_{i+1}\}$ je hotov

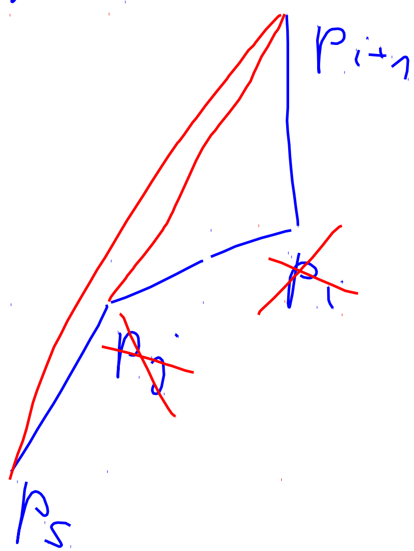
② Body p_j, p_i, p_{i+1} netvoří zatáčku vpravo



V tomto případě vynecháme poslední, tj. p_i

Nicméně musíme zjistit, zda po vynechání p_i poslední tři body tvoří zatáčku vpravo

- pokud ano, skončíme
- pokud ne, vynecháme poslední a zkoumáme poslední tři body, zda tvoří zatáčku vpravo



Tento proces děláme tak dlouho až narazíme na zatáčku vpravo nebo v horním konv. obalu zbudou pouze 2 body.

Časová náročnost algoritmu je $O(n \log n)$

Na vstupu máme n bodů. Časová náročnost algoritmu je funkcí délky n .... $T(n)$

Zápis $T(n) = O(f(n))$

znamená, že $\exists K > 0 \ \forall n \quad T(n) \leq K \cdot f(n)$

V našem případě

$$T(n) \leq K \, n \log n$$

- uspořádání bodů $p_1, p_2, \dots p_n$ trvá čas $O(n \log n)$
- vše ostatní trvá již čas pouze $O(n)$
- do horního konv. obalu každý bod jednou dáme

a nejvýše jedenkrát ho vyndáme

To nám dává pro vytvoření horního konv. obalu čas $O(n)$

V případě, že víme, že konvexní obal bude mít vzhledem k n malý počet vrcholů, je lepší použít jiný algoritmus
- "obalování"

Časová náročnost je $O(n k)$
kde n počet bodů v P a k je počet vrcholů konv. obalu.