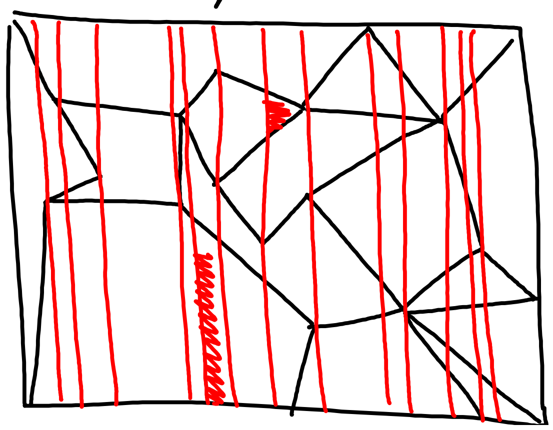
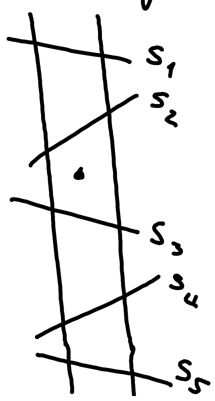
LOKALIZACE BODU

Zadána mapa pomocí dvojitě řetězeného seznamu. Pro tuto mapu chceme najít vyhledávací strukturu $\mathcal{D}$, která pro každý zadaný bod (zadaný souřadnicemi) najde oblast, ve které tento bod leží.

Idea: Rozdělíme danou mapu na menší oblasti, kde vyhledávání bude jednodušší.





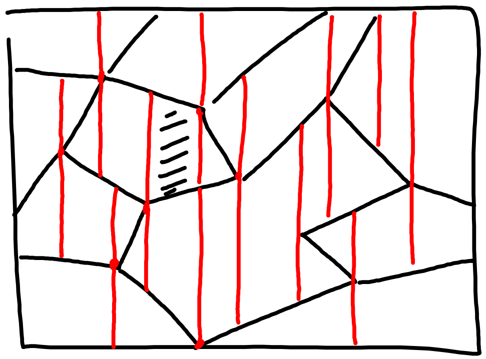
2

V daném pásu máme oblasti uspořádány lineárně pomocí úseček s_1, s_2, s_3, s_4, s_5. Vyhledání lze tedy do provést pomocí lineárního stromu.

Hlavní nevýhoda spočívá v tom, že z rovinného podrozdělení s n úsečkami, uděláme rovinné podrozdělení s $O(n^2)$ oblastmi.

Dělení původní mapy musíme dělat opatrněji – daným koncovým bodem úsečky vedeme vertikální úsečku pouze k nejbližší vyšší hraně a nejbližší nižší hraně daného rovinného podrozdělení
– výsledkem je tzv. LICHOBĚŽNÍKOVÁ MAPA

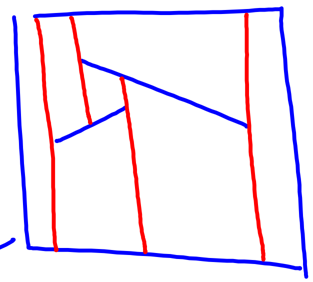
3



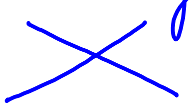
Lichoběžníková mapa vytvořená z rovinného podrozdělení s n úsečkami má nejvýše **3n+1** lichoběžníků.

Lichoběžníkovou mapu lze vytvořit pro každý systém n neprotínajících se úseček umístěných ve čtverci R

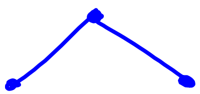
$$S = \{ s_1, s_2, \dots, s_n \}$$



Dvě úsečky se protínají, mají-li společný vnitřní bod



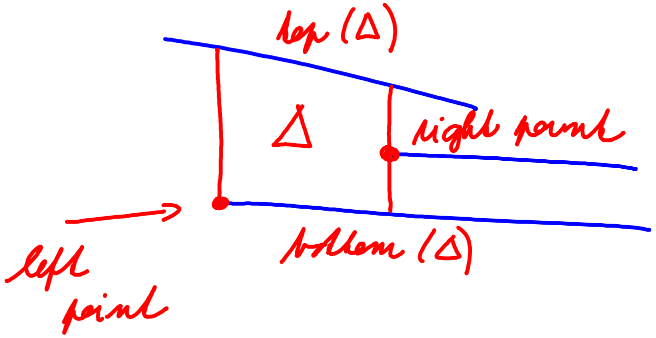
protínají se



neprotínají se

4

Každý lichoběžník v lichoběžníkové mapě je určen 4 prvky



top(Δ) .. úsečka ohraničující Δ shora

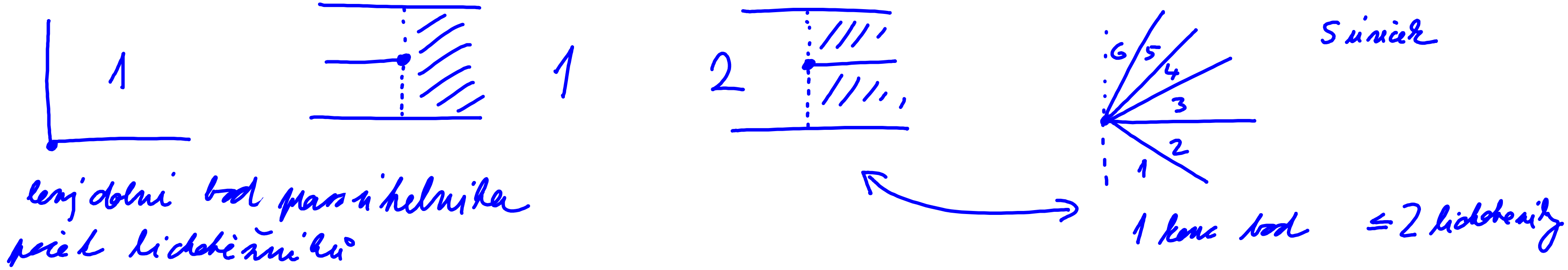
bottom(Δ) ... —— // —— zdola

Předpoklad: Žádné dva koncové body úseček nemají stejnou souřadnici x – (odstraníme na konci)

Věta: Lichoběžníková mapa pro n úseček obsahuje nejvýše 6n+4 vrcholů a 3n+1 lichoběžníků.

(5)

<u>Důkaz.</u> Kolik lichoběžníků je určeno left(Δ) (levým bodem)?



levý dolní bod pravoúhelníka
počet lichoběžníků

$\leq 1 + n + 2n = 3n+1$

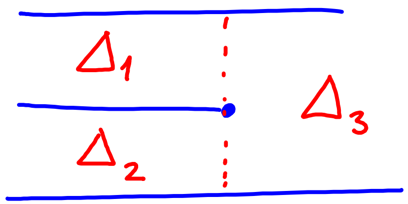
Počet „vrcholů" =

vrcholy R + konc. body úseček + nové body

$\leq 4 + 2n + 2(2n) = 6n+4$

6

Sousední lichoběžník



Δ_3 je pravý horní soused lichoběžníku Δ_1

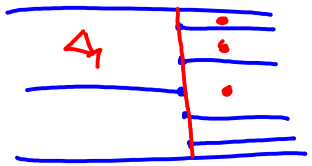
$top(\Delta_1) = top(\Delta_3)$

Δ_3 je pravý dolní soused lichoběžníku Δ_2

$bottom(\Delta_2) = bottom(\Delta_3)$

Δ_1 je levý horní soused lichob. Δ_3

Δ_2 je levý dolní soused lichoběžníku Δ_3



S množina úseček

$T(S)$ lichoběžníková mapa

– ke každému lichoběžníku jsou přiřazeny údaje top, bottom, left, right

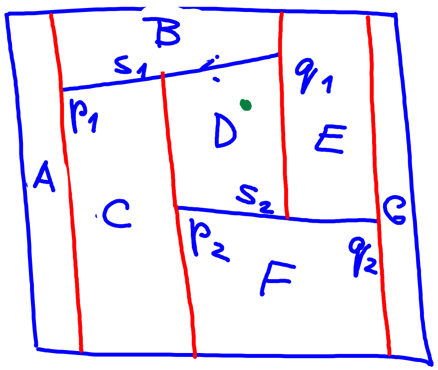
⑦

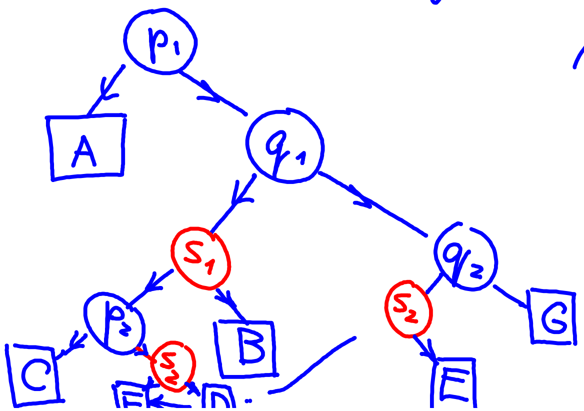
Vyhledávací struktura $\mathcal{D}(S)$ pro lichoběžníkovou mapu $\mathcal{T}(S)$

- orientovaný graf, kde z každého uzlu vychází 2 hrany
- uzly dvou typů: typ x je určen konc. body úseček z S

  typ y je určen úsečkami z S

- listy jsou všechny lichoběžníky dané mapy





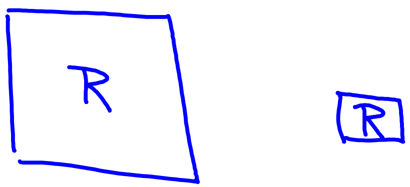
Vyhledávací struktura najde lichoběžník, ve kterém leží daný bod. Pak už je jednoduché, najít L

8

pomocí dvojitě souvislého seznamu oblastí, ve které leží daný lichoběžník.

Protože cesty k listům jsou různě dlouhé, použijeme pro konstrukci vyhledávací struktury náhodnostní algoritmus. Potom bude průměrná délka cesty „dobrá" ($O(\log n)$).

Celkový algoritmus pro konstrukci vyhledávací struktury $\mathcal{D}$ popsán v pseudokódu na str 28



Z $\mathcal{D}(\mathcal{S}_{i-1})$ $\quad \mathcal{S}_{i-1} = \{s_1, s_2, \dots s_{i-1}\}$

vytvoříme $\mathcal{D}(\mathcal{S}_i)$ přidáním s_i.

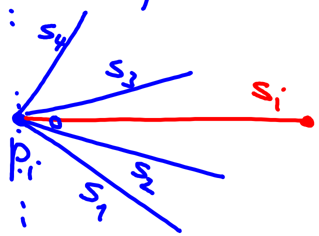
(9)

V $\mathcal{T}(S_{i-1})$ musíme vyhledat lichoběžníky, které protíná úsečka s_i. Úsečka s_i má levý koncový bod p_i, pravý koncový bod q_i

(1) Najdeme lichoběžník Δ_0, ve kterém leží bod q_i.
K tomu použijeme vyhledávací strukturu $\mathcal{D}(S_{i-1})$.

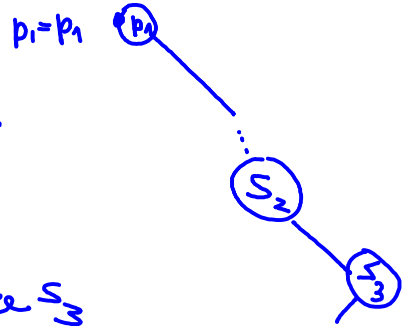
(a) p_i není levým konc. bodem žádné z úseček $s_1, s_2, \ldots s_{i-1}$.
Pak k nalezení Δ_0 použijeme $\mathcal{D}(S_{i-1})$.

(b) p_i je levým konc bodem některé z úseček $s_1, \ldots, s_{i-1}$



V tomto případě najdeme Δ_0 pomocí směrnic úseček s_1, s_2, s_3, s_4 a s_i.

směrnice s_2 < směrnice s_i < směrnice s_3

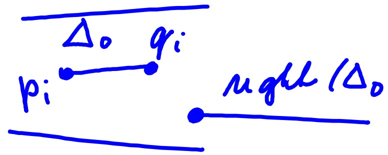


10

Určení dalších lichoběžníků protínaných úsečkou s_i

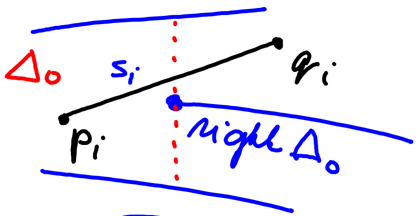
Δ_0 máme

(a) $right(\Delta_0)_x > (q_i)_x$



s_i neprotíná žádný další lichoběžník

(b) $right(\Delta_0)_x < (q_i)_x$



Δ_1 je pravý soused

(i) right(Δ_0) leží pod s_i
pak Δ_1 je horní pravý soused

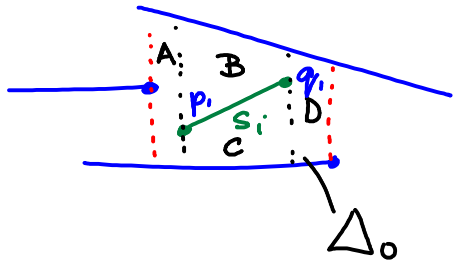
(ii) right(Δ_0) leží nad s_i
pak Δ_1 je dolní pravý soused

viz algoritmus je na str. 29

11

Změny v lichoběžníkové mapě při přechodu od $T(S_{i-1})$ k $T(S_i)$

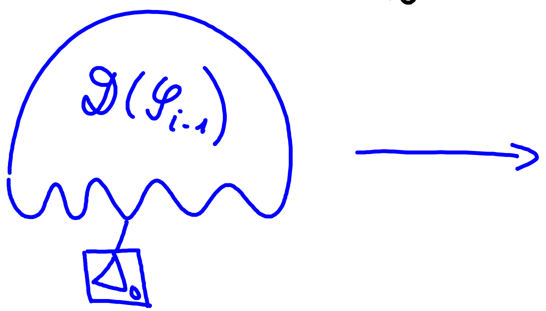
= aktualizace lich. mapy

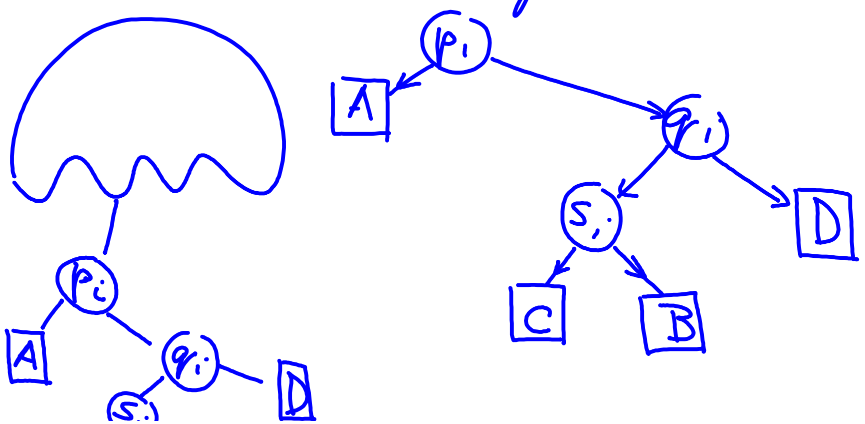


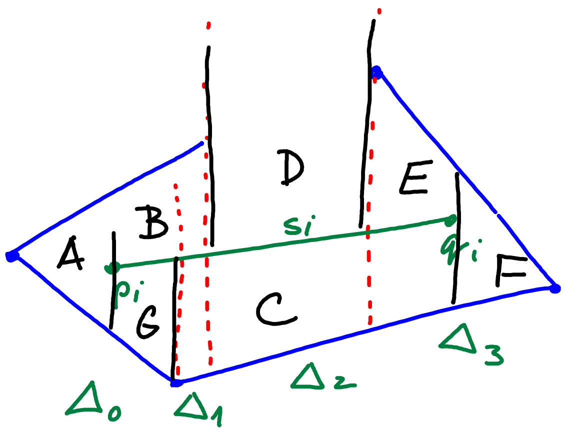
Δ_0 vypustíme z $T(S_{i-1})$

a přidáme A, B, C, D.

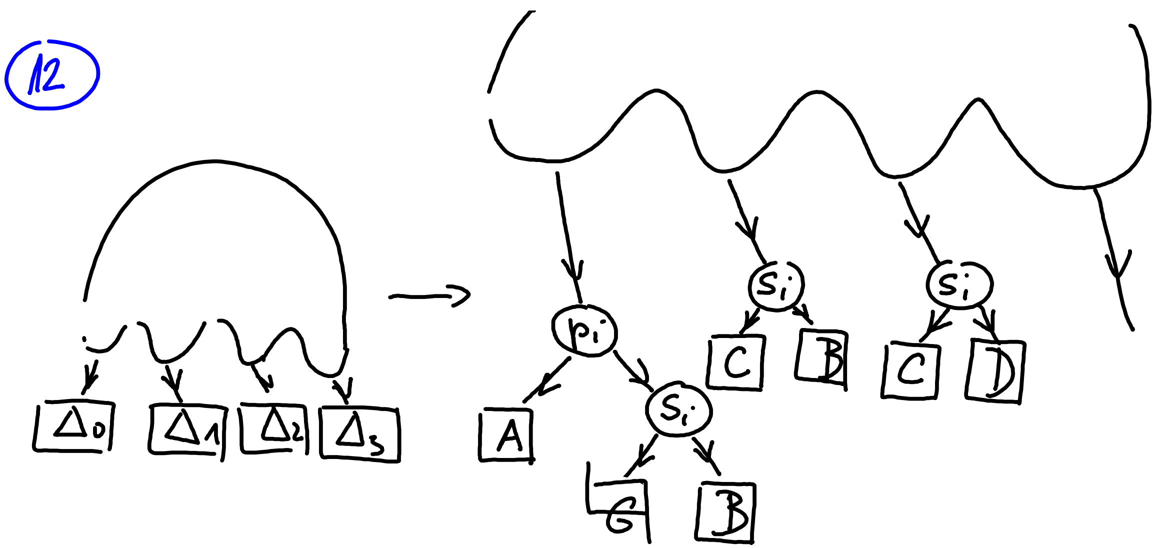
Změna – aktualizace vyhledávací struktury



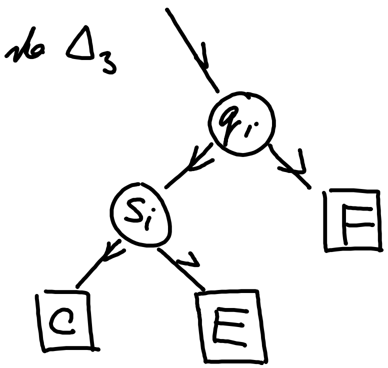




12



nodo Δ_3



(13)

Věta: Očekávaná velikost vyhledávací struktury $\mathcal{D}$ je $O(n)$.

Očekávaný čas konstrukce $\mathcal{T}(\mathcal{S})$ a $\mathcal{D}(\mathcal{S})$ je $O(n \log n)$.

Očekávaný čas vyhledání lichoběžníku, ve kterém leží zadaný bod, je $O(\log n)$