LOKALIZACE BODU

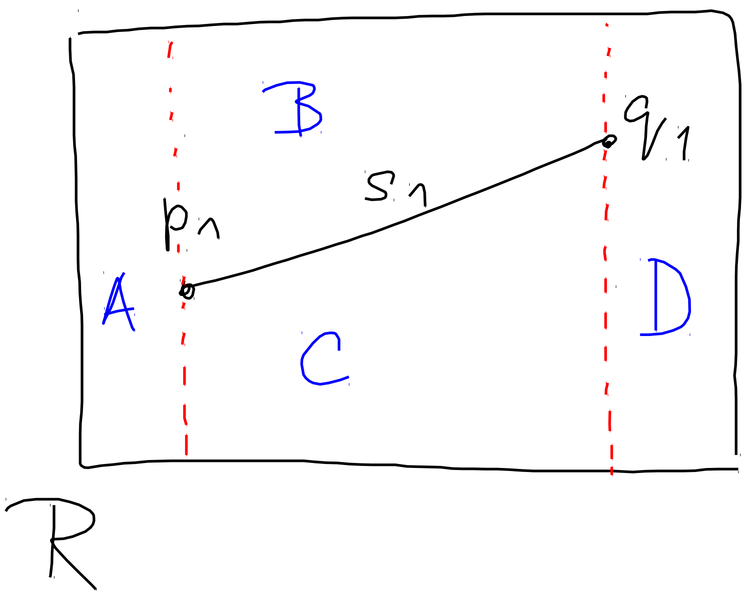
Dáno rovinné podrozdělení, najít vyhledávací strukturu, která určí pro zadaný bod, ve které leží oblasti.

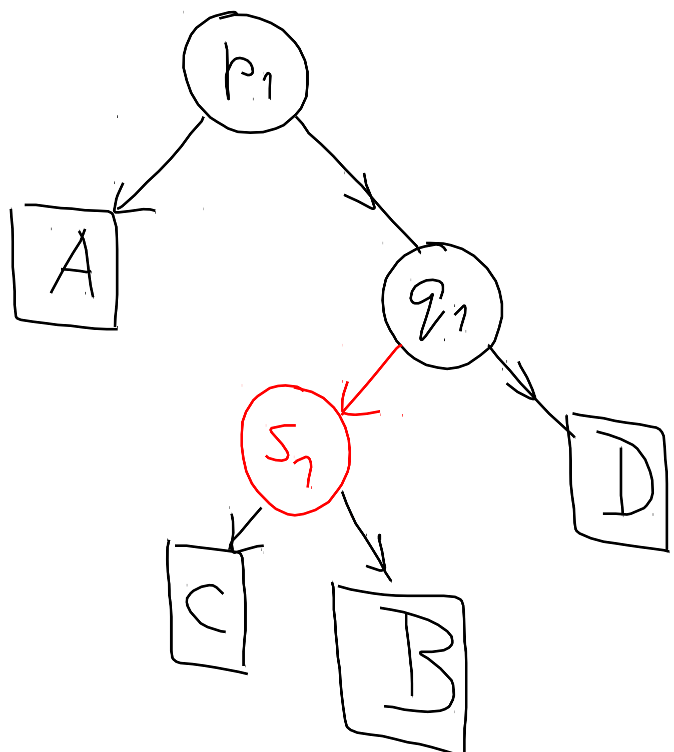
Řešení — dané rovinné podrozdělení „zjemníme", na tzv. lichoběžníkovou mapu a v ní vyhledáváme.

Konstrukci lichoběžníkové mapy a vyhledávací struktury provádíme tzv. přírůstkovým náhodnostním algoritmem.

Předpokládáme, že máme n - úseček $s_1, s_2, \dots s_n$ které se neprotínají ve vnitřních bodech.

Lichoběžníkovou mapu (a vyhledávací strukturu) sestrojujeme postupně pro $1, 2, 3, \dots n$ úseček uspořádaných náhodně.





<u>Predpoklad</u> Žiadne koncové body úsečiek nemajú rovnaké x-ové súradnice

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

$$S_i = \{ s_1, s_2 .. s_i \}$$

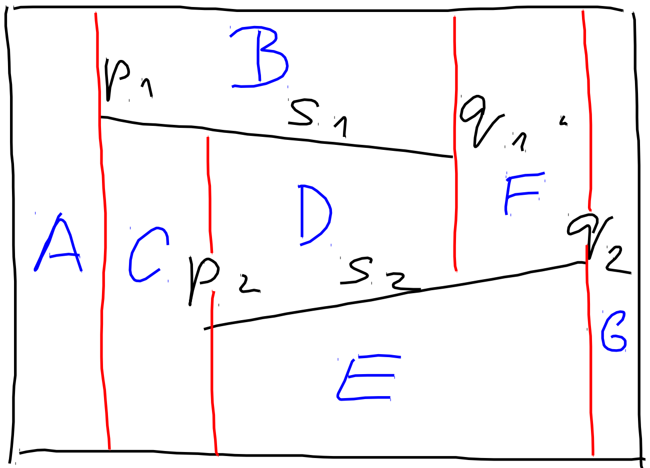
Z $\mathcal{T}(S_{i-1})$ a $\mathcal{D}(S_{i-1})$

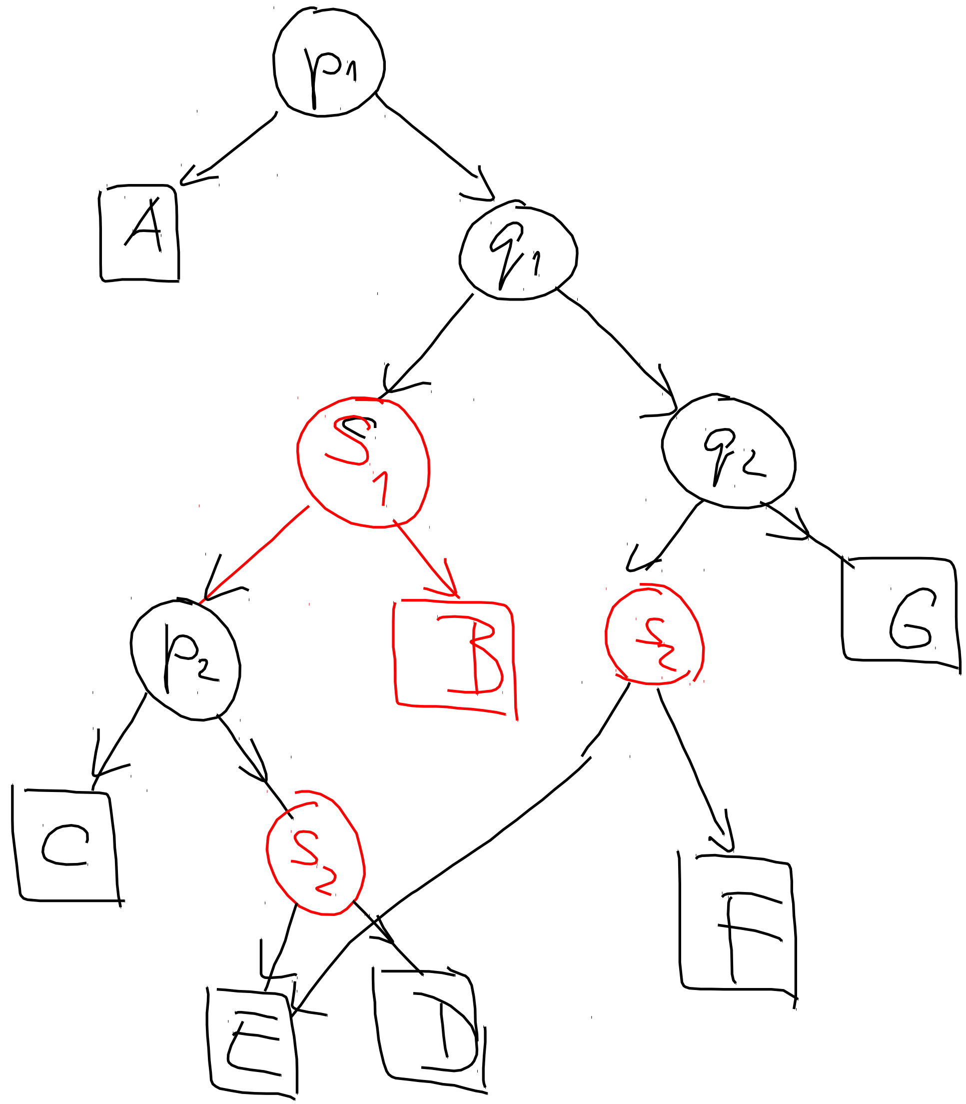
trapezoidal map

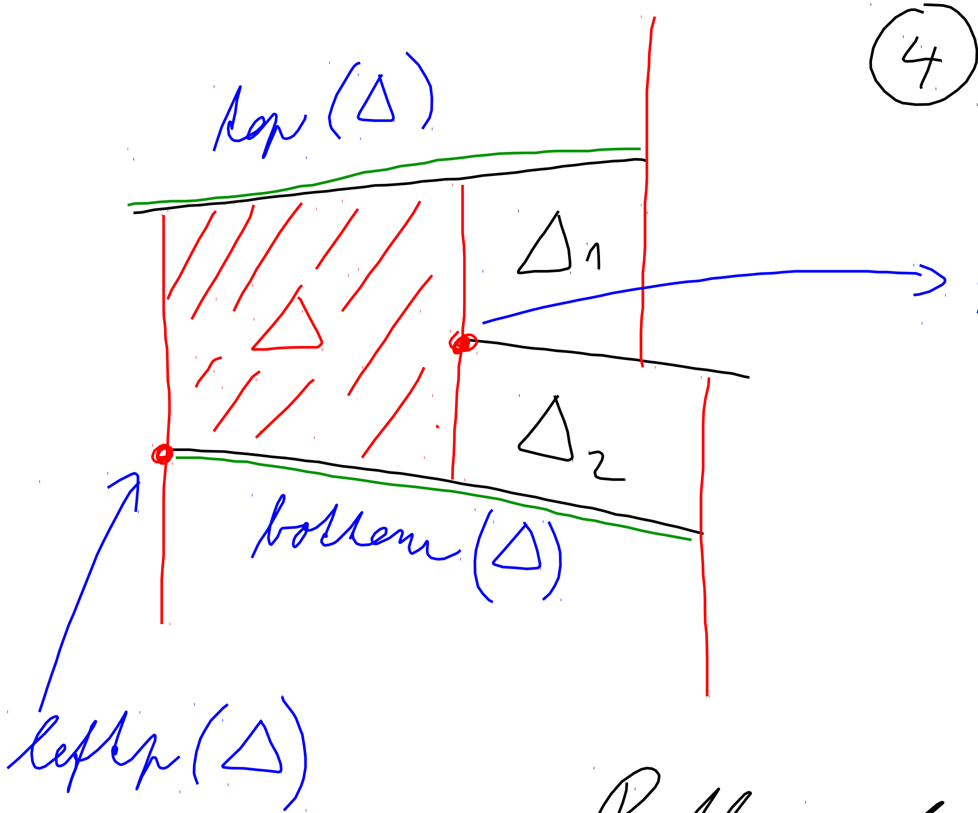
vyhľ. štruktúra (orient. graf)

chceme vytvoriť $\mathcal{T}(S_i)$ a $\mathcal{D}(S_i)$

Lichoběžníková mapa a vyhledávací struktura pro 2 úsečky







Dva lichoběžníky jsou sousední, mají společnou vertikální stranu.

Podle společné další strany (top, bottom) mluvíme o horním a dolním sousedu

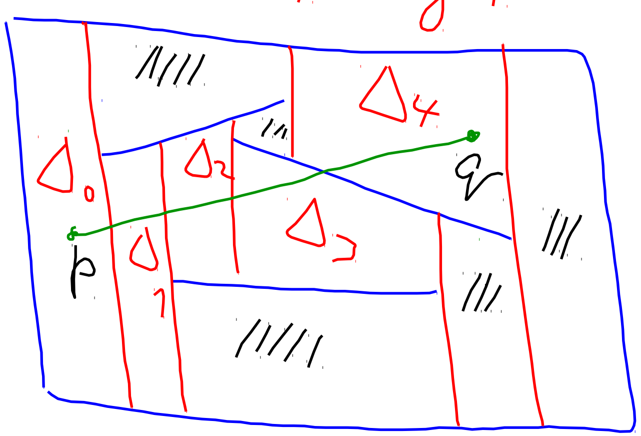
Δ_2 je pravý dolní soused lichoběžníku Δ

Δ je levý horní soused lichoběžníku Δ_1

Přechod od $T(S_{i-1})$ a $\mathcal{D}(S_{i-1})$ k $T(S_i)$ a $\mathcal{D}(S_i)$ se skládá ze 3 algoritmů.

Bak. práce pana Fdrarčnečka jsou explicitně uvedeny všechny 3

1. algoritmus Máme $T(S_{i-1})$ a přidáváme s_i. Algoritmus najde posloupnost lichoběžníků $\Delta_0, \Delta_1, \ldots, \Delta_j$, které s_i protíná zleva doprava.

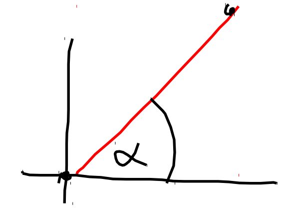


Vstup $T(S_3)$, s_4, $\underline{\mathcal{D}(S_3)}$.

Výstup $\Delta_0, \Delta_1, \Delta_2, \Delta_3, \Delta_4$ z $T(S_3)$.

⑥

Algoritmus první najde Δ_0, ve kterém začíná úsečka $s = s_i$. (Konc. body s jsou p (vlevo), q vpravo.)



směrnice je $\operatorname{tg} \alpha$

① Jestliže p není konc. bod nějaké úsečky z S_{i-1}, pak Δ_0 najdeme pomocí vyhledávací struktury $\mathcal{D}(S_{i-1})$.

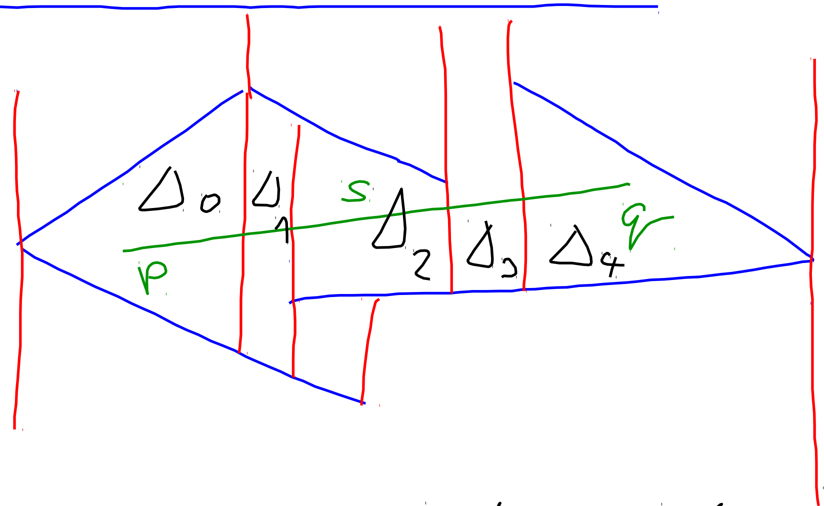
② p je levým konc. bodem nějaké úsečky z S_{i-1}.

zde porov. směrnice s_1 a s

Závěr

$\Delta_0 = B$

Najdeme zobrazování



Je-li q vlevo od right(Δ_0), skončíme.

V opačném případě:

je-li right(Δ_0) nad s, Δ_1 = pravý dolní soused

je-li right(Δ_0) pod s, Δ_1 = pravý horní soused

Takto indukujeme.

Základ algoritmu FOLLOW SEGMENT

Celkový alg. – číslo 28 alg. 29

8

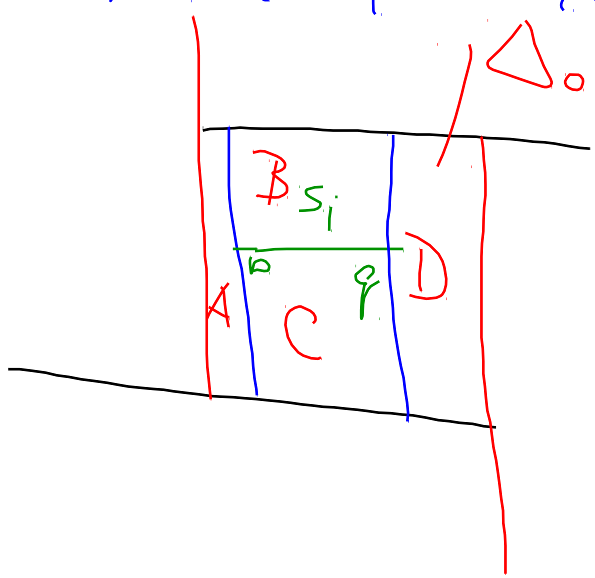
Další dva algoritmy

s_i protíná $\Delta_0, \Delta_1, \dots \Delta_k$ v $\mathcal{T}(S_{i-1})$.

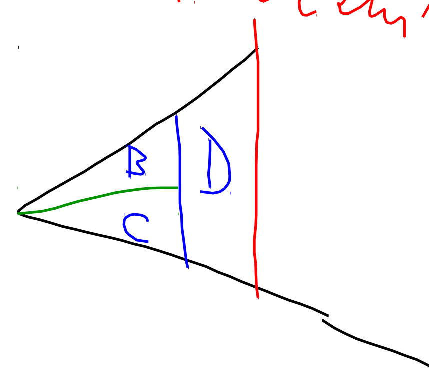
Tj. zrušíme a nahradíme novými lichoběžníky v $\mathcal{T}(S_i)$. Analogické změny provedeme v $\mathcal{D}(S_{i-1})$ a dostaneme $\mathcal{D}(S_i)$.

Pořádně v bak. práci Folvarčného, kde na příkladech

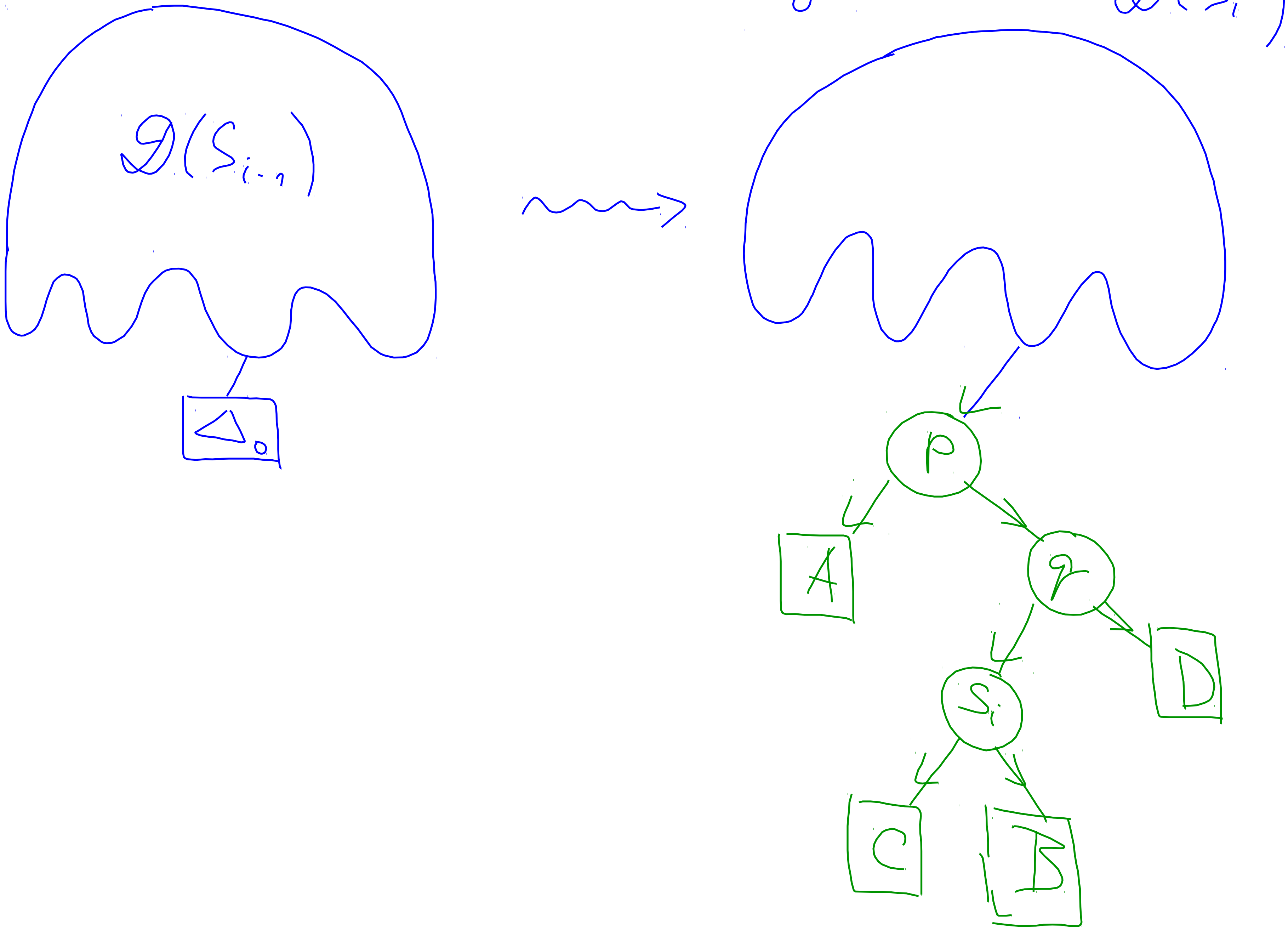
Nechť $k = 0$



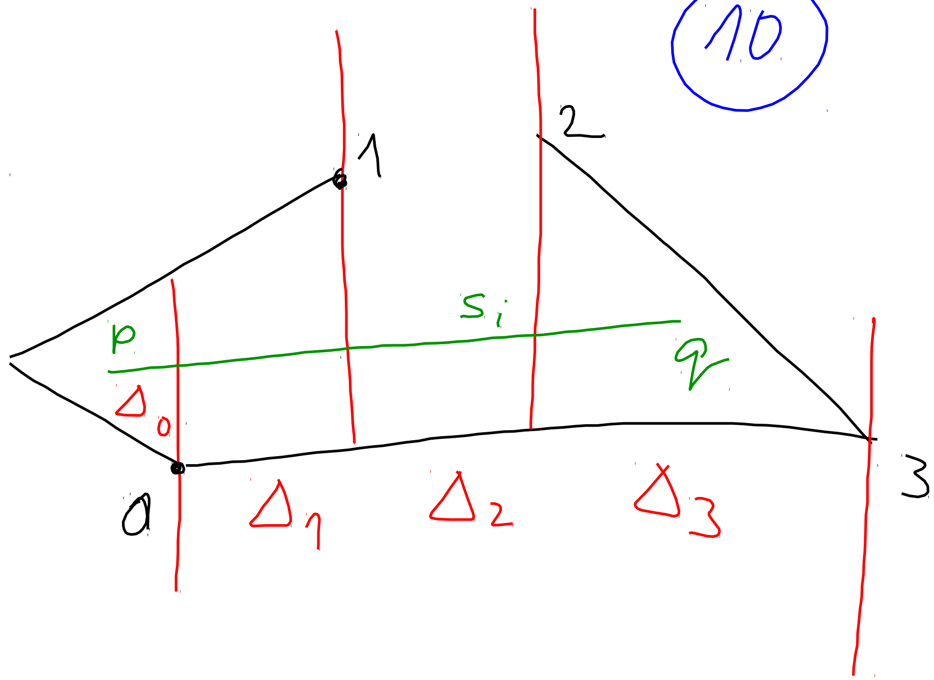
Δ_0 nahradíme v $\mathcal{T}(S_{i-1})$ 4 lichoběžníky A, B, C, D.

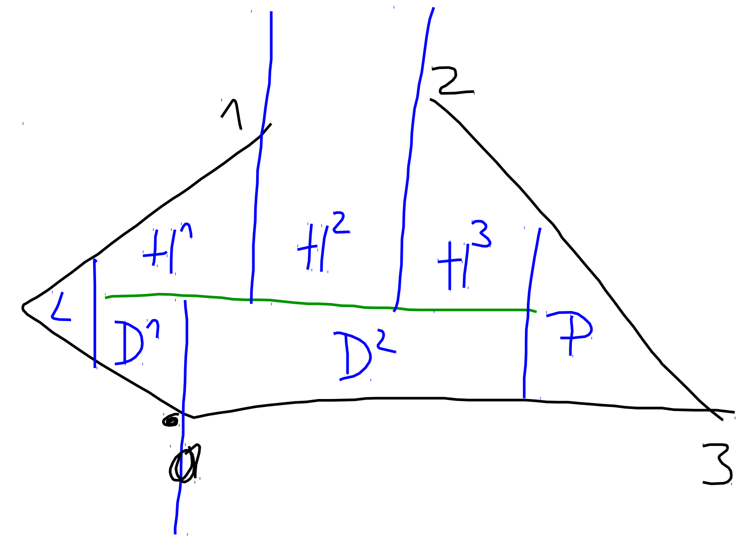


Změna vyhledávací struktury



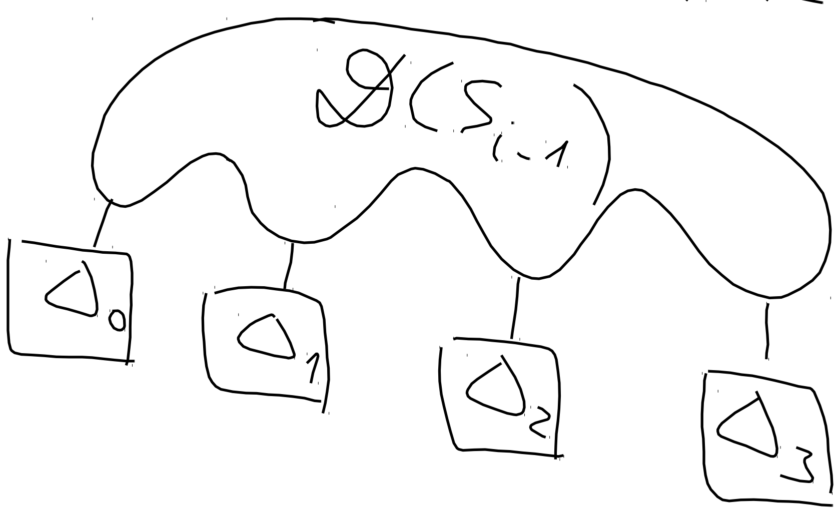
10





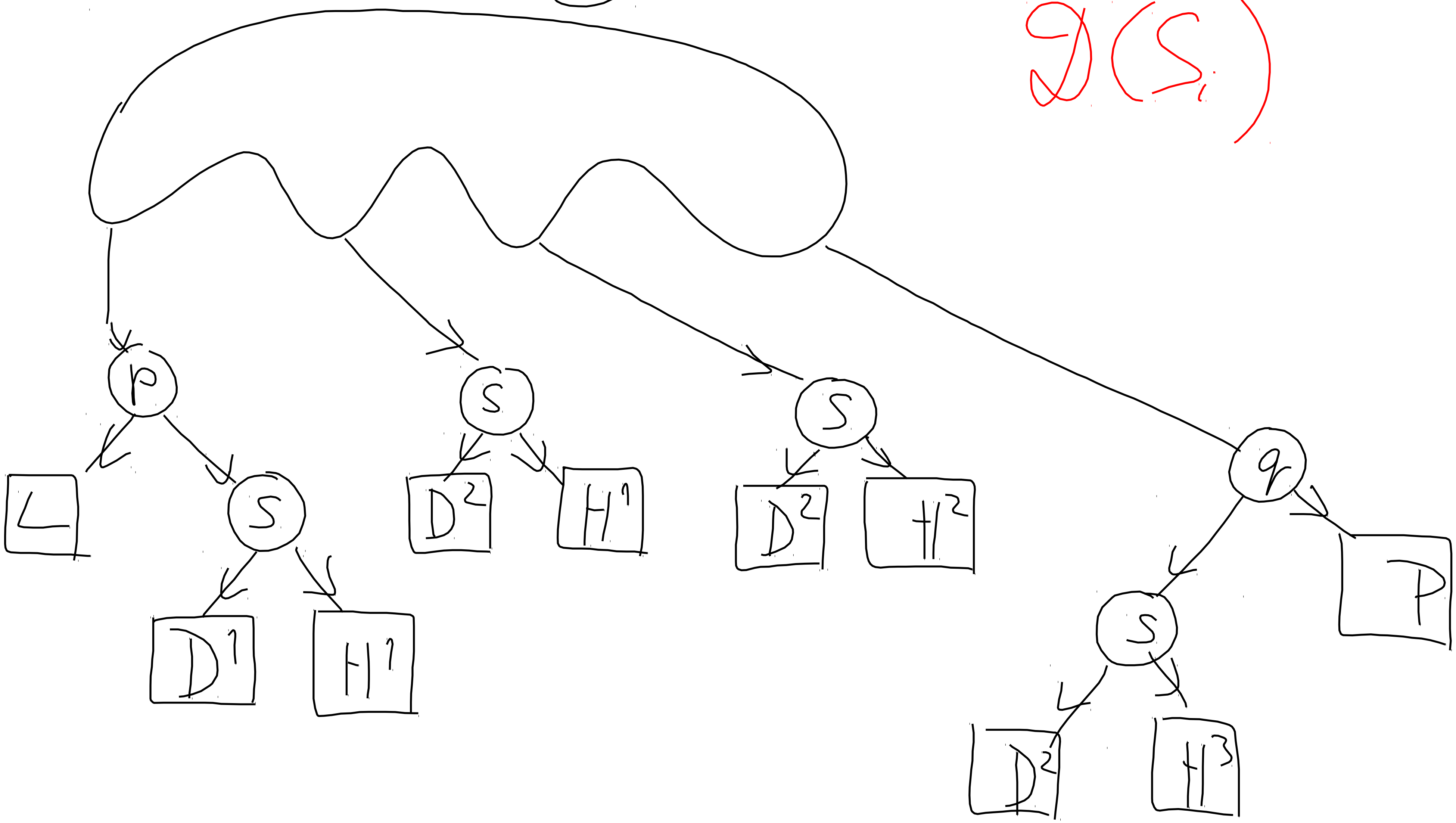
→ průchod od $T(S_{i-1})$ & $T(S_i)$

Průchod od $\mathcal{D}(S_{i-1})$ & $\mathcal{D}(S_i)$



11

$\partial(S_i)$



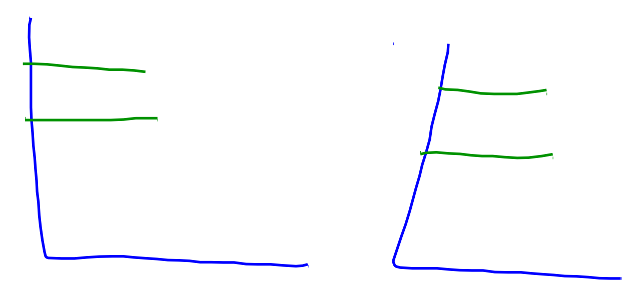
Zjednodušení zadání ve 2 věcech.

(1) Různé koncové body úseček mají různé x-ové souřadnice.

(2) Vyhledávací strukturu D zaměříme pouze na vyhledání bodů, které mají x-ovou souřadnici různou od bodu p_i, q_i.

Tento předpoklad odstraníme zavedením tzv. shear transformation

$$\varphi_\varepsilon : \begin{pmatrix} x \\ y \end{pmatrix} \longmapsto \begin{pmatrix} x+\varepsilon y \\ y \end{pmatrix}$$



Pro $\varepsilon > 0$ dost. malé platí

$$\forall\, i,j \in \{1,2,\dots n\}$$

$$(p_i)_x < (p_j)_x \implies \varphi(p_i)_x < \varphi(p_j)_y$$

Totéž na $q_i, q_j,\ p_i, q_j$

(Pro $2n$ bodů)

$x_i < x_j$ a chceme $x_i + \varepsilon y_i < x_j + \varepsilon y_j$

$y_i - y_j \le 0$ platí $\qquad \varepsilon(y_i - y_j) < x_j - x_i$

$y_i - y_j > 0 \qquad \varepsilon < \dfrac{x_j - x_i}{y_i - y_j} = \varepsilon_{ij}$

$\varepsilon_{ij} = \infty \qquad 0 < \varepsilon < \min\{\varepsilon_{ij}\}$

ε nemusíme počítat, stačí, že existuje

Mějme body p a q.

$p_x < q_x \implies \varphi(p)_x < \varphi(q)_x$ pro vhodné malé ε

$p_x = q_x \implies \varphi(p)_x < \varphi(q)_x$

$\iff p_x + \varepsilon p_y < q_x + \varepsilon q_y$

$\iff \varepsilon p_y < \varepsilon q_y \quad \varepsilon > 0$

$\iff p_y < q_y$

Místo přičítání ε, vezmeme lexikografické uspořádání

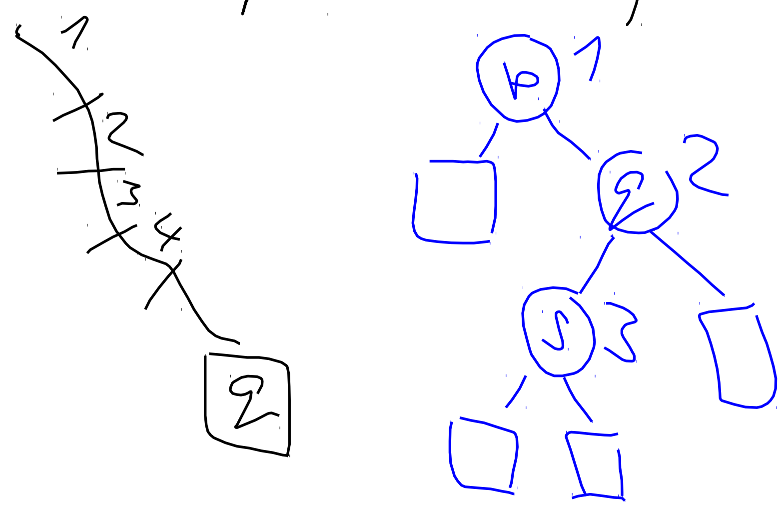
$$\varphi(p)_x < \varphi(q)_x \iff p < q \text{ v lexikografickém uspořádání}$$

Prakticky: v uzlech $\mathcal{D}$, kde pár konc. body úseček porovnáváme lexikograficky.

V uzlech, kde je úsečka, porovnáváme podle souřadnice y.

<u>Věta:</u> Očekávaný čas vyhledávání ve struktuře D vytvořené náhodným uspořádáním množ. X je $O(\log n)$.

Vyhledávání struktury D vytvořené v n krocích. X_i počet uzlů v cestě k bodu q vzniklých v i-tém kroku.



$0 \leq X_i \leq 3$, X_i je náhodná veličina

Očekávaný čas vyhledávání je

$$E\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} E(X_i) \leq$$

$$\leq 3 \cdot p(X_i \neq 0)$$

$p(X_i \neq 0)$ ...... q leží v nějaké

16

Věta: Očekávaný čas vyhledávání ve struktuře D vytvořené náhodným uspořádáním prvků je $O(\log n)$.

Vyhledávání strukturou ~~v~~ D vytvořené v n krocích. X_i počet uzlů v cestě k bodu q vzniklých v i-tém kroku.