Editační vzdálenost

$L(\alpha,\beta)$

baule < hdaule / kule / baule } editační operace: změna, přidání, smazání

↳ Edit. vzdálenost řetězců $\alpha,\beta :=$ min. délka posloupnosti edit. operací, které udělají z α β.

↳ Je to metrika

Operace jsou BÚNO prováděny zleva doprava.

$\alpha = a_1 \dots a_n$, $\beta = b_1 \dots b_m$

$L(\alpha,\beta)$

1. smaže a_1 — $1 + L(a_2 \dots a_n, \beta)$
2. změní a_1 na b_1 — $1 + L(a_2 \dots a_n, b_2 \dots b_m)$
3. před a_1 vloží b_1 — $1 + L(\alpha, b_2 \dots b_m)$
4. ponecháme a_1 a b_1 $(a_1 = b_1)$ — $0 + L(a_2 \dots a_n, b_2 \dots b_m)$

Zkusíme 1…4, vybereme min.

↳ Tohle se dá rekurzivně vyřešit a vrátit nejkratší editační vzdálenost

$L(\alpha_i \dots \alpha_n, \beta_j \dots \beta_m)$

Edit(i,j):

1. Pokud $i > n$: vrátíme $m-j+1$
   Pokud $j > m$: vrátíme $n-i+1$
2. $l_{vloží} \leftarrow 1 + edit(i+1, j)$
3. $l_{smaže} \leftarrow 1 + edit(i, j+1)$
4. $l_{změn} \leftarrow edit(i+1, j+1)$
5. Pokud $\alpha_i \neq \beta_j$: $l_z \leftarrow l_z + 1$
6. Vrátíme min. (l_v, l_s, l_z)

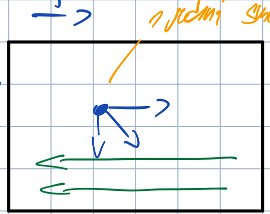
Pomalé pro např.: stejné posloupnosti α i β

$i \in \{1 \dots n+1\}$, $j \in \{1 \dots m+1\}$

Celkem existuje jen $\Theta(n \cdot m)$ různých variant parametrů volání, přestože jich je zavoláno celkem víc

Použiju hashovací tabulku a budu si psát hodnoty

To je návod na nerekurzivní alg.



↗ jediné směry volání

hash. tabulka

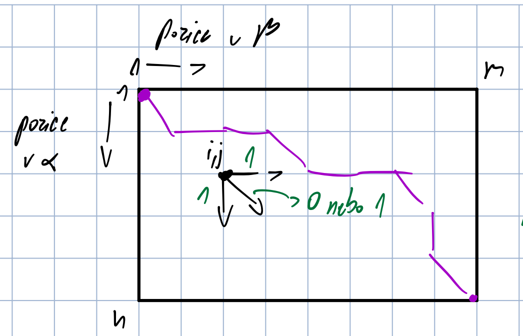
Pokud budu vyplňovat zprava odspodu, zajistím si odpovědi

1. Pro $j = 1 \dots m+1$:
   $T[n+1, j] \leftarrow m-j+1$
   Pro $i = 1 \dots n$:
   $T[i, m+1] \leftarrow n-i+1$
2. Pro $i = n \dots 1$:
3. Pro $j = m \dots 1$:
4. $\delta = \begin{cases} 0 & \text{pokud } \alpha_i = \beta_j \\ 1 & \text{else} \end{cases}$
5. $T[i,j] = \min(1 + T[i+1, j],\ 1 + T[i, j+1],\ \delta + T[i+1, j+1])$

Čas. složitost $= \Theta(n \cdot m)$

Prostor. složitost $= \Theta(n \cdot m)$

Grafový pohled:



nejkratší / nejdelší
Cesta z (1,1) do (n,m)
↓
posl. editací, udělá α z β.

Vytvoříme graf → nejkratší cesta → editační vzdálenost } $\Theta(n \cdot m)$

$\Theta(n \cdot m)$ v DAGu $\Theta(n \cdot m)$

Optimální BST

↗ zohlednění četnosti dotazů

Problém:
Dány klíče $x_1 - x_n$
a váhy $w_1 - w_n \in \mathbb{N}$

Pro BST T na $x_1 - x_n$: hloubky $h_1 - h_n$

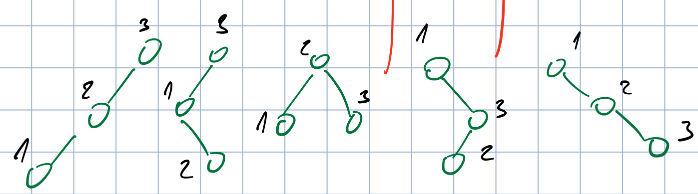
$h_i :=$ # vrcholů na cestě kořen → x_i

cena
$C(T) = \sum_i x_i \cdot w_i$

Cíl: najít $\min(C(T))$ pro nějaký BST T na $x_1 - x_n$.

Např. 1 se ptám 10x
2 1x
3 5x

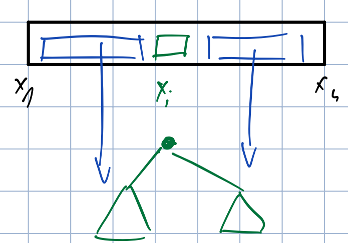
optimum



Ceny: 37 28 31 23 27

vyvážený není nejlepší!

Co kdyby v kořeni opt. stromu bylo x_i



Jestli kořen optimální, pak podstromy taky optimální

$OPT(x_1 - x_n) = OPT(x_1 - x_{i-1}) + OPT(x_{i+1} - x_n) + \sum_{j=1}^{n} w_j$

Kořen ale neznáme, tak zkusíme každý vrchol postupně za kořen

opt. cena BST pro $x_\ell - x_p$

$OptBST(\ell, p)$:

1. Pokud $\ell > p$: return 0.
2. $\min(C_\ell - C_p) + \sum_{i=\ell}^{p} w_i$
   kde $C_i := OptBST(\ell, i-1) + OptBST(i+1, p)$

Je to pomalé

$\ell, p \in \{1 - n+1\}$ → jen $O(n^2)$ podproblémů ↘ v čase $O(n)$ } celkem $O(n^3)$

↳ zavedeme hashování

Nahradíme rekurzi cyklem ... od nejkratších intervalů k nejdelším

Bez rekurze:

1. Pro $l = 1..n$: $T[l, l-1] \leftarrow 0$
2. Pro $d = 1..n$: d = délka intervalu
3. Pro $l = 1..n-d+1$: l := levý okraj
4. $p \leftarrow l+d-1$ p := pravý okraj
5. $T[l,p] \leftarrow \min(C_l .. C_p) + \sum_{i=l}^{p} w_i$

   ↳ $C_i := T[l, i-1] + T[i+1, p]$
6. Vrátíme $T[1,n]$

Čas: $O(n^3)$

prostor: $O(n^2)$

Zapamatujeme si, kde se dosáhlo minimum, to je kořen pro daný interval

Cenu máme, ale jak vypadá strom?

Pak umím vytvořit triviálně strom

Dynamické programování – obecně

Systém podproblémů – stavy
a závislosti mezi nimi → tvoří DAG

Procházíme stavy v topologickém pořadí.

Mějme orientovaný graf s vrcholy $\{1..n\}$ a matici délek hran $L \in \mathbb{R}^{n \times n}$: $L_{ij} = \begin{cases} \text{délka hrany } (i,j) \\ +\infty \text{ pokud hrana neexistuje} \end{cases}$

Chceme matici vzdáleností $D \in \mathbb{R}^{n \times n}$:

D_{ij} = délka nejkratší cesty z i do j.

Umíme:

$n \times$ Dijkstra : $\Theta(n^3)$

$n \times$ Bellman-Ford : $\Theta(n^4)$

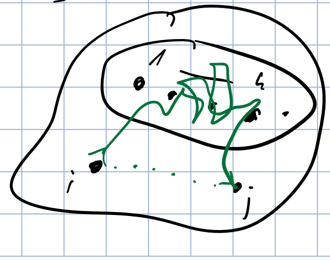
Ukážeme:

Floydův-Warshallův alg. taky $O(n^3)$, ale triviální

Df: D^k_{ij} := délka nejkratšího sledu z i do j, jehož vnitřní vrcholy leží v $\{1..k\}$

↳ matice $D^0, D^1, \dots D^n$

matice L $\qquad$ $= D$

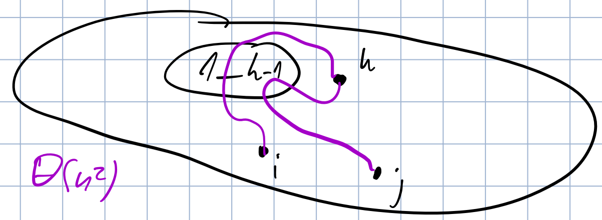


Výpočet D^k z D^{k-1}

$$D^k_{ij} := \min\left(D^{k-1}_{ij},\ D^{k-1}_{ik} + D^{k-1}_{kj}\right)$$

k nepoužito — k použito (Búno) 1x

$\Theta(n^2)$



n kroků: $D^0 \to D^1 \to D^2 \to \dots \to D^n$ čas $\Theta(n^3)$

Nevýhoda: paměť $\Theta(n^3)$

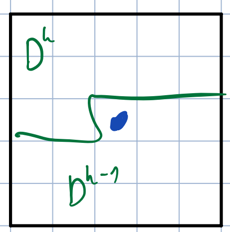
Stačí si pamatovat D^k a D^{k-1} => $\Theta(n^2)$

Nebo můžu přepisovat matici na místě

$D^{k-1}_{ik} = D^k_{ik}$

$D^{k-1}_{kj} = D^k_{kj}$

přepis nevadí



Celkem:

Pro $k = 1 \dots n$:
  Pro $i = 1 \dots n$:
    Pro $j = 1 \dots n$:
      $D_{ij} = \min(D_{ij}, D_{ik} + D_{kj})$

?

Očividně $O(n^3)$