

Řešení úlohy č. 2

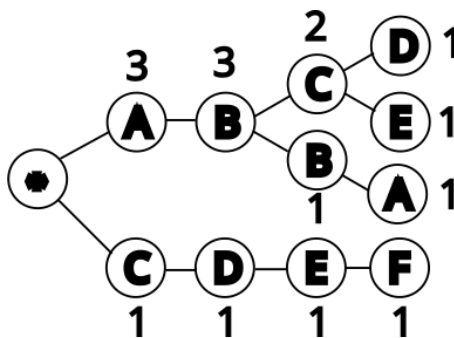
Mapování

Jednotlivé názvy míst si uložíme do prefixového stromu (trie), viz. <https://cs.wikipedia.org/wiki/Trie>. Každá cesta od kořene k listu odpovídá jednomu názvu. Pro každý uzel si ještě uložíme, kolika cest (slov) je součástí. Označme pro uzel v tuto hodnotu jako $c(v)$. Výpočet $c(v)$ lze realizovat buďto přímo při vytváření trie (po každé když přijdu do uzlu přičtu do hodnoty $c(v)$ jedničku a nebo až po vytvoření jedním průchodem (třeba pomocí DFS) - pro listy je $c(v) = 1$ a pro vnitřní uzel je $c(v)$ součtem $c(w_1), \dots, c(w_k)$, kde w_1, \dots, w_k jsou syni v ve stromě. Všimněme si, že hodnota $c(v)$ udává, kolik listů má podstrom zakořeněný v uzlu v .

Například pro vstup

```
15
4
abcd
abba
cdef
abce
6
a 1
b 2
c 3
d 4
e 1
c 1
```

bude vypadat trie nějak takto



Všimněme si, že $c(v) = 1$ právě když cesta z kořene do uzlu v tvoří unikátní prefix napříč všemi názvy. Najítí prefixu tedy lze realizovat přímočarým BFS, které jednoduše modifikujeme tak, aby se zastavilo na uzlu s $c(v) = 1$. Tedy pokud $c(v) = 1$, již do fronty nepřidává žádné sousedy v .

Každý uzel, na kterém jsme se zastavili odpovídá jednomu unikátnímu prefixu. Nyní můžeme najít cenu každého takového prefixu, všechny prefixy seřadit podle ceny a hladově vybírat ty nejlevnější prefixy dokud na to máme tuhu. Máme-li na vstupu M řetězců a každý z nich má délku nanejvýš d (v zadání $d = 100$), trie bude mít v nejhorším případě $O(Md)$ vrcholů. Nejhorší případ nastává tehdy, kdy každé dva názvy začínají na různé písmeno.

Korektnost Pokud by existovala nějaká množina prefixů, která by měla být větší než ta, kterou našel náš algoritmus, tak musí nutně obsahovat nějaký prefix p^* , který náš algoritmus nenašel. Náš algoritmus ale našel všechny prefixy, které mají cenu menší než je cena toho nejdražšího přidaného prefixu. To znamená, že p^* má cenu větší než nejdražší prefix, který našel náš algoritmus. Ale ten jsme odmítli, protože na něj už nemáme tuhu. Tedy předpoklad o existenci takového prefixu p^* vede ke sporu.

Časová složitost BFS zvládneme pustit v čase $O(Md)$ a vybrat unikátní prefixy v nejhorším případě v čase $O(Md)$ (nejhorší případ zde nastává pokud mají každé dvě slova délku d a liší se právě v posledním písmenu (Toto ve skutečnosti nastane jen pokud bychom měli alespoň M znaků v abecedě, pro jednoduchost analýzy toto zanedbejme). Seřadit M slov délky nanejvýš d stihneme buďto v čase $O(dM \log M)$ pomocí standardních řadících

algoritmů (třeba quicksort) a nebo $O(Md)$ využijeme-li přihrádkového řazení. Najít hladově nejlevnější prefixy zvládneme také v čase $O(Md)$. Celkem tedy $O(Md)$.