GRAFOVÉ ALGORITMY - CVIČENÍ 7: KRUSKALŮV A PRIMŮV ALGORITMUS

Převzato z Cormen, Leiserson, Rivest, Stein: Introduction to algorithms. The MIT Press and McGraw-Hill, 2001; kapitola 23.

Příklad 1. Kruskalův algoritmus může vytvořit různé kostry pro stejný graf G v závislosti na seřazení hran podle jejich vah. Ukažte, že pro každou minimální kostru T grafu G existuje takové seřazení hran grafu G, že Kruskalův algoritmus vrací jako výsledek T.

Příklad 2. Mějme graf G reprezentovaný maticí sousednosti. Upravte Primův algoritmus, aby pracoval nad takovýmto vstupem s časovou složitostí $O(V^2)$.

Příklad 3. Je implementace Primova algoritmu pro řídký graf $G = (V,E)$ (tj. $|E| = \Theta(V)$) pomocí Fibonacciho haldy asymptoticky rychlejší než implementace pomocí binární haldy? Změní se situace pro úplný graf, kde $|E| = \Theta(V^2)$? Jaký musí být vztah mezi $|E|$ a $|V|$, aby byla implementace přes Fibonacciho haldu asymptoticky rychlejší než pomocí binární haldy?

Příklad 4. Předpokládejme ohodnocení všech hran grafu celým číslem z intervalu $\langle 1, |V| \rangle$. Jak rychle bude pracovat Kruskalův algoritmus? Co když je ohodnocení hran celé číslo z intervalu $\langle 1, W \rangle$, kde W je nějaká konstanta.

Příklad 5. Předpokládejme ohodnocení všech hran grafu celým číslem z intervalu $\langle 1, |V| \rangle$. Jak rychle bude pracovat Primův algoritmus? Co když je ohodnocení hran celé číslo z intervalu $\langle 1, W \rangle$, kde W je nějaká konstanta.

Příklad 6. Předpokládejte, že ohodnocení hran grafu tvoří uniformní rozdělení nad intervalem $\langle 0, 1)$. Který algoritmus (Kruskalův nebo Primův) bude rychlejší?

Příklad 7. Předpokládejme, že máme již vypočtenu minimální kostru pro graf G. Jak rychle lze aktualizovat minimální kostru při přidání nového uzlu a jeho incidentních hran do grafu G.

Příklad 8. Mějme nový algoritmus pro výpočet minimální kostry stylem "rozděl a panuj", který je popsán následovně. Množinu vrcholů V daného grafu $G = (V,E)$ rozložte na dvě množiny V_1 a V_2 takové, že $|V_1|$ a $|V_2|$ se liší maximálně o jedničku (abs$(|V_1| - |V_2|) \leq 1$, $V_1 \cup V_2 = V$, $V_1 \cap V_2 = \emptyset$). Nechť E_1 (resp. E_2) je množina hran incidentních pouze s uzly z V_1 (resp. V_2). Tento algoritmus rekurzivně spusťte na každém z podgrafů $G_1 = (V_1, E_1)$ a $G_2 = (V_2, E_2)$ pro získání jejich minimálních koster. Nakonec vyberte hranu v E s nejnižším ohodnocením křížící řez (V_1, V_2) a touto hranou spojte dílčí minimální kostry do výsledné minimální kostry grafu G.

Buď dokažte/zdůvodněte korektnost představeného algoritmu nebo uveďte protipříklad, kdy algoritmus selhává.