Josef Doležal

Veta otázky BI-AG1
ZS 2016/17

Acyklický orientovaný graf

zdroj a stok

Acyklický orientovaný graf Uspořádaná dvojice (V, G), kde V je neprázdná množina vrcholů a E množina orientovaných hran taková, že neobsahuje cyklus.

Zdroj Vrchol, do kterého nevede žádná hrana.

Stok Vrchol, ze kterého nevede žádná hrana.

Věta o existenci zdroje v orientovaném grafu

Věta o existenci zdroje v orientovaném grafu

Každý orientovaný graf, který neobsahuje cyklus, má alespoň jeden zdroj.

Věta o existenci zdroje v orientovaném grafu

Věta o existenci zdroje v orientovaném grafu

Každý orientovaný graf, který neobsahuje cyklus, má alespoň jeden zdroj.

Algoritmus topologického uspořádání orientovaného grafu

Algoritmus topologického uspořádání orientovaného grafu

1. Zařaď do fronty všechny vrcholy se vstupním stupněm 0.
2. (dokud není fronta prázdná) Vyber vrchol z počátku fronty
    - \- Vypiš vybraný vrchol
    - \- Pro každého následníka zkontrontroluj, jestli po odstranění hrany má vstupní stupeň 0.
    - \- Pokud má stupeň 0, přidej ho do fronty.

Topologické uspořádání orientovaného grafu

Topologické uspořádání orientovaného acyklického grafu $G = (V, E)$ je takové pořadí vrcholů $v_1, v_2, \ldots, v_n$ grafu G, že pro každou hranu $(v_i, v_j) \in E$ platí $i < j$.

Neorientovaný graf

Neorientovaný graf je uspořádaná dvojice (V, E), kde

1. V je množina vrcholů
2. E je množina hran

Hrana je dvouprvková podmnožina V.

Orientovaný graf

Orientovaný graf

Orientovaný graf G je uspořádaná dvojice (V, E), kde

- V je neprázdná konečná množina vrcholů
- E je množina orientovaných hran

Orientovaná hrana $(u, v) \in E$ je uspořádaná dvojice různých vrcholů $u, v \in V$.
Říkáme, že u je předchůdce v a v je následník u.

Úplný graf K_n

Úplný graf K_n

Úplný graf na n ($n \geq 1$) vrcholech K_n je graf $\left(v, \binom{V}{2}\right)$, kde $|V| = n$.

Úplný bipartitní graf K_n

Nechť $n \geq 1$ a $m \geq 1$. Úplný bipartitní graf $K_{n,m}$ s n vrcholy v jedné partitě a m vrcholy v druhé partitě je graf $(A \cup B, \{\{a, b\} | a \in A, b \in B\})$, kde $A \cap B = \emptyset, |A| = n$ a $|B| = m$.

Kružnice C_n

Kružnice C_n

Nechť $n \geq 1$. Kružnice délky n (s n vrcholy) je graf $(1, \ldots, n, i, i+1 | i \in 1, \ldots, n-1 \cup 1, n)$.

Cesta P_m

Cesta P_m

Nechť $m \geq 0$. Cesta délky m (s m hranami) je graf $(0, \ldots, m, i, i+1 | i \in 0, \ldots, m-1)$.

Doplněk grafu G

Doplněk grafu G

Doplněk $\overline{G}$ grafu $G = (V, E)$ je graf $\left(V, \binom{V}{E} \setminus E\right)$.

Izomorfismus grafů

Izomorfismus grafů

Nechť G a H jsou dva grafy. Funkce $f : V(G) \to V(H)$, která:

- je bijekcí,
- pro každou dvojici $u, v \in V(G)$ platí: $(u, v, \in)\, E(G) \Leftrightarrow f(u), f(v) \in E(H)$.

Automorfismus

Automorfismus G je izomorfismus se sebou samý, tedy $f : V(G) \to V(G)$, která:

- je bijekcí,
- pro každou dvojici $u, v \in V(G)$ platí: $(u, v, \in) \, E(G) \Leftrightarrow f(u), f(v) \in E(G)$.

Stupeň vrcholu $deg_G(v)$

Stupeň vrcholu $deg_G(v)$

Počet hran grafu G obsahujících vrchol v.

Okolí stupně $N_G(v)$

Uzavřené okolí

Množina všech sousedů vrcholu v v grafu G.
Množinu $N_G[v] = N_G(v) \cup \{v\}$ nazveme uzavřené okolí.

Regulární graf

Regulární graf

Graf G je r-regulární, pokud stupeň každého vrcholu je r.
Graf je regulární, pokud je r-regulární pro nějaké r.

Princip sudosti a jeho důsledek

Princip sudosti a jeho důsledek

Pro každý graf $G = (V, E)$ platí

$$\sum_{v \in V} deg_G(v) = 2|E|$$

.

Z tohoto vztahu plyne, že počet vrcholů lichécho stupně je sudý.

Reprezentace grafu

Matice sousednosti

Čtvercová matice $A_G = (a_{ij})_{i,j}^n$ je definována předpisem:

$$\begin{cases} 1 & \{v_i, v_j\} \in E \\ 0 & \text{jinak} \end{cases}$$

Reprezentace grafu

Seznam sousedů

Pro každý vrchol v grafu G uchováváme seznam sousedů (např. spojový seznam). Paměťová složitost je $|V| + 2|E|$.

Indukovaný podgraf

Indukovaný podgraf

Graf H je indukovaný podgraf grafu G, když $V(H) \subseteq V(G)$ a $E(H) = E(G) \cap \binom{V(H)}{2}$.
Podgraf se značí $H \leq G$.

Sled

Doplnit

Cesta v grafu

Cesta v grafu

Podgraf izomorfní nějaké cestě P. Délka cesty je počet hran. V ohodnoceném grafu je pak délka součtem ohodnocení jednotlivých hran.

Souvislá komponenta

Indukovaný podgraf H grafu G, který:

- je souvislý $(\forall u, v \in V : \exists P\,(u, v))$,
- vyvrací existenci souvislého podgrafu F, $F \neq H$ takového že $H \subseteq F$.

V inkluzi se jedná o maximální souvislý podgraf.

DFS

Prohledávání do hloubky

Po výběru počátečního vrcholu p z V spuštěn rekurzivní algoritmus:

- Pokud je p otevřený vrať se ($return$).
- Označ p jako otevřený.
- Pro každého následníka spusť rekurzivně DFS algoritmus.
- Po projití všech následníků označ uzel jako uzavřený.

Orientovaná cesta

$$P_m$$

Nechť $m \geq 0$. Orientovaná cesta s m hranami P_m je graf $(\{0, \ldots, m\}, \{(i, i+1) \,|i \in \{0, \ldots, m-1\}\})$.
(Oproti standardní cestě se jedná o množinu uspořádaných dvojic)

Orientovaná kružnice

C_n

Orientovaná kružnice C_n

Nechť $n \geq 2$. Orientovaná kružnice s n vrcholy je graf

$$(\{1, \ldots, n\}, \{(i, i+1) \,|i \in \{1, \ldots, n-1\}\} \cup \{(n, 1)\})$$

Vstupní stupeň

$$deg_G^+(v)$$

Vstupní stupeň $deg_G^+(v)$

Počet orientovaných hran hran orientovaného grafu G končících ve vrcholu v.

Výstupní stupeň

$$deg_G^+(v)$$

Výstupní stupeň $deg_G^+(v)$

Počet orientovaných hran hran orientovaného grafu G vycházejících z vrcholu v.

Symetrizace

orientovaného grafu

Symetrizace orientovaného grafu

Neorientovaný graf $sym(G) = (V', G')$ kde $V' = V$, a $u, v \in E'$ právě když $(u, v) \in E$ nebo $(v, u) \in E$.

Slabá souvislost

Slabá souvislost

Graf $G = (V, E)$, jehož symetrizace $sym(G)$ je souvislá.

Silná souvislost

Graf, kde pro každé vrcholy $u, v \in V$ existuje orientovaná cesta z u do v
a současně existuje orientovaná cesta z v do u (ne nutně ta samá).

Strom, les a list

Strom, les a list

Strom Graf G, který je souvislý a acyklický.

Les Graf G, který neobsahuje kružnice (nesouvislý, komponenty jsou stromy).

List Vrchol v jehož stupeň $deg_G(v) = 1$.

Tvrzení o existenci listů

Tvrzení o existenci listů

Každý strom T s alespoň 2 vrcholy obsahuje alespoň 2 listy.
Lze dokázat pomocí hledání nejdelší cesty.

Věta o trhání listů

Věta o trhání listů

Je-li $G = (V, E)$ graf na alespoň 2 vrcholech a $v \in V(G)$ je list. Pak:

- G je strom.
- $G - v$ je strom.

Vlastnosti stromů

Vlastnosti stromů

- G je strom.
- Pro každé dva vrcholy $u, v \in V$ existuje právě jedna cesta z u do v.
- G je souvislý a vynecháním libovolné hrany vznikne nesouvislý graf.
- G je souvislý a platí $|V| = |E| + 1$.

Kostra grafu

Kostra grafu

Nechť G je souvislý.
Podgraf K grafu G nazveme kostrou G, pokud
$V(K) = V(G)$ a K je strom.

Vzdálenost dvou vrcholů $d\left(u, v\right)$

Vzdálenost dvou vrcholů $d(u, v)$

Délka nejkratší cesty v G spojující u a v.
Pokud cesta neexistuje (jsou z jiných komponent),
pak $d(u, v) = \infty$.

Prohledávání do šířky

BFS

Prohledávání do šířky BFS

DFS začíná výběrem počátečního vrcholu s a dej mu hodnotu 0, následně:

- Označ všechny vrcholy jako nenalezené.
- Přidej s do fronty.
- Dokud není fronta prázdná
    - Odeber vrchol fronty v a pro každého jeho následníka w:
    - Je-li w nenalezený, označ ho jako nalezený a jeho hodnotu nastav na hodnotu $v + 1$, w přidej do fronty

Vlastnosti kostry BFS

Vlastnosti kostry BFS

Doplnit

Řadící algoritmy

BubbleSort

Funguje na principu probublávání velkých prvků.
Algoritmus vezme dva prvky a pokud jsou ve špatném pořadí, prohodí je. Následně se posouvá o prvek dál.
Ukončuje se ve chvíli, kdy v jednom běhu neproběhlo žádné prohození.
Složitost $O(n^2)$, stabilní, in-place, datově citlivý.

Řadící algoritmy

SelectSort

Funguje na principu vyhledávání nejnižšího prvku. Vstup se rozdělí na seřazenou a neseřazenou posloupnost. V každém kroku se vybere minimum z neseřazené a vloží se na konec seřazené. Volné místo se vyplní sešoupnutím prvků. Složitost $O(n^2)$, nestabilní, in-place a datově necitlivý.

Řadící algoritmy

InsertSort

Na principu řazení vkládáním. Vstup se rozdělí na seřazenou a neseřazenou posloupnost. V každém kroku se vezme první prvek neseřazené posloupnosti a vloží se na správné místo v seřazené.
Složitost $O(n^2)$ (v lepším případě $O(n)$), stabilní, in-place a datově citlivý.

Třídící algoritmus

TopSort

Třídící algoritmus TopSort

- Pro každou hranu (u, v), proveď $D(u){+}=1$. Všechny vrcholy v, které mají $D(v) = 0$ přidej do fronty.
- Dokud není fronta prázdná vezmi a vypiš vrchol v z čela fronty a pro každou hranu směřující z něho do w proveď $D(w){-}=1$, pokud nyní $D(w) = 0$, zařaď ho do fronty.

Řadící algoritmy

Vlastnosti

Pamětová náročnost Rozlišují se In-place a Out-of-place algoritmy.

Stabilita Stabilní, pokud správně seřazené prvky ze vstupu mají stejné pořadí i na výstupu.

Citlivost Určuje, jestli se mění časová složitost na základě vstupu.

Zakořeněný strom

předek, potomek, otec a syn

Zakořeněný strom předek, potomek, otec a syn

Zakořeněný strom Uspořádaná dvojice (T, k), kde $k \in V(T)$ je jeden zvolený vrchol stromu T zvaný **kořen**.

Předek a potomek Leží-li u na cestě z v do kořene, pak je u **předek** a v **potomek**.

Otec a syn Pokud je navíc $\{u, v\} \in E(T)$ hrana, u je **otec** a **syn**.

Binární strom

Strom, který splňuje:

- je zakořeněný,
- každý vrchol má nejvýše dva syny,
- u synů rozlišujeme, který je pravý a který levý.

Binární minimová halda

Struktura tvaru binárního stromu, splňující:

- **Tvar haldy**: Strom má všechny hladiny kromě poslední plně obsazené. Poslední hladina je zaplně zleva doprava.
- **Haldové uspořádání**: Je-li v vrchol a s jeho syn, pak platí $k(v) < k(s)$.

Počet hladin

binární haldy

Počet hladin binární haldy

Binární halda s n prvky má $\lfloor \log n \rfloor + 1$ hladin.

Binární halda

vložení prvku

Binární halda vložení prvku

Binární halda dovoluje vložit prvek na pozici listu. Tímto ale mohlo být porušeno haldové pravidlo. Je tedy potřeba prvek *probublat* na správné místo. Probulání probíhá provnáním s hodnotou v rodiči (pokud je v rodiči větší, prohodí se).

Složitost je $O(\log n)$.

Binární halda

odstranění minima

Binární halda odstranění minima

Odstranit minimum není triviálně možné. Lze ho ale prohodit s nepravějším listem, následně odstranit a list probublat dolů na správné místo.
Složitost je $O(\log n)$.

Binární halda

reprezentace polem

Binární halda reprezentace polem

Pro reprezentaci haldy lze snadno využít pole. Pokud uzly označíme čísly $1, \ldots, n$, pak pro vrchol v s indexem i platí:

- pravý syn má index $2i + 1$,
- levý syn má index $2i$,
- otec má index $\lfloor \frac{i}{2} \rfloor$,
- číslo $i \mod 2$ udává, zda-li v je pravý syn

Binární halda

algoritmus BuildHeap

Binární halda algoritmus BuildHeap

Haldu lze složit v čase $O(n)$ zabubláním prvků, které nejsou listy.
Algoritmus vezme vrcholy $\lfloor \frac{n}{2} \rfloor, \ldots, 1$ a postupně na ně zavolá operaci BubbleDown.

Binární halda

řazení HeapSort

Prvky $x_1, \ldots, x_n$ vložíme do pole a zavoláme na něj *BuildHeap*.
Nyní opakovaně voláme *HeapExtractMin* a hodnoty ukládáme do výstupního pole.
Složitost řazení je $O(n \log n)$.