07 - Základní pojmy teorie grafů

(definice grafu, vlastnosti grafu, charakteristiky uzlů, ohodnocené grafy)

Definice grafu - množina objektů, mezi kterými existují určité vazby spojující tyto objekty.

- Uspořádaná trojice (U, H, f), kde U je množina uzlů, H je množina hran a f: H -> U^2 je incidenční zobrazení počátečního a sousedního koncového uzlu, pokud jsou stejné, jedná se o smyčku.
- U hypergrafu hrany spojují více uzlů

Vlastnosti grafu

- Neorientovaný graf - v základu je graf orientovaný (digraf), jestliže má každá hrana hranu protisměrnou je graf neorientovaný. Symetrizací grafu vzniká neorientovaný graf.
- Prostý graf - graf bez násobných hran (více stejných), opak je multigraf. Prostý graf bez smyček je jednoduchý.
- Prázdný a nekonečný graf - prázdná a nekonečná množina uzlů.
- Diskrétní a úplný graf - žádná hrana a jedna hrana mezi každými dvěmi uzly [(n/2) * (n-1)].
- Bipartitní - množina uzlů je rozdělena na dvě disjunktní třídy (průnik je prázdná množina).
- Regulární - všechny uzly mají stejný stupeň.
- Podgraf - část grafu, pokud obsahuje všechny uzly jedná se o faktor.
- Sled - střídavá posloupnost sousedních uzlů a hran. Pokud se žádná hrana neopakuje, jedná se o tah. Pokud se žádný vnitřní uzel neopakuje jedná se o cestu (cesta je tah).
- Kružnice a cyklus - uzavřená cesta v neorientovaném a orientovaném grafu.
- Souvislý graf - mezi každými dvěmi uzly existuje sled. Každý maximální souvislý podgraf je komponenta.
- Uzlové a hranové řezy - množina uzlů / hran jejichž odstraněním se zvýší počet komponent. Uzlový řez s jedním uzlem je artikulace. Hranový řez s jednou hranou je most.
- Klika a nezávislá množina - Klika je úplný podgraf, nezávislá množina je diskrétní podgraf.

Charakteristiky uzlů

- Uzly - následnící, předchůdci, sousedi uzlu
- Hrany - výstupní okolí, vstupní okolí, okolí uzlu
- Stupně - výstupní stupeň, vstupní stupeň, stupeň uzlu

Ohodnocené grafy

- Uzlové ohodnocení grafu - zobrazení k: U -> R, nazývá se klíč a dá se využít při řazení, vyhledávání či kódování.

- Hranové ohodnocení grafu - zobrazení d: H -> R, nazývá se délka hrany a dá se využít při hledání minimální kostry, nejkratší cesty či toků v sítích.

08 - Reprezentace grafu

(matice sousednosti, matice incidence, výčet sousedů, dynamická reprezentace)

Matice sousednosti - čtvercová matice UxU, u neorientovaného grafu souměrná podle hlavní diagonály. Hodnota 1 značí souseda, hodnota 0 nikoliv. Na diagonále jsou smyčky, multigraf se značí větším číslem.

Matice incidence - obdélníková matice UxH. Hodnota +1 je hrana do uzly, -1 hrana z uzlu (u orientovaného grafu), 0 bez hrany a 2 značí smyčku. Umožňuje zaznamenat hypergraf.

Výčet sousedů - matice obsahují spoustu nul, úspornější je zaznamenat seznam sousedů pro každý uzel, nejlépe pomocí dynamické reprezentace.

Dynamická reprezentace - jednosměrné seznamy uzlů a hran.

```
class Vrchol {
	string klic;
	int ohodnoceni;
	vector<Vrchol*> naslednici; // sousedé vrcholu
	Vrchol *V // následný vrchol
}
class Hrana {
	int ohodnoceni;
	Vrchol ZUzlu, DoUzlu;
	Hrana *H // další hrana v pořadí
}
```

09 - Stromy

(vlastnosti stromů a jejich aplikace, kostra grafu, hledání minimální kostry)

Vlastnosti stromů a jejich aplikace - Strom je souvislý graf, který neobsahuje žádnou kružnici. Pro každé dva uzly tak existuje pouze jedna cesta. Více stromů (komponent) je les. Výška uzlu je počet hran na cestě od kořene, maximální výška uzlu je výška stromu. Arita stromu je maximální počet potomků každého uzlu.

- Prohledávání binárních stromů - level-order, pre-order, in-order, post-order
- Binární vyhledávací strom - hodnoty v levém podstromu jsou menší, hodnoty v pravém podstromu jsou větší než v uzlu.
- AVL strom - samovyvažovací binární vyhledávací strom, výška levého a pravého podstromu se tak vždy liší maximálně o 1.
- Halda - statická reprezentace prioritního stromu, kořen má nejvyšší prioritu, nový list probublává na správnou pozici. Využívá se pro řazení haldou, při něm se vždy vybere kořen (nejvyšší prvek), za něj se dosadí nejnižší list a probublá opět dolů.

Kostra grafu - faktor, který je stromem. Každý strom má jedinou kostru, sebe sama.

Hledání minimální kostry

- Jarníkův algoritmus - růst stromu, vybereme libovolný uzel, který označíme za viditelný, a pak vždy připojujeme nejkratší hranu mezi viditelným a neviditelným uzlem, který se následně stává také viditelným.
- Kruskalův hladový algoritmus - odstraníme všechny hrany, seřadíme je podle velikosti, a pak postupně přidáváme od nejkratší tak, aby nikdy nevznikla kružnice (to se ověřuje pomocí union find algoritmu).
- Borůvkův algoritmus - odstraníme všechny hrany a pak každou komponentu propojíme s jinou pomocí nejkratší hrany, dá se paralelizovat, ale všechny hrany musí mít unikátní ohodnocení.

10 - Grafové algoritmy

(prohledávání grafu, hledání optimálních sledů)

Prohledávání grafu

- BFS - prohledávání do šířky, vyber uzel, expanduj jeho sousedy a vlož je do fronty OPEN, opakuj dokud není fronta prázdná. Možné vylepšení pomocí seznamu CLOSED pro již projité uzly.
- DFS - prohledávání do hloubky, vyber uzel, expanduj jeho sousedy a vlož je do zásobníku OPEN, opakuj dokud není zásobník prázdný. Možné vylepšení pomocí seznamu CLOSED pro již projité uzly.
- Informované prohledávání - použití heuristiky, například A* nebo greedy-search.
- Tarryho algoritmus pro procházení bludiště - DFS, označování dveří IN/OUT, pokud neexistuje východ, projde každou chodbu právě 2x.

Hledání optimálních sledů

- Různé typy optimálních sledů - nejkratší cesta (výchozí), nejdelší cesta (h' = h * (-1)), nejnebezpečnější cesta (maximální součin, h' = -log h), nejširší cesta (minimum šířek, upravený dijkstra).
- Algoritmy pro hledání nejkratší cesty - inicializace je provedena ohodnocením prvního uzlu jako 0, všech ostatních jako nekonečno.
    - Moorův algoritmus - procházení do šířky v neohodnoceném grafu, zapisujeme u každého uzlu vzdálenost od počátku (vzdálenost expandovaného uzlu +1) a předchůdce.
    - Dijkstrův algoritmus - procházení do šířky v nezáporně ohodnoceném grafu, zapisujeme u každého uzlu součet vzdáleností od počátku a předchůdce, procházíme pomocí prioritní fronty.
    - Bellman-Fordův algoritmus - v grafu bez cyklu záporné délky, graf se projde U-1 krát, kdy pro všechny hrany provedeme tzv. relaxaci - změníme hodnotu uzlu, pokud součet hrany a hodnoty předchůdce je menší než původní hodnota a zaznamenáme tohoto předchůdce. Nakonec provedeme relaxace ještě jednou, pokud se nějaký hodnota změní, graf obsahuje cyklus záporné délky.
    - Floyd-Warschallův algoritmus - sestrojíme matici sousednosti délkami hran a postupně se sčítají hodnoty z n-řádku a n-sloupce a nahrazují se jimi vyšší hodnoty na průsečících.

11 - Sítě

(definice sítě a toku, věta o maximálním toku a minimálním řezu, hledání maximálního toku)

Definice sítě a toku

- Síť - orientovaný graf s nezáporně ohodnocenými hranami (kapacity), kde jsou vyznačeny dva uzly, zdroj a stok. Využití v dopravě, potrubí, zatížení datové sítě.
- Tok - hodnota až do velikosti kapacity hrany, která určuje kolik aktuálně hranou protéká, jeho velikost (v celém grafu) je součet toků ze zdroje (na hranách incidentních se zdrojem).

Věta o maximálním toku a minimálním řezu - velikost libovolného toku není větší než velikost libovolného řezu, tedy maximální velikost toku je rovna minimální velikosti řezu.

- Řez - množina hran, po jejichž odstranění by nevedla žádná cesta od zdroje ke stoku. Velikost řezu je rovna součtu kapacit hran v řezu.

Hledání maximálního toku

- Ford-Fulkersonův / Edmonds-Karpův algoritmus - najdi rezervní (někdy též zvanou rozšiřující / zlepšující) cestu (cestu z nenasycených hran) od zdroje do stoku (E.-K. vylepšení - najdi nejkratší rezervní cestu pomocí Moorova algoritmu) a zvyš / sniž (u protisměrných hran) tok všech hran rezervní cesty o její rezervu, opakuj dokud existuje rezervní cesta. Rezervní cesty můžeme hledat i na protisměrných hranách, kde rezerva je inverze a tok snižujeme, díky tomu vždy najdeme maximální tok.
- Další možnosti jsou Dinicův algoritmus nebo Goldbergův algoritmus (operace protlačení vlny a zvednutí uzlu).

12 - Aplikace grafů

(barvení grafu, párování v bipartitních grafech, problémy řešené pomocí teorie grafů)

Barvení grafu - každé 2 sousední uzly mají jinou barvu. Množina uzlů se stejnou barvou je barevná třída, počet barev nutných k obarvení grafu je chromatické číslo (2 pro bipartitní graf / strom, max 4 pro rovinný graf (mapu), |U| u úplného grafu). Barvení grafu je NP-úplný problém, následující algoritmy nezaručují optimální řešení.

- Sekvenční barvená grafu - postupně procházíme uzly a určujeme nejmenší číslo barvy, které lze použít.
- Barvení grafu pomocí nezávislých množin - hledáme postupně co největší nezávislé množiny uzlů, které můžeme obarvit stejnou barvou.
- Barvení grafu slepováním uzlů - spojují se sousední uzly, dokud graf není úplný.

Párování v bipartitních grafech - dvě disjunktní skupiny hran, tedy žádné dvě hrany nemají společný uzel. Slouží například k propojení dělníků a jejich úkolů (úplný bipartitní graf, hledáme maximální párování s minimálním ohodnocením, kde každý pracovník má úkol a každý úkol je řešen).

- Maximální párování - nelze k němu přidat další hranu.
- Největší párování - největší možný počet hran
- Perfektní párování - pokrývá všechny uzly
- Střídavá cesta - její hrany střídavě leží a neleží v párování
- Zlepšující cesta - střídavá cesta jejíž oba krajní uzly jsou volné. Pokud v grafu neexistuje, pak párování je největší.
- Volný uzel - není incidentní s žádnou hranou v párování

Problémy řešené pomocí teorie grafů - problém 7 mostů města Královce, problém obchodního cestujícího, problém obarvení mapy 4 barvami, elektrifikace měst (minimální kostry), vyhledávání, třídění.