Vysoké učení technické v Brně
Fakulta informačních technologií

Hamiltonovská cesta a cyklus v grafu

IAL náhradní projekt 2023/2024

xsvetl07

xskopa17

xmencm00

xhlava60

6. prosince 2023

Obsah

1 Úvod

Cílem projektu bude vytvořit program pro hledání hamiltonovských cest a cyklů v grafu. Jako první krok musíme vymyslet vstupní formát souboru, ve kterém bude graf uložen. Pro umožnění zpracování vstupního souboru bude nutné stanovit jednotný formát. Jednou z možností je použít formát tabulky, kde řádky a sloupce tvoří ID jednotlivých vrcholů grafu. Existující hrany mezi danými dvěma vrcholy mají hodnotu 1, kdežto neexistující hrany mají hodnotu 0. Druhou možností je pro každý vrchol vytvořit pole, do kterého jsou uloženy všechny sousední vrcholy. Pro zpracování vstupního grafu budeme potřebovat zvolit vhodný algoritmus. Pro rychlé ověření můžeme použít hrubou sílu. Toto ale není použitelné pro rozsáhlejší grafy. Další možností bude rotační algoritmus. Ten lze efektivně využít pro grafy s vyšším stupněm. Často využívaný algoritmus je zpětné vyhledávání. Tento algoritmus má nejuniverzálnější použití. Je jednoduchý na implementaci. Dále bude nutné zvolit parametry, pomocí kterých bude uživatel rozhodovat o tom, zda chce nalézt hamiltonovskou cestu či cyklus. A také mít možnost vybrat soubor, ze kterého bude graf načítán. Poslední řešeným problémem bude způsob vypisování nalezených cest/cyklů.

2 Zadání varianty

Cestu v grafu, ve které se vyskytuje každý vrchol právě jednou, nazýváme Hamiltonovou cestou. Má-li tato cesta počátek a konec v jednom jediném vrcholu, pak se jedná o Hamiltonův cyklus v grafu.

Vytvořte program pro hledání Hamiltonovy cesty (pro dva zadané vrcholy) a Hamiltonova cyklu v neorientovaném grafu.

Pokud existuje více řešení, nalezněte všechna. Výsledky prezentujte vhodným způsobem. Součástí projektu bude načítání grafů ze souboru a vhodné testovací grafy. V dokumentaci uveďte teoretickou složitost úlohy a porovnejte ji s experimentálními výsledky.

3 Teorie

3.1 Hamiltonovský graf

Pokud v grafu nalezneme takovou cestu, jenž prochází všemi vrcholy grafu právě jednou s výjimkou počátečního uzlu, který je zároveň uzlem cílovým, tak se jedná o Hamiltonovský graf. Jinými slovy, graf je Hamiltonovský právě tehdy, když obsahuje Hamiltonovskou kružnici (Hamiltonovský cyklus).

Kde Hamiltonovská kružnice je právě nalezená uzavřená cesta. Která začíná v náhodném vrcholu grafu, následně prochází všemi vrcholy právě jednou a končí v počátečním vrcholu. Hamiltonovský graf může obsahovat více kružnic. Podobně je definována i Hamiltonovská cesta, která je otevřená a prochází všemi vrcholy grafu.

3.2 Podmínky Hamiltonovského grafu

Pro potvrzení existence Hamiltonovské kružnice v grafu neexistuje žádná jednoduchá nutná a postačující podmínka. Logicky můžeme předpokládat, že Hamiltonovskou kružnici snadněji nalezneme v grafech s hustou sítí hran než v grafu s nízkým průměrným stupněm vrcholu. Aby vůbec bylo možné uvažovat nad tím, zda je graf Hamiltonovský, je potřeba splnit nutné podmínky.

3.2.1 Nutná podmínka

Graf musí být souvislý a každý vrchol musí mít nejméně druhý stupeň (každý vrchol má minimálně dvě hrany).

Je-li graf G Hamiltonovský graf, tak pro každou neprázdnou vlastní podmnožinu S vrcholové množiny $V(G)$ platí $\omega(G - S) < |S|$, kde ω je počet komponent.

Nutná podmínka se používá spíše k vyvrácení toho, zda je graf Hamiltonovský. Stačí vyvrátit pouze některou z nutných podmínek, a tím můžeme uzavřít hledání Hamiltonovské kružnice. Existují ale i grafy, které splňují nutné podmínky, ale Hamiltonovskými grafy nejsou. Pro potvrzení hamiltonovského grafu se využívají obecně postačující podmínky.

3.2.2 Postačující podmínky

Oreho věta — každá dvojice uzlů nespojených hranou má součet stupňů alespoň n. Graf G s n vrcholy, kde $n \geq 3$. Jestliže pro každé dva nesousední vrcholy u, v grafu G platí $deg(u) + deg(v) \geq n$, tak je graf G Hamiltonovský.

Diaracova věta — každý uzel má stupeň alespoň $\frac{1}{2}n$. Graf G s vrcholy, kde $n \geq 3$. Je-li $\delta(G) \geq \frac{n}{2}$, tak graf G je Hamiltonovský, kde $\delta(G)$ je minimální stupeň grafu.

Pósova věta — pro každé přirozené číslo $j < \frac{1}{2}n$ je počet uzlů, jejichž stupeň nepřevyšuje k, menší než k. Graf G s n vrcholy, kde $n \geq 3$. Jestliže pro každé přirozené číslo $j < \frac{n}{2}$ obsahuje graf G méně než j vrcholů stupně menšího nebo rovného j, tak graf G je Hamiltonovský.

Aby byl graf s $n \geq 3$ uzly Hamiltonovský, stačí splnit jednu z postačujících podmínek. Tyto postačující podmínky ale nemusí nutně každý Hamiltonovský graf splňovat. Příkladem je graf C_n (kružnice o velikosti n) pro $n \geq 5$. Ten nesplňuje postačující podmínky, přesto je Hamiltonovský graf. Protože sám graf je Hamiltonovskou kružnicí.

3.3 Použitelné algoritmy pro nalezení Hamiltonovské kružnice

Problém hledání Hamiltonovské kružnice spadá do skupiny tzv. NP-plných problémů. Jedná se tak o matematický problém, který je považován za jeden z nejtěžších z množiny problémů. Lze jej řešit v polynomiálně omezeném čase na nedeterministickém Turingově stoji. Časová složitost je v lepším případě exponenciální, v horším případě faktoriálová. Hamiltonovská kružnice a cesta se dá použít k vyřešení problému obchodního cestujícího či jezdcova procházka na šachovnici.

3.3.1 Řešení hrubou silou

Systematicky procházíme celý prostor množiny řešení problému. Vyhledávací prostor problému je množina všech možných řešení. Pro problém Hamiltonovské kružnice je velikost vyhledávacího prostoru dána faktoriálem počtu vrcholů daného grafem. V průběhu hledání v úplném grafu je vybrán jeden vrchol z n možných, následující vrchol grafu může být vybrán $n-1$ možnými způsoby, další $n-2$ atd. Časová složitost v nejhorším případě je faktoriální, takže pro větší počet vrcholů je nepoužitelný. Výhodou je, že nalezne opravdu nejlepší řešení a je jednoduchý na implementaci. Lze jej ale využit jen pro grafy s malým množstvím vrcholů.

3.3.2 Heuristické algoritmy

Algoritmy, které dokážou řešit úlohy v polynomiálním (přijatelném) čase. Jsou založené na myšlence, jak by bylo možné danou úlohu vyřešit. Tudíž může být jejich výsledek o něco horší než výsledek přesného algoritmu, ale může najít optimální řešení.

Hladový algoritmus — založen na principu rozhodovacího stromu. Cestu začíná od vrcholu nejvyššího stupně. Následně volí vrcholy podle jejich stupně. Upřednostňuje vrcholy s nižším stupněm, takže se tak často nedostává do slepých uliček.

Rotační algoritmus — postupně přidává vrcholy, aby sestavil Hamiltonovskou cestu a tu následně uzavřel v kružnici. Pokud již nemůže přidat další vrchol, pak použije záměnu pořadí vrcholů a pokouší se dále hledat jinou variantu cesty. Do množiny S si přidává zaměněné vrcholy, aby se předešlo hledání stejných variant. Využívá se spíše v grafech s vysokým stupněm.

3.3.3 Námi zvolený algoritmus

Zpětné vyhledávání (Backtracking) — postupně vytváří kružnici vrchol za vrcholem. Začne ve zvoleném vrcholu a poté se snaží do cesty přidat jeden ze sousedních vrcholů aktuálního vrcholu. Pokud již nelze cestu prodloužit a Hamiltonovská kružnice není hotova, algoritmus se vrací na předchozí vrchol, tak že odstraní aktuální vrchol z cesty.

Obecně exponenciální časová složitost — $O(k^n)$

V horším případě faktoriálová časová složitost — $O(n!)$

4 Implementace

Program je napsán čistě v programovacím jazyku C, pouze pro snadnější překlad jsme do projektu přidali *Makefile* soubor. Projekt byl pro přehlednost rozčleněn do více složek. Kde složka *include* obsahuje hlavičkové soubory, kde nalezneme deklarace použitých funkcí a struktur. Ve složce *src* se nachází zdrojový kód programu. A ve složce *tests* jsou uloženy vstupní grafy pro testování funkčnosti programu.

4.1 Argumenty pro spuštění programu

Program je možno spustit pomocí příkazu:

```
./hamiltonian_algorithm -f filename -s start_node_id -e end_node_id
```

Kde `hamiltonian_algorithm` je název programu, přepínač `-f` pak udává název souboru, ze kterého chceme načíst graf. Přepínač `-s` určuje ID počátečního vrcholu grafu a `-e` je ID koncového vrcholu grafu. Pokud program spustíme bez zadání žádného přepínače na stdout se vypíše HELP menu.

4.2 Načítání vstupního grafu

K načítání vstupního grafu slouží funkce `parse_input()`. Kde po úspěšném otevření vstupního souboru s grafem a úspěšné alokaci struktury `graph_t` dochází k načtení vstupního grafu do pomocného dvourozměrného pole. Pomocí kterého následně proběhne naplnění struktury `graph_t` postupně po řádcích pomocí vnořeného for cyklu. Nakonec dojde k zavření vstupního souboru a vrácení grafové struktury.

4.3 Použité struktury

Pro jednodušší manipulaci s daty jsme použili speciálně definované struktury. Kde struktura `node_t` obsahuje informace o jednom vrcholu grafu (ID vrcholu, stupeň, pole obsahující sousedy vrcholu). Struktura `graph_t` představuje graf načtený ze vstupního souboru (počet vrcholů grafu, pole obsahující vrcholy grafu). A struktura `path_set_t`, do které se ukládají nalezené hamiltonovské cesty (počet nalezených cest, pole obsahující nalezené cesty).

4.4 Funkce pro nalezení cesty grafu

Vyhledávací funkce je volána s parametry `graph`, `start`, `end` a `path`. *Graph* je prohledávaný graf. *Start* je prohledávaný uzel. *End* je konečný uzel cesty. *Path* je dosavadní prohledaná cesta reprezentovaná strukturou `graph_t`, pokud není předána předchozí cesta, nová cesta je inicializována. Prohledávaný vrchol je přidán do cesty, poté jsou vyhodnoceny pro ukončení vyhledávání. Pokud jsou podmínky naplněny, cesta je vypsána. Pokud ukončovací podmínky nejsou naplněny, poté jsou prohledány všechny sousední vrcholy grafu, které ještě nebyly prohledány, voláním vyhledávací funkce nad danými vrcholy.

4.5 Pomocné funkce

Pro výpis chybových zpráv a následné ukončení programu využíváme funkci `error_exit()`. Vstupní argumenty od uživatele načítáme pomocí `parse_cmd()`. Výpis vrcholů grafu spolu s jejich informacemi `print_graph_nodes()`. Výpis hamiltonovské cesty nalezené v grafu `print_path()`. Uvolnění alokované paměti pro graf `free_graph()`.

5 Testování

Pro zjištění funkčnosti našeho programu pro vyhledávání Hamiltonovské cesty jsme si vytvořili vlastní neorientované grafy s různými počty uzlů a vazbami. Vytvořili jsme si celkově přes 40 grafů pro odhalení co největšího množství chyb, které se nachází ve složce *tests*. Při testování jsme testovali grafy, kde se Hamiltonovská cesta nachází, ale i grafy, kde se nenachází. Testy máme rozděleny do složitostí podle počtů uzlů, které se v grafu nachází.

6 Závěr

Cíl projektu se podařilo splnit v plném rozsahu. Program zpracovává vstupní textový soubor s grafem, u kterého jsme se rozhodli použít formát tabulky pro snadnější načítání a předávání dat algoritmu. Druhý formát vstupního souboru (pole sousedních vrcholů pro každý vrchol) jsme zavrhli pro nepřehlednost a následnou složitou manipulaci.

Pro nalezení hamiltonovské cesty a cyklu jsme zvolili algoritmus zpětného vyhledávání. Byl zvolen z důvodu jednodušší implementace a jeho univerzálního použití na různě stupňové grafy. V aplikaci jsme bezpečně ošetřili vstupní argumenty zadávané uživatelem a ověřili, že nedochází k žádnému úniku paměti.

Pro vypsání hamiltonovské cesty je nutné zadat různé vstupní a výstupní ID vrcholu. Pokud chceme vypsat hamiltonovský cyklus zadáme shodné vstupní a výstupní ID vrcholu. Pokud v zadaném grafu neexistuje cesta/cyklus výstup programu zůstane prázdný. V opačném případě se na stdout postupně vždy, na nový řádek, vypíšou všechny nalezené cesty/cykly.

7 Zdroje

Hamiltonovské grafy. Online, Dokumentace projektu. Hradec Králové: Fakulta informačních technologií Vysokého učení technického v Brně, 2020.
Dostupné z: https://theses.cz/id/wy704j/STAG92896.pdf. [cit. 2023-12-05].

Backtracking. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2023.
Dostupné z: https://cs.wikipedia.org/wiki/Backtracking. [cit. 2023-12-05].

Hamitlonovský graf. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2023.
Dostupné z: https://cs.wikipedia.org/wiki/Hamiltonovsk%C3%BD_graf. [cit. 2023-12-05].