Tmatem prce je vizualizace grafovch algoritm, konkrtn prohledvn do hloubky, do ky, Dijkstrv algoritmus a hledn Eulerovsk krunice v neorientovanch grafech (nakreslen grafu jednm tahem).

Vechny algoritmy jsou krokovateln uivatelem, kter zad libovoln graf (viz dle), a pot me krok po kroku sledovat, jak algoritmus probh, a ppadn ho me pustit znovu z jinho potenho vrcholu.

1 Kompilace a sputn

Pro kompilaci je poteba peklada kter um C++11, nap. Clang/LLVM verze 3.5, kter je dostupn v labu. Dle je poteba Qt5, zde by na konkrtn verzi nemlo zleet, ale Qt4 nesta.

Kompilace a sputn se pak provede nsledovn

$ qmake . $ make $ ./build/debug/graphite

Testovno na rznch distribucch Linuxu, vetn labu, a OS X 10.10.

2 Zkladn ovldn

Program se ovld klvesnic i my, kde vechny pkazy jdou zadat bu pes hlavn menu

![hlavni menu](http://i.imgur.com/KJaB5S6.png)

a nebo pomoc klvesov zkratky. Zde je jejich pln seznam:

- Vygenerovn nhodnho grafu ‘Ctrl-R G‘. - Nhodn oritentace hran ‘Ctrl-R D‘. - Vygenerovn nhodnho Eulerovskho grafu ‘Ctrl-R E‘. - Nhodn ohodnocen hran ‘Ctrl-R W‘. - Odstrann orientace hran ‘Ctrl-R U‘.

- Pidn vrcholu ‘A‘. - Spojen dvou vrchol hranou ‘C‘. Je poteba naped vybrat prvn, zmknout ‘C‘, vybrat druh, a zmknout ‘C‘ znovu. - Smazn vrcholu nebo hrany ‘D‘. (nejprve je poteba hranu nebo vrchol vybrat kliknutm myi.) - Oznaen potenho vrcholu ‘S‘. - Start/Restart algoritmu ‘R‘. - Krok algoritmu ‘N‘. - Zmna orientace hrany ‘O‘ (nejprve je poteba hranu vybrat kliknutm myi.) - Nastaven ohodnocen hrany ‘1-9‘, nejprve je ale poteba mt vybran Dijkstrv algoritmus, jinak se ohodnocen hran nezobraz, a pot kliknout na vybran ohodnocen (*ne na hranu*).

Grafy je tak mon uloit do souboru ‘Ctrl-S‘ a znovu nast ‘Ctrl-S‘, piem se zachov i rozloen vrchol v prostoru (pokud je uivatel pesunul.)

3 Pouvn

Nejjednodu je vybrat jeden z piloench graf v souboru, a otevt jej pes ‘File -¿ Open‘, nap. kompletn graf na 5 vrcholech v souboru ‘examples/k5.g‘

![K5](http://i.imgur.com/iYrD1VK.png)

v seznamu algoritm vybrat Eulerovsk tah

![vbr algoritmu](http://i.imgur.com/ewrHxRO.png)

kliknout na libovoln vrchol, vybrat ho jako poten (stisknutm ‘F‘), inicializovat algoritmus (stisknutm ‘R‘), a pot ji krokovat stisknutm ‘N‘.

V libovolnou chvli je mon znovu stiknout ‘R‘, m se algoritmus resetuje a zane pracovat odznova. **Pokud uivatel graf jakkoliv zmn v prbhu algoritmu, je nutn algoritmus resetovat stisktnutm ‘R‘**.

Vtina dleitch informac kter program provd jsou vypisovny do logu v prav stran GUI (a nkter navc na STDOUT). Log v GUI je editovateln text, a lze ho tedy oznait my a smazat.

4 Generovn nhodnch graf

Aby bylo mon aplikaci jednodue pouvat, obsahuje monost vygenerovn nhodnho souvislho grafu (dn ze zabudovanch algoritm nedv smysl vizualizovat na nesouvislch grafech.)

Graf je generovn nsledujcm zpsobem:

- vygeneruje se 10-15 vrchol, kter se postupn spoj hranami, dohromady tvoc jednu velkou cestu - kadmu vrcholu se s pravdpodobnost 2/5 pidl jedna dal nhodn hrana

Takto vygenerovan graf bude vdy souvisl, a dky malmu potu hran i relativn pehledn. Graf je vdy generovan jako neorientovan. Pokud si uivatel peje, me pot nhodn zorientovat hrany (‘Ctrl-R, D‘).

5 Generovn Eulerovskch graf

Protoe pro Eulerovsk grafy mus platit, e kad vrchol m sud stupe, je tak jednoduch nahldnout, e mus leet na njak krunici. Generovn grafu tedy probh tak, e se nejprve vytvo jednovrcholov hraf, a potom se 5-7krt vybere nhodn vrchol z grafu, a pilep se na nj dal krunice dlky 3-5 (pro pehlednost.)

Vsledn graf pak vypad nap. takto

![nhodn Eulerovsk graf](http://i.imgur.com/LQNxfKa.png)

Takto vygenerovan graf m opt vhodu, e je dky menmu potu hran pehledn.

5.1 Rozmstn vrchol

Pi generovn nhodnho grafu jsou vrcholy vdy rozmstny na spirlu, kter se rozvij *zevnit ven*. Pro grafy ve zmnn nhodn generovan grafy je toto rozloen relativn blzko tomu, co by si uivatel mohl pedstatovat, a sta zpravidla pouze pemstit pr vrchol uvnit spirly, aby se pli mnoho hran nekilo.

6 Algoritmy

Vechny algoritmy jsou implementovan jako stavov automat, ktery se stiskem ‘R‘ pesune do potenho stavu, a stiskem ‘N‘ postupn krokuje, a dojde do koncovho stavu, kdy algoritmus dobhl.

Proto jsem zvolil zsobnkovou variantu DFS msto rekurzivn, aby lo jednodue ovldat prbh algoritmu.

7 DFS, BFS

Pro porovnn prohledvn do hloubky a do ky je nejlep zvolit stejn graf, a na nm pozorovat, jak se prbh jednotlivch algoritm li. Oba pouvaj stejnou konvenci, a to e nenavtven vrchol je tmav zelen, oteven je svtle zelen a uzaven je ern.

![barvy vrchol](http://i.imgur.com/CaAOrcu.png)

Jak DFS tak BFS um pracovat s orientovanmi grafy. Orientace hrany se zmn oznaenm hrany my a stiskem ‘O‘. Pro vrcholy ‘A‘ a ‘B‘ se postupn mn typ hrany na ‘A -¿ B‘, ‘A ¡- B‘, a ‘A ¡-¿ B‘.

8 Dijkstrv algoritmus

Dijkstrv algoritmus opt funguje i na orientovanch grafech, priem navc zobrazuje i ohodnocen hran.

![ohodnocen hran](http://i.imgur.com/2d7DzOA.png)

Vha hrany je vdy zobrazena ve smru kam hrana ukazuje, a tedy pro *obousmrn*, resp. *neorientovan* hrany jsou zobrazena ohodnocen dv, kad jednm smrem.

Hranm lze nastavovat hodnoty v rozsahu 1-9, co se provede kliknutm na ohodnocen hrany a stiskem patin klvesy 1-9. Oznaen se zobraz tekovanm tvercem okolo ohodnocen hrany (viz. obrzek).

Samotn dijkstruv algoritmus se pot zobrazuje podobn jako u DFS/BFS. Nenavtven vrcholy jsou tmav zelen, oteven jsou svtle zelen a uzaven jsou ern. Navc se vak u vrchol zobrazuje jejich vzdlenost od potenho vrcholu, a to tak, e se v popisku vrcholu zobraz ‘slo vrcholu / vzdlenost od zdroje‘. Vrcholy zatm neobjeven maj nekonenou vzdlenost, reprezentovno stringem ‘inf‘, viz obrzek.

![Dijkstra prbh](http://i.imgur.com/OWYHOQ7.png)

9 Eulerovsk krunice

Algoritmus pro hledn Eulerovsk krunice se trochu li od prvnch t, a to tm, e funguje pouze na neorientovanch grafech, kde stupe vech vrchol je sud. Pokud je sputn na grafu kter nen Eulerovsk, nemus sprvn fungovat.

Pro implemntaci jsem zvolil Fleuryho algoritmus, kter funguje nsledovn:

1. zani v libovolnm vrcholu 2. vyber hranu kter nen most, pokud jsou vechny mosty tak vyber libovolnou 3. ozna hranu jako smazanou, a pesu se do vrcholu kam hrana ukazuje a opakuj krok 1.

Tento algoritmus je velmi pmoar, a ze vech implementac kter jsem zkouel vede na nejnzornj vizualizaci, a to proto, e bhem svho prbhu me oznaovat hrany kter jsou mosty, a uivatel tak snadno vid, jak se algoritmus rozhoduje kudy pjde.

Na zatku algoritmu sice dn mosty nemohou existovat, ale u po prvnm kroku dojde ke *smazn* jedn hrany, a tm mohou njak mosty vzniknout. Viz obrzek

![graf bez most](http://i.imgur.com/95ubo0l.png)

a po prvnm kroku u mme nalezeny dva mosty.

![mosty jsou oznaeny erven](http://i.imgur.com/Ls741t3.png)

Hledn most je nutn provst po kadm kroku algoritmu, a probh pomoc DFS klasifikace, konkrtn tak, e se prohled cel graf pomoc DFS, najdou se vrcholy, kter le na njak krunici (pomoc zptnch hran v DFS stromu), a ty co nele jsou oznaen jako mosty.

Sloitost algoritmu je tedy kvadratick, protoe pro kad krok je nutn provst cel DFS na odhalen most. Existuj sice alternativn algoritmy, kter jsou ve vsledku rychlej, ale jejich vizualizace je asto velmi matouc, zatmco Fleuryho algoritmus je velmi pmoar.