Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informatiky a kvantitativních metod

Vizualizace dat

Bakalářská práce

Autor: Martin Čapek
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Ondřej Klapka

Hradec Králové Leden 2017

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně pod vedením Ing. Ondřeje Klapky a uvedl jsem všechny použité zdroje.

.....

V Hradci Králové dne 24.4.2017 Martin Čapek

Poděkování:

Děkuji vedoucímu bakalářské práce Ing. Ondřeji Klapkovi za vstřícný přístup a užitečné rady, které mi pomohly při zpracování bakalářské práce.

Anotace

Tato bakalářská práce se zabývá problematikou reprezentace dat v grafické podobě. Cílem je prozkoumat a otestovat možnosti vizualizace a analýzy dat získaných ze sociálních sítí prostřednictvím 3D grafů. Stručně představit historii a více popsat některé sociální sítě. Objasnit důvody použití samotné vizualizace, prozkoumat a porovnat vizualizace ve 2D a 3D. Popsat několik základních metod pro analýzu dat ze sociálních sítí a uvést několik příkladů, kde se analýza takovýchto dat využije. Dále pak otestovat a představit několik existujících zástupců softwaru, které lze použít pro řešení zmíněné problematiky. Výsledkem této bakalářské práce je aplikace pro jednoduchou vizualizaci dat a otestování základních algoritmů.

Annotation

Title: Data visualization

This bachelor thesis deals with the representation of the data in graphical form. The aim is to explore and test possibilities of the visualization and analysis of data collected from social networks via 3D graphs. Briefly introduce the history of social networks and describe some of those in more detailed way. Explain the reasons for using the visualization itself, examine and compare the visualizations in 2D and 3D. Describe some basic methods for analyzing data from social networks and list several examples where the analysis of such data is useful. Then test and introduce several representatives of existing software that can be used to solve the mentioned problems. The result of this thesis is a simple application for data visualization and testing of basic algorithms.

Obsah

Seznam obrázků

Seznam pojmů

2D ..... dvourozměrné zobrazení

3D ..... třírozměrné zobrazení

API ..... aplikační programové rozhraní

Layout ..... grafické rozvržení

Plugin ..... modul aplikace rozšiřující funkčnost

Framework ..... sada nástrojů, usnadňující vývoj aplikace

Metrika ..... číselné ohodnocení části grafu pro její konkrétní vlastnost

Open-source ..... licenční označení softwaru – zdrojový kód je volně dostupný

Closed-source ..... licenční označení softwaru – zdrojový kód není dostupný

GNU GPL ..... druh licenčního označení svobodného softwaru

LGPL ..... druh licenčního označení svobodného softwaru

BSD ..... druh licenčního označení svobodného softwaru

CSV ..... formát souboru tabulkových dat

JSON ..... formát souboru pro organizovaná data

GDF ..... formát souboru pro ukládání dat grafů

GLSL ..... OpenGL Shading Language

Úvod

Lidský mozek musí každý den zpracovat nepředstavitelné množství informací a dat. Tyto informace a data jsou získávány pomocí smyslových orgánů a zhruba přes 80 % těchto informací je vnímáno zrakem. A z tohoto důvodu mozek dokáže daleko lépe a rychleji zpracovat vizuální informace než jen obyčejný text. Právě proto se při učení nebo při prezentování různých projektů často užívá různých diagramů nebo mentálních a konceptuálních map. Vizualizované informace jsou tedy lépe zapamatovatelné a pochopitelné, což podporuje tvrzení, že jeden obrázek vydá za tisíc slov. Cílem vizualizace může tedy být snazší pochopení zkoumané problematiky a vyvozování závěrů, lepší zapamatovatelnost, názornost anebo jen pro efekt a zaujetí cíleného publika. Vizualizace se uplatní v mnoha vědních oborech.

Často využívaným námětem pro vizualizaci se v poslední době staly sociální sítě. Ještě před nástupem počítačů Jacob L. Moreno, sociolog a psychiatr, studoval vztahy a interakce členů v rozdílných skupinách. Jako první vytvořil grafickou reprezentaci sociální sítě užitím výsledků z jeho pozorování, které začalo okolo roku 1930. Jacob L. Moreno je považován za jednoho ze zakladatelů analýzy sociálních sítí. O několik desítek let později se tento obor stal nesmírně populární, protože s příchodem internetu a webových sociálních sítí se daleko snáz získávají cenná data. 1.1.[14]1.1.[15]

Milióny uživatelů denně navštěvují tyto webové služby a sdílejí nejrůznější informace a obsah. Analýza a následná vizualizace takovýchto dat je tedy nesmírně užitečná například při tvorbě politických průzkumů, v marketingu a obchodování a ve spoustě dalších odvětvích. Obyčejný graf nebo tabulka není zrovna ideální způsob, jak reprezentovat a následně analyzovat různá data od miliónů uživatelů. Reprezentace pomocí 3D grafů nabízí mnoho výhod, jak bude popsáno v následujících kapitolách.

Tato bakalářská práce je rozdělena do čtyř hlavních kapitol a má jak teoretickou, tak i praktickou část. První kapitola představí vizualizaci dat, tedy o co se jedná, stručnou historii vizualizace, základní terminologii a další informace ohledně tvorby grafových vizualizací. V další části budou představeny některé sociální sítě. Tato část je zaměřena především na jejich historii a základní informace. Třetí kapitola se zabývá analýzou sociálních sítí. Budou stručně představeny nejpoužívanější algoritmy pro

analýzu a také existující programy, které umožňují vizualizace a analýzy různých sítí (samozřejmě i sociálních). Poslední kapitola je vyhrazena pro popis implementace aplikace pro vizualizaci sítí.

1 Vizualizace dat

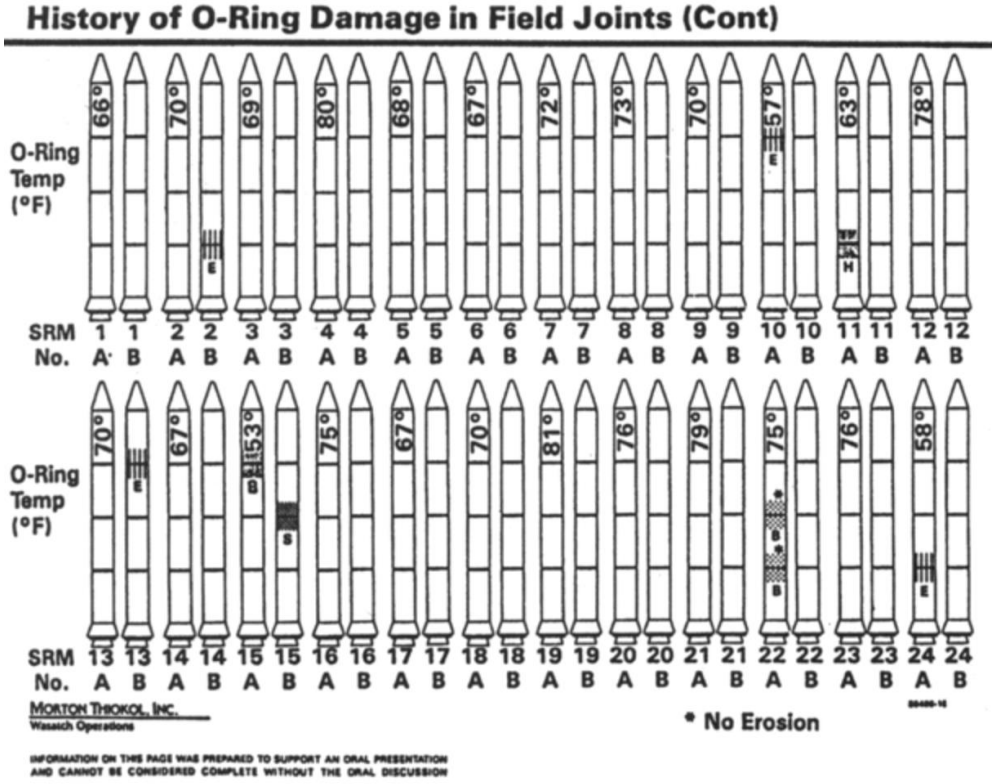
Definic pro vizualizaci dat je mnoho a často se velmi rozcházejí ve svém znění. Vizualizaci lze tedy definovat takto: "Vizualizace dat je proces zkoumání dat a informací po jejich převedení do grafické podoby. Jejím cílem je pochopení zkoumaných jevů a vniknutí do problému. Proto o vizualizaci mluvíme též jako o vizuální analýze dat." [1]. Je to velmi obecná definice. Mezi nejčastější druhy vizualizací patří například různé diagramy, mapy, grafické symboly anebo grafy. Tato práce je zaměřena na vizualizaci pomocí grafů. Vizualizace mají uplatnění snad v každém vědeckém oboru. Například při plánování interních odkazů webu, mapování sociálních sítí, dopravy, biochemické sítě prvků nebo rozložení populace je velice výhodná vizualizace grafem.

První dochované vizualizace, vyryté do kamene nebo napsané na papyru[1]se datují až několik tisíc let před naším letopočtem. Takovéto vizualizace vyprávěly příběhy tehdejších lidí nebo poskytovaly návody k lovu a jiným činnostem následujícím generacím. Postupem času však získaly vizualizace daleko větší význam a mnohokrát vedly i k záchraně životů. Například roku 1854, když v Londýně vypukla epidemie cholery, doktor John Snow zaznamenával místa úmrtí Londýnských obyvatel, postižených touto chorobou do mapy města. Po zaznamenání přibližně 500 úmrtí byl schopen podle mapy určit epicentrum nákazy – vodní zdroj na Broad street. Zamezením přístupu k tomuto vodnímu zdroji omezil šíření nákazy a díky tomu se podařilo zachránit mnoho lidských životů. 1.1.[14]

Naopak jsou i případy, kde neefektivní nebo špatně pochopená vizualizace vedla ke ztrátě lidských životů. Eduard Tufte, autor knihy Visual explanations: images and quantilities, evidence and narrative (1997), ve svém díle uvádí jako příklad tragédii amerického raketoplánu Challenger (28. leden 1986), který se po 73 vteřinách letu vznítil a zemřelo tak všech 7 členů posádky. Tufte za viníka označuje neefektivní, a tedy i špatně pochopenou vizualizaci. Inženýři z firmy Morton Thiokol, která dodávala těsnění raketoplánu, se snažili přesvědčit NASA2, že správná funkce dodaného těsnění je závislá na teplotě – čím nižší teplota, tím vyšší pravděpodobnost disfunkce. Na obrázku jsou znázorněny rakety z předchozích 24 letů jejich teploty při startu a rozsahy

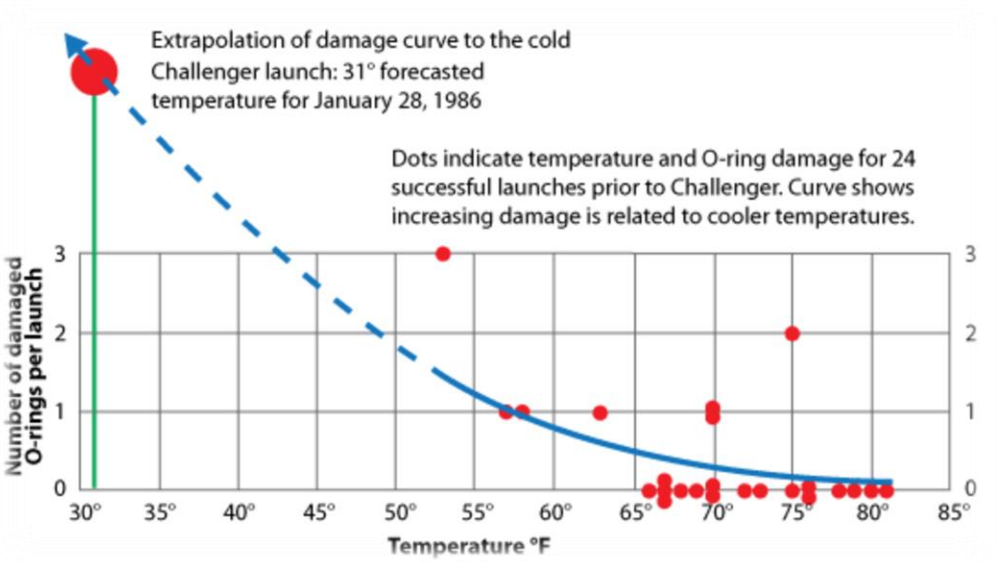
[1] Psací materiál ve starověkém Egyptě
[2] Národní úřad pro letectví a kosmonautiku

poškození. Z této vizualizace ale není zřejmý závěr analýzy, kterou se inženýři snažili předat NASA. 1.1.[16]1.1.[15]



Obr. 1 - Ukázka vizualizace vytvořené firmou Morton Thiokol, 1.1.[16]

Meteorologická stanice předpovídala na 28. ledna 1986 velice nízké teploty. V době startu 11:38 bylo naměřeno 2,2 ˚C (36 °F). Tato teplota je tedy o 9,4 °C (17 °F) nižší než dosud nejnižší teplota při startu raketoplánu. Pokud by inženýři zvolili jinou formu prezentace dat, je možné, že by se jim podařilo přesvědčit NASA o odložení startu, dokud by nebyly příznivější podmínky počasí. Tufteho efektivnější řešení, které prezentuje stejná data tedy teploty při startu a defekty raket ukazuje **Chyba! Nenalezen zdroj odkazů.**.



Obr. 2 - ukázka efektivní vizualizace podle Eduarda Tufteho, 1.1.[16]

Toto jsou samozřejmě jen extrémní příklady, ne každá vizualizace zachraňuje nebo maří životy, ale špatné pochopení rozhodně může vést ke špatným rozhodnutím.

1.1 Vizualizace formou grafu

Mnohé problémy v matematice, informatice, ale i v jiných vědních oborech lze abstrahovat prostřednictvím grafů. Z matematického hlediska můžeme graf *G* definovat jako uspořádanou dvojici množin *(V, E)*, kde *V* je nějaká neprázdná množina a *E* je množina dvoubodových podmnožin množiny *V*. Vrcholy *V* (vertices) představují samotné objekty a množina hran *E* (edges) reprezentuje propojení mezi vrcholy. Z pohledu například mapy by tedy vrcholy byly města a hrany cesty mezi nimi. Takto je definován neváženy a neorientovaný graf. V případě, že prvky v množině hran *E* jsou uspořádány, tak se jedná o orientovaný graf. Graf může (ale nemusí) být i vážený. Vážené grafy mají definovanou ohodnocovací funkci *w*, která každé hraně $e \in E$ přiřadí číslo, zvané váha hrany, případně může existovat ohodnocovací funkce i pro vrcholy.

Bipartitní graf je takový graf, jehož vrcholy lze rozdělit na dvě disjunktní množiny tak, že žádný vrchol z první množiny není spojen hranou s vrcholem ve druhé množině.

Úplný graf je graf, ve kterém jsou každé dva vrcholy spojeny hranou. Počet jeho hran je $m = n * \frac{n-1}{2}$.

Stupeň vrcholu představuje počet vrcholů, které jsou s tímto vrcholem spojeny hranou neboli počet jeho sousedů. U orientovaných grafů se ještě rozlišují vstupní a výstupní hrany.

Cesta je posloupnost vrcholů a hran $P = \{v_0,e_1,v_1...e_n, v_n\}$, kde vrcholy v_0 až v_n jsou navzájem různé. V orientovaných grafech se rozlišuje i směr cesty. Délka cesty je počet hran, které posloupnost obsahuje.

Uzavřená cesta neboli kružnice je cesta, která má počátek i konec ve stejném vrcholu a platí tedy $v_0 = v_n$.

Komponenta je souvislá část grafu a mezi vrcholy z různých komponent neexistuje žádná cesta. 1.1.[17] 1.1.[19]

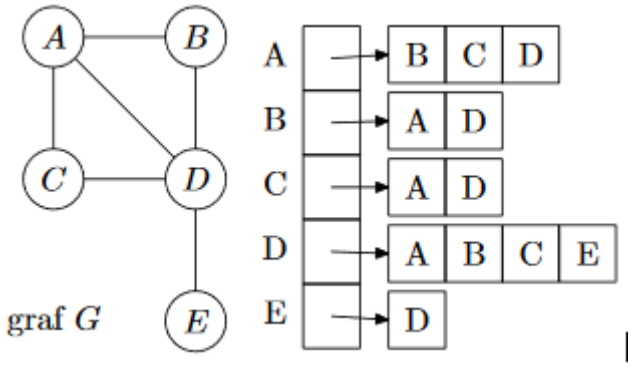
1.2 Reprezentace v počítači

Grafy mohou být v počítači reprezentovány několika způsoby. Výběr konkrétního způsobu závisí především tom, jak se bude s grafem dále pracovat. Graf může být buď orientovaný, tedy hrana v prvním vrcholu začíná a ve druhém končí, a proto platí $(u, v) \neq (v, u)$. Orientované grafy odlišujeme graficky pomocí šipek, určující směr na hranách. Příkladem pro orientovaný graf může být znázornění struktury odkazů webu. Na stránce A může být odkaz na stránku B, ale nemusí tomu tak být i naopak. Druhým typem je graf neorientovaný, tedy graf, ve kterém nezáleží pořadí vrcholů a platí $(u, v) = (v, u)$.

Jedním ze způsobu, jak reprezentovat graf v počítači je zaznamenání počtu všech vrcholů a vytvoření seznamu hran. Tento způsob není příliš vhodný pro další práci a analýzu.

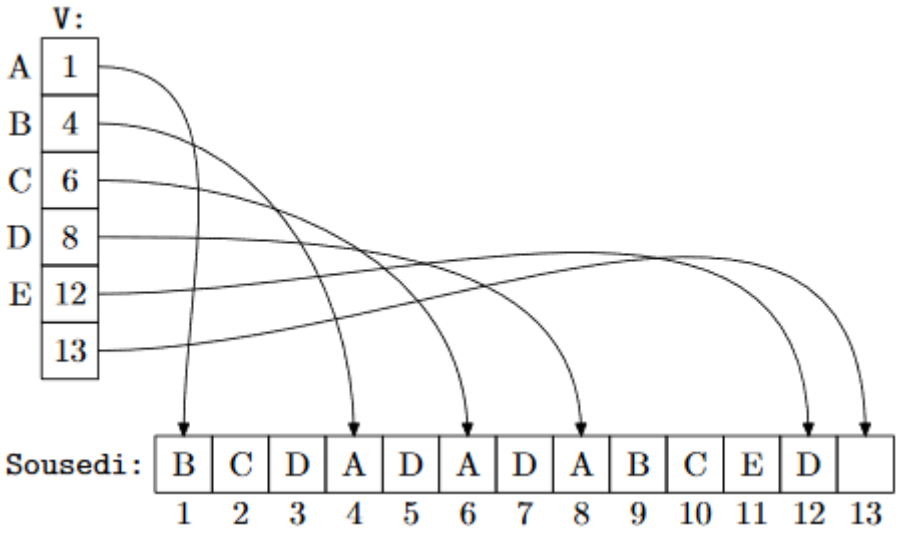
Dalším způsobem je vytvoření matice sousednosti. Jedná se o matici, která obsahuje pouze hodnoty 0 nebo 1 a má velikost $n \times n$. Matice nese informace o tom, které vrcholy spolu sousedí, tedy jestli pro každou dvojici vrcholů (u, v) existuje hrana. Matice reprezentuje pouze orientované grafy. Abychom reprezentovali i neorientovaný graf, musíme ho nejprve převést na orientovaný. Převodu docílíme tak, že každou neorientovanou hranu uv nahradíme dvojicí šipek uv a vu jdoucích proti sobě. Tento způsob se využívá velice často, a to i přes jeho vysokou paměťovou náročnost, která je n^2 (počet vrcholů). V případě jednoho miliónu vrcholů by tedy bylo potřeba 10^{12} hodnot.

Seznam sousedů je opět určen jen pro orientované grafy, takže u neorientovaného grafu musíme stejným způsobem, jako bylo popsáno v předchozím odstavci, převést na orientovaný. Pro každý vrchol *v* si vytvoříme seznam jeho sousedů tzn. vrcholy, do kterých z vrcholu *v* vede hrana.



Obr. 3 - Neorientovaný graf a jeho převedení na seznam sousedů, 1.1.[17]

Pro zvýšení efektivity jednotlivé seznamy spojíme za sebe do jednoho pole *Sousedi*. Vytvoříme ještě pole *V*, které obsahuje ukazatele na začátky seznamů v poli *Sousedi*, abychom věděli, kde končí seznam sousedů předchozího vrcholu. Seznam sousedů vrcholu *i* bude končit o jednu pozici dříve, než začíná seznam sousedů vrcholu *i+1*. Abychom mohli určit, kde končí seznam sousedů posledního vrcholu, tak rozšíříme obě pole o 1 tzn. přidáme fiktivní vrchol s prázdným seznamem vrcholů



Obr. 4- ukázka spojení seznamů sousedů a přidání fiktivního vrcholu, 1.1.[17]

Nejčastější operace s grafy jsou testování, zda jsou dva konkrétní vrcholy spojeny hranou a procházení všech sousedů konkrétního vrcholu. Matice sousednosti je

ideální pro testování dvou vrcholů pro sousednost, protože testuje v konstantním čase. V případě, že v algoritmu je potřeba projít všechny sousedy konkrétních vrcholů, je nejlepší zvolit reprezentaci seznamem sousedů. Je tedy vždy nutné vědět, co v grafu chceme dělat. 1.1.[17]1.1.[17]

1.3 Rozmístění vrcholů

Jeden z důležitých aspektů efektivní vizualizace grafů je rozmístění jeho vrcholů. Existuje mnoho druhů grafů, a tak je i mnoho rozdílných požadavků na grafové prvky a jejich uspořádání. K nejpoužívanějším druhům layoutů patří například stromové[3], planární[4], ortogonální[5] nebo tzv. silově řízené, na které se zaměřuje následující odstavec.

Silově řízené algoritmy se i ve většině českých textech vyskytují v nepřeložené podobě, tedy force-directed. V ideálním případě by vrcholy měly být rozmístěny tak, aby se hrany co nejméně křížily, a to i při vyšším počtu vrcholů a aby hrany byly stejně dlouhé a symetrické. V neposlední řadě by měl graf i dobře vypadat. Jedná se o systém, využívající principů fyziky. Mezi jeho prvky působí přitažlivé, ale i odporové síly, vrcholy si lze představit jako elektricky nabité částice, které se navzájem odpuzují a hrany jako pružiny, které tyto částice propojují.1.1.[18]1.1.[18] Existuje mnoho variací force-directed algoritmu, které se zaměřují na konkrétní situace.1.1.[20]

1.3.1 Eades

Variace force-directed algoritmu od Petera Eadese se zaměřuje především na estetickou stránku a je určen spíše pro menší grafy okolo 30 vrcholů. 1.1.[20]

Eades přirovnává vrcholy ke kovovým kruhům a hrany k pružinám. Nejprve jsou vrcholy (kovové kruhy) rozmístěny náhodně, poté začnou působit síly pružin, které přemisťují vrcholy, dokud se nedostane celý graf do rovnovážného stavu. Eadsův algoritmus používá k výpočtu sil logaritmy a vzorec vypadá následovně:

$$c1 * log \frac{d}{c_2},$$

[3] Graf je souvislý a neobsahuje kružnici (tři navzájem spojené vrcholy)
[4] Graf je zobrazen v rovině a existuje takové zobrazení, kde se nekříží žádné dvě hrany
[5] Graf je zobrazen v pravoúhlé projekci

kde proměnná d zde představuje délku pružiny a c_1 a c_2 jsou konstanty. Síly, kterými se vrcholy odpuzují, se vypočte: $\frac{c_3}{d^2}$, c_3 je další konstanta a d je vzdálenost mezi vrcholy. Algoritmus by se dal shrnout následovně:

```
1. Rozmísti vrcholy na náhodné pozice
2. Opakuj M krát
     a. - vypočítej síly pro každý vrchol
     b. - přemísti vrchol o c4 * (vypočtená síla vrcholu)
3. Vykresli graf
```

Hodnoty $c_1 = 2$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0.1$, jsou vhodné pro většinu grafů. S těmito parametry se graf dostane do rovnovážného stavu přibližně po 100 opakováních, tedy $M = 100$. 1.1.[19]1.1.[20]1.1.[20]

1.3.2 Fruchterman-Reingold

Roku 1991 bylo publikováno rozšíření předchozího algoritmu, které navrhli Fruchterman a Reingold. Vrcholy grafu jsou zde popisovány jako samostatné částice, které se navzájem přitahují i odpuzují. Přitažlivé a odpudivé síly se nyní počítají následovně:

$$f_a(d) = \frac{d^2}{k}, f_r(d) = -\frac{k^2}{d},$$

kde d představuje vzdálenost mezi dvěma vrcholy, k je optimální vzdálenost mezi vrcholy a je definována jako

$$k = C\sqrt{\frac{area}{numberOfVertices}}.$$

Tento algoritmus je podobný jako Eadsův, odlišuje se v zavedení „teploty“, která ovlivňuje výpočet nové pozice vrcholu. Teplota je pro všechny vrcholy stejná. Nejprve se vrcholy přemisťují o velkou vzdálenost (teplota je vysoká), a jak se postupně upravuje rozmístění vrcholů, jsou prováděny stále menší změny (teplota se snižuje). Tato technika je obecně nazývána simulované žíhání[6]. 1.1.[20]

6 Druh tepelného zpracování kovů, prováděné kvůli zlepšení některých jeho vlastností

```
area:= W * L; {W and L are the width and length of the frame}
G := (V, E); {the vertices are assigned random initial positions}
k := sqrt(area/|V|);
function f_a(x) := begin return x^2/k end;
function f_r(x) := begin return k^2/x end;
for i := 1 to iterations do begin
    {calculate repulsive forces}
    for v in V do begin
        {each vertex has two vectors: .pos and .disp
        v.disp := 0;
        for u in V do
            if (u ≠ v) then begin
            {δ is the difference vector between the positions of the two vertices}
                δ := v.pos − u.pos;
                v.disp := v.disp + (δ/|δ|) * f_r(|δ|)
            end
    end
    {calculate attractive forces}
    for e in E do begin
        {each edges is an ordered pair of vertices .vand.u}
        δ := e.v.pos − e.u.pos;
        e.v.disp := e.v.disp − (δ/|δ|) * f_a(|δ|);
        e.u.disp := e.u.disp + (δ/|δ|) * f_a(|δ|)
    end
    {limit max displacement to temperature t and prevent from displacement
outside frame}
    for v in V do begin
        v.pos := v.pos + (v.disp/|v.disp|) * min(v.disp, t);
        v.pos.x := min(W/2, max(−W/2, v.pos.x));
        v.pos.y := min(L/2, max(−L/2, v.pos.y))
    end
    {reduce the temperature as the layout approaches a better configuration}
    t := cool(t)
end
```

Obr. 5 - Ukázka pseudokódu Fruchterman-Reingoldova algoritmu, 1.1.[20]

Tento algoritmus rozmisťuje vrcholy jinak než Eadsův. V 1.1.[20] byl algoritmus aplikován na graf o zhruba 40 vrcholech, nicméně přijatelného výsledku lze dosáhnout i při několika stech vrcholech. 1.1.[19]1.1.[20]1.1.[20]

1.4 Možnosti zobrazení grafů

Dvourozměrné zobrazení nebo zkráceně 2D popisuje realitu pouze pomocí dvou rozměrů - výšky a šířky. Výsledná scéna nemá objem, a tak se všechny její body nacházejí v jedné rovině. Přidáním dalšího rozměru – hloubky (objemu) získáme tedy trojrozměrné zobrazení, které nám umožňuje vytvořit realisticky vypadající scénu. V dnešní době je 3D technologie stále limitována zobrazovacími zařízeními. Samozřejmě existují už 3D monitory, projektory, 3D televize, hologramy[7] dokonce se před několika

[7] Výstupní zařízení obrazu, které umožňuje zobrazit trojrozměrnou strukturu obrazu

lety na trhu objevila zařízení pro virtuální realitu, ale bohužel nejsou ještě dostatečně rozšířené a cenově přístupné pro širší veřejnost. 1.1.[22] Je tedy potřeba převést trojrozměrnou scénu na dvourozměrnou, aby ji monitor dokázal zobrazit v reálném čase. Tento proces se nazývá renderování, což je tedy tvorba reálného obrazu či scény na základě počítačového modelu.

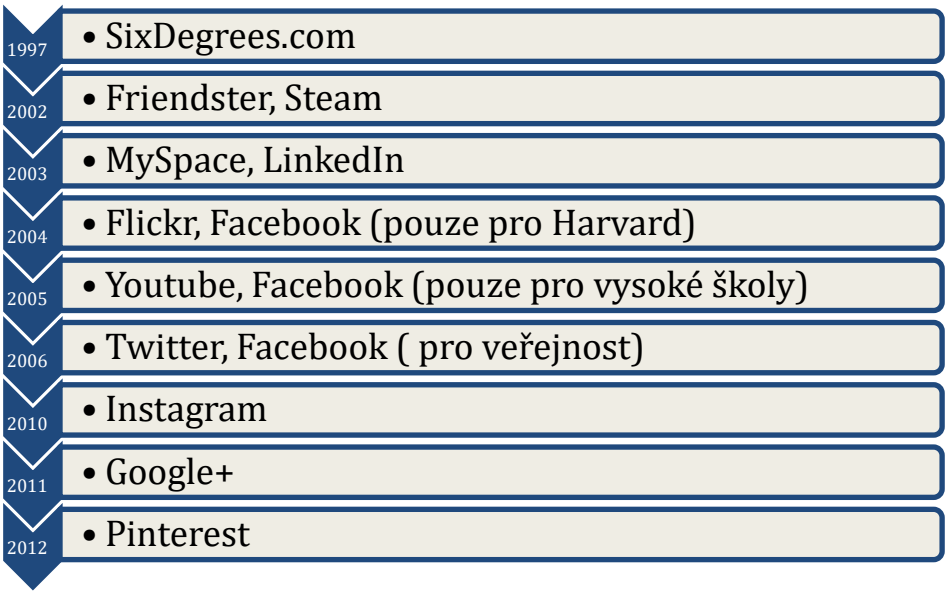
Nedá se říci, že trojrozměrná reprezentace je lepší než dvojrozměrná, nicméně při užití tří dimenzí se nám nabízí spousta dalších možností a prostoru, jak interpretovat data. Použití dalšího rozměru – hloubky, je při určitých typech grafu velice žádoucí. Dále nám umožní si graf lépe esteticky navrhnout, což pomáhá cílenému publiku si dané informace lépe zapamatovat nebo pochopit, anebo graf jen zkrátka lépe upoutá pozornost a publikum zaujme. Ne vždy je ale vhodné použít trojrozměrné zobrazení. Graf by pak mohl být až příliš složitý, nepřehledný nebo by mohl zkreslovat zobrazovaná data. Při volbě zobrazení je tedy nutné vzít v potaz to, jaká data jsou vizualizována i potřeby uživatele.

K úplnému zkoumání trojrozměrného grafu nám ve většině případech pouze jeden úhel pohledu stačit nebude. Je tedy nutné mít možnost podle potřeby natáčet, přibližovat, oddalovat scénu anebo se po scéně přímo pohybovat kamerou. K práci ve 3D zobrazení na profesionální úrovni existují i 3D myše, které umožňují pohyb po 3 osách i rotaci kolem nich. Toto speciální vstupní zařízení ocení především profesionální grafici a architekti. 1.1.[28]

2 Sociální sítě

Sociální síť lze definovat jako webovou službu, umožňující uživatelům vytvořit si svůj profil, navazovat spojení s dalšími uživateli a sdílet různé typy informací. Pro mnoho uživatelů se sociální sítě už staly nedílnou součástí jejich života. Na následujících řádcích bude uvedeno několik příkladů.

Roku 1997 byla, dle výše uvedené definice, spuštěna první sociální síť SixDegrees.com, která umožňovala uživatelům vytvořit si svůj profil, utvářet seznam přátel a komunikovat mezi sebou prostřednictvím zpráv. Tato sociální síť neměla veliký úspěch hlavně z důvodu, že na internetu v tuto dobu ještě nebylo mnoho uživatelů. 1.1.[13] Postupem času vzniklo mnoho podobných projektů. Významnějším z nich byl web Friendster, který byl spuštěn roku 2002, o rok později byly spuštěny MySpace a LinkedIn. V roce 2004 vznikla dnes nejznámější a nejpoužívanější sociální síť Facebook. Původně byl Facebook určen pouze pro studenty Harvardovy univerzity, postupem času se ale rozšířil i na další univerzity a roku 2006 byl Facebook otevřen pro veřejnost. Dnes má již přes 1,86 miliardy aktivních měsíčních uživatelů a toto číslo neustále narůstá. 1.1.[12] 1.1.[32] Roku 2006 také vyšla služba Twitter.



Obr. 6 - časová osa vzniku nejznámějších sociálních sítí, [Zdroj: autor práce]

2.1 Twitter

Twitter je velice oblíbená mikroblogovací platforma, která byla vypuštěna do světa roku 2006. Dnes má Twitter přes 315 miliónů aktivních měsíčních uživatelů, kteří každý den sdílejí okolo 500 miliónů příspěvků. Jeho zakladateli jsou programátoři Jack Dorsey, Noah Glass, Biz Stone, Evan Williams.

Twitter umožňuje sdílet krátké textové zprávy, tzv. tweety, dlouhé maximálně 140 znaků. Podle zakladatelů Twitteru bylo toto omezení zavedeno ze dvou důvodů. Jedním důvodem je, aby bylo možné tweety odesílat a číst ve formě SMS zpráv. Omezení na 140 znaků textu a dalších 20 znaků pro jméno uživatele, který tweetoval, tedy 160 znaků, což je maximální délka SMS[8] zprávy. Dalším důvodem je samotný text. Uživatel si musí více promyslet obsah tweetu, když se nemůže tolik rozepisovat. Pro sdružení tweetů do skupin nebo přiřazení k určitému tématu se používají tzv. hashtagy. Hashtag se zapisuje pomocí znaku # a libovolného textu například #visualization nebo #twitter. Není omezené množství hashtagů pro příspěvek, a tak lze reagovat na mnohá témata jedním příspěvkem.

Z Twitteru lze získat několik druhů dat. Například seznam uživatelů, kteří sledují určitého uživatele a informace o nich, veškeré tweety uživatele, tweety podle města nebo státu nebo všechny tweety podle konkrétního hashtagu a mnoho dalších druhů dat. Získání těchto dat z Twitteru a exportovat je do souboru bez placení desítek dolarů za softwarové licence a bez použití Twitter API je těžký úkol. Snad žádný software pro data mining[9] a monitoring sociálních sítí není zdarma anebo je značně omezen v základní verzi. Nejslibněji se jevil program NodeXL, který i v základní verzi dokáže importovat různá data z Facebooku, Twitteru, Flickeru anebo z Youtube, ale základní verze nepovoluje žádný export dat. Další programy vracejí jen omezené množství záznamů (~100). 1.1.[23]

V případě Twitter API se nabízí i použití Console. Po autorizaci Twitter účtem se zobrazí jednoduché rozhraní, pomocí kterého lze manipulovat s účtem (např. mazat a přidávat příspěvky, měnit nastavení apod.) nebo o něm získat určitá data (např. ID followerů, seznamy tweetů, základní informace o jiných twitter účtech atd.) ve formátu JSON.

---

[8] Služba krátkých textových zpráv (Short Message Service)
[9] Získávání potencionálně užitečných dat (nejen ze sociálních sítí)

Tento výstup slouží spíše jako ukázka dat, které lze získat z Twitter API. Pro získání komplexnějších dat lze použít některou z mnoha volně dostupných knihoven a vytvořit si vlastní Twitter aplikaci. V případě Javy lze použít například Twitter4j.

2.2 Facebook

Facebook je v současnosti bezkonkurenčně nejnavštěvovanější sociální síť. Zakladatelem a výkonným ředitelem Facebooku je Mark Zuckerberg. Facebook zaměstnává okolo 17000 lidí a má 1.86 miliardy měsíčních aktivních uživatelů (31. prosinec 2016), kteří každý den přidávají nový obsah, který lze analyzovat. 1.1.[32]

Na Facebooku se vytváří společenství ve formě "přátelství". Se svými přáteli pak lidé sdílí nejrůznější informace. Může se jednat například o fotografie domácích mazlíčků, politické názory, svatební oznámení, aktuální pocity nebo jakýkoliv jiný obsah. Uživatelé přidávají tyto příspěvky na tzv. zeď, ke které mají většinou přístup jen jejich přátelé. Pro získání takových dat přes Facebook API by bylo potřeba speciání oprávnění od každého uživatele. V roce 2007 bylo umožněno si vytvořit veřejný profil (např. pro organizace nebo zájmové skupiny). Data z těchto profilů jsou dostupná přes Facebook API i bez speciálních oprávnění. 1.1.[23]

Stejně jako v případě Twitter API je dostupná Console i pro Facebook API. Dále i mnoho knihoven pro mnohé programovací jazyky. Pro Javu je to například knihovna Facebook4J.

3 Analýza grafu

Strukturovaná data ze sociálních sítí, získaná formou dotazníků nebo přímo z databází na internetu, lze analyzovat pomocí základních grafových algoritmů.

Často zmiňovaným pojmem v analýze tohoto druhu je metrika, která představuje zobecnění vzdálenosti mezi vrcholy. Lze počítat metriky pouze pro konkrétní vrchol, skupiny vrcholů anebo pro celou sociální síť. Existuje mnoho metod analýzy grafů, v následujícím odstavci je stručný přehled těch nejvyužívanějších. 1.1.[19]1.1.[21]

3.1.1 Centralita

Centralita je jednou ze základních metod analýzy. Tato metoda nám poskytuje odpovědi na otázku, které z vrcholů v grafu jsou nejdůležitější nebo nejvlivnější. Existuje několik typů centralit a každý typ může určit důležitost vrcholu jinak. 1.1.[19]1.1.[21]

3.1.1.1 Centralita stupně vrcholu (degree centrality)

Pravděpodobně nejjednodušším typem je centralita stupně vrcholu. Jedná se tedy o počet hran napojených na konkrétní vrchol. V případě orientovaných grafů se berou v potaz jak výstupní, tak i vstupní hrany, protože oba údaje mohou být za určitých podmínek užitečné.

Z pohledu sociálních sítí lze tak uvažovat, že lidé s největším počtem propojení s jinými lidmi by měli mít největší vliv. 1.1.[19]1.1.[21]

3.1.1.2 Centralita podle vlastního vektoru (Eigenvector centralita)

Jedná se o rozšíření předchozí metody. V předchozím případě se důležitost nebo vliv určoval podle počtu sousedících vrcholů. Eigenvector centralita už ale nenahlíží na všechny sousední vrcholy jako na stejně důležité. Výsledná centralita vrcholu je tedy součet vlivu a počtu jeho sousedních vrcholů. Nejdůležitější je tedy ten vrchol, který má nejvíce sousedů anebo nejdůležitější sousední vrcholy (nebo obojí).

Zprvu přiřadíme stejnou hodnotu centrality všem vrcholům např. $x_i=1$. Tato hodnota je nevyhovující pro měření centrality, ale použijeme ji při výpočtu $x_i' = \sum A_{ij} X_j$, kde A je matice sousednosti. Po úpravách x splňuje v limitě rovnici $A_x = k_1 * x$, kde k_1 je její největší vlastní číslo (k je takové číslo, pro které platí

$A_x = k * x$, kde x je vlastní vektor). Centralita pro vrchol i se dá tedy spočítat jako $x_i = k_1^{-1} \sum(A_{ij} X_j)$.

Nastává ale potíž s orientovanými grafy, které mají vrcholy s nulovou vstupní centralitou, to řeší následující Katzova variace centrality přidáním vhodných konstant. 1.1.[19]1.1.[21]

3.1.1.3 Katz centralita

Jde o variaci Eigenvector centrality. Hlavní princip této metody spočívá v tom, že každý vrchol zvyšuje centralitu každému sousednímu vrcholu o hodnotu své vlastní centrality.

Katz řeší problém Eigenvectorové centrality s nulovým vstupním stupněm tak, že do rovnice přidal kladné konstanty α a β. Upravená rovnice vypadá následovně $x = \alpha A x + \beta 1$, kde *1* je vektor obsahující samé jedničky. Beta je také často rovna 1, protože nás nezajímají samotné hodnoty centralit, ale pouze rozdíly hodnot mezi jednotlivými vrcholy. Výsledná rovnice pak vypadá takto $x = (I - \alpha A)^{-1} * 1$. Tento vzorec vyžaduje invertování matice, což je pro velké sítě velmi výpočetně náročné. 1.1.[19]1.1.[21]

3.1.1.4 Pagerank

Katzova variace má jednu vlastnost, která je při použití určitých dat nežádoucí. V případě, že existuje vrchol s velmi vysokou (v porovnání s ostatními vrcholy) Katz centralitou, tak vrchol "rozdá" všem sousedním vrcholům mnoho bodů centrality a výsledky tak budou téměř znehodnocené. PageRank tento problém řeší tak, že vrchol zvyšuje centralitu sousedů o hodnotu, úměrnou jeho centralitě, dělenou počtem sousedů. Upravený vzorec tedy vypadá takto $x_i = \alpha \sum A_{ij}(X_j / k_j^{out})$, kde k_j^{out} je výstupní stupeň vrcholu *j*. 1.1.[19]1.1.[21]

3.1.1.5 Huby a autority

Předchozí metriky měří důležitost vrcholu podle toho, kolik a jak moc významných vrcholů s ním sousedí. Nicméně v některých sítích může vrchol mít vysokou centralitu jen proto, že ukazuje na jiné vrcholy s vysokou centralitou.

Například existuje mnoho webových stránek, které obsahují pouze odkazy na jiné užitečné stránky na konkrétní téma.

Proto je možné rozlišovat dva typy vrcholů – autority a huby. Vrcholy, které obsahují užitečné informace, jsou autority a huby nám poskytují informace o tom, kde autority najít. Autorita může být i hub a naopak. Huby a autority mohou existovat pouze u orientovaných grafů.

Tento nápad byl prvně publikován Kleinbergem, který ho využil při tvorbě algoritmu, zvaného hyperlink-induced-topic search (HITS), který vypočítává centralitu zvlášť pro huby (podle toho, na kolik autorit vrchol odkazuje) i autority (podle toho, kolik hubů na něj odkazuje). 1.1.[19]1.1.[21]

3.1.1.6 Centralita blízkosti (Closeness centrality)

Předchozí centrality neberou v potaz strukturu sítě, což může vést k tomu, že přiřadí vysokou hodnotu vrcholu, který je důležitý jen z pohledu svých sousedů, ale z pohledu celé sítě tomu už tak nemusí být. Closeness centralita měří důležitost vrcholu podle průměrné hodnoty vzdálenosti od všech ostatních vrcholů v grafu.

Průměrná vzdálenost ℓ není měřítko centrality ve stejném smyslu, jako u předchozích centralit, jelikož dává malé hodnoty centrálnějším vrcholům a vysoké hodnoty méně centrálním vrcholům, což je pravý opak od ostatních. Z tohoto důvodu je tato centralita počítána jako inverzní hodnota.

Důležitý vrchol podle této metriky má tedy dobrý přístup k informacím o ostatních vrcholech nebo může ostatní vrcholy rychleji ovlivňovat.

Průměrná vzdálenost vrcholu x_i od ostatních vrcholů se vypočítá pomocí vzorce

$$\ell_i = \frac{1}{n}\sum d_{ij},$$

kde n představuje počet vrcholů v grafu a d_{ij} je nejkratší cesta mezi vrcholy x_i a x_j. Vzorec pro samotnou centralitu je pak $C_i = 1/\ell_i$. 1.1.[19]1.1.[21]

3.1.1.7 Centralita mezilehlosti (Betweenness centrality)

Hodnota centrality pro vrchol je počet nejkratších cest mezi každými dvěma vrcholy v grafu, na kterých hodnocený vrchol leží.

Betweenness centralitu pro vrchol x_i lze spočítat pomocí vzorce $B_i = \sum(n^i_{st} / g_{st})$, kde g_{st} je počet všech nejkratších cest mezi dvojicemi vrcholů x_s a x_t, a n^i_{st} je počet nejkratších cest, které navíc vedou přes vrchol x_i. 1.1.[19]1.1.[21]

3.1.2 Shlukovací koeficient (Clustering Coefficient)

Většina sítí, a to včetně sociálních sítí, se dále dělí do různých skupin nebo komunit. Ne vždy lze tyto komunity snadno rozpoznat. Například v síti přátelství může vztah mezi dvěma sousedy vybraného člověka vypovídat něco o místu bydliště těchto osob. Pokud jsou všichni tři spojeni hranami, tak existuje šance, že bydlí blízko sebe. 1.1.[19] Toto řeší shlukovací koeficient.

Shlukovací koeficient je tedy metrika, která udává pravděpodobnost (z rozsahu 0 až 1) toho, zda dva sousední vrcholy vybraného vrcholu jsou také sousedy neboli počítá hustotu uzavřených cest o délce 3 (trojice spojených vrcholů). Lze dále rozlišovat globální a lokální shlukovací koeficient.

Globální shlukovací koeficient pracuje s počtem všech trojúhelníků v grafu. Trojúhelník představuje smyčku délky tři nebo uzavřenou cestu délky 2, tedy tři vrcholy, z nichž je každý s každým spojen hranou. Spočítají se všechny cesty délky 2 a zaznamenává se, které jsou uzavřeny a které nejsou. Výsledky podělíme a získáme shlukovací koeficient, tedy

$$C = (\frac{počet\ uzavřených\ cest\ délky\ 2}{počet\ cest\ délky\ 2}) .$$

Lokální je poměr dvojic sousedů vrcholu i, kteří jsou sami spojeni ku všem dvojicím jeho sousedů. Tedy

$$C_i = (\frac{počet\ dvojic\ vrcholů\ i, které\ josu\ spojeny}{počet\ dvojic\ sousedů\ vrcholu\ i}) .$$ 1.1.[19]1.1.[21]

3.1.3 Hierarchické shlukování (Hierarchical clustering)

Hierarchické shlukování není samostatný algoritmus, ale třída algoritmů s mnoha variacemi a alternativami. Je to technika, ve které začínáme s jednotlivými vrcholy v síti a spojujeme je do skupin na základě jejich podobnosti. V každém kroku je spočítána podobnost všech dvojic zatím vzniklých skupin a nejpodobnější skupiny jsou pak spojeny do jedné.

Existuje mnoho možností, na základě kterých lze určovat podobnost. To dodává metodě jistou flexibilitu a může tak být přizpůsobena na konkrétní problémy. Míra podobnosti porovnává pouze dvojice vrcholů a lze ji vytvořit třemi způsoby:

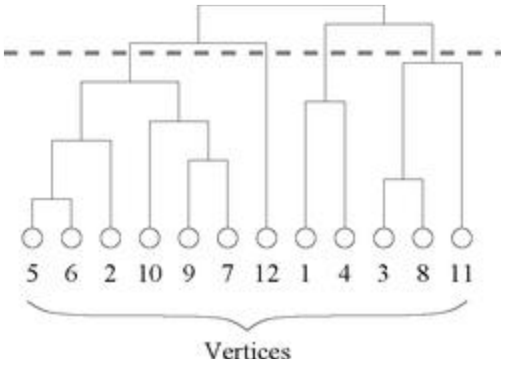
1. jednotlivým propojením (single-linkage) - podobnost mezi dvěma skupinami je definována jako podobnost dvojice jejich nejpodobnějších vrcholů
2. úplným propojením (complete-linkage) - podobnost dvou skupin je rovna podobnosti dvojice nejméně podobných vrcholů
3. průměrným propojením (average-linkage) - podobnost dvou skupin je rovna průměrné podobnosti všech dvojic vrcholů

Nejvhodnější metoda ve většině případech je průměrné propojení, protože bere v potaz celou skupinu, a nejen extrémy jako ve dvou zbylých případech. Metodu hierarchického shlukování lze shrnout takto:

1. Vyber způsob určování podobnosti a urči ji pro všechny dvojice vrcholů
2. Přiřaď každý vrchol do vlastní skupiny (každý vrchol vlastní skupinu).
3. Najdi dvě skupiny s největší podobností a spoj je do nové skupiny
4. Vypočítej podobnost mezi nově sestavenými skupinami a všemi ostatními za použití jedné z výše uvedených metod (jednotlivé propojení, úplné propojení, průměrné propojení)
5. Opakuj od bodu 3 dokud všechny vrcholy nebudou přiřazeny do jedné skupiny

Výsledek této metody je znázorněn ve formě dendrogramu[10]. Lze tak najít vnořené komunity uvnitř větších. 1.1.[19]1.1.[21]

[10] Druh diagramu, zobrazuje kroky shlukové analýzy



Obr. 7 - Ukázka dendrogramu, 1.1.[21]

3.2 Software

V následujících odstavcích bude představeno několik zástupců softwaru pro vizualizaci grafů. Nástrojů existuje celá řada, a ačkoli spousta z nich je komerčních, následující kapitola představí několik programů, které komerční nejsou, a přesto umožňují tvorbu a následnou analýzu rozsáhlejších grafů sociálních sítí. Výsledné grafy mohou obsahovat obrovské množství grafových elementů a je tedy nutné vybrat vhodný nástroj, který umožní rychle a pohodlně vytvářet, procházet anebo manipulovat s takovými grafy. Všechny představené programy jsou desktopové a umožňují tvorbu velice působivých vizualizací. Bude se jednat o programy Gephi, Cytoscape, Wandora, Tulip a Pajek.

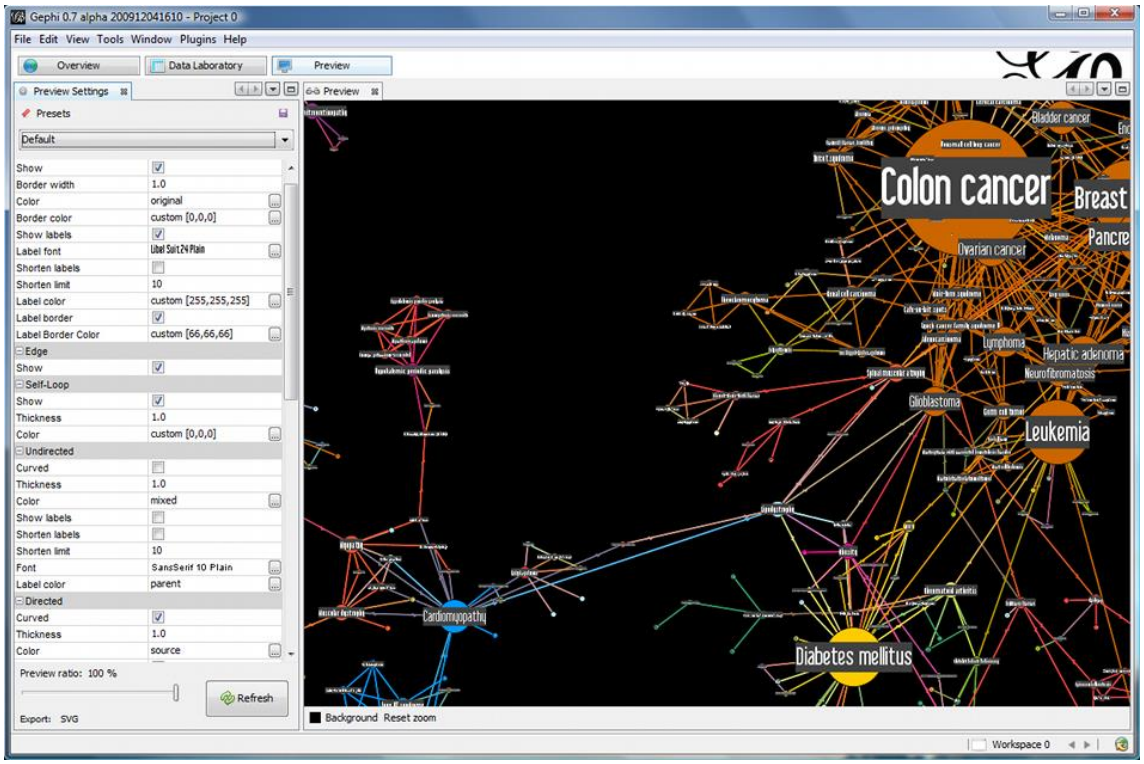
3.2.1 Gephi

Gephi je jeden z nejpoužívanějších open-source nástrojů pro vizualizaci. Získal přezdívku „Photoshop pro grafy“. Byl vyvinut studenty na UTC (University of Technology of Compiègne) ve Francii. Je navržen pro vizualizaci grafů a komplexních systémů. Aplikace byla napsána v jazyce Java na NetBeans platformě a vydána 31. července 2008. Gephi má spoustu možností, jak při tvorbě, tak i při následném procházení a analýze grafu, tyto možnosti se dají dále rozšířit různými pluginy, které díky rozsáhlé komunitě rychle přibývají. 1.1.[5]

Gephi zvládne vytvořit graf až s 1 000 000 vrcholy a hranami, které se dají přehledně spravovat, a i takto rozsáhlý graf prochází a analyzuje velice rychle díky užití OpenGL knihovny. K přehlednosti a lepšímu čtení v grafu dopomáhají barvy, změna velikosti vrcholů, vytváření skupin a oddílů, filtrování podle zadaných kritérií nebo

změna layoutu. Z vytvořeného grafu lze dále generovat statistiky např. hustota, hodnost, modularita, průměr, nejkratší cesta mezi dvěma vrcholy atd. Importovat data lze z mnoha formátů – GEXF, GDF, GML, GraphML, Pajek NET, GraphViz DOT, CSV, UCINET DL, Tulip TPL a Netdraw VNA. 1.1.[4]

Malým nedostatkem Gephi je dokumentace a nápověda funkcí a konkrétních parametrů, která mnohdy chybí úplně a někdy je jen velice stručně a nedostatečně popsána. 1.1.[5][3] 1.1.[9]



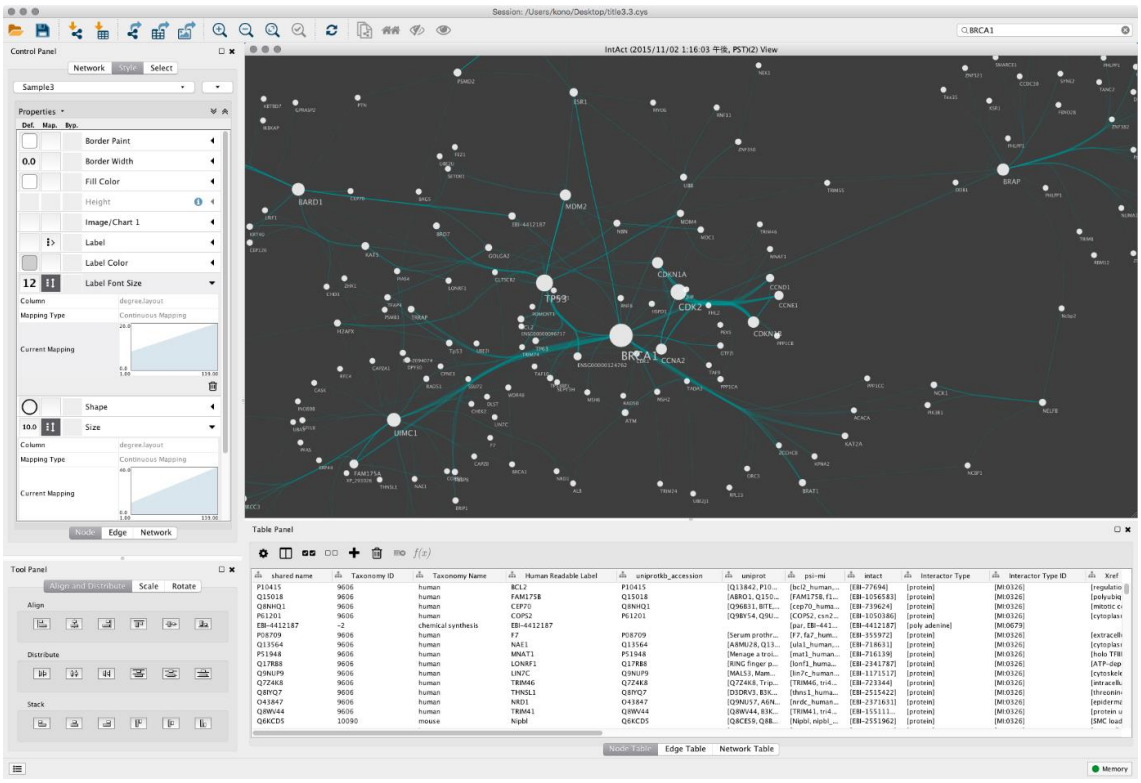
Obr. 8 - Ukázka rozhraní Gephi se vzorovým grafem nemocí, 1.1.[4]

3.2.2 Cytoscape

Cytoscape je open-source nástroj pro vizualizace komplexních grafových sítí, a to především pro biologický výzkum, ale lze jej využít i pro vizualizaci a následnou analýzu v mnoha jiných oborech. Byl vytvořen v institutu systémové biologie (Institute of Systems Biology) v Seattlu roku 2002 pod LGPL licencí. Je napsán v Javě za použití Piccolo frameworku a existuje i Javascriptová verze (Cytoscape.js) pro užití na webu. V současné době je nejaktuálnější verzí Cytoscape 3.2.1, která byla vydána v únoru 2015. Vývoj tedy stále probíhá, a to především v podobě nových pluginů, které se zaobírají různými doménami, jako například vizualizace sociálních sítí a webových portálů, ale hlavně se jedná o pluginy pro systémovou biologii. 1.1.[6] [4]

Cytoscape je podobný projekt jako Gephi, má podobné rozhraní i většinu funkcí jako například nabídka z mnoha rozvržení zobrazení, změna vlastností vrcholů a hran jako jsou velikost, barva, průhlednost, tvar a podobně. Od Gephi se odlišuje svým zaměřením na systémovou biologii. Lze tedy importovat z mnoha formátů, spojených s tímto odvětvím jako například BioPAX (Biological Pathways Exchange), PSI (Proteomics Standards Initiative), SBML (Systems Biology Markup Language), GO (Gene Ontology), OBO (Open Biomedical Ontologies) a také běžné formáty jako GML, XGMML, GraphML, KGML a další. 1.1.[6] 1.1.[7]

Největší výhodou Cytoscape je rozsáhlá komunita, která navrhuje a vytváří nové pluginy pro různá zaměření, a naopak jeho nevýhodou je zpětná nekompatibilita mezi pluginy jednotlivých verzí. 1.1.[9] Z tohoto důvodu stále ještě převládá počet uživatelů Cytoscape verze 2.x nad 3.x. verzí.

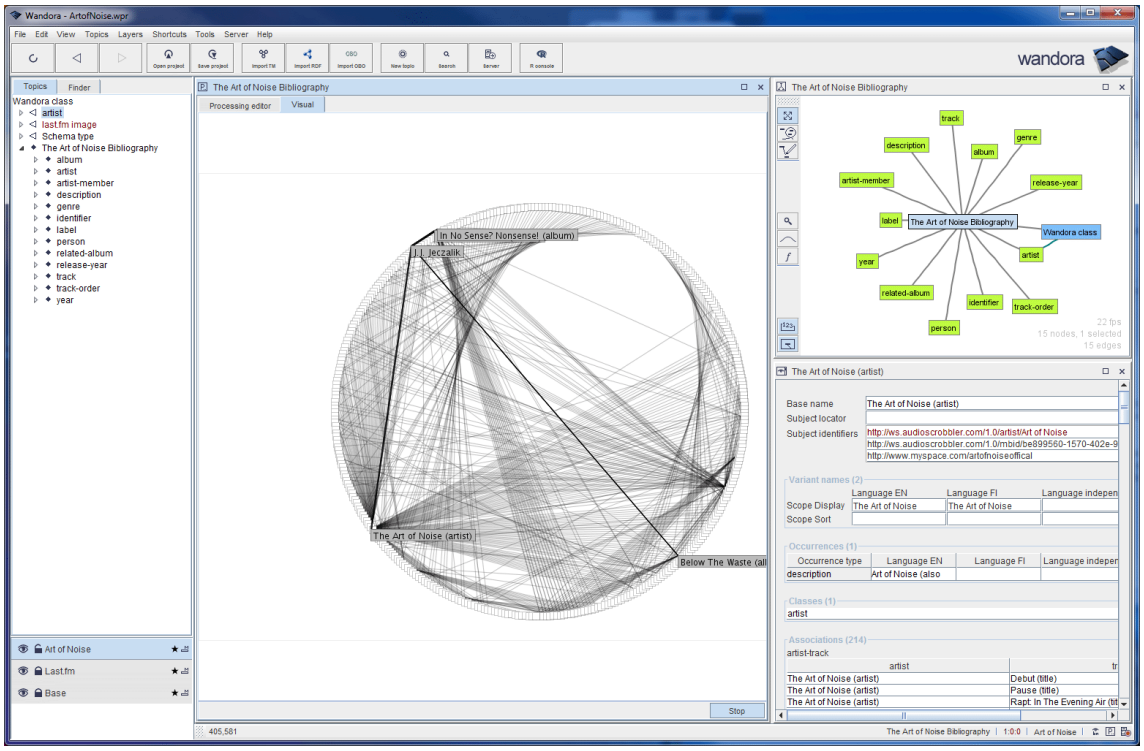


Obr. 9 – Ukázka rozhraní Cytoscape s grafem proteinové interakce, 1.1.[6]

3.2.3 Wandora

Wandora je open-source nástroj pro extrakci dat, jejich následnou správu, vizualizaci v podobě grafů a jejich publikaci. Je napsána v jazyce Java pod GNU GPL Version 3 licencí. Byla vydána roku 2002 a její vývoj stále probíhá. Poslední verze byla vydána v srpnu 2015.**Chyba! Nenalezen zdroj odkazů.**1.1.[8]1.1.[9]

Mimo již zmiňované funkce, Wandora umožňuje vytvářet mapy námětů (Topic maps). Data je možné importovat z formátů RDF (Resource Description Framework), XML, XTM, OBO (Open Biomedical Ontologies) a extraktovat data z mnoha internetových zdrojů jako například Twitter, YouTube, IMDB a podobně. Možnosti vizualizace zde nejsou tak propracované, jako u předchozích programů, ale má spíše základní funkce – přiblížení, změna perspektivy, barev, layoutu. Při vizualizaci více prvků v grafu je aplikace velmi pomalá, a tedy téměř neovladatelná.



Obr. 10 - Ukázka rozhraní Wandora, 1.1.[8]

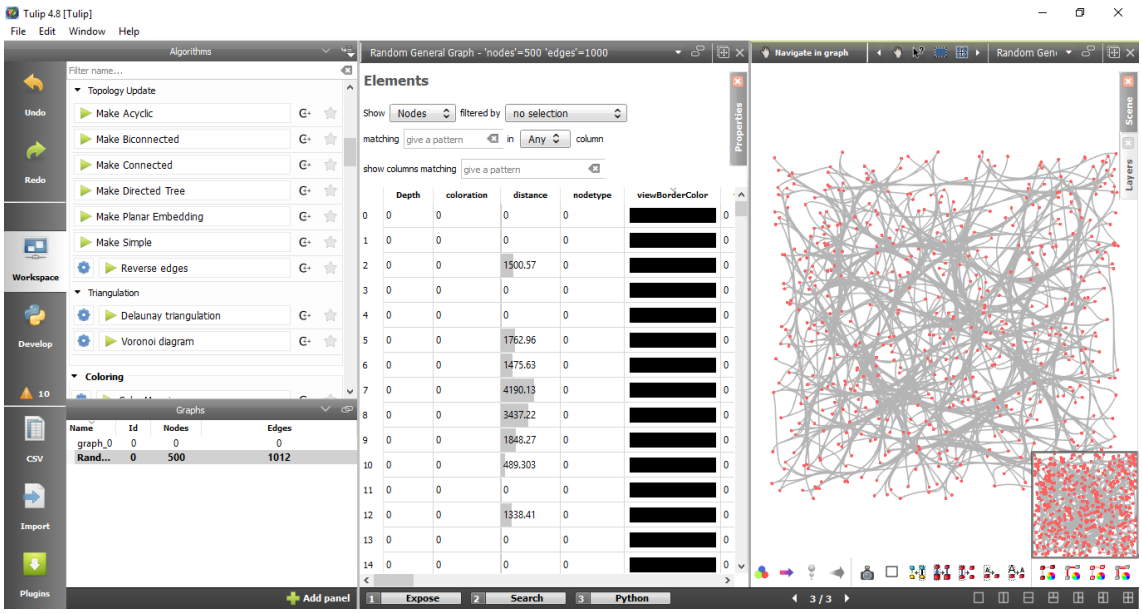
3.2.4 Tulip

Tulip je framework zaměřený na analýzu a vizualizaci grafů. Byl napsán v C++ Davidem Auberem v roce 2003 pod LGPL. Současná poslední verze je 4.8.0, která vyšla v říjnu 2015. Tulip používá knihovnu OpenGL, dokáže vytvářet a pracovat s grafy, které mají i více než 1 000 000 grafových elementů v reálném čase. 1.1.[9]1.1.[10]

Při prvním spuštění aplikace překvapí svým neobvyklým rozložením panelů funkcí, ale rozhraní je velmi rychlé a přehledné. Aplikace umožňuje upravovat velké skupiny grafových elementů najednou (velikost, barva, vzor apod.), aplikovat různé algoritmy pro tvorbu layoutu (stromové rozložení, force-directed, hierarchické rozložení

atd.), filtrovat zobrazení vrcholů a hran grafu podle zadaných parametrů. K jeho funkcím dále patří i základní analýza – počítání metriky, nejkratší cesty od jednoho vrcholu k dalšímu vrcholu atd.

Samotnou aplikaci lze dále rozšířit pomocí dalších pluginů a zmiňované funkce provádí velice rychle i při velkém množství grafových prvků.



Obr. 11 - Ukázka rozhraní Tulip, [Zdroj: autor práce]

3.2.5 Pajek

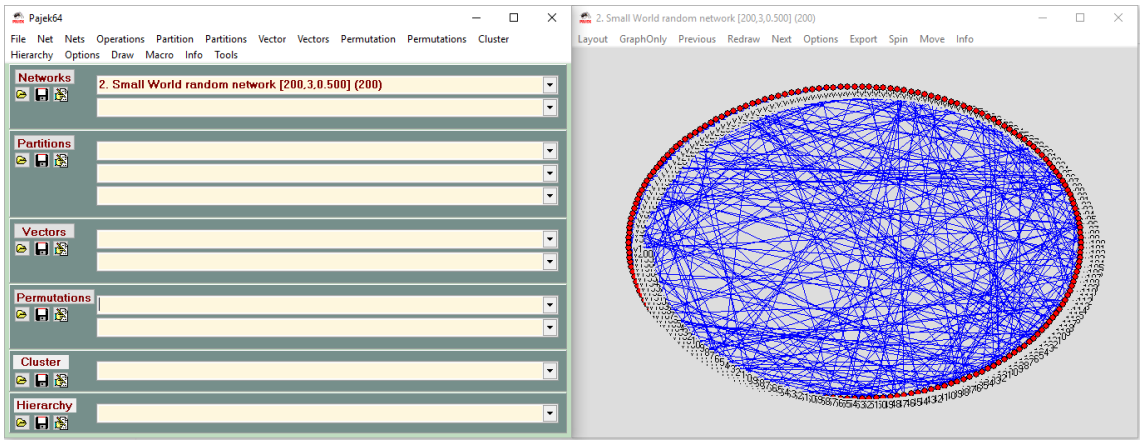
Pajek je aplikace pro analýzu a vizualizaci rozsáhlých grafových sítí. Byl vyvinut roku 1996 v jazyce Delphi (Pascal) a jeho autoři jsou Andrej Mrvar a Vladimir Batagelj. Pajek je určen pro operační systém Windows, ale za použití emulátoru[11] se dá spustit i na jiných platformách. Jedná se o closed-source aplikaci, volně dostupnou pro nekomerční použití. Současná verze je Pajek 4.07, která vyšla 1. prosince 2015, takže vývoj stále probíhá.

Vzhled aplikace je zcela jiný, než jaký je možné vidět u výše zmiňovaných nástrojů. Rozhraní je rozděleno do dvou oken. Hlavní okno je určeno pro manipulaci s daty. Pajek umí pracovat se šesti typy dat – Network (celé grafové sítě), Partition (do jaké skupiny patří konkrétní vrchol), Vector (početní hodnoty vrcholů), Cluster (podmnožina vrcholů), Permutation (pro změnu uspořádání grafu) a Hierarchy (uspořádané skupiny a vrcholy). [9] [11] Grafy se vykreslují až po stisku tlačítka

[11] Software, umožňující běh programů na jiném systému, než pro který byly programy navrhovány

"Draw" a ne okamžitě, jako u předchozích nástrojů, což má kladný vliv na rychlost programu. K procházení grafů dopomáhají například funkce změny layoutu, přiblížení, otáčení, přidávání nebo odebírání popisků k jednotlivým vrcholům, měnit barvy nebo přepínat mezi 2D a 3D zobrazením. Po analytické stránce má aplikace mnoho možností. [7]

Mezi hlavní výhody patří časté updaty, propracovaná dokumentace, rychlý chod aplikace a bohaté možnosti analýzy, naopak, co by se mohlo aplikaci vytknout, je přílišná složitost, špatná interaktivita s uživatelem, a ačkoliv vzhled aplikace nemá vliv na funkčnost, tak se design téměř nezměnil od její první verze (1996).



Obr. 12 – Ukázka rozhraní Pajek, [Zdroj: autor práce]

4 Návrh vlastního řešení

V této kapitole bude popsán postup při tvorbě aplikace pro vizualizaci dat, získaných z online kurzu Social Media Analysis. 1.1.[33]

Po nainstalování a vyzkoušení několika nástrojů, řešící stejnou problematiku vizualizace, se nejvíce žádanými vlastnostmi staly přehlednost, intuitivnost a jednoduchost aplikace. Protože pokud se s programem nepracuje pohodlně, tak ani nezáleží, jak kvalitně je naprogramován a kolika funkcemi disponuje. Pokud existuje v tomto ohledu lepší alternativa, tak si většina uživatelů zvolí právě tu. Samozřejmě čím více funkcí aplikace má, tím je těžší tohoto dosáhnout, ale to by měla vyřešit podrobná dokumentace programu, která bohužel často chybí.

4.1 Požadavky

4.1.1 Funkční požadavky

- Možnost číst data ze souboru.
- Reprezentace dat v podobě síťových grafů.
- Rozmisťovat vrcholy podle algoritmu
- Možnost zobrazit data v tabulce a vyhledávat v nich.

4.1.2 Mimofunkční požadavky

- Rychlost.
- Jednoduchost.
- Interaktivita.

4.2 Použité technologie

4.2.1 Java

Jedním z nejpoužívanějších programovacích jazyků je Java. Je to objektově orientovaný jazyk, který lze využít při tvorbě desktopových, mobilních nebo i webových aplikací. Je bezpečný, multiplatformní a výkonný. Díky jeho rozšířenosti existuje mnoho knihoven, které programátorům usnadňují práci. 1.1.[29]

4.2.2 JOGL

JOGL je zkratkou pro Java Bindings for OpenGL. Jedná se o standardizované aplikační rozhraní pro práci s 2D a 3D grafikou v Javě. JOGL bylo prvně publikováno roku 2003, je šířeno jako Open source pod licencí BSD a jeho autory jsou bývalí studenti MIT[12] – Ken Russell a Chris Kline. Později byl vyvíjen ve firmě Sun Microsystems v herním oddělení. OpenGL je dnes běžně implementováno i přímo na hardwarové úrovni, tedy na grafických kartách. 1.1.[25]

Rozhraní nabízí mnoho funkcí pro tvorbu a zpracování geometrických primitiv (body, úsečky, polygony). OpenGL používá 4 hlavní metody: Init(), Display(), Reshape() a Dispose(). Metoda init() je volána při každém startu aplikace a její pomocí se nastaví počáteční stav. Metoda display() se volá při každém překreslení scény, tedy neustále od startu až po ukončení aplikace. Pokaždé, když změníme velikost okna OpenGL aplikace, tak se zavolá metoda reshape(), která přizpůsobí velikost scény změněnému oknu aplikace. Poslední metoda dispose() signalizuje listeneru, že zdroje, které OpenGL využívá, má uvolnit. Volá se tedy při ukončování aplikace.

4.3 Struktura navržené aplikace

Aplikace je rozdělena do 7 tříd:

- App – stará se o vytvoření okna a spuštění aplikace.
- DataForm – načte data ze souboru, vytvoří vrcholy a ukáže je v tabulce, přiřadí vrcholům počáteční pozice.
- Renderer – vykresluje scénu.
- Node – reprezentuje konkrétní vrchol.
- Edge – reprezentuje hranu mezi dvěma vrcholy.
- Graph – uchovává logiku grafu tedy všechny vrcholy a hrany.
- FRLayout – obsahuje Fruchterman-Reingold algoritmus pro výpočet ideálních (aby se co nejméně křížily hrany) pozic vrcholů

[12] Massachusettský technologický institut – soukromá výzkumná universita

4.4 Import dat

Jedním z funkčních požadavků na výslednou aplikaci je import dat ze souboru. V případě, že by vstupní data měly podobu tabulky, tak bychom mohli použít například formát CSV (comma seperated values), který se vyznačuje hlavně svou jednoduchostí a přenositelností. Jelikož se jedná o graf, tak bychom museli použít dva CSV soubory, kde jeden by obsahoval vrcholy a druhý hrany.

Vhodnějším řešením je formát GDF (Geographic Data Files), který je velmi podobný CSV. GDF soubor je rozdělen do dvou sekcí, jedna pro vrcholy a druhá pro hrany. Každá sekce začíná řádkem s názvy sloupů. Data jsou zapisována po řádcích a oddělena oddělovači (většinou ","). 1.1.[31] Data tedy mohou vypadat následovně:

```
nodedef>name VARCHAR,sex VARCHAR
Joseph,male
Katrina,female
Fiora,female
edgedef>node1 VARCHAR,node2 VARCHAR
Joseph,Katrina
Katrina,Fiora
Fiora,Joseph
```

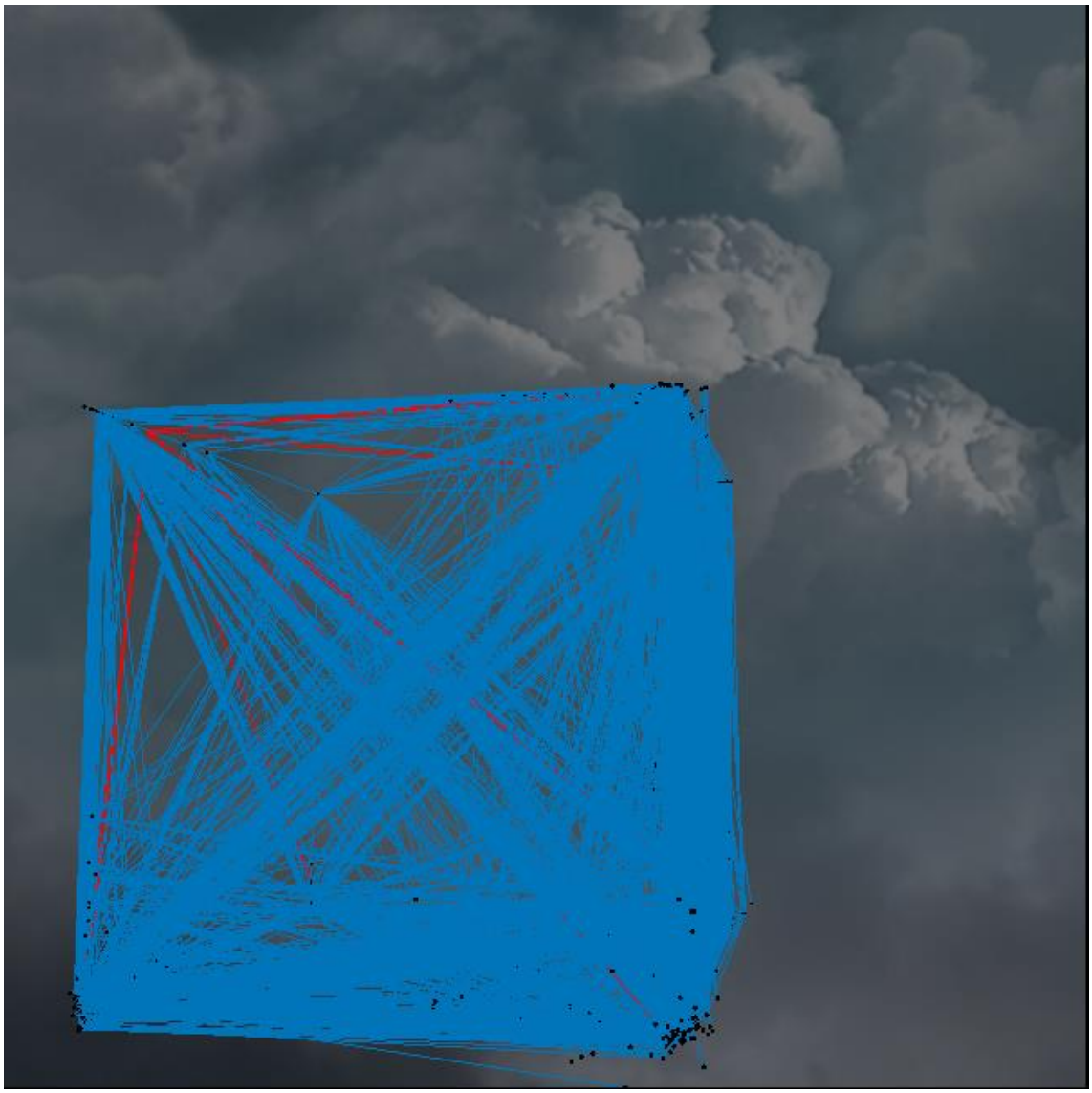
4.5 Skybox

Skybox je metoda, pomocí které lze relativně snadno vytvořit iluzi okolního světa ve 3D scéně. Je velmi často využíván při tvorbě počítačových her a simulací. Jedná se tedy o "krabici", na jejíchž šesti vnitřních stěnách (někdy pěti stěnách – podstava nemusí být vždy vhodná) jsou textury, které tvoří prostředí scény (např. obloha, příroda, pokoj apod.). Textury jsou pozicované tak, aby na sebe perfektně navazovaly a pozorovatel (kamera) je napevno umístěn do středu skyboxu, takže pozorovatel není schopen poznat, že se pohybuje v krychli.

Mimo skyboxu lze použít i skydome (někdy skysphere). Myšlenka skydomu je stejná jako u skyboxu, tedy iluze prostředí v nekonečnu, ale má několik nevýhod oproti skyboxu, takže se nepoužívá v takové míře. Mezi hlavní nevýhodu lze počítat zvýšené množství polygonů, které je potřeba vykreslit, tedy snížení výkonu.

Zvýšit efektivitu skyboxu a dodat tak scéně co nejreálnější iluzi prostředí lze dosáhnout mnoha způsoby. Například přidáním pohybující se textury mraků, přidaný světelný zdroj u textury slunce nebo přechod mezi dnem a nocí, tedy dva pomalu prolínající se skyboxy apod.

Užití propracovaného skyboxu ve vytvářené aplikaci by nebylo příliš efektivní, protože by to odpoutávalo pozornost od grafu a aplikace by tak mohla být nepřehledná. Nicméně jednoduchý skybox pomáhá uživateli se ve scéně lépe orientovat a z estetického hlediska je to rozhodně lepší varianta než pouze bílé nebo černé pozadí. 1.1.[24]

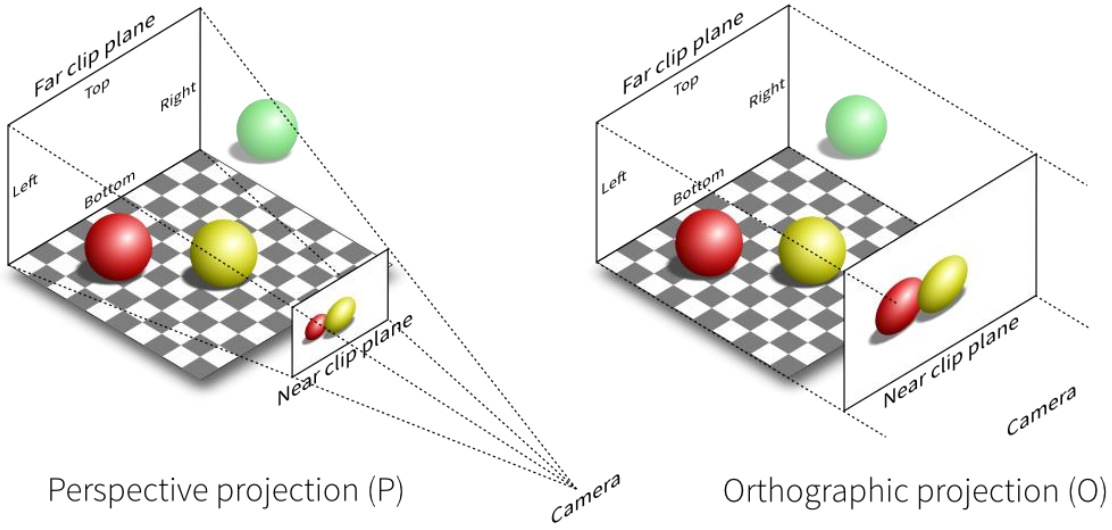


Obr. 13 - Ukázka grafu a použití skyboxu, [Zdroj: autor práce]

4.6 Kamera

Pro úplné zkoumání 3D scény je nezbytné správné nastavení kamery tedy umístění pozorovatele a jeho pohyb ve scéně. OpenGL nastavuje kameru prostřednictvím dvou matic – projection (definuje charakteristiky kamery) a modelview (nastavuje pozorovatele a grafické objekty ve scéně).

Nejprve nastavíme projekční matici. K dispozici jsou dva základní typy projekce – perspektivní a ortografická. Hlavní rozdíl je ten, že perspektivní projekce deformuje objekty a tvary podle vzdálenosti (čím vzdálenější od pozorovatele, tím je objekt menší), protože tato projekce představuje komolý jehlan, naopak ortografická projekce je vnímána jako pravoúhlý hranol, takže objekty se zvyšující se vzdáleností svou velikost nemění. Ortografická projekce najde uplatnění hlavně u aplikací typu CAD, kde tolik nezáleží na realističnosti scény, ale klíčové jsou skutečné rozměry objektů. 1.1.[30]



Obr. 14 - Porovnání perspektivní a ortogonální projekce, 1.1.[26]

Ve vytvářené aplikaci je použita perspektivní projekce, dle následujícího kódu:

```
gl.glMatrixMode(GL2.GL_PROJECTION);
gl.glLoadIdentity();
glu.gluPerspective(45, width/height, 0.1f, 10000.0f);
```

První řádek definuje matici, která bude upravena, další řádek matici nastaví na jednotkovou matici a pak následuje nastavení perspektivy. Atributy metody jsou *GLdouble fovy* (pozorovací úhel ve směru osy y), *GLdouble aspect* (poměr výška/šířka okna), *GLdouble zNear* (minimální viditelná vzdálenost), *GLdouble zFar*(maximální viditelná vzdálenost).

ModelView matice, jak lze odvodit z názvu, je spojení dvou matic – modelové (model) a pohledové (view). Pohledová matice nastavuje pozici a orientaci kamery a modelová slouží pro nastavení pozic grafických primitiv. V kódu, stejně tak jako u projekční matice, musíme nejprve určit, která matice bude upravena, a poté se použijí některé z mnoha transformačních funkcí, které OpenGL nabízí (např. funkce pro rotaci, posunutí, změna velikosti...). Matice se pak přinásobí k aktuální *ModelView* matici a získáme výslednou scénu. 1.1.[25]1.1.[26]1.1.[27]

K rozpohybování scény se většinou používá metoda *gluLookAt*, která má jako atributy 3D souřadnice. Atributy *eye* specifikují pozici kamery, *center* určuje střed scény a *up* určuje vektor, směřující nahoru. Zapisuje se následovně:

```
gluLookAt(eyeX, eyeY, eyeZ, centerX, centerY, centerZ, upX, upY, upZ);
```

Lze použít i jiný přístup, například tzv. „rozhlížení“. Pozorovateli se neustále nuluje poloha, takže se nepohybuje, zato veškeré objekty ve scéně (mimo skyboxu) ano. Využívá se zde kombinace funkcí *glTranslatef()* and *glRotate4f()*. Tento přístup je ale méně optimalizován, proto je v aplikaci použito *gluLookAt*.

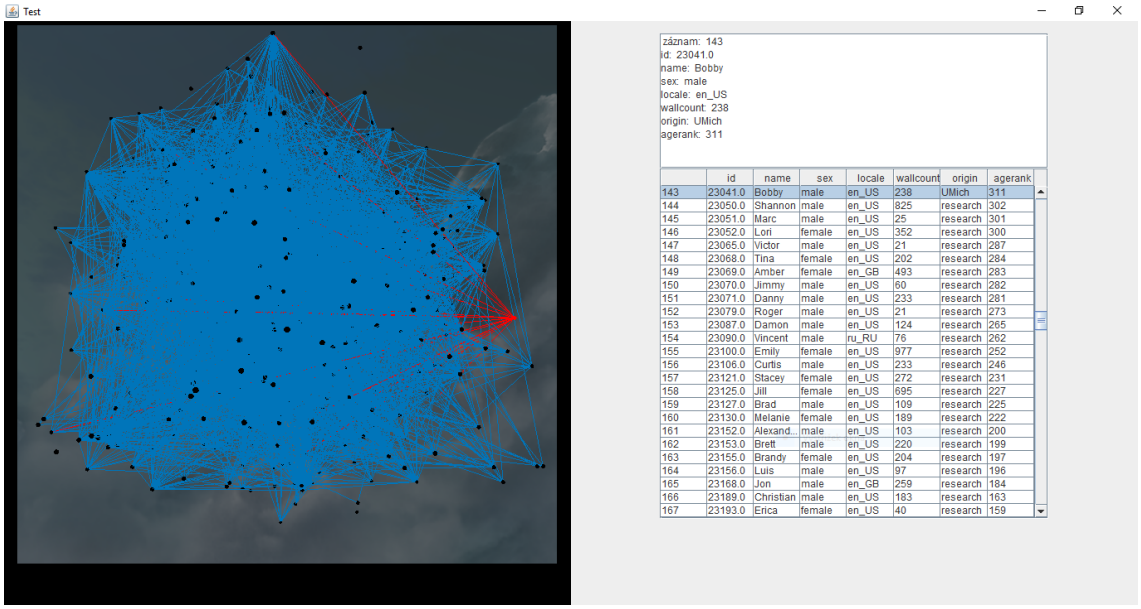
Dále se použijí Listenery, které změní tyto souřadnice např. stiskem klávesy nebo myše apod.

4.7 Layout grafu

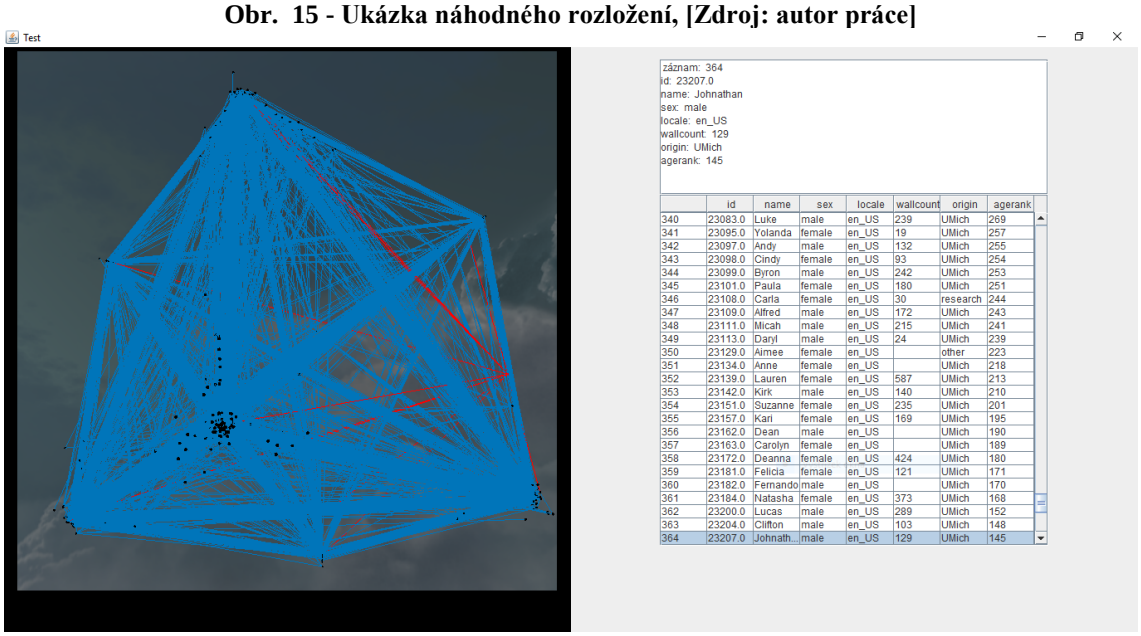
Pro rozložení vrcholů v grafu byl použit Fruchterman-Reingold algoritmus z pseudokódu[13] v 1.1.[20], který byl upraven pro 3D přidáním Z souřadnice. Před použitím algoritmu musí být vrcholům přiřazeny náhodné počáteční pozice. Algoritmus má upravit pozice vrcholům tak, aby se jejich hrany co možná nejméně křížily, nicméně

[13] Neformální jazyk pro popis algoritmů, který je nezávislý na programovacím jazyce

při vizualizaci dat ze sociálních sítí se může stát, že bude obrovský nepoměr hran a vrcholů, jako například v použitém datasetu - 388 vrcholů a 3598 hran. V tomto případě se hrany křížit budou, ale lze vidět výrazné zlepšení oproti náhodnému rozložení.



Obr. 15 - Ukázka náhodného rozložení, [Zdroj: autor práce]



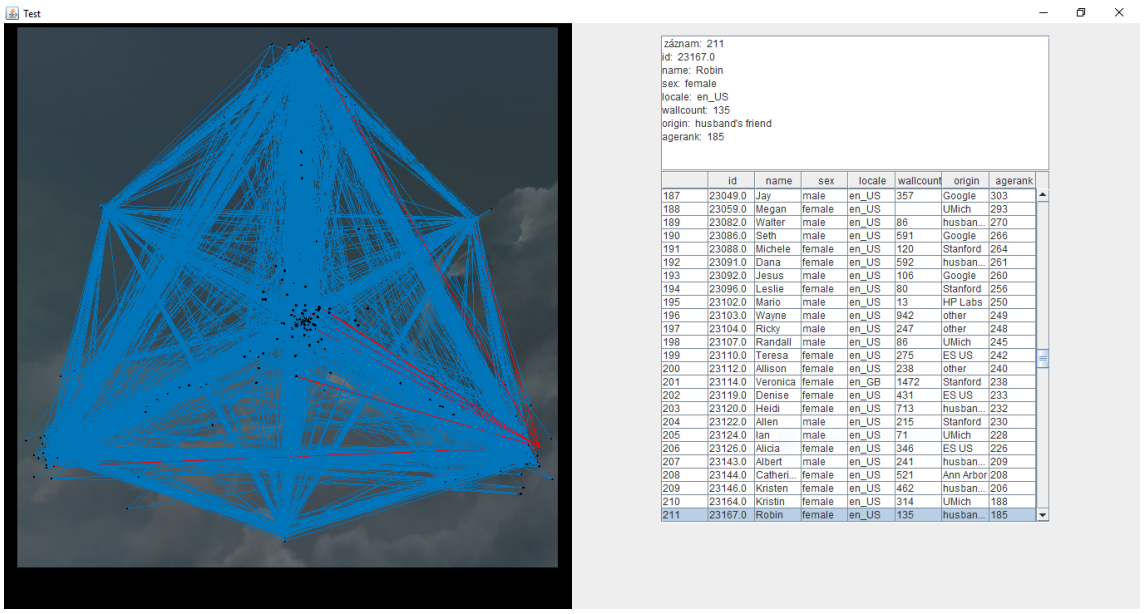
Obr. 16 - Ukázka rozložení pomocí FR. algoritmu, [Zdroj: autor práce]

4.8 Popis vytvořené aplikace

Po spuštění aplikace se zobrazí okno pro výběr souboru (GDF) s daty. Data mají pevnou strukturu a byla získána v rámci kurzu Social Media Analysis 1.1.[33].

Po načtení dat se zobrazí okno, v jehož levé části je scéna s grafem a v pravé části je tabulka s daty jednotlivých vrcholů. Ve scéně se lze pohybovat prostřednictvím kláves

W (dopředu), A (vlevo), S (dozadu), D (vpravo) a rozhlížet pomocí levého tlačítka myše. Klávesou N se zobrazí nápověda a klávesa R slouží pro reset původní pozice pozorovatele. Po kliknutí na záznam v tabulce se v textovém poli nad tabulkou vypíše v čitelnější podobě a zároveň se ve scéně barevně odliší, a to i všechny hrany, napojené na tento vrchol. Vrcholy lze vybírat i ve scéně pravým tlačítkem myši. Po vybrání vrcholu ve scéně se ukáže jeho detail v textovém poli nad tabulkou.



	id	name	sex	locale	wallcount	origin	agerank
187	23049.0	Jay	male	en_US	357	Google	303
188	23059.0	Megan	female	en_US		UMich	293
189	23082.0	Walter	male	en_US	86	husban...	270
190	23086.0	Seth	male	en_US	591	Google	266
191	23088.0	Michele	female	en_US	120	Stanford	264
192	23091.0	Dana	female	en_US	592	husban...	261
193	23092.0	Jesus	male	en_US	106	Google	260
194	23096.0	Leslie	female	en_US	80	Stanford	256
195	23102.0	Mario	male	en_US	13	HP Labs	250
196	23103.0	Wayne	male	en_US	942	other	249
197	23104.0	Ricky	male	en_US	247	other	248
198	23107.0	Randall	male	en_US	86	UMich	245
199	23110.0	Teresa	female	en_US	275	ES US	242
200	23112.0	Allison	female	en_US	238	other	240
201	23114.0	Veronica	female	en_GB	1472	Stanford	238
202	23119.0	Denise	female	en_US	431	ES US	233
203	23120.0	Heidi	female	en_US	713	husban...	232
204	23122.0	Allen	male	en_US	215	Stanford	230
205	23124.0	Ian	male	en_US	71	UMich	228
206	23126.0	Alicia	female	en_US	346	ES US	226
207	23143.0	Albert	male	en_US	241	husban...	209
208	23144.0	Catheri...	female	en_US	521	Ann Arbor	208
209	23146.0	Kristen	female	en_US	462	husban...	206
210	23164.0	Kristin	female	en_US	314	UMich	188
211	23167.0	Robin	female	en_US	135	husban...	185

Obr. 17 - Celkový pohled na aplikaci, [Zdroj: autor práce]

4.9 Shrnutí výsledků

Byla vytvořena aplikace pro jednoduchou vizualizaci dat ze sociálních sítí, která rozmisťuje vrcholy podle Fruchterman-Reingold algoritmu. Aplikace byla vytvořena dle výše uvedených požadavků.

První problém, který musel autor řešit, bylo určení typu dat, která budou vizualizována a způsob, jakým je získat. Bylo vyzkoušeno několik programů pro získávání dat ze sociální sítě Twitter, ale výsledky byly značně omezené a nevyhovující, poté za užití Twitter API a knihovny Twitter4J byla získána data relací s autorovým Twitter profilem. Získaná data by ve výsledku formovala asi 200 grafových entit, což je pro testování také nevyhovující. Nakonec byl použit dataset z kurzu 1.1.[33], který obsahuje data získaná z Facebooku. Dataset obsahuje 388 vrcholů a 3598 hran.

Dalším problémem bylo vybírání vrcholů přímo ve scéně. Výběr vrcholu ve 3D scéně není tak triviální záležitost, jako ve 2D, protože musíme pouze pomocí souřadnic

x a *y* na obrazovce získat objekty v prostoru. Nabízí se hned několik způsobů – GL_SELECT[14], Ray casting[15] nebo barevný výběr.

V aplikaci byla použita metoda barevného výběru. Základní myšlenka je taková, že každé grafické primitivum, které chceme vybírat, se vykreslí jiným odstínem barvy (běžným okem nerozeznatelný rozdíl). Pak se prostřednictvím funkce glReadPixels a zadaných souřadnic vrátí informace o vybraném pixelu, tedy i barevný odstín a lze tak snadno určit konkrétní vrchol.

[14] OpenGL funkce pro výběr, která je často špatně implementována grafickými ovladači a dnes je již označena jako zastaralá (deprecated).
[15] Algoritmus pro výběr – „vystřelí" paprsek ze zadané *x*,*y* souřadnice do prostoru, detekuje kolize paprsku s objekty a seřadí je.

5 Závěry a doporučení

S neustále rostoucí popularitou sociálních sítí se nabízí mnoho důvodů, proč z nich zachycovat a následně analyzovat data. V teoretické části bakalářské práce je tedy představena problematika vizualizace a analýzy dat ze sociálních sítí.

V praktické části byl proveden průzkum, ve kterém jsou stručně představeny vybrané aplikace pro vizualizaci. Celkem se jedná o 5 aplikací – Gephi, Cytoscape, Wandora, Tulip, Pajek. Praktická část dále obsahuje popis použitých technologií a metod při tvorbě aplikace pro vizualizaci dat.

Jelikož aplikace nabízí pouze základní možnosti vizualizace, tak se nabízí mnoho možností rozšíření do budoucna. Nejspíše nejžádanějším rozšířením bude přímé napojení na některou API konkrétní sociální sítě, protože drtivá většina programů pro získání dat ze sociálních sítí nejsou zdarma. Takovéto rozšíření by velice urychlilo a usnadnilo práci s aplikací.

Dalším možným rozšířením je implementace jiných algoritmů pro rozmisťování vrcholů a možnost přepínaní mezi nimi. Například spectrální layout, cirkulační layout anebo jiná variace force-directed layoutu.

Dále přidání výše popsaných algoritmů pro síťovou analýzu. Tedy například výpočet shlukovacího koeficientu a různých druhů centralit.

Aplikaci je také možné vylepšit z hlediska optimalizace. Díky použití Display listu (optimalizační metoda) dokáže současná aplikace bez znatelného snížení rychlosti vykreslování vizualizovat okolo 3000 vrcholů. Přepsání části pro vykreslování v jazyce GLSL by mohlo tento počet i několikanásobně navýšit.

Po teoretické stránce by se práce mohla dále rozšířit například o méně využívané algoritmy pro rozmisťování vrcholů.

6 Seznam použité literatury

[1] Jiří Sochor, Bedřich Beneš, Petr Felkel, Jiří Žára. Vizualizace. Location: FEL ČVUT Praha, 1997, p. 197

[2] James J. Thomas, Kristin A. Cook. Illuminating the Path: The research and Development Agenda for Visual Analytics. Location: National Visualization and Analytics Center, (2005), pp. 7-9

[3] Riccardo Mazza. Introduction to Information Visualization. Location: Springer-Verlag London, 2009, p. 3

[4] Mathieu Bastian., Sebastien Heymann, Mathieu Jacomy. "Gephi: an open source software for exploring and manipulating networks." Internet: gephi.org/features, [Přístup získán 2.11.2015]

[5] Kim Rees. "Data Visualization Review: Gephi, Free Graph Exploration Software. Information Aesthetics." Internet: http://infosthetics.com/archives/2010/07/review_gephi_graph_exploration_software.html, 7.10.2010, [Přístup získán 2.11.2015].

[6] Paul Shannon, Andrew Markiel, Owen Ozier, Nitin S. Baliga, Jonathan T. Wang, Daniel Ramage, Nada Amin, Benno Schwikowski, Trey Ideker. "Cytoscape: a software environment for integrated models of biomolecular interaction networks." Internet: www.cytoscape.org/what_is_cytoscape.html, 2015 [Přístup získán 11.11.2015].

[7] Mike Bergman. "Cytoscape: Hands-down Winner for Large-scale Graph Visualization." Internet: www.mkbergman.com/415/cytoscape-hands-down-winner-for-large-scale-graph-visualization/, 28.1.2008 [Přístup získán 11.11.2015].

[8] Aki Kivelä, Olli Lyytinen. "wandora the knowledge management application." Internet: wandora.org/wiki/, 6.8.2015 [Přístup získán 13.11.2015].

[9] Sorn Jarukasemratana, Tsuyoshi Murata. "Recent Large Graph Visualization Tools: A Review". Information and Media Technologies, vol.8, pp.944-960, 15.12.2015.

[10] David Auber, Patrick Mary. "Tulip, Better Visualization Through Research." Internet: tulip.labri.fr/TulipDrupal/, 2.11.2015 [Přístup získán 24.11.2015].

[11] Andrej Mrvar, Vladimir Batagelj. (2015-12-1). Pajek and Pajek-XXL - Programs for Analysis and Visualization of Very Large Networks. [online]. Available: mrvar.fdv.uni-lj.si/pajek/pajekman.pdf [Přístup získán 30.11.2015].

[12] Lucie Hušková. Novinky z Facebooku. Internet: http://newsfeed.cz/v-1-ctvrtleti-2016-stoupl-pocet-uzivatelu-facebooku-na-165-miliard/, 29.4.2016 [27.2.2017].

[13] Michal Krčmář. Sociální sítě a jejich vývoj – pohled do historie. Internet: http://objevit.cz/socialni-site-vyvoj-pohled-do-historie-t22280, 5.3.2013 [Přístup získán 27.2.2017].

[14] Michael Goldberg. 7 Great Visualizations from History. Internet: http://data-informed.com/7-great-visualizations-history, 28.2.2014 [Přístup získán 5.3.2017].

[15] Alexander Paul Hare, June Rabson Hare."J.L.Moreno". Journal of Group Psychotherapy. Místo: Sage Publications London(1989).

[16] Marek, Tomáš. Efektivní vizualizace dat se zaměřením na základní typy grafů. Brno 2014. Magisterská diplomová práce. Masarykova univerzita. Fakulta filozofická. Vedoucí práce Mgr. Jan Boček.

[17] Jakub Černý. Reprezentace grafu. Internet: http://algoritmy.eu/zga/, [Přístup získán 5.3.2017].

[18] Adam, Drda. Demonstrace algoritmů pro kreslení svazů. Olomouc 2015. Bakalářská práce. Univerzita Palackého v Olomouci. Přírodovědecká fakulta. Vedoucí práce Jan Konečný.

[19] Miloš, Kudělka. Analýza a vizualizace sociální sítě. Praha 2011. Diplomová práce. Univerzita Karlova v Praze. Matematicko-fyzikální fakulta. Vedoucí práce RNDr. František Mráz CSc., KSVI.

[20] Stephen G. Kobourov: Force-Directed Drawing Algorithms Handbook of Graph Drawing and Visualization, Chapman and Hall / CRC,2004

[21] M. E. J. Newman. Networks: An Introduction. Location: Oxford University Press Inc., New York, (2010)

[22] Dmitri, Danilov. 3D Graph Exploration. Tatru. Master thesis. University of Tatru. Faculty of mathematics and computer science. Supervisor: Ulrich Norbisrath, PhD.

[23] Aleš, Jiránek. Analýza a vizualizace vztahů nad daty ze sociálních sítí. Praha 2016. Diplomová práce. Vysoká škola ekonomická v Praze. Fakulta informatiky a statistiky. Vedoucí práce Ing. Michal Šebesta.

[24] Přispívatelé MediaWiki. "Skybox tutorial". Internet: https://sidvind.com/wiki/Skybox_tutorial, 27.1.2012 [Přístup získán 23.3.2017].

[25] Přispívatelé MediaWiki. "JOGL". Internet: http://jogamp.org/jogl/www/, [Přístup získán 2.4.2017].

[26] Nicolas P. Rougier. "Modern OpenGL". Internet: https://glumpy.github.io/modern-gl.html, [Přístup získán 2.4. 2017]

[27] Kai Ruhl. "JOGL (Java OpenGL) Tutorial". Internet: http://www.land-of-kain.de/docs/jogl/, 1.4.2009 [Přístup získán 2.4. 2017]

[28] Margaret, Rouse. 3D mouse. Internet: http://whatis.techtarget.com/definition/3D-mouse, 1.10.2014 [Přístup získán 8.4.2017]

[29] Oracle. Go Java. Internet: https://go.java/index.html, [Přístup získán 8.4.2017]

[30] David Procházka, Tomáš Koubek, Jana Andrýsková. “Programování grafických aplikací s využitím OpenGL a OpenCV” Internet: https://is.mendelu.cz/eknihovna/opory/index.pl?opora=1607, [Přístup získán 9.4.2017].

[31] Charles Iliya Krempeaux. GDF: A CSV Like Format For Graphs. Internet: http://changelog.ca/log/2013/03/09/gdf, 9.3.2013 [Přístup získán 9.4.2017].

[32] Facebook Newsroom - Company Info. Internet: http://newsroom.fb.com/company-info/, 31.12.2016 [Přístup získán 9.4.2017].

[33] The Data Science Community. Internet: https://datasciencebe.com/2014/10/08/coursera-social-media-analysis-michigan-univerity/, [Přístup získán 1.2.2017].

7 Přílohy

Příloha 1 - CD se zdrojovými kódy aplikace

Univerzita Hradec Králové
Fakulta informatiky a managementu
Akademický rok: 2015/2016

Studijní program: Aplikovaná informatika
Forma: Prezenční
Obor/komb.: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

PŘEDKLÁDÁ:	ADRESA	OSOBNÍ ČÍSLO
Čapek Martin	Zámečnická 481, Trutnov	I1301251

TÉMA ČESKY:

Vizualizace dat

TÉMA ANGLICKY:

Data visualization

VEDOUCÍ PRÁCE:

Ing. Ondřej Klapka - KIKM

ZÁSADY PRO VYPRACOVÁNÍ:

Cílem této práce je prozkoumat a otestovat možnosti vizualizace dat pomocí 3D grafů. Objasnit důvody použití, prozkoumat a porovnat vizualizace ve 2D a 3D. Dále pak představit několik zástupců softwaru na vizualizaci a na závěr prakticky vyzkoušet nějaký algoritmus.

Osnova
1. Prozkoumat problematiku a metody vizualizace dat ve formě grafů
2. Představit existující nástroje pro vizualizaci grafů
3. Navrhnout vlastní nástroj pro zvolenou oblast použití
4. Navržené řešení vhodnou technologií implementovat
5. Zhodnotit dosažené výsledky
6. Seznam zdrojů
7. Přílohy

SEZNAM DOPORUČENÉ LITERATURY:

Illuminating the Path: The research and Development Agenda for Visual Analytics, National Visualization and Analytics Center (2005)

Podpis studenta: ..... **Datum:** .....

Podpis vedoucího práce: ..... **Datum:** .....