

UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY



DIPLOMOVÁ PRÁCA

Priestorová vizualizácia grafových štruktúr

2011

Lubomír Lábaj

Priestorová vizualizácia grafových štruktúr

Evidenčné číslo: e8f356ba-25f7-4988-a73d-c4fe6bbff798

DIPLOMOVÁ PRÁCA

Bc. Ľubomír Lábaj

**UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
KATEDRA APLIKOVANEJ INFORMATIKY**

9.2.9 aplikovaná informatika

Mgr. Matej Novotný PhD.

Bratislava 2011



Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

ZADANIE ZÁVEREČNEJ PRÁCE

Meno a priezvisko študenta: Bc. Ľubomír Lábaj

Študijný program: aplikovaná informatika (Jednoodborové štúdium, magisterský II. st., denná forma)

Študijný odbor: 9.2.9. aplikovaná informatika

Typ záverečnej práce: diplomová

Jazyk záverečnej práce: slovenský

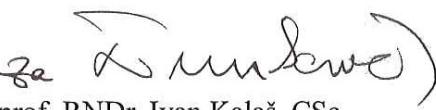
Názov: Priestorová vizualizácia grafových štruktúr

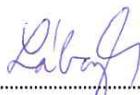
Ciel: Návrh, implementácia a demonštrácia vlastnej techniky pre interaktívnu vizualizáciu vzťahov a hierarchií. Výsledná vizualizácia je trojrozmerná, kladie popri vzťahoch dôraz aj na atribúty uzlov.

Vedúci: Mgr. Matej Novotný, PhD.

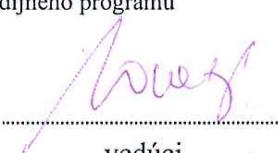
Dátum zadania: 10.11.2009

Dátum schválenia: 05.05.2011


prof. RNDr. Ivan Kalaš, CSc.
garant študijného programu



.....
študent



.....
vedúci

Čestné prehlásenie

Čestne prehlasujem, že som túto diplomovú prácu vypracoval samostatne na základe odporúčaní a pripomienok vedúceho diplomovej práce a len s pomocou literatúry a podkladových materiálov, ktoré uvádzam v priloženom zozname literatúry.

.....
Bc. Ľubomír Lábaj

Pod'akovanie

V prvom rade by som sa chcel pod'akovat' vedúcemu mojej diplomovej práce Mgr. Matejovi Novotnému PhD. za jeho čas, odborné rady a podporu, ktorú mi poskytol počas tvorby diplomovej práce.

Taktiež sa chcem pod'akovat' blízkej rodine a kruhu kamarátov, ktorí mi počas tohto obdobia vychádzali v ústrety a podporovali ma.

V neposlednom rade d'akujem, okrem iného aj za vytlačenie diplomovej práce, môjmu bratovi Ing. Ondrejovi Lábajovi.

Abstrakt

Pokrok a informatizácia spôsobuje príval obrovského množstva digitálnych informácií, vzhľadom k tomu sa zvyšuje aj dopyt po nových prístupoch v oblasti vizualizačných techník. Ako naznačuje názov práce, jej témovej je vizualizácia grafových štruktúr v 3D priestore. V prvej časti práce sa venujeme predstaveniu grafových štruktúr, ich vizualizačným technikám a jednotlivým aspektom, ktoré ovplyvňujú kvalitu výsledku v 2D aj 3D priestoroch. V druhej časti je predstavený návrh a implementácia vlastnej aplikácie pre 3D vizualizáciu grafových štruktúr, pričom využívame nadobudnuté skúsenosti a kladné aspekty techník predstavených v prvej časti. Samotná aplikácia sa vyznačuje rozšírením pružinových algoritmov na sférickú geometriu, reprezentáciou entít využitím 3D glyfov a reprezentáciu vzájomných vzťahov niekoľkými typmi 3D kriviek spolu s prístupom ich zväzovania, ktorý zvyšuje ich čitateľnosť aj pri komplexnejšej štruktúre grafu. Implementácia bola uskutočnená s využitím Borland C++ Builder 6.0 a knižnice OpenGL. V záverečnej časti práce sa nachádza zhodnotenie výsledkov práce a ich demonštrácia.

Kľúčové slová: vizualizácia, hierarchia, siet, graf, pružinový algoritmus, glyf, guľa.

Abstract

Due to progress and informatization there is a vast number of digital information which results in the increased demand for new approaches in the area of visualization techniques. As the topic suggests, the subject of this dissertation is visualization of graph structures in 3d space. In the first part of the work, the introduction to graph structures, their visualization techniques and particular aspects that influence the quality of the outcome in 2d and 3d space is provided. In the second part, design and implementation of my own application for 3d visualization of graph structures is described, applying the knowledge and positive aspects of the techniques that were introduced in the first part. The application alone is characteristic for the expansion of spring algorithms on spherical geometry, representation of entities by 3d glyphs and visualization of the relationships via several types of 3d curves together with edge bundling approach, that increases their readability even within more complex graph structure. The implementation was performed using Borland C++ Builder 6.0 and Open GL library. The last part contains the analysis and demonstration of the results achieved.

Keywords: visualization, hierarchy, network, graph, spring algorithm, glyph, sphere.

Obsah

Obsah.....	3
Úvod	5
1. Vizualizácia grafových štruktúr	6
1.1. Graf	6
1.1.1. Strom	7
1.1.2. Siet'	7
1.2. Typy vizualizácií stromov a sietí.....	8
1.2.1. Horizontálne a vertikálne stromy	8
1.2.2. Radiálne stromy.....	9
1.2.3. Stromové mapy	9
1.2.4. Radiálna konvergencia	10
1.2.5. Arc diagram.....	10
1.2.6. Maticový diagram	11
1.3. Pružinový model.....	12
1.3.1. Pružinové systémy a elektrické sily	13
1.3.2. Fruchterman – Reingold algoritmus.....	14
1.3.3. Kamada – Kawai algoritmus	15
1.4. Zväzovanie hrán	16
1.4.1. Pružinový model zväzovania hrán	17
1.4.2. Zväzovanie hrán na základe geometrie	18
1.4.3. Hierarchické zväzovanie hrán	19
1.5. Glyfy	20
1.5.1. Chernoffove tváre.....	21
1.5.2. Hviedzicový glyf	21
1.5.3. Profilový glyf	21
1.6. Vizualizácia a 3D priestor	22
1.6.1. Depth cues	22
1.6.2. 3D grafové štruktúry	23
1.6.3. Anaglyf.....	24
2. Návrh metód	25
2.1. Základná charakteristika.....	25
2.2. Sférická geometria	26
2.3. Pružinové algoritmy v neeuklidovskej geometrii	28
2.4. Antipódne body	31

2.5.	Vizualizácia entít – 3D glyf.....	31
2.6.	Vizualizácia vztahov	34
2.7.	Zväzovanie hrán	34
2.8.	Zlepšenie vnímania 3D priestoru.....	37
3.	Implementácia	38
3.1.	Vytvorenie vrcholov a ich glyfov	38
3.2.	Vytvorenie hierarchie a priradenie riadiacich vrcholov	40
3.3.	Zobrazovanie hrán	42
3.4.	Pružinový model a jeho modifikácia	43
3.5.	Ovládanie aplikácie	45
3.5.1.	Myš a klávesnica	45
3.5.2.	Hlavné nastavenia – karta „Main“.....	46
3.5.3.	Nastavenia vrcholov – karta „Nodes“	46
3.5.4.	Nastavenia hrán – karta „Edges“	47
3.5.5.	Nastavenia hierarchie – karta „Hierarchy“.....	48
3.5.6.	Ovládanie pružinového modelu – karta „Spring model“	48
4.	Demonštrácia.....	49
5.	Záver.....	59
	Zoznam použitej literatúry	60
	Prílohy	63

Úvod

Žijeme v dobe v ktorej sa informácie stávajú nesmierne dôležité, a môžeme povedať že hýbu svetom. To bola pravda aj v minulosti, ale technický pokrok až pomerne nedávno dosiahol stupeň kedy viac už nie je problém informácie uchovávať v digitálnej podobe bez toho aby sme si lámalí hlavu, kol'ko miesta zaberajú. Informatizácia a rozvoj internetu umožňuje prístup k obrovskému množstvu informácií, ktoré je možné sledovať, merat' a zaznamenávať, čo prispieva ku komplexným dátovým množinám, s mnohorozmernými dátami. Extrahovať z takého množstva dát užitočné informácie je pre človeka neľahký úkon. Veľkú pomoc v takých prípadoch predstavuje prevod do grafickej podoby v ktorej sme schopní intuitívne vnímať aj komplexné štruktúry. Z grafického vyobrazenia dokážeme ľahko rozpoznať vlastnosti ako podobnosť, symetria a identifikovať unikátne, z radu vyčnievajúce, charakteristiky. A práve to je hlavným cieľom vizualizácie dát, sprostredkovat' informácie ukryté v mohutnej štruktúre s hromadou textu a čísel, pri pohľade na ktorú človek nie je schopný vyvodiť závery podobného typu.

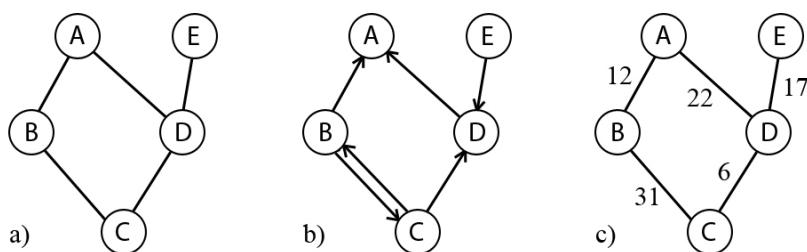
Kapitola 1

1. Vizualizácia grafových štruktúr

Grafy sa často využívajú na reprezentáciu a vizualizáciu dát, v ktorých grafová štruktúra odzrkadľuje vzájomné vzťahy entít. Hlavným dôvodom je zrozumiteľné sprostredkovanie tejto informácie, a jednoduchá identifikácia vzťahov a väzieb aj pre dátové množiny s väčším množstvom dát. Príkladom sú vizualizácie medziľudských vzťahov, dopravných spojení medzi mestami, databázových systémov, sietovitej štruktúry, máp webových stránok a podobne.

1.1. Graf

Definícia z matematiky znie, že graf pozostáva z množiny vrcholov a hrán medzi dvojicami vrcholov. Takto zadané grafy môžeme rozdeliť na dve triedy na základe toho ako definujeme hrany. Ak medzi dvoma vrcholmi existuje hrana a nerozlišujeme, aký ma smer, hovoríme o neorientovanom grafe (obr.1a). Príkladom môže byť vizualizácia v ktorej vrcholy označujú ľudí a hrana medzi nimi reprezentuje súrodenecký vzťah. Ak však medzi dvojicou vrcholov zohľadňujeme aj smer konkrétnej hrany, a môžeme po nej cestovať iba v jednom smere hovoríme o orientovanom grafe (obr.1b).



Obr. 1 – Príklady grafov, a) neorientovaný, b) orientovaný, c) váhovaný.

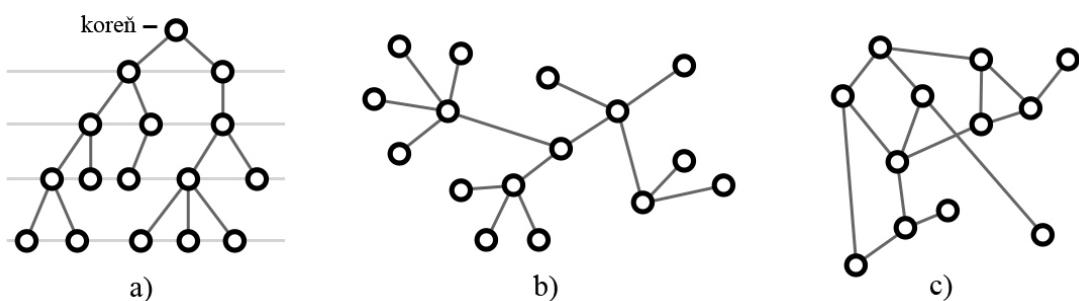
Pri tomto type grafu preto záleží, v akom poradí sú zapísané vrcholy definujúce hranu medzi nimi. Orientovaná hrana môže reprezentovať napríklad reláciu odosielateľ - príjemca.

Samozrejme existujú aj hybridné grafy kde môžu existovať aj orientované aj neorientované hrany, slučky a viacnásobné hrany rovnakého typu, nebudeme ich však konkrétnie menovať.

Hrany okrem smeru môžu mať definované aj váhy vo forme reálneho čísla (obr.1c). Ak všetky majú rovnakú hodnotu, môžeme ju zanedbať, alebo k nej pristupovať v zmysle, že každá je hodnoty 1. Za vzdialenosť dvoch vrcholov sa považuje minimálny počet hrán, ktoré je potrebné prejsť na ceste medzi nimi. Ak váhy hrán sú rôzne, vzdialenosť určuje cesta, s najmenším súčtom váh prejdených [1]. Veľkú podmnožinu grafov tvoria stromy a siete.

1.1.1. Strom

Stromy patria do skupiny acyklických grafov, čo v preklade znamená, že medzi každými dvoma vrcholmi existuje práve jedna cesta. Rovnako však môžeme rozlišovať dva typy stromov v závislosti od orientácie hrán. Orientovaný strom začína v „koreni“ často označovanom *root*. Zároveň ako jediný vrchol v strome nemá rodiča (predchodcu). Všetky ostatné vrcholy majú práve jedného rodiča (obr. 2a). Tieto stromy nazývame hierarchické a sú typické napríklad pre reprezentáciu súborového systému, kapitol v knihe, personálnej štruktúry organizácie. Často sú reprezentované pomocou dátových štruktúr, kde každý vrchol nesie informáciu o svojom rodičovi a potomkoch, takzvaná rodič–potomok (parent - child) reprezentácia. Neorientovaný strom sa nazýva kostra grafu a v princípe neobsahuje nadradené vrcholy, a teda nemá konkrétnie definovaný začiatok [2] (obr. 2b).



Obr. 2 – a) hierarchický strom, b) neorientovaný strom, c) siet'.

1.1.2. Sieť

Siete vo všeobecnosti predstavujú cyklické grafy. Pre informácie organizované v sietovitej štruktúre poskytujú prirodzenú cestu vizualizácie. Hrany opäť môžu byť orientované, alebo neorientované v závislosti od vzťahu, ktorý reprezentujú[3] (obr. 2c).

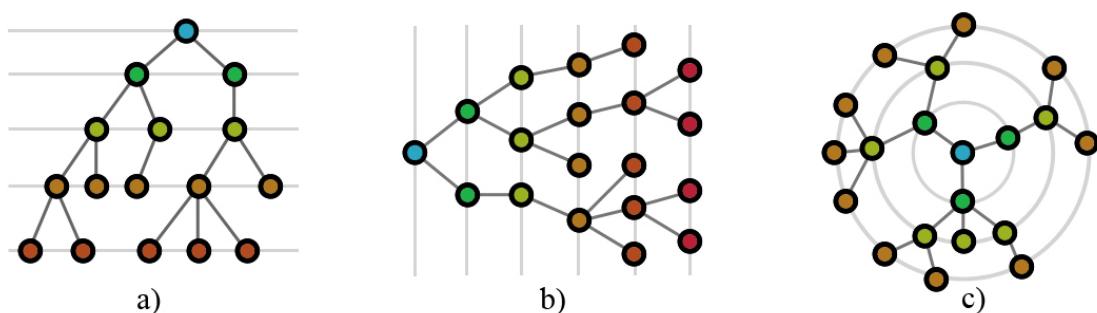
1.2. Typy vizualizácií stromov a sietí

Existuje veľa techník, ktoré slúžia k vizualizácii grafových štruktúr a v závislosti, ktorú z nich si zvolíme, bude aj získaný výsledok. Každá z techník môže sprostredkovovať a odhaliť iné skutočnosti.

Vizualizačné techniky hierarchických stromov sa vo väčšine prípadov líšia od vizualizácie kostier a sietí. Dôvodom je orientovaný vzťah rodič–potomok a poznatok o počiatočnom vrchole, koreni, z ktorého sa strom rozvetvuje. Na základe toho vieme určiť aj hĺbku vrcholov. Všetky tieto dodatočné informácie dokážeme využiť pri návrhu špecifickej techniky vizualizácie. V tejto kapitole predstavíme niekoľko typov vizualizačných techník grafov, medzi nimi aj tú, ktorá je vhodná pre väčšinu grafových štruktúr a bude predmetom skúmania v ďalších častiach práce.

1.2.1. Horizontálne a vertikálne stromy

Obe vizualizačné techniky využívajú rovnaký princíp, a slúžia na zobrazovanie hierarchických stromov. Vrcholy stromu nachádzajúce sa v rovnakej hĺbke v obidvoch prípadoch ležia v jednej rovinej linii. Rozdielom je iba smer ktorým strom expandujeme. Horizontálne stromy vykresľujeme zľava doprava, vertikálne zhora nadol, prípadne v prevrátenom smere (obr. 3a,3b). Keďže väčšinou šírka zobrazovacej plochy je väčšia ako jej výška, vertikálne stromy sú vhodnejšie pre plytké stromy, ktoré sa však rýchlo rozvetvujú, naopak horizontálne pre hlbšie s malým faktorom vetvenia. Oba spôsoby však neefektívne využívajú plochu, pretože pre každú úroveň stromu je k dispozícii rovnako veľký priestor. Vzhľadom k tomu, že so zväčšujúcou hĺbkou sa vo väčšine prípadov zvyšuje aj počet vrcholov, to nie je najlepšie riešenie.



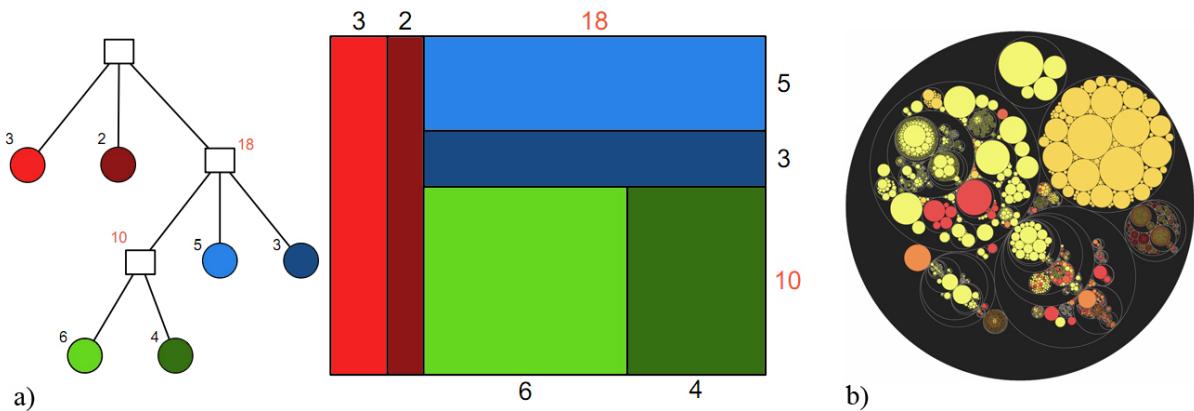
Obr. 3 – a) vertikálny strom, b) horizontálny strom, c) radiálny strom.

1.2.2. Radiálne stromy

Tento typ vizualizácie využíva rovnaký princíp ako horizontálne a vertikálne stromy. Strom sa však nevetví iba v jednom smere, ale všetkými. Vrcholy v rovnej hĺbke sú rovnako vzdialé od koreňa, čiže ležia na spoločnej kružnici. Rovnú líniu z predošej vizualizácie vystriedali sústredné kružnice. Využitie priestoru je oveľa efektívnejšie, pretože ako graf rastie do hĺbky, zväčšuje sa aj priemer kružníc (obr. 3c).

1.2.3. Stromové mapy

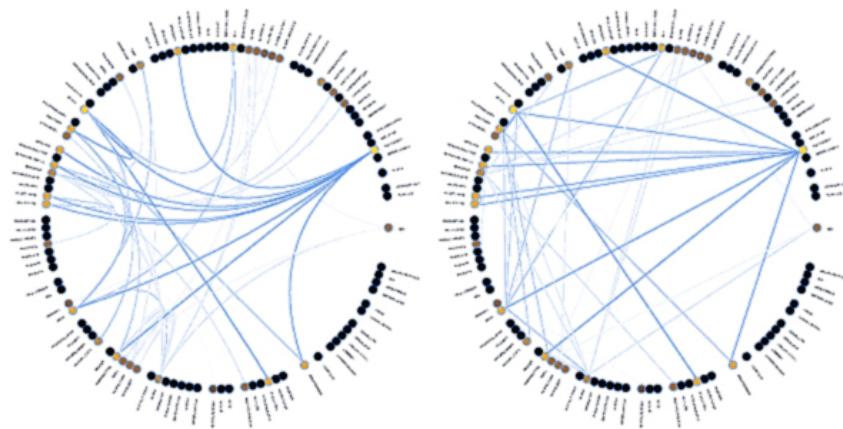
Využíva techniku známu ako „space-filling“. Doslovným prekladom je vyplňanie priestoru a jej efektivita spočíva v stopercentnom využití zobrazovacieho priestoru. Každý vrchol v strome je reprezentovaný ako určitá oblasť plochy, ktorá sa rekurzívne delí na menšie v závislosti od jeho synov, ktorí nesú informáciu o tom akú časť priestoru majú zabrať [38]. Príklad takéhoto delenia môžeme sledovať na obrázku 4a. Vrcholy grafu sú ohodnotené vzhľadom k hodnotám listov. V prvom kroku sa celá plocha, ktorá predstavuje koreň, rozdelí horizontálne na tri časti, keďže má troch synov. Následne sa rekurzívne predelia vertikálne tie vrcholy, ktoré sa vetvia ďalej. Zároveň sa zachováva pomer delenia priestoru v závislosti od hodnoty vrcholov [3][22]. Tento rekurzívny postup pritom nie je viazaný iba na obdlžnikovú geometriu, rovnako môžeme použiť napríklad kružnice (obr. 4b), alebo iné útvary, ktoré budú dodržiavať uvedené podmienky [23].



Obr. 4 – a) vytvorenie stromovej mapy, b) stromová mapa vytvorená pomocou kružníc [3].

1.2.4. Radiálna konvergencia

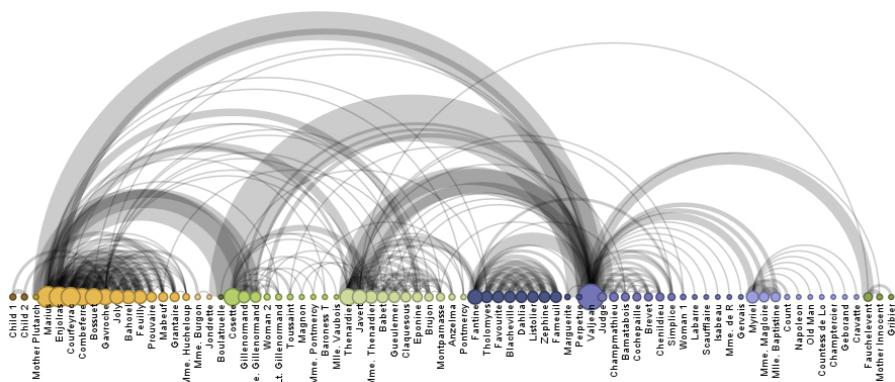
Tento typ slúži na vizualizáciu vzťahov entít, ktoré spolu vytvárajú komplexnú sieť. Príkladom môže byť vizualizácia skupiny používateľov sociálnej siete (napr. facebook.com), ktorá zobrazuje, ktorí z nich sú navzájom priatelia. Entity sú rovnomerne rozložené po obvode kružnice, a hrany smerujú jej vnútrom. Vizualizujú sa bud' pomocou rovných alebo zakrivených čiar, často s využitím čiastočnej prieľadnosti (obr. 5 [4]).



Obr. 5 – Radiálna konvergencia SchemaBall [4].

1.2.5. Arc diagram

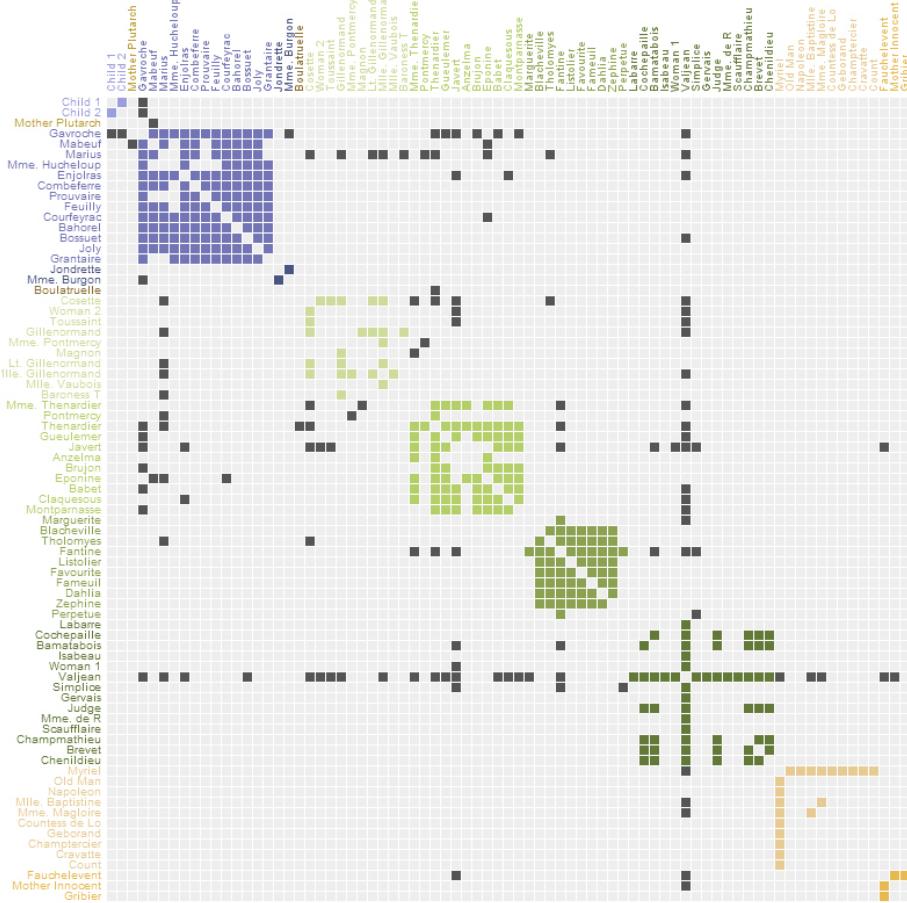
Principiálne podobná radiálnej konvergencii, využívajú však iba jednorozmerné rozmiestnenie entít pozdĺž spoločnej osi. Vzťah medzi dvoma prvkami reprezentuje kružnicový oblúk a ich vzájomná vzdialenosť zároveň určuje jeho priemer (obr. 6). Aj keď jednorozmerné rozloženie neodhaluje celkovú štruktúru siete tak dobre ako dvojrozmerné, môže pomáhať odkrývať grafové kliky a mosty. Dôležitú úlohu však zohráva usporiadanie rozmiestnenia entít [5].



Obr. 6 – Arc diagram [5].

1.2.6. Maticový diagram

Častou reprezentáciou grafu je matica susednosti, v ktorej hodnota políčka $[i,j]$ vyjadruje vzťah medzi prvkami i a j . Matica pritom dokáže uchovávať aj orientované vzťahy, pretože pre vyjadrenie vzťahu medzi prvkami i a j máme k dispozícii dve políčka $[i,j]$ a $[j,i]$. Ak je vzťah neorientovaný obidve políčka majú rovnakú hodnotu. Samotný maticový diagram je v podstate iba vykreslenie spomínanej matice do štvorcovej mriežky kde políčka vyfarbujeme v závislosti od konkrétneho vzťahu (obr. 7 [5]). Aj keď potrebné miesto pre vyobrazenie exponenciálne rastie s počtom prvkov, jeho výhodou je upustenie od reprezentácie vzťahov pomocou čiar, vďaka čomu nehrozí ich vzájomné kríženie a aj pri komplexnejších grafoch dokážeme identifikovať presne každú z nich [24]. Práve kríženie hrán spôsobuje chaos vo vizualizácii komplikovaných grafov. Na druhej strane, sledovanie cesty medzi vrcholmi je omnoho komplikovanejšie. Výsledok podobne ako pri arc diagramoch závisí od vhodného usporiadania prvkov.

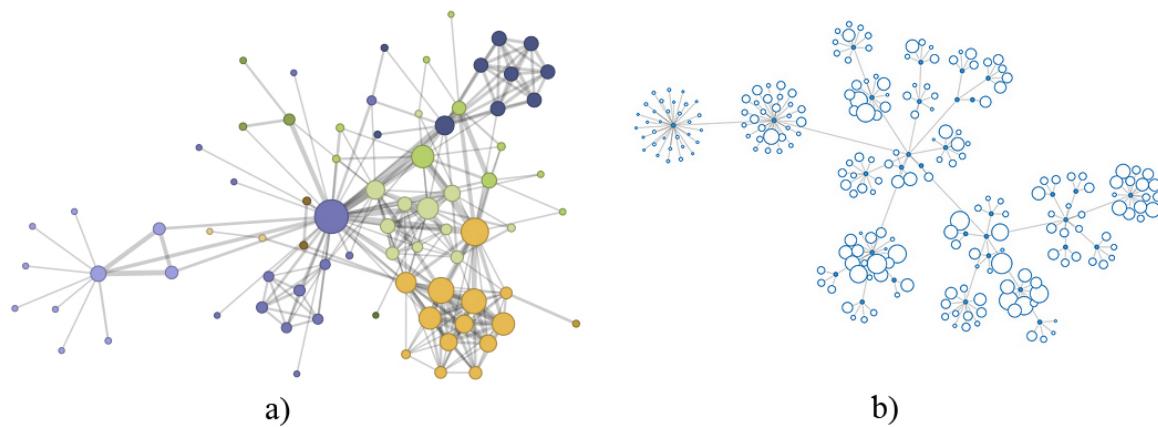


Obr. 7 – Maticový diagram [5].

1.3. Pružinový model

Pružinové, alebo silou riadené algoritmy patria do kategórie, ktorá poskytuje pravdepodobne najväčšiu flexibilitu pri zobrazovaní neorientovaných grafov. Špecifickým znakom je, že pri dosahovaní rozmiestnenia vrcholov sa využíva informácia, ktorá je implicitne zakomponovaná v každej štruktúre grafu na rozdiel od spomínaných metód, ktoré vytvárali usporiadane rozmiestnenie entít na základe vopred stanovených pravidiel [25].

Základná idea je vnímať graf ako fyzikálny systém v ktorom jednotlivé vrcholy na seba vzájomne pôsobia silami. Vrcholy grafu si môžeme predstaviť ako tuhé telesá s nenulovou hmotnosťou, a hrany ako mechanické pružiny natiahnuté medzi nimi, ktoré vrcholy pritáhujú k sebe alebo odpudzujú od seba. Odpudivé sily zabezpečujú rovnomernejšie rozmiestnenie vrcholov v priestore a zároveň zabranujú aby skončili na rovnakom mieste (obr. 8 [5]).



Obr. 8 – Pružinový algoritmus aplikovaný na: a) siet, b) neorientovaný strom [5].

Flexibilita týchto algoritmov môže mať pozitívny aj negatívny účinok na konečný výsledok. Dôvodom je fakt, že takto definovaný fyzikálny systém nemá iba jedno lokálne minimum v ktorom pôsobiace sily dosahujú rovnovážny stav. Aplikáciou rovnakého algoritmu môžeme pre jeden graf vytvoriť nekonečne veľa rozmiestnení. Možnosťou tiež je, že raz dosiahnuté rozmiestnenie sa nám už nikdy nepodarí zrekonštruovať. Všetko to závisí od komplexnosti štruktúry grafu a implementácie algoritmu. Vo všeobecnosti sú však grafy vykreslené týmto algoritmom pekné na pohľad, zrozumiteľné, majú tendenciu neobsahovať množstvo prekrížených hrán a odhalujú symetriu. Obzvlášť dobré výsledky dosahujú pri zobrazovaní neorientovaných stromov (obr. 8b). Horšie sú na tom zložité siete s veľkým množstvom hrán, v ktorých nie je možné zabezpečiť, aby sa hrany vzájomne nekrízili, čo

vedie k chaotickému výsledku aj pri stredne veľkých grafoch. Konkrétnie riešeniu tohto problému sa budeme ešte venovať. Kríženie hrán nie je jediným problémom, ktorý predurčuje použitie týchto algoritmov pre malé a stredne veľké grafy s rátovo stovkami vrcholov. Ďalším dôvodom je výpočtová náročnosť, pretože dosiahnutie rovnovážneho stavu si vyžaduje iteratívne počítanie pôsobiacich síl na každý z vrcholov a ich postupné posúvanie dovtedy, pokým energia v systéme neklesne pod veľmi malú hodnotu blízku nule. Existujú však aj pružinové algoritmy pre zobrazovanie grafov s tisícami vrcholov [26]. Vyžaduje si to však vytvoriť niekoľko vrstiev abstrakcií grafu, od najjednoduchšej až po zložitejšie s malými detailmi. Pružinový algoritmus sa najprv aplikuje na najjednoduchší model grafu, a neskôr sa využije detailnejšia úroveň abstrakcie na ktorú sa použije algoritmus už iba lokálne. Takto postupuje každou úrovňou abstrakcie, až kým nedosiahneme poslednú, ktorou je pôvodný graf [6]. Existuje niekoľko rôznych variácií pružinových modelov, priblížime si niektoré z nich a presnejšie definujeme ako prebieha samotný algoritmus a výpočet pôsobiacich síl.

1.3.1. Pružinové systémy a elektrické sily

S prvým modelom tohto typu prišiel Eades v roku 1984 [6]. Vrcholy vnímal ako železné prstence a hrany ako pružiny natiahnuté medzi nimi. Na začiatku sú vrcholy grafu náhodne rozmiestnené v priestore tak aby žiadne dva nemali rovnakú pozíciu, potom tento systém uvoľníme a čakáme, pokým nedosiahne rovnovážny stav. Sily pôsobiace na vrcholy, medzi ktorými sa nachádza pružina, sa počítajú podľa Hookovho zákona. Ukázalo sa však, že klasické lineárne pružiny sú príliš silné, ak sú vrcholy od seba veľmi vzdialené, preto boli nahradené logaritmickými pružinami, ktorých silu Eades definoval ako

$$c_1 * \log(d/c_2)$$

kde c_1 a c_2 sú konštandy a d označuje aktuálnu dĺžku pružiny. Okrem príťažlivých síl sú v systéme zakomponované aj odpudivé sily medzi vrcholmi, ktoré nie sú prepojené hranou. Tie sú ekvivalentom sily pôsobiacej medzi elektricky nabitymi časticami s rovnaký nábojom, popísané Coulombovým zákonom, v algoritme definované ako

$$c_3 / \sqrt{d}$$

c_3 je opäť vhodne zvolená konštanta a d vzdialosť medzi vrcholmi. Hlavným cieľom pri návrhu tohto algoritmu bolo získanie rozloženia, ktoré by zachovalo rovnakú dĺžku hrán a odzrkadlovalo symetriu [6][7].

1.3.2. Fruchterman – Reingold algoritmus

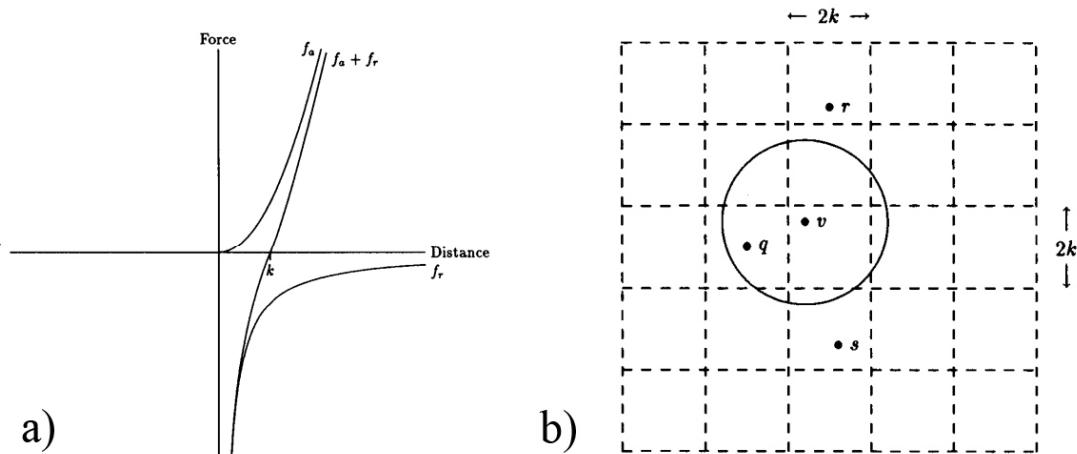
Fruchterman a Reingold rozšírili pôvodnú myšlienku Eadesovho algoritmu. Pridali podmienku, ktorá zabezpečuje aby rozmiestnenie vrcholov v priestore bolo rovnomerné. Vrcholy grafu pritom vnímajú ako atomické častice. Rovnako sa medzi nimi nachádza príťažlivá (spojené hranou) F_a a odpudivá sila (nespojené hranou) F_r , ktoré definujú ako

$$F_a(d) = d^2/k \quad F_r(d) = -k^2/d$$

Veličina d predstavuje vzdialenosť medzi vrcholmi, a k je konštanta určujúca optimálnu vzdialenosť vrcholov pre dosiahnutie rovnomerného rozloženia. Konštanta k je definovaná ako

$$k = c * \sqrt{S/n}$$

kde S predstavuje (v 2D) obsah plochy zobrazovacieho priestoru, n je počet vrcholov a c je vhodne zvolená konštanta. Veľkosť pôsobiacich síl vzhľadom na vzdialenosť vrcholov je znázornená na obrázku 9a.



Obr. 9 – a) pôsobenie síl vzhľadom k vzdialnosti vrcholov, b) štvorcová mriežka určujúca pre vrchol v množinu vrcholov ktorého ho odpudzujú [8].

Okrem zmeny definície síl, zaviedli aj takzvanú „teplotu“ systému, ktorá kontroluje maximálnu vzdialenosť o ktorú môžeme vrcholy v danej iterácii posunúť. Táto „teplota“ s časom klesá, čiže postupne ako sa rozloženie vrcholov dostáva do finálnej podoby, klesá aj vzdialenosť o ktorú môžeme vrcholy posúvať. Tento prístup je špeciálnym prípadom techniky simulovaného žiľania [27]. Je potrebné podotknúť, že oba spomenuté algoritmy v každej iterácii počítajú odpudivé sily medzi každou dvojicou vrcholov nespojených hranou,

a príťažlivé medzi dvojicami prepojenými hranou. Na redukovanie náročnosti výpočtu odpudivých sú preto Fruchterman a Reingold navrhli obmedziť vzdialenosť vrcholov, pre ktorú sa ešte zohľadňujú, pretože bolo preukázané, že z princípu nemajú veľký vplyv na výsledok.

Neskôr preto obohatili algoritmus o štvorcovú mriežku, ktorou rozdeľovali priestor. Veľkosť strany štvorca definujú ako $2*k$. Pre každý vrchol sa v tomto prípade zohľadňujú odpudivé sily iba s vrcholmi ležiacimi v susedných políčkach mriežky. Prinášalo to však skreslenie spôsobované štvorcovou oblasťou výberu vrcholov. Zaviedli preto ešte dodatočnú kruhovú oblasť s rovnaký polomerom, ako bol rozmer políčka mriežky a vrcholy ležiace v relevantných susedných políčkach sú ešte dodatočne otestované (obr. 9b) [8].

1.3.3. Kamada – Kawai algoritmus

Kamada and Kawai vo svojom prístupe stále využívajú Eadesov model s využitím pružín, ale k pôsobiacim silám v systéme pristupujú inak. Algoritmus sa snaží rozmiestniť vrcholy v priestore do konfigurácie, v ktorej euklidovská vzdialenosť $l_{i,j}$ každej dvojice vrcholov, odzrkadľuje ich teoretickú vzdialenosť $d_{i,j}$ v štruktúre grafu. Na začiatku algoritmu je preto nevyhnutné vypočítať dĺžku najkratšej cesty medzi každou dvojicou vrcholov. Pružinový model musí byť prispôsobený pre tento prístup, pretože k znižovaniu energie dochádza vtedy, ak sa minimalizuje rozdiel medzi euklidovskou a teoretickou vzdialenosťou. V systéme sa nenachádzajú separované príťažlivé a odpudivé sily, dva vrcholy sa navzájom priťahujú alebo odpuzujú, iba v závislosti od rozdielu medzi geometrickou a teoretickou vzdialenosťou. Keďže teoretická vzdialenosť dvoch vrcholov v grafe hovorí o počte hrán najkratšej cesty, je nevyhnutné zaviesť konštantu, ktorá určuje ideálnu dĺžku jednej hrany z dôvodu porovnávania s euklidovskou vzdialenosťou. Celkovú energiu v systému E môžeme definovať ako

$$E = \sum_{i=1}^{n-1} \sum_{j=i+1}^n \frac{1}{2} k_{i,j} (|p_i - p_j| - l_{i,j})^2$$

kde $l_{i,j} = L * d_{i,j}$, L predstavuje spomínanú ideálnu dĺžku pružiny medzi vrcholmi i, j a $k_{i,j} = K/d_{i,j}^2$ kde K je konštanta pružiny.

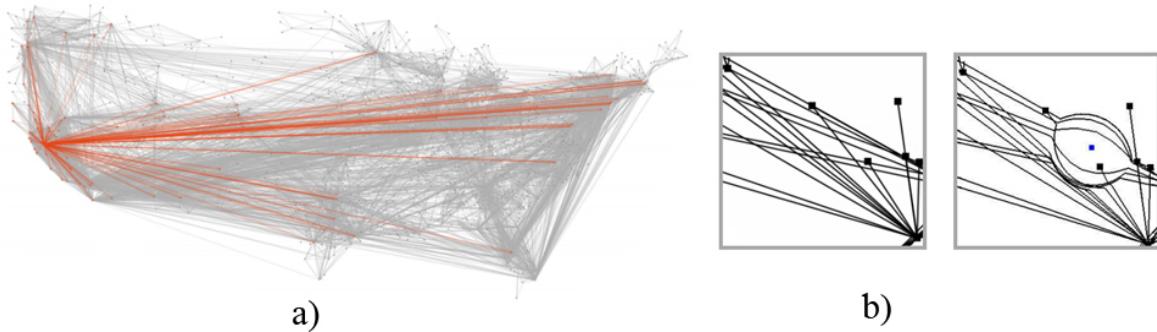
V každej iterácii sa pre každý vrchol s využitím Newton-Raphson metódy [28] vypočíta, o koľko klesne energia v systéme jeho posunutím. Vždy posunieme iba jeden vrchol, ktorý maximalizuje jej pokles [9].

1.4. Zväzovanie hrán

Vizualizácia komplexnejších grafov často trpí grafickým neporiadkom, čo v podstate znamená, že obrazovka je zahltená grafickou informáciu natoľko, až celá vizualizácia stráca svoj význam, pretože nie sme schopní rozlišovať jednotlivé prvky alebo vzťahy medzi nimi (obr. 10a [11]). Existuje niekoľko rôznych cest ako tomu môžeme zabrániť.

V zásade poznáme tri základné kategórie metód, podľa toho či sú orientované na vzhľad, priestorovú transformáciu, alebo sú iba dočasné.

Medzi techniky orientované na vzhľad patrí napríklad vzorkovanie, filtrovanie, zoskupovanie dát, ale aj použitie transparentnosti, zmena veľkosti bodov, a zvyšovanie rozlíšenia. K dočasným technikám patrí využitie animácie. Do kategórie priestorových transformácií spadá pixel plotting, space filling, dimensional reordering, topological distortion a point/line displacement. Posledná so spomenutých pod kategórií sa týka predovšetkým grafových štruktúr, preto sa jej budeme venovať bližšie [10].



Obr. 10 – a) zahltená vizualizácie siete [11], b) použitie „šošovky“ pre dočasné odklonenie hrán [12].

Algoritmy pružinových modelov, ako už bolo spomenuté sa zameriavajú na rozmiestňovanie vrcholov grafu, aby bolo dosiahnuté čo najlepšie rozloženie, ktoré by odzrkadľovalo jeho konkrétnu štruktúru a zároveň sa maximalizovala jej čitateľnosť. Táto vlastnosť ich zaraďuje do techník priestorových transformácií, konkrétnie do pod-kategórie, ktorá sa sústredí na vrcholy.

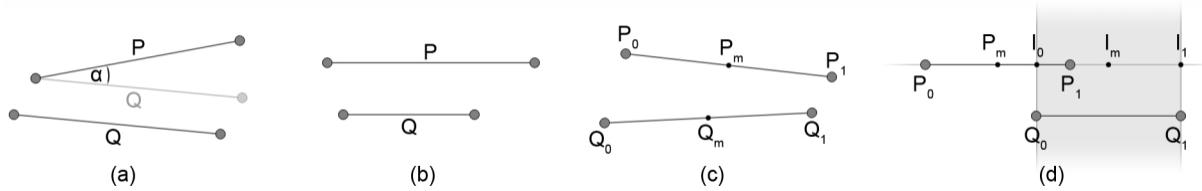
Druhou pod-kategóriou sú techniky, ktoré sa starajú o tvar hrán. Pre väčšie grafy, najmä komplexnejšie siete, sa stávajú kritické práve hrany, ktoré s narastajúcim počtom majú tendenciu veľmi rýchlo pohltiť priestor, a vzájomne sa krížia. Vzniká chaos, ktorý spôsobuje neschopnosť rozlišovať a vnímať ich kontinuitu. Na výber máme z dvoch možností, podľa toho či sa rozhodneme použiť permanentnú transformáciu alebo iba dočasnú. Dočasné transformácie využívajú analógiu šošovky pri pohľade cez ktorú, sa hrany v lokálnej časti

grafu odkláňajú z jej centra pozornosti alebo sa ich počet umelo znižuje [29][36]. Vďaka tomu môžeme odhaliť informáciu, ktorú prekrývali (obr. 10b [12]) [10]. Pozícia vrcholov pritom ostáva nemenná.

Ukážeme si niektoré techniky, ktoré využívajú permanentnú transformáciu hrán. Ich veľkou výhodou je, že výsledok je trvalý, preto nevyžadujú stálu interakciu, čím sa stávajú obzvlášť dôležité najmä pre statické vizualizácie.

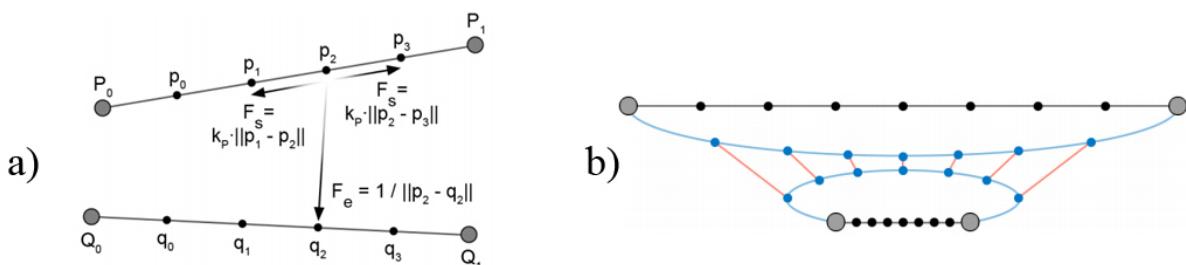
1.4.1. Pružinový model zväzovania hrán

Podobne ako pružinové algoritmy na rozmiestňovanie vrcholov, využíva analógiu pružín z fyzikálneho systému. Patrí medzi výpočtovo zdĺhavejšie a náročnejšie na implementáciu, pretože je nutné rozlíšiť podľa viacerých kritérií, na ktorých miestach sa majú pružiny aplikovať, na druhej strane však prináša graficky veľmi dobré výsledky. Pružiny sa umiestňujú iba medzi dvojice hrán, ktoré sú si „podobné“.



Obr. 11 –Kritériá podobnosti: a) uhlopriečka, b) pomer veľkostí, c) relevantnosť pozícii, d) viditeľnosť [11].

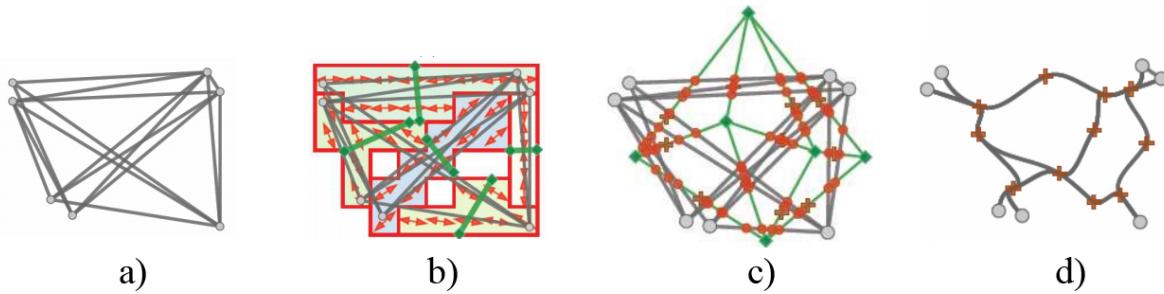
Podobnosť charakterizujeme na základe vzájomnej vzdialenosťi hrán, orientácie respektíve uhlu ktorý zvierajú, pomeru veľkostí a posledným kritériom je takzvaná viditeľnosť, ktorá zisťuje či kolmý priemet stredového bodu hrany a vzhľadom na ňu, skončí na hrane b alebo mimo nej (obr. 11). Ak sa o dvoch hranách rozhodne, že sú podobné, tak sú obe rovnomerne rozdelené n bodmi (obr. 12a), a medzi dvojice bodov s rovnakým indexom, z ktorých každý leží na vlastnej hrane je natiahnutá pružina, čo spôsobí zakrivenie hrán a ich vzájomné priblíženie (obr. 12b) [11].



Obr. 12 – a) aplikácia pružiny a výpočet pôsobiacich síl, b) demonštrácia zakrivenia hrán [11].

1.4.2. Zväzovanie hrán na základe geometrie

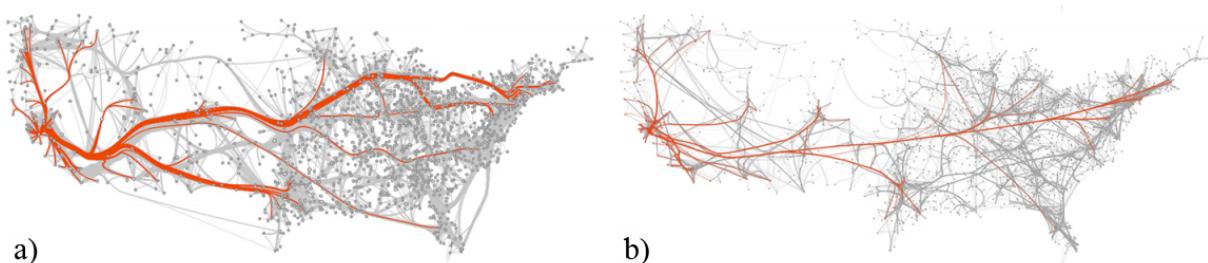
Cieľom je vytvorenie zhlukov relevantných hrán do formy, ktorá pripomína vyznačenie ciest v mapách. Ideou je využitie kontrolnej mriežky na detekciu zhlukov hrán, ktoré môžeme zviazať. Algoritmus pozostáva z troch hlavných krokov, vytvorenia kontrolnej mriežky, vytvorenia zviazaných hrán a samotnej vizualizácie.



Obr. 13 – Postup zväzovania hrán od pôvodného po výsledný graf [13].

Po vygenerovaní mriežky sa v jednotlivých políčkach vyhodnotí dominantný smer hrán, ktoré ho križujú. S využitím určitej prahovej hodnoty uhlového rozdielu, sú políčka s rovnakým dominantným smerom spojené do jednej oblasti. Pre každú samostatnú oblasť sa vygeneruje hrana, ktorá je kolmá na smer v danej oblasti, pričom je snaha zachovať deliaci pomer na dve približne rovnaké časti (obr. 13b). Premostením vytvorených hrán, s využitím Delaunayovej triangulácie, vzniká siet, ktorá určuje, ktoré hrany grafu majú tvoriť zväzok. Ak hrana siete križuje hrancu grafu, vytvorí na nej kontrolný bod (obr. 13c). Všetky kontrolné body vytvorené prienikom rovnakej hrany siete sú potom spojené do jediného bodu. Hrany grafu môžu byť pretaté viacerými hranami siete, a získať tak niekoľko kontrolných bodov. Pri vykreslovaní finálnej vizualizácie ich potom využívame ako riadiace vrcholy interpolačnej krivky medzi vrcholmi tvoriacimi hranu (obr. 13d) [13].

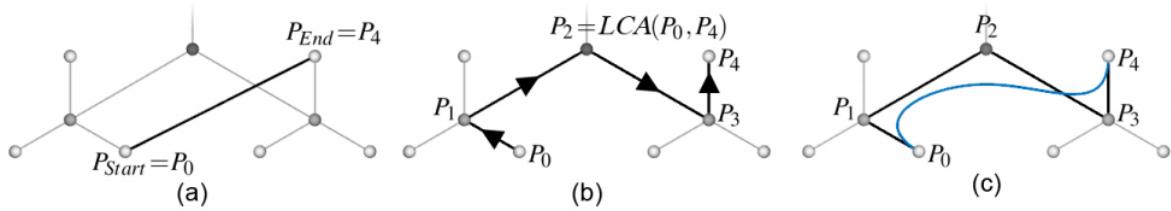
Rozdiel vo výsledku finálnej vizualizácie pri použití oboch doteraz spomínaných techník je demonštrovaný na obrázku 14 [12]. Obidve boli aplikované na graf z obrázka 10a.



Obr. 14 – Aplikované: a) zväzovanie hrán na základe geometrie, b) pružinový model zväzovania [12].

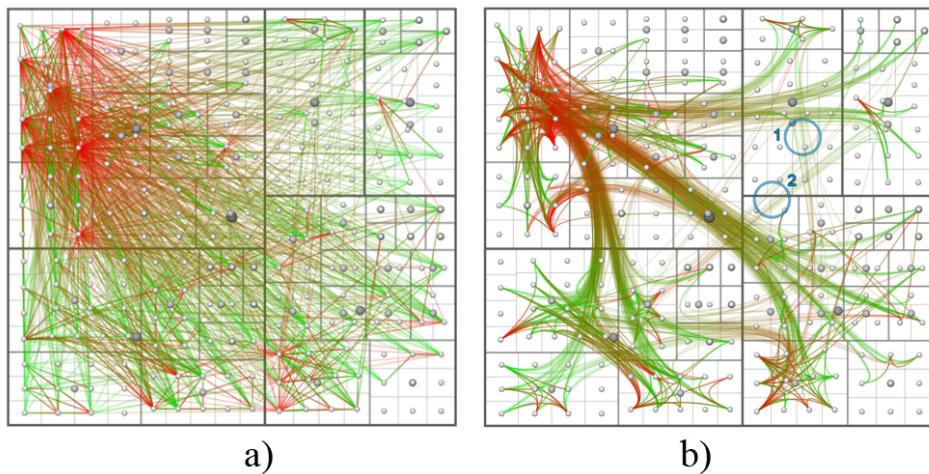
1.4.3. Hierarchické zväzovanie hrán

Tento prístup je určený pre hierarchické štruktúry, v ktorých existuje pre každú dvojicu vrcholov práve jedna cesta, ktorou vieme medzi nimi cestovať⁷. Informácia tohto typu je implicitne obsiahnutá v štruktúre a jednoducho ju dokážeme extrahovať⁸. Princípom je využiť pozície vrcholov, ktoré sa nachádzajú na ceste z medzi vrcholmi P_{start} a P_{end} , ako riadiace vrcholy krivky vykreslenej medzi nimi (obr. 15 [14]). $LCA(P_{start}, P_{end})$ označuje najbližšieho spoločného predchodcu oboch vrcholov.



Obr.15 – Postup vytvorenia riadiacich vrcholov a následné vykreslenie bázierovou krivkou [14].

Hlavným kladom je jednoduchá implementácia a využitie v rôznych už existujúcich typoch vizualizácií stromov, ktoré sa sústredia na efektivitu rozmiestnenia vrcholov, ako sú horizontálne a vertikálne stromy, radiálne stromy, balónové stromy a stromové mapy. Ďalšou z výhod je možnosť aplikácie rôznych typov kriviek, ktoré môžeme efektívne a jednoducho kontrolovať⁹, pre dosiahnutie lepšieho výsledku, bez absencie opakovaného vykonania výpočtu riadiacich bodov. V predchádzajúcich technikách je pre dosiahnutie iného výsledku potrebné opakovat¹⁰ celý postup odznova. Obrázok 16 zobrazuje aplikáciu tejto metódy na vizualizačnú techniku stromovej mapy, v ktorej za pozíciu vrcholu považujeme stred konkrétnej oblasti, ktorá ho reprezentuje [14].

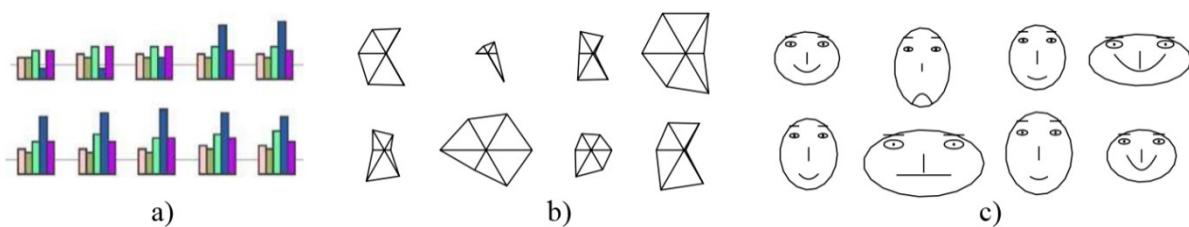


Obr. 16 – Stromová mapa a vykreslené hrany: a) pred aplikáciou, b) po aplikácii, algoritmu [14].

1.5. Glyfy

Vizualizácia grafov často nie je iba záležitosť rozmiestnenia vrcholov a vyobrazenia ich vzájomného vzťahu. Pri určitých dátových setoch je dôležité porovnávať entity medzi sebou, aby sme odhalili informáciu, ktorú hľadáme. Vtedy vstupujú do vizualizácie glyfy.

Glyfy patria medzi techniky vizualizácie viacozmerných dát. Samotný glyf je grafická reprezentácia jednej entity, kde jej atribúty ovplyvňujú grafické parametre navrhnutého glyfu. Predpokladajme, že máme databázu v ktorej entity sú ľudia a ich atribúty sú vek a pohlavie. Ako glyf môžeme použiť štvorec, kde vek bude ovplyvňovať veľkosť štvorca, a pohlavie farbu ktorou je vyfarbený. Človek dokáže v tomto prípade ľahko rozoznať, ktoré entity sú si podobné, prípadne sa vymykajú priemeru, takzvaný „outliers“. Často krát hľadáme práve informácie tohto typu. Ich hlavná sila oproti vizualizačným technikám ako scatter plot [30], alebo paralelné súradnice [31], spočíva v dobrej čitateľnosti a rýchлом vnímaní podobností aj pri viac ako trojdimenzionálnych dátach. Počtu parametrov sa samozrejme medze nekladú, veľmi komplexné glyfy však ľahko stratia svoj význam. Ak je počet atribútov príliš veľký, je vhodnejšie zvážiť výber určitého počtu z nich, ktoré majú najväčší vplyv vzhľadom na žiadaný výsledok vizualizácie. Rovnako potrebné je myslieť na správnu škálu mapovaných atribútov aby pri mapovaní hraničných hodnôt nevznikali neprimerane veľké/malé glyfy, a vzhľadom k nim nebolo potrebné upravovať veľkosť všetkých glyfov, čo môže viest' k zhoršenej schopnosti rozlišovať podobnosti, v krajných prípadoch k úplnej neschopnosti.



Obr. 17 – Glyfy: a) Profilový, b) Hviezdicový, c) Chernoffove tváre [2].

Do základnej množiny grafických atribútov na ktoré je možné mapovať konkrétné hodnoty patria pozícia (2D alebo 3D), veľkosť (dĺžka, obsah, objem), tvar, orientácia, materiál (farba, jas, transparentnosť, textúra), štýl čiar (hrúbka, vzor: napr. bodkovaná) aj dynamika (rýchlosť pohybu a jeho smer, trasenie, atď.) [2].

1.5.1. Chernoffove tváre

Patria k najznámejším typom glyfov a ich princíp je založený na vysokej citlivosti človeka vnímať rozdiely v črtách ľudskej tváre, ktorú sme si počas vývoja vybudovali. Jednotlivé hodnoty atribútov ovplyvňujú veľkosť, orientáciu a zakrivenie grafických prvkov, ktoré reprezentujú časti tváre (oči, ústa, nos, oboče atď.) (obr. 17c). Akokoľvek sa často prezentujú ako príklad použitia glyfov, v praxi sa paradoxne veľmi neuplatnili.

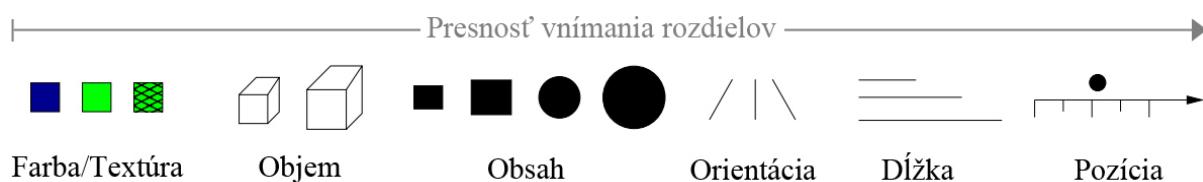
1.5.2. Hviezdicový glyf

Podobne ako paralelné koordináty, hviezdicový glyf tiež mapuje každý z atribútov na vlastnú os. V tomto prípade však všetky začínajú v spoločnom bode a koncové body sú rovnomerne rozmiestnené po celom obvode kružnice, ktorej polomer určuje dĺžku osi. Lomená čiara prechádza každou osou v bode nanesenej hodnoty a na konci sa vracia do bodu v ktorom začala. Tým vzniká polygón pripomínajúci hviezdu (obr. 17b).

1.5.3. Profilový glyf

Tento typ glyfu mapuje každý atribút na samostatný obdĺžnik, pričom jeho farba charakterizuje o aký atribút ide, a samotná hodnota určuje jeho výšku. Počet atribútov stanovuje počet obdĺžnikov, ktoré sú rozmiestnené pozdĺž spoločnej osi. V podstate využíva rovnaký princíp ako stĺpcový diagram (obr. 17a).

Častou kritikou glyfov je problém, že niektoré grafické atribúty sa vnímajú ľahšie a sú zrozumiteľnejšie ako iné. Ak zoberieme príklad tvárových glyfov, tak dĺžku úst rozoznávame oveľa presnejšie ako zakrivenie nosa alebo polomer hlavy. Preto je dôležité pri návrhu glyfu sústrediť sa aby jeho črty boli intuitívne a ľahko vnímateľné. Ak majú rôznu zložitosť vnímania, tak mapovanie atribútov je potrebné robiť v takom poradí, aby najdôležitejšie boli najpresnejšie vnímateľné. Na obrázku 18 je niekoľko grafických čŕt, ktoré sú usporiadané vzhľadom k tomu ako presne dokážeme merat' a vnímať ich rozdiely [3][19][37].



Obr. 18 – Grafické črty usporiadané podľa miery schopnosti vnímať ich rozdiely [3].

1.6. Vizualizácia a 3D priestor

Žijeme v trojdimenzionálnom priestore (ak neberieme v úvahu čas) z tohto dôvodu je pre nás jeho vnímanie intuitívne a môžeme to využiť v náš prospech pri tvorbe vizualizácie. Otázkou ostáva či je výhodnejšie využívať trojdimenzionálny priestor pri vizualizácii dát, oproti dvojdimenzionálному, teda či dokáže zrozumiteľne zobraziť väčšie množstvo dát. 3D priestor je očividne bohatší na informácie a ponúka viac možností ako 2D priestor, keďže ten je jeho podmnožinou. Nemusí to však prinášať iba pozitívny efekt, a často sa využíva z nesprávnych dôvodov. Informácie, ktoré nám umožňujú vnímať 3D priestorovosť sa nazývajú „depth cues“.

1.6.1. Depth cues

Môžeme ich rozdeliť do troch skupín:

- Monokulárne statické
- Monokulárne dynamické
- Binokulárne

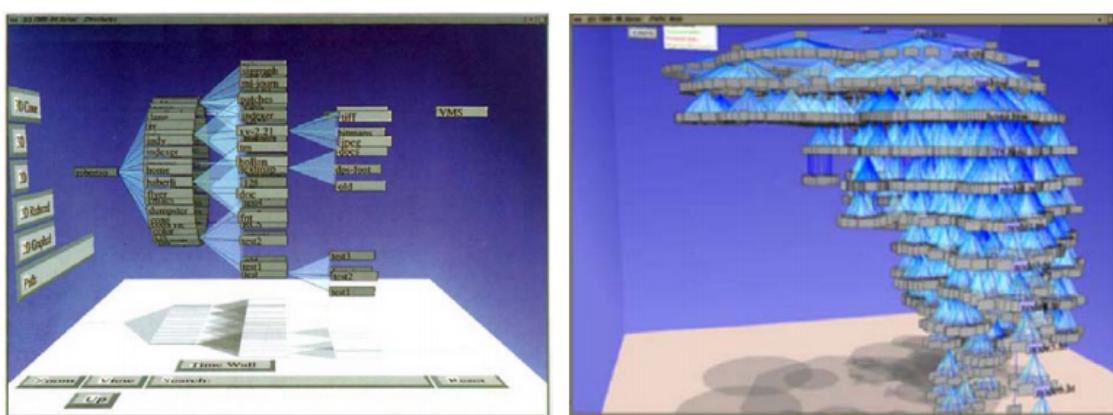
Monokulárne statické – Nachádzame ich v statických obrázkoch, a vnímame ich aj pri pohľade iba jedným okom. Využívajú sa v 2D vizualizačných technikách pre dosiahnutie efektu tretieho rozmeru. Patria sem: lineárna perspektíva, relatívna veľkosť, vzájomná poloha, osvetlenie, vrhanie tieňa, hĺbka ostrosti atď. [38].

Monokulárne dynamické – Vnímame ich v pohyblivom obrazze, a opäť postačí pohľad iba jedným okom. Jednou z informácií je pohybový parallax, ktorý vnímame napríklad keď sledujeme krajinu pri jazde vlakom. Blízke objekty sa javia pohybujúce oveľa rýchlejšie ako vzdialené. Druhou je kinetický hĺbkový efekt, ktorý môžeme demonštrovať na príklade drôtu poohýbaného do komplexnej 3D štruktúry. Pri pohľade na statický obrázok často nedokážeme určiť presný tvar tohto objektu, ak ale sledujeme rotujúci sa objekt v 3D priestore, jeho štruktúra je zjavná. Aby sme teda dokázali vnímať komplexné štruktúry, vyžaduje si to pohyb v danom priestore, preto je nevyhnutné zabezpečiť tento aspekt vo vizualizácii, buď formou interakcie, alebo animácie, napríklad spomínanej rotácie.

Binokulárne – Pri tejto kategórii využívame, že oči človeka sú od seba vzdialené v priemere 6,4cm a sledujú scénu z rôznych pozícií a pod odlišným uhlom. Tento fakt spôsobuje, že mozog dostáva dva mierne rozličné pohľady, z ktorých dokáže zistiť relatívne vzdialenosť objektov. Tento jav sa nazýva stereoskopická disparita, a bezpochyby patrí k najdôležitejším aspektom, ktoré nám umožňujú vnímať 3D priestor. Podobné podmienky dokážeme simulovaliť aj v počítačovej grafike, ak dokážeme zabezpečiť aby každé oko dostávala svoj vlastný, náležite správny, obraz. Hovoríme vtedy o stereo zobrazovaní. Samotné vygenerovanie dvojice správnych obrazov nie je jediný problém. Zabezpečiť aby obe oči zároveň sledovali iba svoj obraz je zatiaľ najväčšia výzva stereo zobrazovania. Aj keď existuje niekoľko techník, každá z nich má svoje pre a proti. Efektívnejšie sú stále relatívne málo dostupné alebo vyžadujú externé zariadenia, prístupnejšie techniky sú naproti tomu menej efektívnejšie a často je nimi negatívne ovplyvnený vizuálny efekt.

1.6.2. 3D grafové štruktúry

Často štruktúra informácií je natoľko komplexná, že vyvstáva otázka či nie je ideálnejšie zobrazovať ich v troch rozmeroch. Najčastejším dôvodom je kríženie hrán. Jedinou štruktúrou pre, ktorú komplexnosť nespôsobuje veľké problémy sú stromy, pretože ich stále môžeme zobraziť v 2D priestore ako planárne grafy. Aj napriek tomu existujú aj varianty stromov v 3D priestore. Príkladom je kužeľový strom, o ktorom jeho tvorcovia tvrdia, že dokáže zobraziť približne 1000 vrcholov bez spôsobenia chaosu, čo je omnoho viac ako pri 2D projekciách (obr. 19 [2]).



Obr. 19 – Kužeľový strom [2].

Vyžaduje však neustálu interakciu užívateľa a nutnosť otáčať vizualizáciu. Všeobecnejšie grafové štruktúry sa však len málokedy dokážu vyhnúť kríženiu hrán, a práve preto by mali ľažiť z tretej dimenzie. Podľa štúdie Sollenbergera and Milgrama v roku 1993 [32], bolo preukázané, že pohybový parallax, a stereo zobrazenie nám umožňujú zrozumiteľne vnímať viac informácií, v prípade sietí takmer trojnásobok. Zvyšujú najmä schopnosť správne rozpoznávať pozíciu hrán v priestore.

1.6.3. Anaglyf

Patrí k najstarším technikám stereo zobrazovania, zároveň aj k najpopulárnejším práve pre jeho dostupnosť a využitie aj pri statických fotografiách. V dnešnej dobe ju už vytláčajú nové modernejšie prístupy, ktoré poskytujú vernejší zážitok, avšak stále patrí k najdostupnejším, pretože nevyžaduje dômyselnosť externé zariadenia ak nerátame ľahko dostupné okuliare, ani nekladie podmienky na zobrazovací display. Nevýhodou je menej dokonalý výsledok spôsobený predovšetkým skreslením farieb.

Farebné skreslenie spôsobuje princíp na ktorom je anaglyf postavený. Oba obrazy pre ľavé aj pravé oko sa nachádzajú v jednom obraze. Trik spočíva v použití dvoch farebných komplementárnych filtrov, ktorými sa prekrýva ľavý a pravý obraz. K najčastejšie používaným patrí spojenie červeného a azúrového filtra, pretože dokáže zobrazovať aj farebné obrazy. Pôvodný červený a modrý filter boli určené pre čiernobiele obrazy. Oba obrázky sú najskôr prekryté navzájom opačnými filtrami, čo spôsobuje odstránenie určitých farebných zložiek v závislosti od jeho farby. Spojením týchto dvoch obrázkov do jedného potom vzniká takzvaný anaglyf. Pre dosiahnutie 3D efektu sú potrebné okuliare s rovnakými farebnými filtrami, ktoré zabezpečujú, že každé oko vidí iba svoj obraz [15].

Kapitola 2

2. Návrh metód

Obsahom tejto kapitoly je idea návrhu našej aplikácie pre vizualizáciu grafových štruktúr. Bližšie si špecifikujeme charakter aplikácie, požiadavky na ňu, a zároveň vysvetlíme a odôvodníme výber použitych prístupov. Taktiež sa budeme venovať vlastným technikám, ktoré prezentujeme ako originálny prínos.

Hlavným cieľom je priniesť nové a inovatívne techniky do vizualizácie grafových štruktúr, ktorých skúmanie obohatí množinu vedomostí týkajúcej sa tejto tematiky a zároveň jej praktické využitie poskytne nový, užitočný pohľad na rôzne dátové sety.

2.1. Základná charakteristika

Kedže navrhovaná technika musí byť schopná vizualizovať rôzne grafové štruktúry, pružinové algoritmy poskytujú vďaka svojej univerzálnosti pravdepodobne najlepší smer ktorým sa môžeme vydať. Aj keď ich tematika je už pomerne rozsiahlo zdokumentovaná, drívavá väčšina prác sa venuje ich použitiu v euklidovskej geometrii. Základom sú aplikácie v rovine ale v súčasnosti sa objavujú aj techniky rozširujúce sa do troch rozmerov [33]. Triviálne rozšírenie do 3d pomocou pridania ďalšieho rozmeru však vedie k neprehľadnej vizualizácii. Väčšie a komplexnejšie štruktúry ale dokážeme vnímať v priestore intuitívne, ak kopírujú tvar niektorého známeho priestorového útvaru, napr. kužeľa [2] alebo gule. Z tohto dôvodu sme sa rozhodli aplikovať pružinový model rozloženia vrcholov grafu na sférickom povrchu. Výber gule ako vhodnej primitívy, na ktoré rozložíme graf nebol náhodný. Samotný guľový povrch poskytuje viacero výhod, ktoré predstavíme.

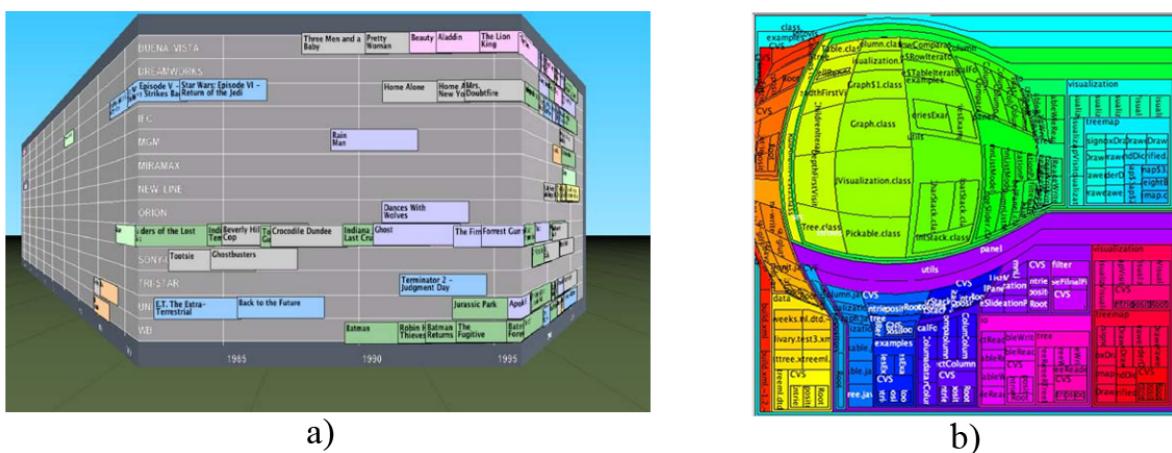
Vzťah medzi entitami budú reprezentovať krivky s rôznou farbou a priehľadnosťou v závislosti od vzťahu. Aplikácia bude schopná zobrazovať aj orientované aj neorientované vzťahy. Zároveň bude na vzhľad kriviek aplikovaný algoritmus zväzovania hrán, ktorého úlohou bude zvýšiť prehľadnosť cele vizualizácie aj s vyšším počtom vzťahov. Jednotlivé entity budú reprezentované 3D glyfom, ktorý bude odzrkadľovať hodnoty ich atribútov a pomáhať odhalovať podobnosti medzi nimi.

Ked'že naša vizualizácia nie je určená pre konkrétny typ dát, všetky vstupné dáta budú upravené do správnej podoby na základe definície vstupného formátu. Základom vstupu bude matica susednosti, reprezentujúca hrany v grafe (vzťahy medzi entitami) a zoznam samotných entít spolu s hodnotami ich atribútov. Bližšia špecifikácia vstupného formátu sa nachádza v prílohe č.1.

Pre zvýšenie celkovej prehľadnosti vizualizácie budú v aplikácii implementované rôzne techniky, ktoré pomáhajú vnímať 3D priestorovosť či už v pohyblivej alebo statickej vizualizácii.

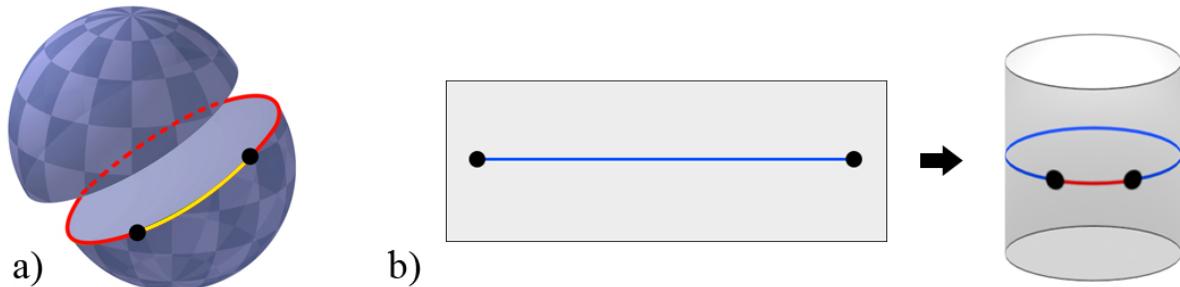
2.2. Sférická geometria

Ako sme už spomínali, výber sférickej geometrie neboli náhodný. Jednou z hlavných predností vizualizácie na povrchu gule je fakt, že jeho zakrivenie automaticky zobrazuje informáciu, ktorá je centrom záujmu vo vyššom detaile a s narastajúcou vzdialenosťou od stredobodu pozornosti sa detail zmenšuje a zobrazované informácie sa zhustujú. Výsledkom je možnosť lepšie sa sústrediť na špecifickú časť dát, ale zároveň si zachovať aj globálny obraz. Tento prístup sa vo vizualizácii nazýva Focus+Context [34]. Touto vlastnosťou disponuje aj hyperbolická geometria. Lamping s kol. vo svojej štúdii vizualizácie hierarchických štruktúr zobrazovaných v hyperbolickej geometrii, vyzdvihujú práve túto vlastnosť a jej prínos v zmysle, že práve vďaka nej je schopná poňať aj hierarchie s veľkým množstvom vrcholov [16]. Pri klasických 2D vizualizačných technikách v euklidovskej geometrii, je potrebné tento efekt dodatočne simulať, napríklad formou „fish-eye“ šošovky [3] (obr. 20b) alebo perspektívnej steny (obr. 20a) [3] [38].



Obr. 20 – a) perspektívna stena, b) fish-eye pohľad [3].

Ďalšou zaujímavou vlastnosťou gule je jej povrch, ktorý v podstate môžeme vnímať ako ohraničenú dvojrozmernú plochu, ktorá však nemá žiadne rohy. Ohraničenú preto, že veľkosť povrchu je určená v závislosti od polomeru gule, a vieme ju presne určiť. Zároveň ale nemôžeme doraziť na jej koniec, pretože ak budeme po jej povrchu kráčať stále rovnakým smerom, eventuálne skončíme na mieste v ktorom sme odštartovali a môžeme v rovnakom smere kráčať nekonečne dlho. Silou riadené algoritmy v 2D alebo 3D priestore často narážajú svoje limity pri rozmiestňovaní vrcholov v rovine/priestore, kedy sú inak navzájom si blízke vrcholy odtlačené odpudivými silami do neprirodzené vzdialenosť polôh, prípadne neefektívne zapĺňajú grafický display. Je nevyhnutná ich presnejšia konfigurácia, a rovnako zabezpečiť aby nikam „neušli“ teda, aby vizuálne ľažisko rozmiestnenia ležalo približne v strede obrazovky. Aj na guľovom povrchu bude nutné zabezpečiť aby dĺžka hrán bola primeraná v závislosti na počte vrcholov aby sa zachovalo ich rovnomerné distribuovanie po celom povrchu. Výhodou však je, že graf nemôže opustiť rozmery plochy, čiže odpadá potreba definovať okraje za ktoré sa vrcholy nesmú posunúť alebo zabezpečiť aby jeho centrum bolo v strede plochy.



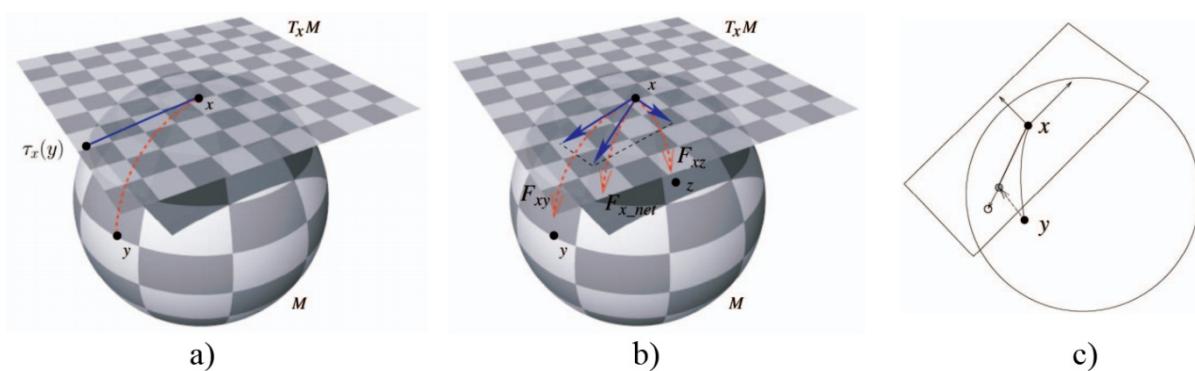
Obr.21 – a) veľká kružnica (červená čiara), najkratšia vzdialosť (žltá čiara) [35], b) zakrivenie obdĺžnikovej plochy do valcovej a vyznačená zmena veľkosti vzdialenosť.

Zo spomenutých vlastností povrchu vyplýva ešte jedna dôležitá charakteristika, ktorou je vzdialenosť dvoch bodov ležiacich na jej povrchu. Pritom využijeme vedomosť, ktorá hovorí, že ľubovoľnými dvoma bodmi ležiacimi na povrhу gule, dokážeme preložiť kružnicu, ktorá má rovnaký polomer ako guľa. Nazýva sa veľká kružnica (great circle) (obr.21a). Je nevyhnutné nájsť ju ak sa snažíme zistiť najkratšiu vzdialosť, respektíve najkratší kružnicový oblúk, medzi spomínanými bodmi. Body získanú kružnicu rozdeľujú na dva kružnicové oblúky, najkratšia vzdialosť medzi nimi, je dĺžka kratšieho z nich. Z tohto vysvetlenia ale vyplýva, že nech už preložíme bodmi akúkoľvek kružnicu, vždy dostaneme pre ich vzdialenosť dve rôzne hodnoty.

Naproto tomu v euklidovskej geometrii vždy existuje práve jedna vzdialenosť. Reálne využitie tejto skutočnosti sa najlepšie demonštruje na príklade obdĺžnikovej plochy, ktorej zahnutím a spojením vytvoríme valec (obr. 21b). Na ľavom obrázku, vidíme dva body umiestnené pri opačných okrajoch 2D plochy. Ich vzdialenosť je takmer rovnako veľká ako šírka celej plochy, ak však plochu stočíme do valca, vidíme, že body sa nachádzajú hned vedľa seba a ich najkratšia vzdialenosť je niekoľkonásobne menšia. Ak by šlo o konkrétnu vizualizáciu, vykreslená hrana v prvom prípade smeruje naprieč celou grafickou plochou a zaberá jej pomerne veľkú časť. Práve takéto hrany sú pôvodcami „neporiadku“ a znížujú čitateľnosť celej vizualizácie. Pri guľovej ploche funguje rovnaký princíp, ale vo všetkých smeroch. Guľová plocha teda umožňuje pre rovnaký graf väčšiu variabilitu rozmiestnenia vrcholov, a teda potenciálne lepší výsledný stav pružinového modelu, ako v prípade jej plošnej 2d analógie.

2.3. Pružinové algoritmy v neeuklidovskej geometrii

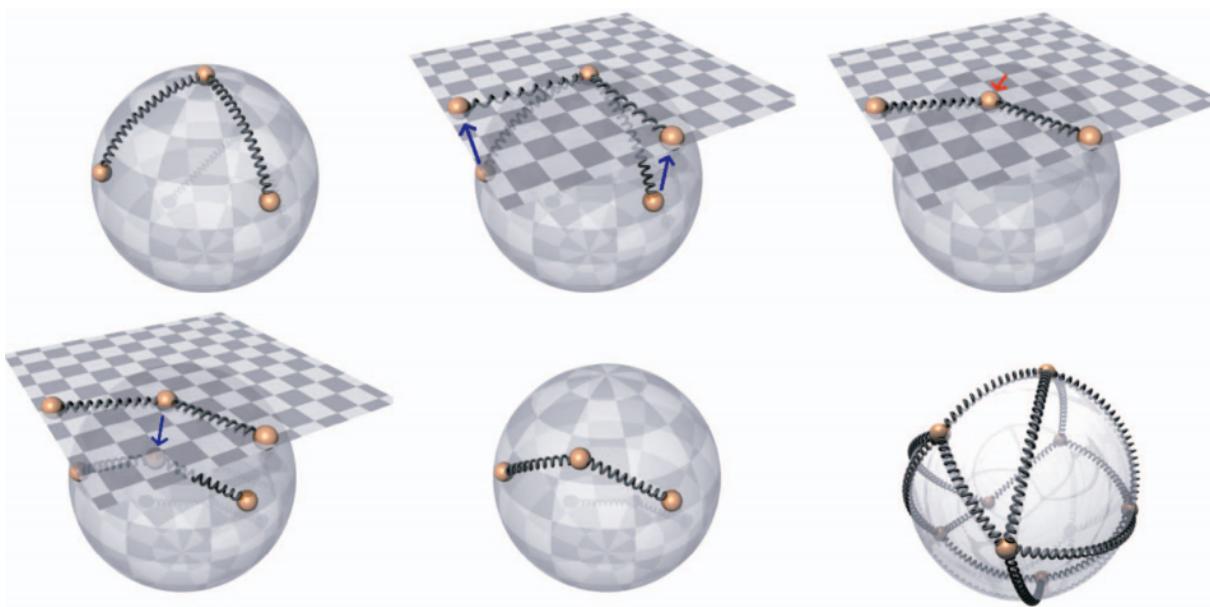
Mimo euklidovského priestoru, sa často na vizualizácii grafov, najmä hierarchií, využíva hyperbolický priestor, ktorý voči nemu poskytuje určité výhody, ako napríklad spomínaný „fish-eye“ efekt. Aj napriek nižšiemu počtu vizualizácií v inom ako euklidovskom priestore, existujú štúdie, ktoré sa venujú tematike presunu algoritmov z euklidovskej do neeuklidovskej geometrie. Jedným z dôvodov je zvýšená schopnosť vnímať komplexnejšie grafy v 3D priestore ak ležia na povrchu jednoduchých, alebo všeobecne známych geometrických útvarov ako kužeľ, guľa, valec a podobne [17].



Obr. 22 – a) znázornenie priemetu a vrcholu y a kružnicového oblúka xy do dotyковej roviny v bode x, b) výpočet síl pôsobiacich na x, c) priemet bodu y do dotykovej roviny a úprava vzdialenosť od x [18].

Existuje niekoľko rôznych princípov ako uskutočniť prevod pružinových algoritmov medzi euklidovskou a neeuklidovskou geometriou, často krát však ide o špecifický prevod pre konkrétny typ geometrie alebo priestoru. Kobourou a Wampler navrhli techniku, ktorá nie je obmedzovaná na konkrétny objekt, ale je aplikovateľná na akúkoľvek geometriu, ktorá patrí do kategórie Riemannovej [18]. Základný princíp ich metódy, si popíšeme na guli, pretože je najviac relevantná s touto prácou a v zásade to neovplyvní pochopenie všeobecného prístupu.

V euklidovskej geometrii vnímame ako vzdialenosť dvoch bodov, dĺžku úsečky, ktorá ich spája. Podobne môžeme pristupovať aj k chápaniu vzdialosti dvoch bodov, ktoré ležia na povrchu gule. V jej prípade však ich vzdialosť určuje najkratší kružnicový oblúk. Vo všeobecnosti môžeme teda uvažovať o definícii vzdialosti ako o najkratšej spojitej krvke, ktorá v danej geometrii spája zadané body (obr. 22a [18]). Objasnenie tohto faktu je dôležité, pretože táto technika spolieha na rozšírenie metodiky z euklidovskej geometrie na Riemannovu.



Obr. 23 – Grafické znázornenie celého postupu posunutia centrálneho vrcholu [18].

Základným kameňom je využitie dotykovej plochy, v mieste vrcholu grafu, o ktorom práve pri iterovaní pružinového systému rozhodujeme v ktorom smere ho posunieme a aký veľký bude samotný posun. Nazvime ho centrálny vrchol. Po nájdení dotykovej roviny v mieste centrálneho vrcholu všetky ostatné vrcholy do tejto roviny kolmo premietneme v smere normálového vektora plochy v centrálnom vrchole. (obr. 22c [18]). Tým získame, že všetky vrcholy grafu sú transformované do euklidovskej roviny, v ktorej už jednoducho dokážeme vypočítať výslednú pôsobiacu silu. Predtým je ale potrebné upraviť vzdialenosť

všetkých vrcholov od centrálneho vrcholu, pretože priemetom sa skreslili. Napríklad vrcholy ležiace na odvrátenej strane gule, ktoré sú v skutočnosti veľmi vzdialené od centrálneho vrcholu, sa po premietnutí nachádzajú hned vedľa neho. Je preto nutné vypočítať skutočnú vzdialenosť každého vrcholu od centrálneho priamo na povrchu gule a následne podľa nich upraviť aj ich vzdialosti v rovine dotykovej plochy (obr. 22c [18]). Potom už môžeme pristúpiť k samotnému riešeniu pôsobiacich síl klasickým spôsobom charakteristickým pre euklidovskú rovinu (obr. 22b [18]). Po získaní pozície bodu v rovine, ktorý reprezentuje novú pozíciu pre centrálny vrchol, je potrebné premietnuť ju späť na povrch gule. Idea je rovnaká ako pri premietaní bodov do dotykovej roviny. Po premietnutí opäť musíme upraviť dĺžku kružnicového oblúka aby odpovedala vzdialosti v dotykovej rovini, čím získame novú pozíciu pre centrálny vrchol (obr. 23 [18]). Celý tento postup je potrebné opakovať pre každý vrchol, ktorý sa snažíme posunúť a dotyková plocha musí byť definovaná práve v ňom.

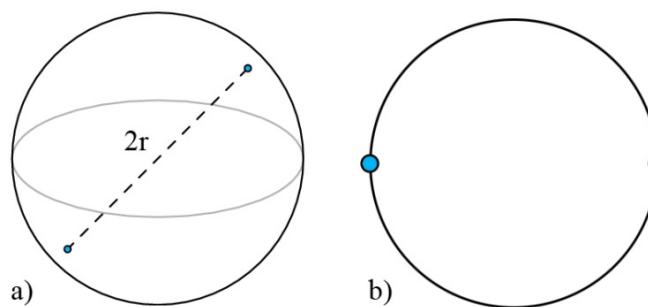
Táto technika funguje bezchybne, avšak v prípade gule existujú dva prípady, ktoré musíme riešiť individuálne. Jeden z nich nastáva vtedy, keď veľkosť posunutia centrálneho vrcholu v dotykovej rovine presiahne veľkosť polomeru gule na ktorej sa vrcholy nachádzajú. V takej situácii by spätný kolmý priemet neskončil na povrchu gule. V tomto prípade je najlepšie skrátiť vektor posunutia na hodnotu menšiu ako je polomer a po premietnutí predĺžiť kružnicový oblúk do adekvátej dĺžky. Druhý problém spôsobujú body, ktoré sa nachádzajú presne na opačnej strane gule, a ich priemet do dotykovej roviny končí na rovnakej pozícii akú má centrálny vrchol. Vtedy medzi nimi nedokážeme identifikovať žiadny vektor, a preto nemôžeme upraviť ich vzájomné pozície do správnej vzdialenosťi. Kobourou a Wampler v tomto prípade navrhujú napevno určiť vektor posunutia, ktorý sa použije. Vo všeobecnosti však nezáleží aký vektor použijeme, môžeme využiť ktorýkoľvek ležiaci v dotykovej rovini (viď. kapitola 2.4).

Konečná veľkosť guľového povrchu nemusí na výsledok kvality rozloženia vrcholov pôsobiť iba pozitívne. Keďže systém sa snaží minimalizovať svoju celkovú energiu, v prípade konečne veľkého priestoru sa môže dostať do situácie kedy už neexistuje dostatok miesta aby sa rozťahol do žiadanej podoby. Ak máme k dispozícii nekonečný priestor, systém má neobmedzenú voľnosť, to však vedie k napríklad už spomínamej neprehľadnosti oblaku bodov v 3D. Vhodné by preto bolo nájsť určitý kompromis medzi obmedzeniami a voľnosťou. V našom prípade sme doteraz vnímali povrch gule ako konečne veľkú 2D oblasť. Na rozdiel od euklidovského 2D priestoru, máme k dispozícii ďalší extra parameter, ktorý dosiaľ nijako nevyužívame a to vzdialenosť vrcholov od stredu gule. Všetky vrcholy v štandardnom návrhu našej vizualizácie ležia v rovnakej „výške“. Tento extra rozmer môžeme využiť na zníženie

energie systému tým, že vrcholy na ktoré je vyvíjaný tlak a nemajú už priestor aby sa vzdialili v rámci plochy gule, začneme vysúvať do vyššej vzdialenosť od povrchu gule. Vďaka tomu dosiahneme zníženie pôsobiacich sín v danom vrchole. Systém je stále ohraničovaný veľkosťou guľového povrchu, poskytujeme mu však určitý stupeň voľnosti, ktorý môže zlepšiť kvalitu finálneho rozloženia. Navrhovaná technika však obsahuje aj obyčajný model, aj modifikovaný využívajúci výšku, aby sme dokázali porovnať dosiahnuté výsledky.

2.4. Antipódne body

V prípade gule, dvojicu bodov ležiacich na jej povrchu nazývame antipódne, ak ležia presne oproti sebe, teda ich priama spojnica má dĺžku rovnakú, ako je priemer gule, a zároveň prechádza jej stredom (obr. 24a). Príkladom takej dvojice bodov je južný a severný pól gule. Veľkú kružnicu na ktorej ležia delia na dva rovnako veľké oblúky (obr. 24b). Problémom týchto bodov je však nájdenie veľkej kružnice, lepšie povedané nájdenie presne jednej, pretože ich vieme zstrojitiť nekonečne veľa. Pre akékoľvek iné dva body, ktoré nie sú antipódne dokážeme zstrojitiť práve jednu veľkú kružnicu a jeden z kružnicových oblúkov je vždy kratší ako druhý, na základe čoho vieme určiť aj smer interpolácie medzi nimi. Tento fakt zaraďuje antipódne body do prípadov v ktorých nezáleží v ktorom smere budeme medzi bodmi interpolovať a ktorú veľkú kružnicu si zvolíme, keďže ich je nekonečne veľa a obidva kružnicové oblúky sú rovnako veľké.

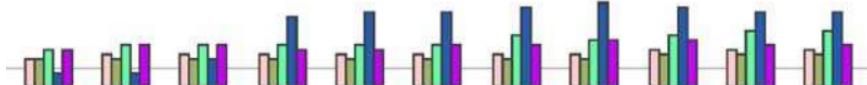


Obr. 24 – Antipódne body znázornené: a) na povrchu gule, b) na kružnici.

2.5. Vizualizácia entít – 3D glyf

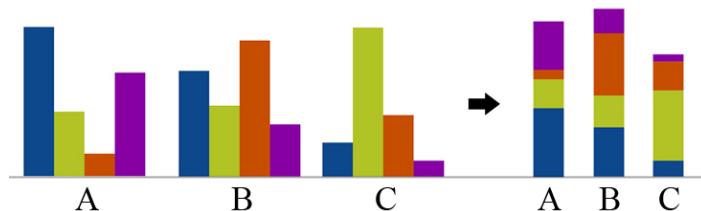
Pri návrhu glyfu sa sústredíme predovšetkým na jednoduchosť a využijeme grafické črty, ktoré dokážeme čo najpresnejšie vnímať aj v 3D priestore. Za základ sme si preto zvolili profilový glyf, ktorý má veľmi jednoduchý ale účelný dizajn (obr. 25) [2].

Jednou zo spomínaných vlastností ľudského vnímania, je rôzna citlivosť vnímať a určovať mieru rozdielnosti grafických čít. Cleveland a McGill pri svojich experimentoch dokázali, že rozdiely v dĺžke pozdĺž spoločnej osi dokážeme rozoznať rýchlejšie a s väčšou presnosťou ako rozdiely v orientácii, uhle, veľkosti či farbe [1]. Práve vďaka tomu profilové glyfy umožňujú presnejšie vnímanie rozdielov v rámci atribútov glyfu, ale zároveň aj medzi glyfmi samotnými.



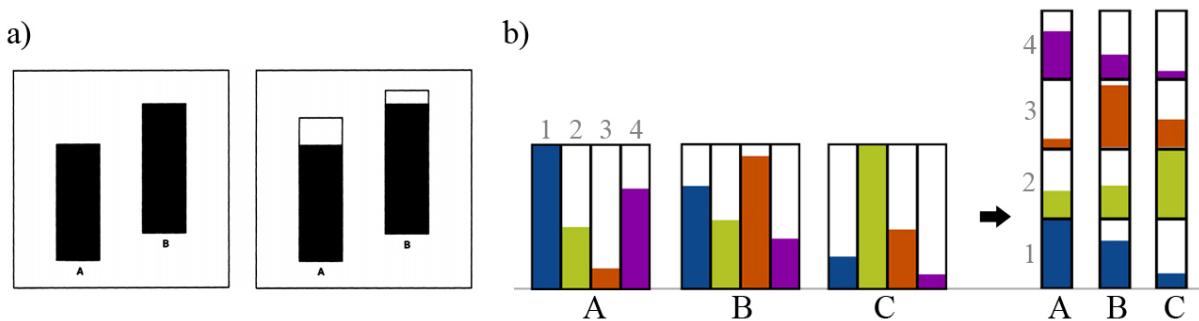
Obr.25 – Profilový glyf [2].

Transformáciou profilového glyfu do 3D priestoru, získame ďalší rozmer, ktorým je jeho hrúbka. Z princípu ako sa vytvára profilový glyf, však nemá veľký význam využiť ju na mapovanie ďalšieho atribútu, pretože hodnoty by už viac nemenili iba pozdĺž jednej osi. Ak by sme obdlžníkový priestor určený na mapovanie jedného atribútu rozšírili na kváder so štvorcovou podstavou, nemusíme ich viac usporadúvať vedľa seba iba v jednom smere, ale napríklad aj dvoch, teda umiestniť ich do mriežky. Obidva spôsoby sú však neekonomicke z hľadiska priestoru, pretože vrcholy grafu môžu byť rozmiestnené iba na povrchu gule, tým pádom by takéto glyfy rýchlo pohltili priestor. Ako ušetríť miesto v podobnom prípade sa môžeme inšpirovať v reálnom svete, ak prirovnáme glyfy ležiace na povrchu gule, k budovám na povrchu zeme. Ak potrebujeme vytvoriť veľa užitočného priestoru vzhľadom k obmedzenej ploche, najlepším spôsobom je stavať výškové budovy. Aj keď veľkosť zemského povrchu je limitovaná, môžeme tvrdiť, že výškou nie sме obmedzovaní. Rovnaký princíp môžeme aplikovať na zmenu glyfu. Jednotlivé stĺpce, preto nebudeme ukladať vedľa seba ale na seba, smerom do výšky. Transformácia profilového glyfu do tejto podoby nielen šetrí miestom, ale zároveň je zachovaná jeho podstata, nanášania hodnôt a porovnávania rozdielov iba v jednom smere spoločnej osi (obr. 26).



Obr. 26 – Transformácia rozloženia stípcov profilového glyfu z horizontálnej do vertikálnej podoby.

Môžeme si všimnúť, že vnímanie rozdielov medzi glyfmi navzájom je však v tomto prípade sťažené, pretože hodnoty rovnakých atribútov začínajú v rôznych výškach v závislosti od hodnoty predchádzajúceho atribútu. Z tohto dôvodu navrhujeme modifikovať profilový glyf tak ako je to naznačené na obrázku 27b. Veľkosť každého stĺpca je vyznačená hranicou. Následným vyskladaním týchto stĺpcov, je vnímanie rozdielov oveľa intuitívnejšie a presnejšie, nielen preto, že všetky rovnaké atribúty začínajú v rovnakej výške, ale dodatočná informácia o veľkosti celého stĺpca nám umožňuje registrovať aj malé rozdiely v hodnotách. Na tento fakt poukázali aj Cleveland a McGill pri využívaní stĺpcových diagramov a môžeme ho pozorovať na obrázku 27a [19]. Vidíme, že bez použitia hraničnej obálky len ľahko vnímame rozdiel, naopak s ním je zmena okamžite zjavná.



Obr. 27 – a) Vnímanie rozdielu hodnôt stĺpcového diagramu bez a s hraničnou obálkou [19], b) profilový glyf s využitím hraničnej obálky a jeho transformácia.

Samotná výška jednotlivých stĺpcov bude v aplikácii voľne nastaviteľná, rovnako ich počet. Aj keď množstvo atribútov vstupných dát nebude limitované, pravdepodobne bude určená hranica, kol'ko ich môžeme na glyf naniest'. Príliš veľké množstvo by mohlo spôsobiť ich neprimeranú veľkosť'.

Nie všetky typy atribútov však dokážeme preniesť do číselnej podoby, a nanášať na glyf. Ak vrcholy grafu reprezentujú ľudí, jedným z atribútov môže byť meno človeka, z čoho vyplýva, že pravdepodobne každý vrchol bude mať unikátnu hodnotu. Pri vizualizácii grafov môžeme naraziť na niekoľko podobných typov informácií, ktoré nie je efektívne prenášať do číselnej formy. Z tohto dôvodu aplikácia bude schopná zobrazovať tieto informácie formou textu, ktorý sa zobrazí v blízkosti príslušného vrcholu a na glyf budeme nanášať iba atribúty, ktorých porovnávanie na základe číselnej hodnoty má význam. Atribúty vrcholu preto budú rozdelené do niekoľkých kategórií podľa typu obsahu a definované vo vstupnom súbore. Ďalšou informáciou, ktorá sa využíva v týchto typoch vizualizácie je fotografia alebo obrázok, reprezentujúca vrchol. Preto ju bude možné zobraziť rovnako ako text.

2.6. Vizualizácia vzťahov

Vzťahy budú reprezentovať krivky, ktorých základnými atribútmi budú farba, hrúbka a priehľadnosť. K samotnej vizualizácii kriviek však môžeme pristupovať dvoma rôznymi spôsobmi. Budť budeme využívať na ich zobrazovanie iba samotný povrch gule a teda využijeme krivky so sférickou lineárhou interpoláciou, alebo ho môžeme opustiť a aplikovať klasické krivky s obyčajnou lineárhou interpoláciou.

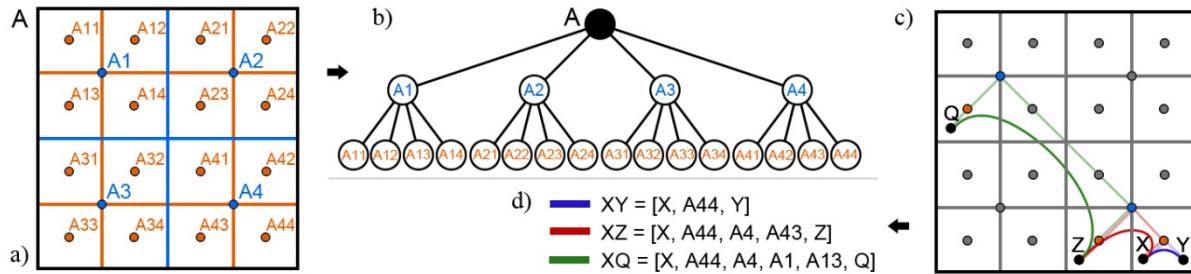
Kladným aspektom prvého spôsobu je, že by ťažil zo spomínamej vlastnosti povrchu, kde informácie zobrazované v centre pozornosti sú znázornené vo vyššom detaile. Nevýhodou je však, že budú okupovať rovnaký priestor ako vrcholy grafu, čo pri veľkom počte vrcholov môže spôsobovať neschopnosť pozorovať hrany. Klasické krivky opúšťajú povrch gule, a môžu smerovať jej vnútom alebo vonkajškom, obidve alternatívy však prenechávajú povrch gule iba pre zobrazovanie vrcholov. Identifikácia obyčajných kriviek bude pravdepodobne zložitejšia, pretože môže vytvárať komplexné 3D štruktúry, avšak informácie ako pohybový paralax nám uľahčia ich vnímanie. Oba spôsoby môžu ponúkať efektívne riešenie, preto budú implementované a skúmané.

Aplikácia nebude obmedzená iba na zobrazovanie neorientovaných vzťahov, ale aj orientovaných. Z tohto dôvodu bude potrebné navrhnuť ovládanie farieb hrán a ich hrúbky tak aby boli schopné odzrkadľovať rôzne vzťahy. Nie všetky vzťahy musia byť rovnaké a môže ich existovať niekoľko typov, preto aplikácia bude ponúkať nastavovanie atribútov hrán samostatne pre každý typ vzťahu. Jedným z aspektov, ktorý by mal pomôcť rozlišovať orientované vzťahy od neorientovaných je animácia farby v smere orientácie vzťahu.

2.7. Zväzovanie hrán

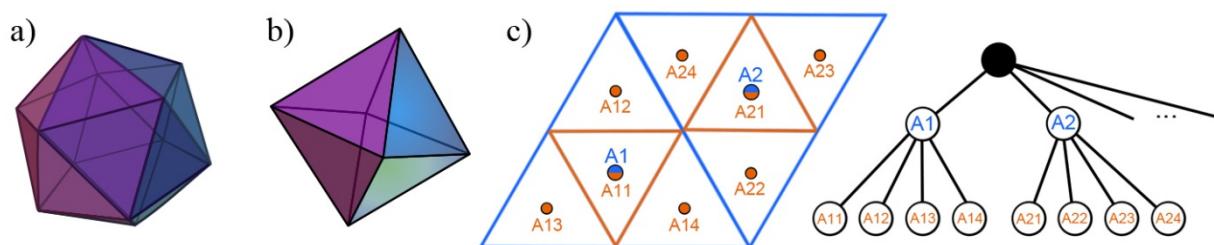
Nech už zvolíme akýkoľvek typ čiar na reprezentáciu hrán, pri komplexných grafoch s vysokým počtom hrán by sme dosiahli zahľatie priestoru pomerne rýchlo. Rozhodli sme sa preto navrhnuť a implementovať vhodný algoritmus zväzovania hrán založený na princípe podobnom hierarchickému zväzovaniu. Ako už bolo spomenuté, tento prístup využíva hierarchickú štruktúru stromov, a pozície jeho vrcholov na definíciu riadiacich vrcholov krivky. Naša aplikácia však nebude zobrazovať iba stromové štruktúry ale grafy všeobecne. Z tohto dôvodu nemôžeme využiť vlastnú hierarchiu grafu a pozície jeho vrcholov. Namiesto toho vytvoríme novú stromovú hierarchiu, ktorá bude reprezentovať rekurzívne rozdelenú zobrazovaciu plochu na stále menšie rovnako veľké oblasti s rovnakým tvarom. Každý vrchol

hierarchie reprezentuje jednu oblasť a nesie informáciu o pozícii jej stredu. Príklad vytvorenia hierarchie sa nachádza na obrázku 28. Vľavo vidíme rekúzívne rozdelenú štvorcovú plochu a vpravo hierarchiu z nej vytvorenú. Konkrétny obrázok vytvára veľmi podobnú hierarchiu, aká by vznikla pri aplikácii hierarchického zväzovania na stromovú mapu pre strom, v ktorom každý jeho vrchol má štyroch rovnako veľkých synov.



Obr. 28 – a) rekúzívne rozdelená štvorcová plocha A so znázornenými stredmi, b) hierarchia vytvorená na základe rozdelenia na obrázku a), c) znázornené krivky pre hrany XY, XZ a XQ, d) popis hrán a ich riadiacich vrcholy.

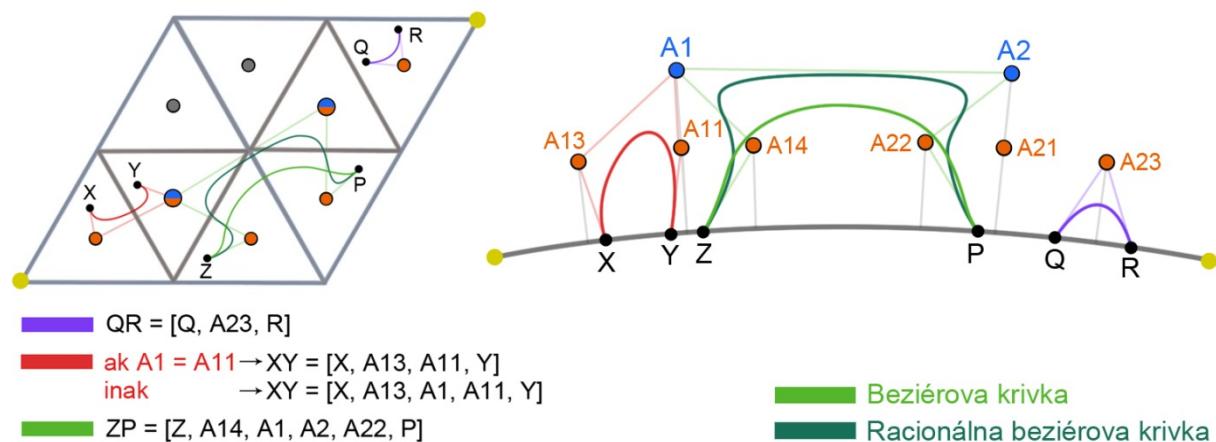
Po vytvorení hierarchie je postup vytvorenia riadiacich vrcholov pre hranu medzi dvoma vrcholmi nasledovný. Pre oba vrcholy musíme najskôr zistiť v ktorej z najmenších oblastí sa nachádzajú, teda určiť listy v hierarchii, ktoré reprezentujú tieto oblasti. Potom musíme nájsť v hierarchickom strome cestu medzi nimi. Vrcholy nachádzajúce sa na tejto ceste poslúžia ako riadiace vrcholy krivky. K nim samozrejme ešte pridáme aj pozície konkrétnych dvoch vrcholov, ktoré určujú počiatočný a koncový bod krivky. Všimnime si, že koreň hierarchie neobsahuje žiadny riadiaci vrchol. Preto, ak cesta medzi listami vedie aj cez koreň jednoducho ho ignoruje. Vďaka tomu získajú takéto hrany väčšiu voľnosť a nebudú sa všetky snažiť tlačiť do stredu zobrazovacej plochy respektíve k jednému bodu. Druhým dôvodom je, že nie pre každý priestor respektíve plochu dokážeme určiť jej stred, príkladom je aj samotný povrch gule.



Obr. 29 – a) Dvadsaťsten, b) Osemsten, c) prerozdeľovanie na základe trojuholníkov spolu s vygenerovanou hierarchiou.

Na prerozdeľovanie guľového povrchu budeme musieť využiť aj iný geometrický útvar ako štvorec. Často sa na reprezentáciu gule využívajú pravidelné mnogosteny, konkrétnie osemsten a dvadsaťsten (obr. 29). Oba tieto objekty sú zložené z rovnostranných trojuholníkov a dokážeme zostrojiť guľu, ktorá sa bude dotýkať všetkých ich vrcholov. Vďaka tomu tieto objekty poskytujú vhodné riešenie, pretože každý rovnostranný trojuholník dokážeme rekurzívne rozdeliť na štyri menšie rovnako veľké rovnostranné trojuholníky. Všetky novovo vytvorené vrcholy trojuholníkov ešte potrebujeme posunúť tak aby sa tiež dotýkali gule. Toto posunutie má za následok mierne skreslenie tvaru trojuholníkov, môžeme ho však považovať za zanedbateľné, pretože má nepostrehnuteľný vplyv na výsledok.

Určenie trojuholníka v ktorom sa vrchol nachádza prevedieme jednoduchým testom. Vrchol spojíme so stredom gule čím vznikne úsečka, a pre každý z trojuholníkov testujeme, či priesečník danej úsečky a roviny, ktorú určujú jeho vrcholy leží v jeho vnútri alebo nie. Postupujeme od najväčších, smerom k najmenším, čím sa znižuje výpočtová náročnosť, pretože d'alej kontrolujeme už iba synov prečierného trojuholníka.



Obr. 30 – vľavo pohľad na krivky zhora, vpravo pohľad z boku s riadiacimi vrcholmi v rôznej výške nad povrhom gule.

Výhodou navrhovanej techniky je možnosť kontrolovať hĺbku hierarchie, čím môžeme určovať počet riadiacich vrcholov a stupeň detailov. Rovnako môžeme meniť aj vzdialenosť riadiacich vrcholov od stredu gule. Pre krivky využívajúce sférickú lineárnu interpoláciu vyžadujeme, aby všetky ležali na povrchu gule, pri klasických krivkách však nemusia. Môžeme teda riadiace vrcholy oddiaľiť alebo priblížiť k jej stredu, čím odkloníme krivky buď do vnútorného, alebo okolitého priestoru gule.

Výsledok samotného zviazania závisí od typu krivky, ktorý využijeme pri interpolácii medzi riadiacimi vrcholmi. Ako demonštruje obrázok 30, aplikáciou racionálnej bázierovej

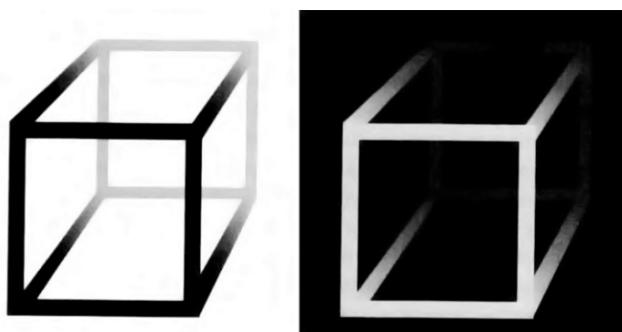
krivky, pri ktorej môžeme nastavovať váhu riadiacich vrcholov, dokážeme docieľiť väčšie a presnejšie zakrivenie vzhľadom na pozície riadiacich vrcholov, čo vo výsledku spôsobí užšie zväzky hrán, zaberajúce menej miesta. Obidva typy krviek aj s bázierovou krvkou so sférickou interpoláciou preto budú predmetom skúmania.

2.8. Zlepšenie vnímania 3D priestoru

Pre dosiahnutie lepšieho vnímania 3D priestoru a uvedomenia hĺbky, v akej sa objekty nachádzajú najmä v statickej vizualizácii, budú v aplikácii implementované tri rôzne techniky. Prvou z nich je takzvaný atmosférický vplyv, ktorý je založený na princípe, v ktorom intenzita farby a jasu objektov sa zo zväčšujúcou vzdialenosťou od pozorovateľa znižuje (obr. 31) [20]. Tento efekt umožní presnejšie vnímanie hĺbky najmä v statických obrázkoch, keď vizualizácia nebude v pohybe.

Druhou technikou bude stereo zobrazovanie pomocou techniky anaglyfu. Poskytuje pomerne slušné výsledky, okrem okuliarov nemá špeciálne nároky a keďže zobrazuje obidva obrazy do jedného statického obrázka, môžeme vytvoriť aj stereo fotografie vytvorené vizualizácie. Konkrétnie sa zameriame na anaglyf s využitím červeného a zeleno-modrého filtra, ktorý dokáže zobrazovať aj farebné obrazy. Tento typ farebného anaglyfu využíva princíp v ktorom finálny RGB obrázok obsahuje červený kanál ľavého obrázka zatiaľ čo modrý a zelený je extrahovaný z pravého obrázka, prípadne prevrátené, v závislosti od usporiadania filtrov v okuliaroch.

Tretou je posilnenie hĺbkového efektu úpravou veľkostí objektov. Objekty bližšie ku kamere budú zväčšené, zároveň čo vzdialenejšie zmenšené a obidve skreslenia sú nad rámec bežnej perspektívy, aby hĺbkový efekt bol výraznejší.



Obr. 31 – Atmosférický vplyv [20].

Kapitola 3

3. Implementácia

Obsahom tejto kapitoly je implementácia riešení prezentovaných v predošlej kapitole. Postupne popíšeme všetky časti navrhovanej techniky, algoritmické riešenia definovaných návrhov a rozhodnutia, ktoré pritom bolo potrebné učinit'. Objasníme aj vzniknuté problémy s nimi súvisiace a prístupy, ktoré viedli k ich odstráneniu.

3.1. Vytvorenie vrcholov a ich glyfov

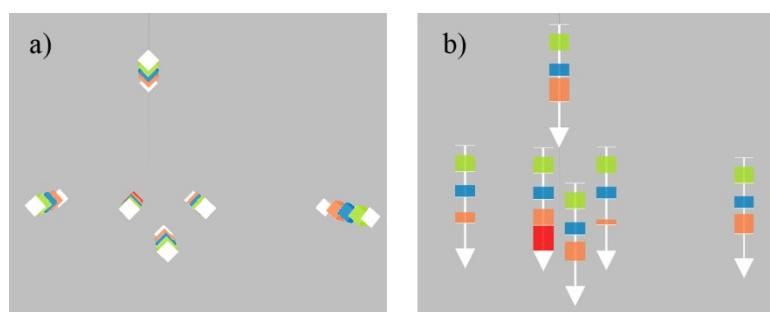
Po načítaní vstupného súboru sa ako prvé vytvoria vrcholy. Každý vrchol obsahuje informáciu o jeho sférických aj karteziánskych súradničach. Stred gule, na ktorej vrcholy ležia, je umiestnený v počiatku súradnicového systému, teda v bode [0,0,0]. Tento fakt nám zjednoduší niektoré výpočty čím sa zároveň zníži počet operácií, pretože karteziánske súradnice vrcholu môžeme automaticky využívať ako vektor smerujúci zo stredu gule k nemu. Vrcholy sa vytvárajú na náhodných, navzájom rôznych súradničach. To či vrcholy neležia veľmi blízko seba nemusíme riešiť, pretože o finálne rozloženie sa postará pružinový model.

Každý vrchol obsahuje aj informáciu o hodnotách definovaných atribútov. Po načítaní hodnoty atribútu sa automaticky vygeneruje jeho normalizovaná hodnota v intervale <0,1> na základe jeho hornej a dolnej hranice. Atribúty typu 1 a 5 (textové reťazce) dostávajú automaticky hodnotu 0 a hodnota atribút typu 2 (boolova premenná) už je vlastne normalizovaná takže ju iba prepíšeme. Týmto získame jednoduchý prístup k reálnym aj normalizovaným hodnotám, ktoré využijeme pri zostrojovaní glyfu. Atribúty typu 1 a 5 sa teda textové reťazce a obrázky sa vizualizujú samostatne pomocou techniky „billboardov“, čo znamená, že objekty sú vykreslené stále tak aby boli natočené čelom ku kamere. Okrem glyfu je teda vedľa vrcholov možné zobrazovať aj text a obrázky.

Počet atribútov, ktoré môžeme naniest' na glyf určuje globálna premenná GLYF_NUM_FLOORS. Jej hodnotu je možné nastavovať v intervale od 1 do 10. Pri viac ako 10 atribútoch by už glyf začal strácať svoj význam. Jeho telo je rovnomerne rozdelené na

„podlažia“, ktorých výšku určuje GLYF_FLOOR_HEIGHT a každé je vyplnené do výšky na základe normalizovanej hodnoty atribútu ktorý reprezentuje. Pre každé podlažie je možné určiť atribút, ktorého hodnotou sa vyplní a zároveň aj jeho farbu. Medzi podlažiami sa nachádza tenká stena, ktorá označuje koniec jedného podlažia a začiatok druhého. Využitím tohto stropu zabezpečíme presnejšie vnímanie hodnôt atribútov (viď. kapitola 2.5). Iba strop však nastačil, pretože ak podlažie nebolo vyplnené v celej svojej výške, tak glyf bol nespojity. Z tohto dôvodu sme potrebovali nejako prepojiť podlahu so stropom. Jedným riešením bolo podopriť strop v každom rohu pomocou stĺpu. Tento spôsob však zaberá veľa grafickej informácie a zároveň často stĺp zacláňa pri pozorovaní hodnoty atribútu v danom podlaží. Riešením preto bolo využitie iba jedného stĺpu, ktorý sa nachádza v strede. Tým pádom už nedokáže tento stĺp zacláňať, práve naopak, samotný stĺp je čiastočne zakrytý do určitej výšky podľa hodnoty atribútu.

Hotový glyf je uložený do display listu a je transformovaný do pozície vrcholu na základe jeho sférických súradníc. Je teda v danom bode kolmo orientovaný vzhľadom na povrch gule. Vo väčšine prípadov je čitateľnosť glyfov v spojení s interakciou dobrá, t'ážkosti však nastávajú ak chceme vizualizáciu pozorovať z väčšej blízkosti, kedy sa v pozorovanej oblasti zníži zakrivenie povrchu gule, a v podstate sa pozérame na glyfy priamo zhora (obr. 32a). V tomto prípade je preto možné vytvoriť iba dvojrozmerné verzie glyfu, ktoré zobrazujeme pomocou techniky „billboardov“, podobne ako pri obrázkoch a texte (obr. 32b). Užívateľ má preto na výber medzi 2D a 3D glyfom, a kedykoľvek sa môže medzi nimi prepínat. Na obrázku 32, môžeme pozorovať zlepšenie situácie využitím 2D verzie glyfu.



Obr. 32 – a) pohľad na 3D glyf zhora, b) 2D glyf.

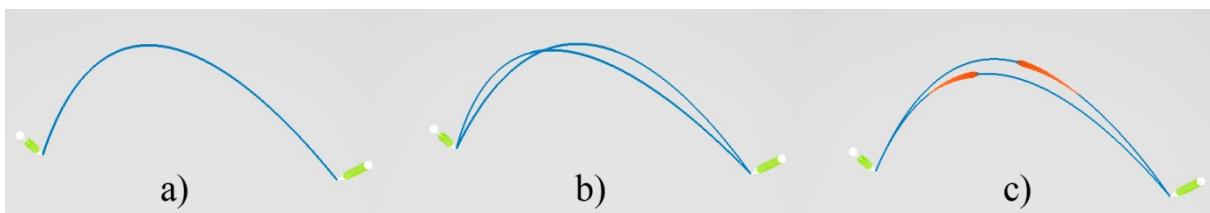
Glyf ma okrem iného aj nastaviteľnú šírku. To však môže v hustejších vizualizáciach s väčším množstvom vrcholov k neschopnosti rozlišovať, ktorý glyf patrí ku ktorému vrcholu. Preto je glyf umiestnený na prevrátený kužeľ, ktorého špička smeruje priamo do vrcholu ku ktorému je určený. V prípade 2D glyfov a obrázkov je kužeľ nahradený trojuholníkom.

3.2. Vytvorenie hierarchie a priradenie riadiacich vrcholov

Ako už bolo spomínané, pre vytvorenie hierarchie riadiacich vrcholov využívame dvadsaťsten a osemsten. Dvadsaťsten sa skladá z 20 trojuholníkov preto má koreň až 20 synov. Pri využití osemstenu je ich iba 8. Hierarchia je uložená vo forme spájaného zoznamu, v ktorom každý trojuholník, ak nie je listom, odkazuje smerníkmi na 4 ďalšie trojuholníky, ktoré vznikli jeho rozdelením. Zároveň obsahuje aj informáciu o jeho vrcholoch, stredovom a svojej hĺbke v hierarchii. Hĺbku hierarchie dokážeme kontrolovať pomocou premennej `tesslvl` (tessellation level), ktorá hovorí o tom, koľko krát sa má každý z trojuholníkov v prvej úrovni hierarchie rekurzívne rozdeliť. Ak je `tesslvl` 1, tak všetky počiatočné trojuholníky sa rozdelia na 4 ďalšie.

Ďalším faktorom, ktorý ovplyvňuje dosiahnutý výsledok zväzovania, je nastavovanie výšky riadiacich vrcholov nachádzajúcich sa na rovnakej úrovni v hierarchii. Pod výškou teraz rozumieme ich vzdialenosť od stredu gule. Pre každú úroveň hierarchie si preto ukladáme túto informáciu, a môžeme ju ľubovoľne meniť. Na začiatku sa všetky úrovne nachádzajú v rovnakej výške dĺžky polomeru gule, teda ležia na jej povrchu. Ich posunutím do vnútra gule dokážeme teda krivky odkloniť od jej povrchu smerom do vnútra. Rovnako ich však môžeme vysunúť ďalej, tak aby sa nachádzali mimo gule. Tým pádom budú krivky odklonené opačným smerom, čo môže byť užitočné ak vizualizácia obsahuje väčšie množstvo vrcholov a glyfy nám zacláňajú vo výhľade. V tomto prípade však nastáva problém, pretože nie všetky hrany sa budú nachádzať úplne mimo gule. Stále môžeme nájsť také, ktorých riadiace vrcholy nútia krivku pretínať guľu. Príkladom môže byť hrana, ktorej dva riadiace vrcholy sú antipódne body, teda nech ich vzdialime akokoľvek ďaleko ich spojnica vždy vedie stredom gule. Nevyhnutným riešením je preto pridať na začiatok hierarchie ďalšie riadiace vrcholy, ktoré budú pomáhať premostiť riadiace vrcholy v podobných prípadoch. Priestor preto rovinou, ktorá delí guľu na severnú a južnú pologuľu, rozdelíme na dva pol priestory a ako riadiace vrcholy budeme využívať póly gule. Ich vysunutím do vyšej výšky vznikne dobré premostenie dvoch riadiacich vrcholov ležiacich na odvrátených stranach gule. Spravíme ich preto synmi koreňa, a jeho pôvodných synov priradíme pod póly, na základe toho v ktorom polpriestore sa nachádzajú. Týmto by sme však docielili úplné odstránenie problému, iba v prípade ak cesta medzi vrcholmi v hierarchii vedie len cez jeden z pólov. Ak by cesta viedla cez oba póly opäť sa dostaneme k tomu istému problému. Preto sa vrátime k pôvodnej hierarchii a modifikujeme ju iným spôsobom v ktorom do koreňa hierarchie pridáme nasledujúcu podmienku, ktorá sa aplikuje v momente keď ním prechádza cesta medzi

vrcholmi. Ak teda taký prípad nastane, koreň sa pozrie na oboch svojich synov, ktorí sú súčasťou cesty a na základe ich pozícií sa rozhodne aký/aké kontrolné vrcholy budú pridané. Ak obaja patria do rovnakého pol priestoru, vloží sa do cesty iba jeho riadiaci vrchol. Táto časť podmienky simuluje presne predošlú modifikáciu hierarchie s pridaním pólom. Ak však ležia v opačných pol priestoroch, pridanie iba jedného riadiaceho vrcholu nebude stačiť. Vytvoríme preto 4 ďalšie body, ktoré budú ležať v rovine oddelujúcej dané pol priestory, rovnomerne rozmiestnené v celom 360 stupňovom okolí gule. Tieto nám pomôžu lepšie premostiť hrany smerujúce medzi pol priestormi. Spolu s pólmi ich využijeme nasledovne. Pred syna koreňa, v ktorého pod strome sa nachádza prvý vrchol hrany, pridáme ako ďalší riadiaci vrchol pól na základe pol priestoru v ktorom leží. Pred druhého syna, v ktorého pod strome sa nachádza druhý vrchol hrany, priradíme zase jeden zo 4 bodov ležiacich v deliacej rovni a to ten ktorý je k nemu najbližšie. Pridanie týchto dvoch extra bodov, by malo zaručiť aby už žiadne krvky nepretínali guľu. Nesmieme však zabúdať na to, že riadiace vrcholy musia mať určitú minimálnu výšku aby bolo odklonenie dostatočne veľké.



Obr. 33 – Zobrazovanie dvoch rovnakých navzájom opačných hrán: a) rovnaké riadiace vrcholy, b) náhodne posunuté riadiace vrcholy, c) zobrazenie orientácie s využitím hrúbky a farby gradientu.

Všeobecným problémom vytvárania hrán na základe riadiacich vrcholov v danej hierarchii je, že nijako nezohľadňuje orientáciu hrán. Ak medzi dvojicou vrcholov existuje viac ako jedna hrana, nezávisle od ich smeru, všetkým krvkám budú priradené rovnaké riadiace vrcholy a teda budú všetky ležať v sebe (obr. 33a). Nie je preto možné rozlísiť koľko rôznych hrán sa medzi vrcholmi nachádza. Riešením je riadiace vrcholy jemne odkloniť v náhodnom smere z ich pôvodnej pozície, samozrejme okrem pozície samotných vrcholov medzi ktorými sa hrana nachádza. Po vygenerovaní riadiacich vrcholov pre výslednú krvku aplikujeme na každý z nich posunutie o malú konštantnú vzdialenosť v náhodne vygenerovanom smere. Tým sa zabezpečí, aby dve hrany, ktoré prechádzajú rovnakými riadiacimi vrcholmi, nebudú úplne rovnaké, teda nebudú navzájom v sebe ležať (obr. 33b). Posunutie je natoľko malé, že neovplyvní efektívnosť a celkový výsledok zväzovania.

3.3. Zobrazovanie hrán

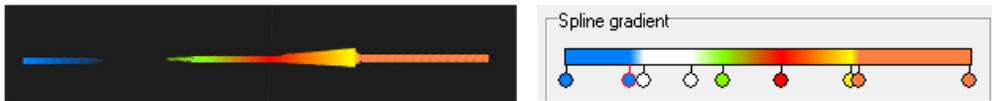
Po vytvorení riadiacich vrcholov krivky, sa môžeme venovať samotnému zobrazovaniu hrán. Okrem grafických atribútov čiar, môžeme využiť aj rôzne typy kriviek pre dosiahnutie odlišných výsledkov. Aplikácia preto obsahuje 3 rôzne typy bázierových kriviek: štandardnú, sférickú a racionálnu (váhovanú).

Sférická bázierová sa líši od štandardnej nahradením lineárnej interpolácie za sférickú lineárnu interpoláciu a vyžaduje, aby všetky riadiace vrcholy ležali na povrchu rovnakej gule. Algoritmus funguje na rovnakom princípe, ako pri štandardných krivkách, nevyužíva však pri interpolácii rovné spojnice medzi riadiacimi vrcholmi, ale najkratšie kružnicové oblúky, ktoré ich spájajú. Preto v prípade ak sa nachádzajú riadiace vrcholy v rôznych výškach, sú automaticky premiestnené tak, aby sa dotýkali gule. Problém nastáva v prípade, ak dva za sebou idúce riadiace vrcholy sú antipódne body. Vtedy algoritmus nedokáže určiť, ktorý je najkratší kružnicový oblúk medzi nimi, lebo ich dokážeme zstrojiť nekonečne veľa. Je preto nevyhnutné pridať medzi ne ďalší riadiaci vrchol. V princípe môžeme zvoliť úplne ktorýkoľvek bod ležiaci na povrchu gule okrem daných dvoch riadiacich vrcholov. S využitím malého náhodného posunutia riadiacich vrcholov (perturbácie ich súradníc) však tento prípad nastane iba veľmi zriedkavo.

Racionálna bázierová krivka sa odlišuje možnosťou zadávať riadiacim vrcholom rôznu váhu. Váha vrcholu určuje aký veľký vplyv bude mať na zakrivenie krivky, respektíve nakoľko sa k nemu krivka bude snažiť priblížiť. Ak sú váhy všetkých riadiacich vrcholov rovnaké, tvar krivky je totožný s klasickou bázierovou krivkou. Zvyšovaním váh určitých vrcholov dokážeme docieliť výraznejšie zakrivenie a priblíženie k riadiacim vrcholom, aké nedokáže poskytnúť štandardná krivka. Racionálna bázierová krivka vďaka tomu vytvára užšie a presnejšie zväzky hrán, zaberajúce menej priestoru.

Pre získanie schopnosti odlišovať rôzne typy hrán, môžeme každému typu nastaviť iný druh krivky, respektíve rovnakú krivku s rôznymi parametrami. Hlavným prvkom odlíšenia typov je však využitie farby, priehľadnosti a hrúbky čiar. Aby bola schopnosť ovládať tieto grafické atribúty čo najlepšia, aplikácia obsahuje farebný prechod - gradient, medzi ktorého riadiacimi bodmi lineárne interpolujeme všetky tieto nastavenia. Body, ktorými gradient definujeme, môžeme ľubovoľne pridávať/mazať a upravovať. Každý z týchto bodov nesie informáciu o farbe, priehľadnosti a hrúbke čiary. Vďaka tomu dokážeme vytvárať množstvo kombinácií nastavení všetkých parametrov na dosiahnutie želaného výsledku (obr. 34). Orientované hrany tak nemusia byť vizualizované iba pomocou farebného prechodu, ale aj

hrúbkou, kedy sa hrana môže postupne v jednom smere zužovať (vid' obr. 34 – žltý riadiaci bod s vyššou hrúbkou). Nastavením priehľadnosti zase môžeme vytvoriť napríklad prerušované čiary (vid'. obr.34 – transparentné biele riadiace body).



Obr. 34 – Vpravo gradient, vľavo odpovedajúci výsledok v aplikácii.

Ďalším prvkom, ktorý pomáha odhaliť smer hrán, je animovanie celého gradientu so všetkými jeho nastaveniami. Na gradiente môžeme vhodným zadaním a nastavením riadiacich bodov vytvoriť napríklad šípku v smere orientácie hrany. Následnou animáciu gradientu tak šípka cestuje po celej dĺžke hrany a intuitívnym spôsobom pomáha odhaľovať jej smer a zároveň aj počiatočnú a finálnu destináciu. Ak však v niektorom vrchole končí a zároveň aj začína hrana rovnakého typu (teda má rovnaké grafické atribúty), pohyb šípky zanikajúcej na konci jednej hrany, a zároveň objavujúcej sa šípky na začiatku druhej hrany, človek vníma ako plynulý prechod a zdá sa, že šípka precestovala z jednej hrany na druhú. Tento efekt tak sprostredkúva klamlivú informáciu. Z tohto dôvodu sme náhodne poposúvali fázy pohybu, aby sa nestávalo, že animácia hrán rovnakého typu začína a končí v rovnakom momente. Asynchronný pohyb však nepriniesol zlepšenie, práve naopak, pre veľký počet rôznych pohybov nebolo možné sústredit sa ani na jednotlivé hrany ani na viaceré hrán. Ukázalo sa, že sledovanie pohybu viacerých je oveľa jednoduchšie, ak ich animácie sú zosynchronizované.

3.4. Pružinový model a jeho modifikácia

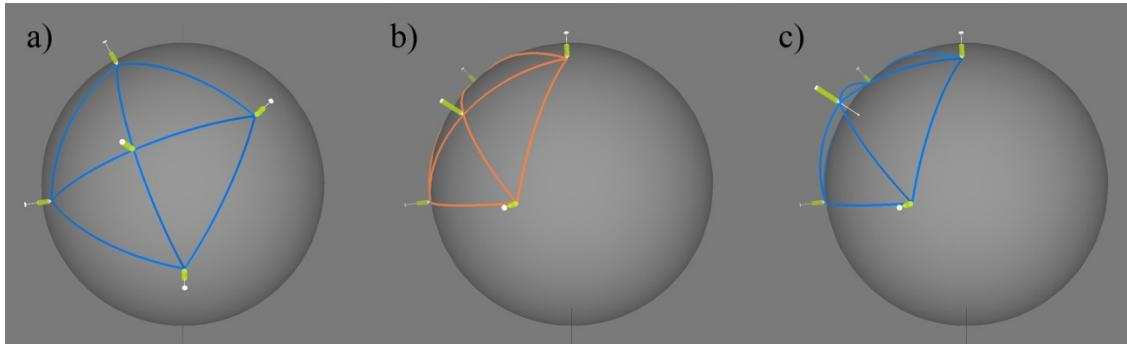
Spôsob prevodu vrcholov z povrchu gule do dotykovej roviny a ich spätné premietnutie, je podrobne vysvetlené v kapitole 2.3, preto sa budeme venovať už iba konkrétnemu definovaniu pôsobiacich síl a najmä modifikácií tohto modelu pridaním vysúvania vrcholov nad povrch gule. Sily v systéme sú modelované tak ako ich vo svojom modeli definoval Eades (vid' kapitolu 1.3.1). Teda medzi vrcholmi prepojenými hranou je umiestnená logaritmická pružina, ktorej silu počítame na základe Hookovho zákona, a odpudivé sily medzi vrcholmi nespojenými hranou určujeme podľa Coulombovho zákona. Vzhľadom k tomu, že veľkosť guľového povrchu je konečná, bolo potrebné určiť maximálnu ideálnu dĺžku pružiny na základe počtu vrcholov v grafe. Na jej výpočet sme využili formulu

definovanú vo Fruchterman – Reingold algoritme, $k = c * \sqrt{S/n}$. Kde k predstavuje ideálnu dĺžku pružiny, n označuje počet vrcholov a S veľkosť plochy. V našom prípade je S veľkosť guľového povrchu, čiže $S = 4\pi r^2$. V aplikácii je k reprezentované ako `maxSpringLength`. Dĺžka hrán na vstupe môže byť rôzna, je však normalizovaná na hodnoty v intervale $(0,1)$. Ich reálna dĺžka sa preto určuje nasledovne. Ak dĺžka hrany na vstupe je x tak jej skutočná dĺžka je určená ako `maxSpringLength * x`, teda `maxSpringLength` určuje maximálnu reálnu dĺžku pružiny. Aktuálnu dĺžku pružiny medzi dvoma vrcholmi určuje najkratší kružnicový oblúk medzi nimi. Dĺžku kroku kontroluje konštantá `timeStep` a postupné tlmenie pôsobiacej sily zabezpečuje konštantá `damping`, inak by systém nikdy nedosiahol rovnovážny stav.

Teraz prejdeme k modifikácii v ktorej vrcholom dovolíme vystúpiť aj nad povrch gule. Aby sme dokázali rozhodnúť, ktorý z vrcholov má byť vysunutý, každý z nich nesie informáciu o napätí (`stress`). Napätie v danom vrchole vyjadruje, aký veľký tlak naňho vyvíjajú s ním spojené pružiny. Pod napäťom v práci vyjadrujeme iba odpudivo pôsobiace sily pružín, teda v prípade ak je vzdialenosť medzi vrcholmi spojenými hranou menšia, ako v skutočnosti má byť. Veľkosť napäťia vo vrchole počítame nasledovne. Pre každú hranu, ktorá je s ním spojená, zistíme, aká je jej aktuálna dĺžka (`actualEdgeVal[i][j]`) a porovnáme ju s jej ideálnou dĺžkou definovanou na vstupe (`constEdgeVal[i][j]`). Ak je aktuálna dĺžka menšia ako ideálna, tak je veľkosť ich rozdielu zarátaná do napäťia vrcholu. Tým získame hodnoty napäťia v každom vrchole a využijeme ich pri rozhodovaní, ktoré vrcholy majú byť vysunuté a do akej vzdialenosťi, nasledovne. Opäťovne porovnáme všetky vrcholy, ktoré sú spojené hranou, ktorá spôsobila nárast napäťia v oboch vrcholoch. Aby sme dosiahli uvoľnenie napäťia medzi nimi vysunieme iba jeden z nich a vyberieme, ten ktorého `stress` je vyšší. Hodnotu napäťia, ktorú daná hrana spôsobuje uložíme do `stressTop` vybraného vrcholu, ak už tam nie je vyššie číslo a zároveň o ňu znížime `stress` druhého vrcholu. Tým zabezpečíme, že napätie každej hrany bude zohľadnené vždy iba v jednom z jej vrcholov. Po ukončení cyklu tak každý vrchol, ktorého `stress` je väčší ako 0 vysunieme do výšky na základe jeho hodnoty v `stressTop`.

Ked'že v premennej `stressTop` každého vrcholu ukladáme maximálnu hodnotu napäťia spôsobeného jednou sním spojenou hranou, po prejdení všetkých sa v nej nachádza údaj o koľko ho musíme vysunúť aby všetky tieto hrany mali aspoň optimálnu dĺžku. Dĺžku hrany po vysunutí vrcholov tak môžeme vnímať ako dĺžka kružnicového oblúka medzi nimi

(`actualEdgeVal[i][j]`), plus vzdialenosť o ktorú boli vysunuté (`i.stressTop`, `j.stressTop`). Na obrázku 35a/35b môžeme pozorovať situáciu, v ktorej stredový vrchol je tiesnený zo všetkých strán. Všetky hrany v grafe majú rovnakú ideálnu dĺžku, hrany sním spojené sú však v tomto prípade kratšie. Premietnutie vzniknutého napäťia do jeho výšky sa nachádza na obrázku 35c.



Obr. 35 – a) pôvodné rozmiestnenie grafu, b) pôvodné rozmiestnenie z iného pohľadu, c) vysunutie vrcholov na základe ich napäťia.

Samotná výška sa však nepremieta do výpočtového modelu síl, ten stále zohľadňujeme iba aktuálnu vzdialenosť vrcholov. Výpočet výškového vysunutia sa preto opakuje po každej iterácii. Tento spôsob teda nie je fyzikálne korektný, avšak v práci sme sa snažili hlavne o vytvorenie prototypu na báze empirického modelu, aby sme vedeli rozhodnúť či navrhovaná modifikácia má potenciál a kladne ovplyvňuje výsledok rozloženia.

3.5. Ovládanie aplikácie

Pracovnú plochu aplikácie tvorí zobrazovacie okno, napravo od ktorého sa nachádza hlavné ovládacie menu rozdelené na piatich samostatných kartách. Každá z nich slúži na ovládanie špecifickej časti vizualizácie (viď. Príloha č.3).

3.5.1. Myš a klávesnica

Myšom je možné vizualizáciu otáčať (potrebné držať ľavé tlačidlo myši) a približovať (potrebné držať pravé tlačidlo myši). Pri držaní klávesy „ctrl“ na klávesnici, môžeme ľavým tlačidlom myši posúvať pozície vrcholov na povrchu gule, a pravým tlačidlom ich označovať. Pri označení vrcholu/vrcholov sa automaticky všetky ostatné hrany, ktoré s ním/nimi nie sú spojené zobrazia šedou farbou. Hodnoty atribútov posledného označeného

vrcholu môžeme pozorovať na karte „Nodes“ v časti definovania glyfu. Pre zrušenie označenia vrcholov slúži kláves „z“.

3.5.2. Hlavné nastavenia – karta „Main“

Karta obsahuje hlavné grafické nastavenia, ktoré však priamo neovplyvňujú atribúty samotnej vizualizácie. V hornej časti sa nachádza tlačidlo „Load new file“, po ktorého stlačení je možné vybrať textový súbor v ktorom sa nachádza uložená vizualizácia. Pre samotnú vizualizáciu môžeme nastavovať farbu pozadia na akom sa bude zobrazovať (Background), ďalej môžeme povoliť/zakázať zobrazenie gule na ktorej budú rozložené vrcholy grafu (Sphere) a rovnako môžeme meniť aj jej farbu. Jej zobrazovanie je vhodné v prípadoch keď krivky ležia priamo na povrchu alebo smerujú mimo nej. Nachádza sa tu aj nastavenie stereo zobrazovania (Stereo view), pri ktorom môžeme okrem zapnutia meniť aj ohniskovú vzdialenosť. Implicitne je ohniskový bod umiestnený na povrch gule. Je však možné pre dosiahnutie lepšieho 3D efektu, posunúť ho bližšie alebo ďalej. Ďalej môžeme meniť veľkosť zorného uhla (Field of view) a povoliť/zakázať atmosférický jav, na ktorého dosiahnutie využívame OpenGL hmlu (Fog). Počiatočnú/ konečnú hranicu intervalu určujúcu 0% / 100% hustotu hmly je možné približovať a vzdialovať, pre dosiahnutie silnejšieho alebo slabšieho atmosférického vplyvu.

3.5.3. Nastavenia vrcholov – karta „Nodes“

Hlavnou časťou týchto nastavení je konfigurácia glyfu (Glyph). Počet atribútov, ktoré glyf zobrazuje kontrolujeme pomocou „Number of floors“. Zároveň môžeme nastavovať aj jeho výšku („Height“), šírku („Width“) a jeho tvar na základe počtu strán („Number of sides“). V prípade ak je počet strán 2, zobrazí sa 2D verzia glyfu, ktorá je natočená smerom ku kamere. V časti „Edit floor“ môžeme pre každé podlažie glyfu („Floor“) definovať, ktorý z vrchových atribútov („Attribute“) bude reprezentovať a zároveň aj jeho farbu („Color“). Vedľa týchto nastavení sa nachádza aj obrázok, ktorý zobrazuje aktuálne nastavenie glyfu. Môžeme teda pozorovať, ktorý z atribútov sa v glyfe už nachádza, aká je jeho farba a pozícia. Vybrať môžeme iba z atribútov typu 2,3,4 (číselné hodnoty a booleove premenné). Pre zobrazenie atribútov typu 1 (textových reťazcov) slúži sekcia „Caption“. Pre ich zobrazovanie je potrebné zaškrtnúť „Display caption“. Samotnému textu môžeme nastaviť veľkosť fontu („Font size“), farbu („Color“) a priehľadnosť („Opacity“). Podobne ako textové reťazce sa osobitne zobrazujú aj atribúty typu 5, čiže obrázky v časti „Image“.

3.5.4. Nastavenia hrán – karta „Edges“

Na tejto karte sa nachádzajú nastavenia grafických atribútov hrán. Prvým globálnym nastavením je počet segmentov krivky („Number of spline segments“) a aplikuje sa pre úplne všetky typy hrán. Implicitne je nastavený na 50, avšak pri racionálnej bézierovej krivke bude vo väčšine prípadov potrebné zvýšiť ich počet, pre dosiahnutie lepsieho výsledku. Vzhľadom k tomu, že krivky v niektorých prípadoch smerujú okolo gule, môžeme určiť či sa majú napájať ku glyfu na spodku („Connect at: bottom“) alebo na vrchu („Connect at: top“), čím zvýšime celkovú prehľadnosť. V zvyšku karty sa už nachádzajú nastavenia pre vybraný typ hrán. „Edge type“ slúži na výber typu hrany, ktorého grafické atribúty chceme upravovať. Všetky ďalšie nastavenia už potom ovplyvňujú iba tento hrany tohto typu. „Spline type“ určuje aký typ interpolácie sa využije pri tvorbe kriviek. Celkovo je na výber z 5 rôznych možností:

- „Bézier“ – štandardná bézierova krivka
- „Spherical bézier“ – sférická bézierová krivka
- „Rational bézier“ – racionálna bézierova krivka
- „Arc from Node to Node“ – kružnicový oblúk priamo medzi vrcholmi grafu
- „Straight line from Node to Node“ – rovná čiara spájajúca priamo dva vrcholy grafu

„Adjust“ slúži iba na ovládanie váh riadiacich vrcholov v prípade výberu racionálnej bézierovej krivky. Hlavné nastavenia grafických atribútov hrany sa nachádzajú v časti „Spline gradient“. Každý gradient je určený minimálne dvoma krajnými riadiacimi bodmi. Pravým kliknutím do oblasti gradientu sa v tomto mieste vytvorí nový riadiaci bod. Okrem krajných riadiacich bodoch, ich môžeme ľubovoľne posúvať a meniť poradie. Ich celkový počet je obmedzený na 50. „Gradient control points attributes“ obsahuje nastavenia označeného riadiaceho bodu gradientu. „Color“ umožňuje nastaviť jeho farbu, „Opacity“ priečenosť, „Line width“ hrúbku čiary. Tlačidlo „delete“ slúži na vymazanie označeného bodu. V časti „Spline animation“ môžeme zapnúť animovanie gradientu („Animate color“, „Animate line width“) a nastaviť aj rýchlosť animácie („speed“). Je potrebné spomenúť, že rýchlosť animácie je ovplyvnená aj počtom segmentov krivky.

3.5.5. Nastavenia hierarchie – karta „Hierarchy“

Na tejto karte sa nachádzajú nastavenia ovplyvňujúce hierarchiu riadiacich vrcholov. Hĺbku hierarchie, teda počet rekurzívnych rozdelení trojuholníkov určuje „Levels“. Nula znamená, že žiadny z počiatočných trojuholníkov nie je už rozdelený na menšie, a hierarchia má vtedy iba jednu úroveň. „Type“ určuje, ktorý z útvarov bude použitý pre vytvorenie hierarchie. Riadiace vrcholy sú implicitne tvorené stredmi trojuholníkov, v „Control points“ však môžeme toto nastavenie zmeniť, aby sa za riadiaci vrchol vybral jeden z vrcholov trojuholníka, ktorý je najbližšie k danému vrcholu grafu. To poskytuje možnosť vytvoriť ďalšie variácie hierarchií. Výšku riadiacich vrcholov ovládame v „Height of control points“. „Level“ určuje úroveň riadiacich vrcholov v hierarchii, ktorých výšku modifikujeme pomocou posúvača („Height“). Na presmerovanie kriviek aby išli mimo gule je potrebné zaškrtnúť políčko v časti „Extra control points“. Rovnako je možné upravovať aj výšku týchto dodatočných riadiacich vrcholov. Veľkosť náhodného posunutia riadiacich vrcholov ovláda posúvač v „Randomize positions“. Jeho posunutím úplne na začiatok sa žiadne posunie neaplikuje.

3.5.6. Ovládanie pružinového modelu – karta „Spring model“

Na začiatku po načítaní vstupného súboru je potrebné pružinový model spustiť tlačidlom „Start“. Kedykoľvek ho môžeme zastaviť tlačidlom „Stop“. Ak chceme vytvoriť nové rozloženie vrcholov, tlačidlom „New layout“ sa všetky vrcholy opäťovne náhodne rozmiestnia po povrchu. Tlačidlo „Next iteration“ dovoľuje sledovať účinok pružinového modelu po samostatných iteráciách. Každým stlačením prebehne jedna iterácia výpočtu. Vlastnosti modelu je možné meniť v časti „Layout attributes“. Aj keď výpočet maximálnej dĺžky pružiny je automatizovaný, aplikácia dovoľuje ovplyvňovať jej dĺžku pre dosiahnutie uspokojivejších výsledkov. Posúvač „Max spring lenght“ umožňuje upraviť jej dĺžku v rozmedzí 10% až 200% pôvodnej dĺžky. Veľkosť príťažlivých a odpudivých síl v systéme kontrolujú posúvače „Attractive force“ a „Repulsive force“. Dôležitým atribútom pružín je ich tuhosť („Damping“), ktorá určuje ako rýchlo dochádza k tlmeniu ich energie. Znížením tejto hodnoty, budú pružiny strácať svoju energiu oveľa rýchlejší. Využitie nami definovaného stresu v rozložení vrcholov sa povoľuje v časti „Use stress“. Ak chceme aby boli vrcholy premiestnené výšky na základe ich stresovej hodnoty je potrebné zvoliť možnosť „Yes – displace nodes“. Druhá možnosť „Yes – display stress“ dovoľuje hodnoty pozorovať, bez toho aby sa menila výšková pozícia vrcholov.

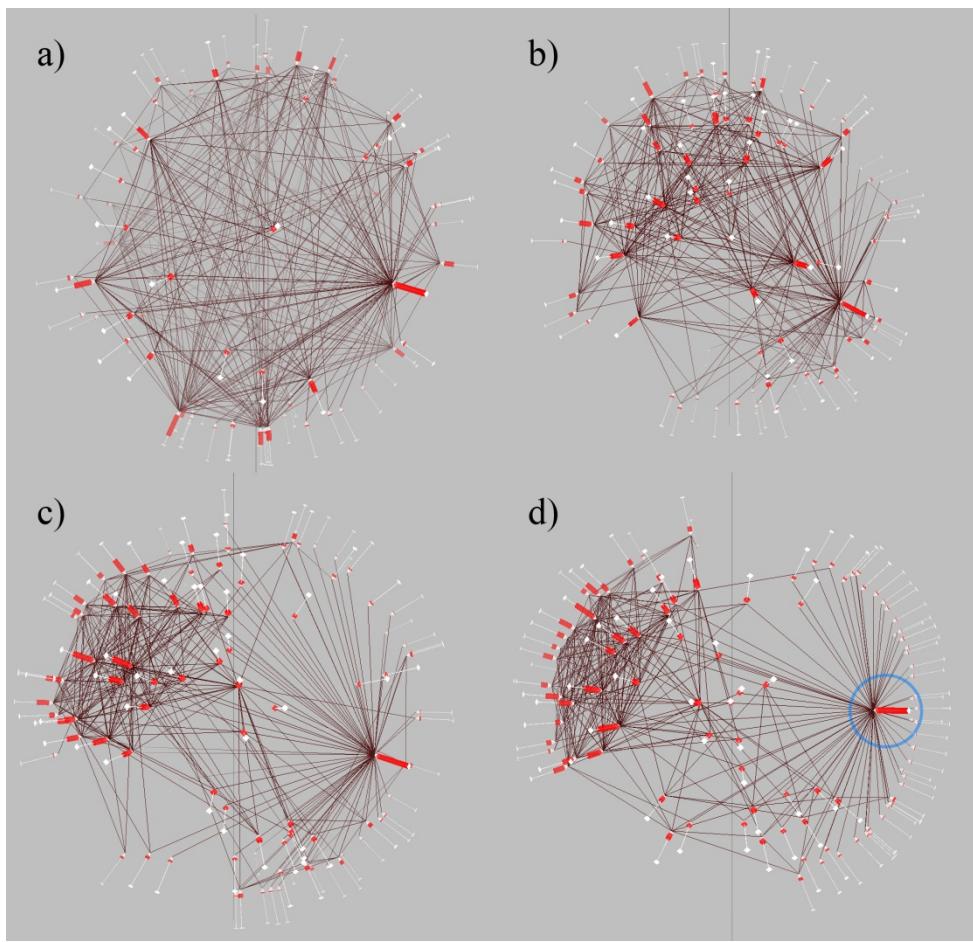
Kapitola 4

4. Demonštrácia

Reálne výsledky navrhnutej techniky sme testovali na dátovej množine emailovej komunikácie americkej energetickej spoločnosti Enron, ktorá patrila medzi popredné svetové spoločnosti v oblasti dodávky energie. Z dôvodu odhalenia finančných podvodov však v roku 2002 vyhlásila bankrot. Počas vyšetrovania Federal Energy Regulatory Commission zverejnila emailovú komunikáciu spoločnosti pre verejnosť. Dáta obsahujú komunikáciu zhruba medzi pol miliónom emailových adres.

Použité dáta, ktoré sme konvertovali na náš vstup, sú definované formou siete, v ktorej vrcholy reprezentujú emailové adresy. Ak medzi dvoma adresami prebehla emailová komunikácia, nachádza sa medzi nimi neorientovaná hrana. Konkrétnie názvy emailových adres neboli súčasťou dát, preto boli vygenerované vo forme „index_vrcholu@enron.com“ a definované ako atribút vrcholu typu 1 s názvom email. Zároveň každý vrchol obsahuje informáciu, s ktorou komunikoval (exchanged_emails), ktorú budeme nanášať na glyf.

V prvom príklade na obrázku 36 sa nachádza ukážka rozmiestnenia vrcholov pomocou pružinového modelu. Vizualizácia zobrazuje prvých 100 emailových adres a ich vzájomnú komunikáciu. Celkovo sa štruktúre nachádza 324 hrán. Horná hranica atribútu exchanged_emails (v glyfe reprezentovaný červenou farbou) je nastavená na 50 a hrany sú zobrazené priamym spojením vrcholov. V prvej fáze sú vrcholy náhodne rozmiestnené (obr.36a). Vidíme, že v tomto štádiu je vizualizácia neprehľadná, aj keď už teraz, pomocou glyfov vieme ľahko povedať, s ktorými adresami prebiehala najväčšia komunikácia. Avšak, až po aplikácii pružinového algoritmu sú odhalené kľúčové informácie (obr. 36d). V konečnej konfigurácii sa v pravej časti separoval od zhluku vrchol s najväčšou hodnotou atribútu exchanged_emails (vyznačený modrým krúžkom). Konkrétnie ide o emailovú adresu „2@enron.com“. Zároveň sa okolo neho rozmiestnila skupina vrcholov, ktoré sú spojené hranou iba s ním.

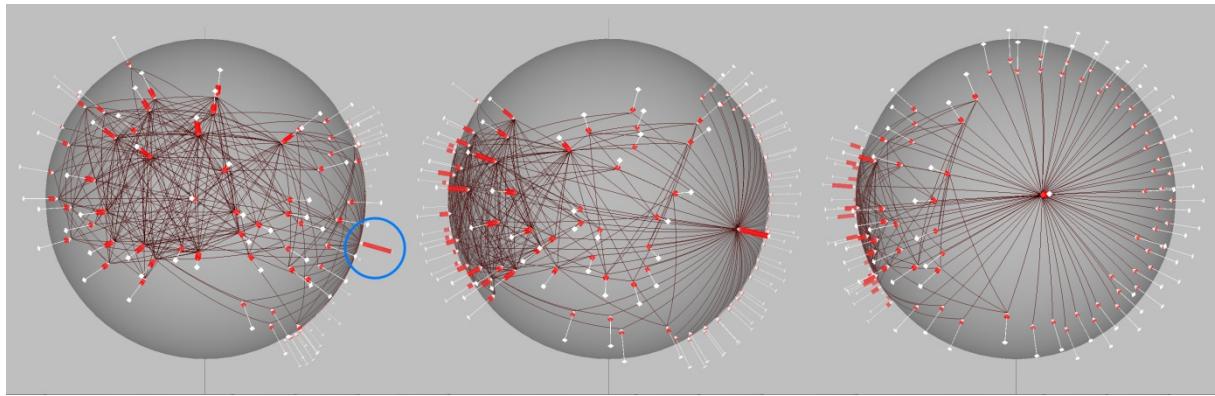


Obr. 36 – Fázy pružinového modelu: a) počiatočná konfigurácia, b),c) medzifázy, d) konečná konfigurácia.

Z vizualizácie tak vieme vyvodiť, že z daných 100 adres, veľká skupina výlučne komunikovala s adresou „2@enron.com“ a zároveň táto adresa komunikovala s najväčším počtom rôznych adres, ktorých bolo aspoň 50, na základe výšky hodnoty atribútu `exchanged_emails` v glyfe.

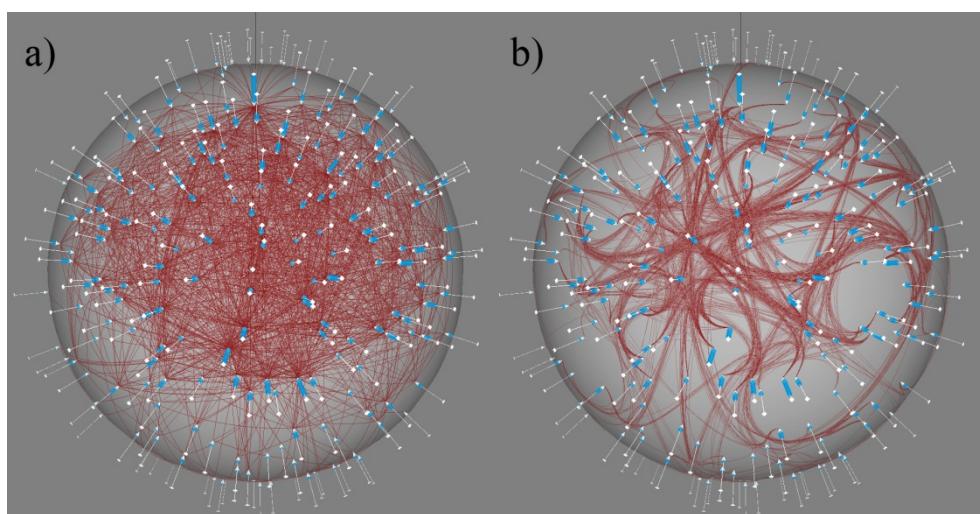
Obrázok 37 zobrazuje rovnaký vstupný súbor, hrany sú však reprezentované kružnicovými oblúkmi a zobrazujeme aj guľu, na ktorej sú vrcholy rozložené. Obsahuje tri rôzne pohľady na vizualizáciu s rôznym natočením gule. Obrázok demonštruje ako zakrivenie guľového povrchu pomáha zobrazovať informácie v centre pozornosti s vyšším detailom a zároveň zachováva prehľad aj o informácii v okrajových častiach. V prvom pohľade (ľavý obrázok) sledujeme zhluk vrcholov a hrán. Aj napriek neprehľadnej situácii, vďaka tomu, že sa zhluk nachádza v centre pohľadu, dokážeme rozlišovať jednotlivé hrany. Postupne ako guľu otáčame, pozorujeme ako sa hrany pôvodného zhluku začínajú vizuálne zhustovať. Aj keď detekcia samostatných hrán už nie je možná, stále vidíme takmer celý zhluk a jeho približnú štruktúru. Ako vyzerá štruktúra grafu, ktorú momentálne nepozorujeme objasňujú aj

glyfy. Keďže vystupujú mimo povrchu gule na základe ich vzhľadu, dokážeme určiť aké typy vrcholov sa nachádzajú v okrajových častiach pohľadu a aký je ich počet. V prvom pohľade tak vyznačený vrchol hodnotou svojho atribútu jasne napovedá, že k nemu smeruje veľké množstvo hrán. Zároveň vidíme že v danej oblasti je pomerne osamotený. Druhý a tretí pohľad najlepšie demonštruje ako sa mení vzhľad informácie, ktorá sa dostáva z okrajovej časti do centra záujmu, konkrétnie na zhľuku okolo vrcholu „2@enron.com“.



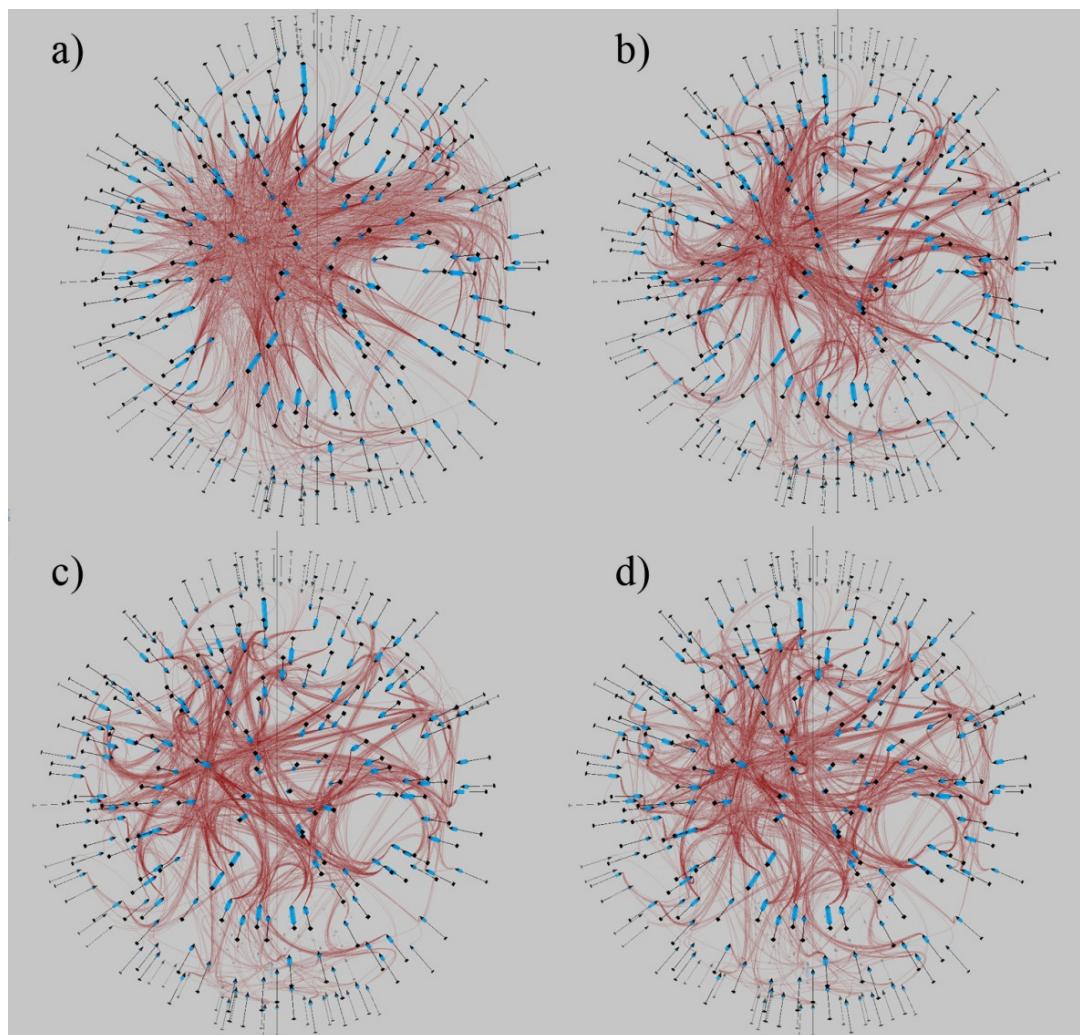
Obr. 37 – Pootočenie vizualizácie a demonštrácia vlastnosti gule „focus + context“.

V doterajších ukážkach išlo o pomerne jednoduchý graf preto sme mohli využiť reprezentáciu hrán pomocou rovných spojníc a kružnicových oblúkov. V nasledujúcich príkladoch sme zväčšili vstupné dátá na 250 emailových adres, v ktorom sa nachádza celkovo 1859 hrán. Na obrázku 38a pozorujeme, že kríženie hrán je už príliš veľké, vzhľadom k tomu už kružnicové oblúky nie sú postačujúce.



Obr. 38 – Zobrazenie dátového setu o veľkosti 250 vrcholov a 1859 hrán. Hrany reprezentované: a) kružnicovými oblúkmi, b) sférická bázierova krivka s využitím hierarchie hĺbky 3.

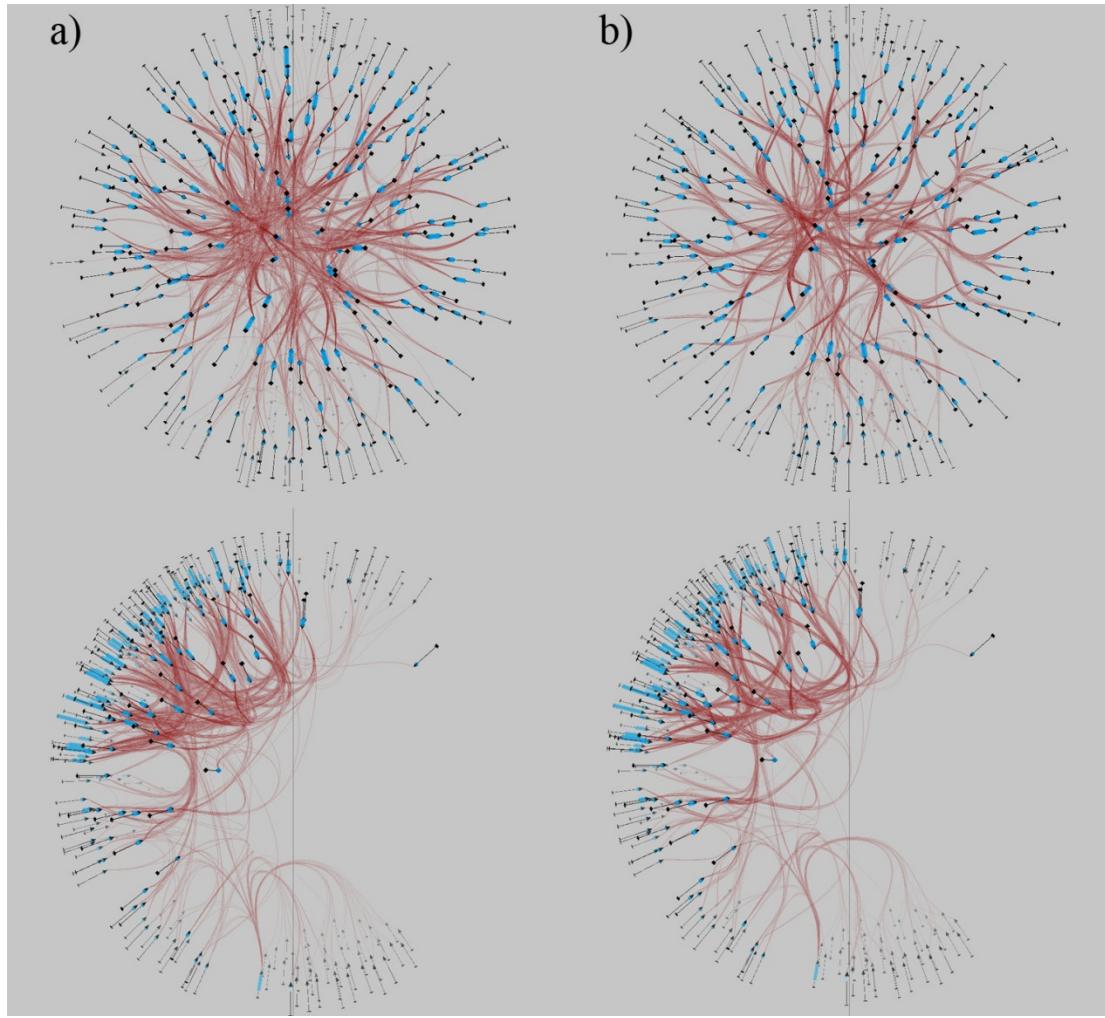
Obrázok 38b zobrazuje identickú situáciu, hrany sú však reprezentované sférickou bézierovou krvkou, ktorej riadiace vrcholy určuje hierarchia hlbky 3 vytvorená na základe osemstenu. Hrany v tomto prípade netvoria chaotickú neprehľadnú štruktúru, ale formujú čitateľnejšie zväzky. Farba hrán je čiastočne prieľadná, zväzky z väčšou hustotou hrán preto dokážeme odlišovať na základe intenzity farby. Rovnakým princípom, na základe intenzity, vieme určiť aj približný počet hrán spojených s určitým vrcholom, pretože všetky tesne pred vrcholom vytvoria úzky zväzok, a vchádzajú aj vychádzajú z neho iba v jednom smere.



Obr. 39 – Vplyv hlbky hierarchie na tvar krviek interpolovaných bézierovou krvkou. Počet rekurzívnych delení trojuholníkov: a) 0, b) 1, c) 2, d) 3.

Ako vplýva hlbka vytvorenej hierarchie (počet rekurzívnych delení oblastí) na vzhľad bézierových krviek znázorňuje obrázok 39. Všetky riadiace vrcholy v tomto príde ležia na povrchu gule, samotná krvka nie úplne. Môžeme pozorovať, že zvýšením rekurzívneho delenia z 2 na 3 už pre danú vizualizáciu neprinieslo zlepšenie, preto nemá význam používať hlbšie hierarchie. Najväčší rozdiel v kvalite je dosiahnutý v prvých dvoch rekurzívnych

deleniach. Zároveň pozorujeme, že hĺbka hierarchie podstatne ovplyvňuje celkový výsledok zväzovania. Čím kratšie hrany vizualizácia obsahuje, teda čím sú bližšie k sebe vrcholy spojené hranou, tým je vhodnejšie zvoliť väčšiu hĺbku, pretože je vysoká pravdepodobnosť, že sa budú nachádzať v menšom spoločnom trojuholníku a riadiace vrcholy priradené hrane medzi nimi sa budú nachádzať bližšie. Pri nízkej hĺbke hierarchie sú vrcholy nútene využívať veľmi vzdialené riadiace vrcholy, čo dramaticky ovplyvní ich dĺžku a odklonenie.

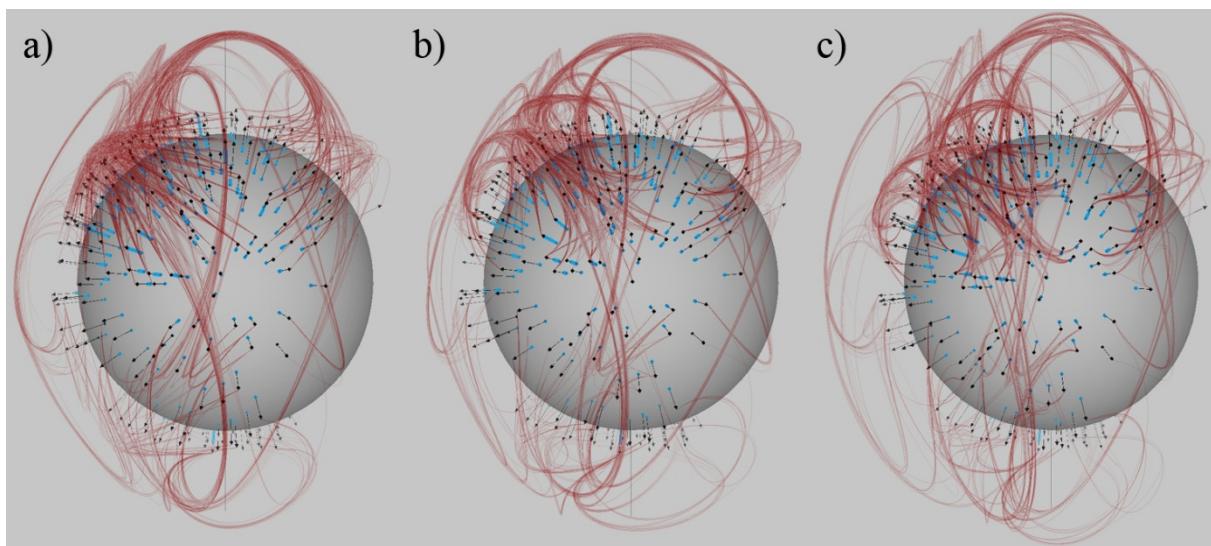


Obr. 40 – Riadiace vrcholy priblížené k stredu gule a interpolácia medzi nimi bázierovou krivkou: a) štandardnou, b) racionálnou. (Hore pohľad spredu, dole pohľad z boku).

Na obrázku 40 pozorujeme ako riadiace vrcholy ovplyvnia tvar kriviek a výslednú štruktúru vizualizácie, v prípade ak sú priblížené k stredu gule. V príklade boli riadiace vrcholy priblížené k stredu na základe ich hĺbky v hierarchii, presnejšie čím bola ich hĺbka menšia tým bližšie boli posunuté k stredu. Vzhľadom k tomu kratšie hrany s menším počtom riadiacich vrcholov sú odklonené oveľa menej hrany medzi vrcholmi, ktorých cesta v hierarchii vedie cez koreň. Vďaka tomu sa zlepšila prehľadnosť, pretože odklonenie

krátkych a dlhých hrán je teraz výrazne rozdielnejšie a zväčšil sa aj priestor, ktorý môžu okupovať. Zároveň však vzniká komplexnejšia 3D štruktúra, pre ktorej uvedomenie je nevyhnutná interakcia s vizualizáciu. Obrázok 40b demonštruje ako racionálna bézierová krivka pomáha vytvoriť užšie a lepšie rozoznateľné zväzky oproti štandardnej na obrázku 40a.

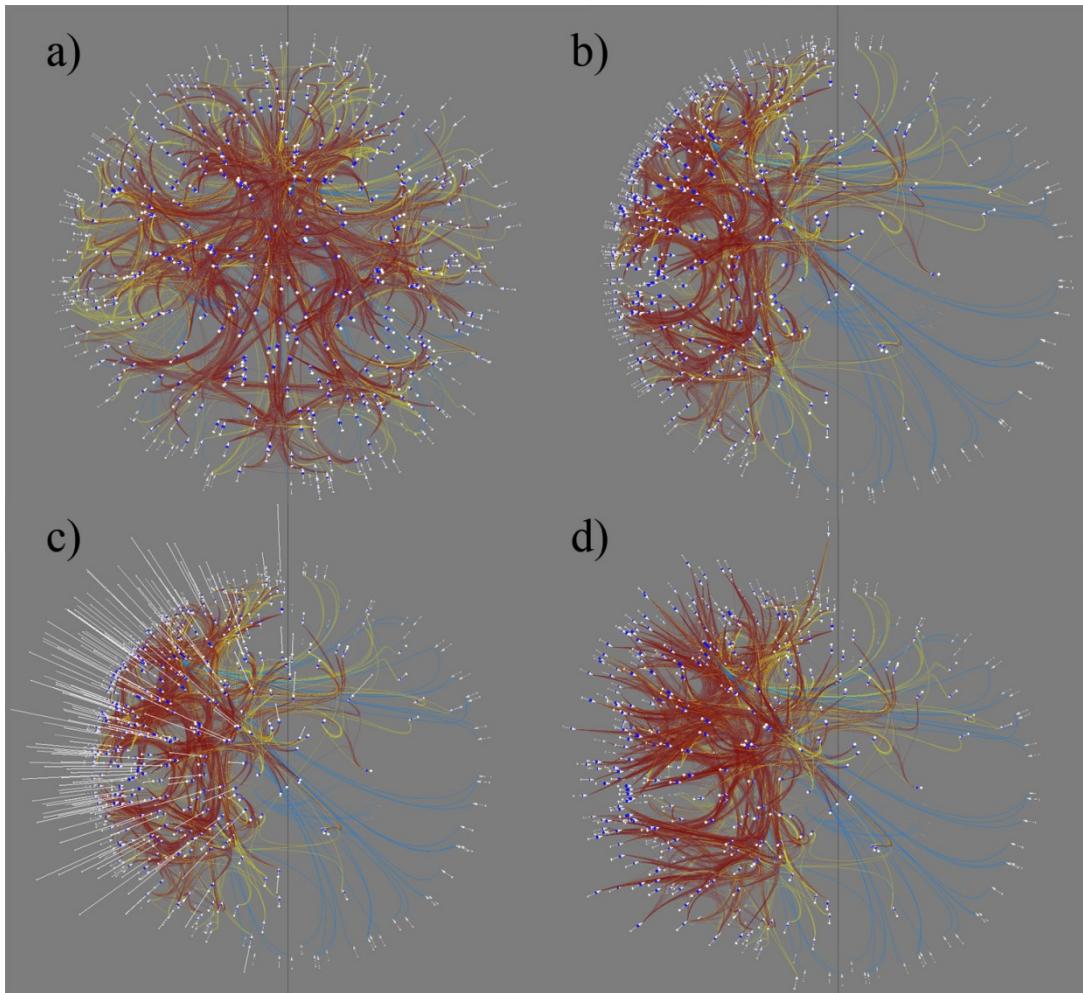
Opačný prípad v ktorom riadiace vrcholy vzdialujeme od povrchu, tak aby krivky smerovali okolo gule je zobrazený na obrázku 41. Krivky sú v tomto prípade omnoho viac odklonené a predĺžené, z dôvodu pridania dodatočných riadiacich vrcholov, ktoré zabraňujú ich kríženiu s guľou. Táto štruktúra je oproti predošej lepšie čitateľná v statických obrazoch, čiastočne však zhoršuje čitateľnosť glyfov z dôvodu ich prekrytie.



Obr. 41 – Riadiace vrcholy vzdialené od stredu gule a interpolácia medzi nimi racionálnou bézierovou krivkou. Počet rekurzívnych delení trojuholníkov v hierarchii: a) 0, b) 1, c) 2.

Modifikácia pružinového modelu nám umožňuje upravovať tvar kriviek a výslednú štruktúru vizualizácie aj zapojením vysúvania samotných vrcholov grafu, na základe ich hodnoty napäťia. Ukážka sa nachádza na obrázku 42. Vizualizácia v tomto prípade zobrazuje komunikáciu medzi 500 emailovými adresami, s celkovým počtom 6030 hrán. Okrem toho boli vytvorené tri rôzne typy hrán, na základe hodnôt atribútu `exchanged_emails` vo vrcholoch. Prvý typ tvoria hrany, v ktorých aspoň jeden z krajných vrcholov má hodnotu tohto atribútu maximálne 1, čiže daná emailová adresa komunikovala iba s adresou, ktorú reprezentuje druhý koncový vrchol hrany (modrá krivka). V druhom type je hranica posunutá na hodnotu atribútu v intervale 2 až 10 (žltá krivka). V treťom type oba z vrcholov majú hodnotu atribútu väčšiu ako 10 (červená krivka). Vďaka tomu dokážeme aj v komplexnejšej štruktúre intuitívne identifikovať, ktoré z emailových adres komunikujú najmenej/najviac a

zároveň aj s kým. Všetky vrcholy prepojené červenými krivkami reprezentujú adresy, komunikujúce minimálne s ďalšími desiatimi adresami. Žlté, reprezentujú adresy s menšou frekvenciou komunikácie. Pozorujeme, že väčšina modrých kriviek smeruje z vrcholov nachádzajúcich sa mimo zhluku, smeruje k jedinému vrcholu.

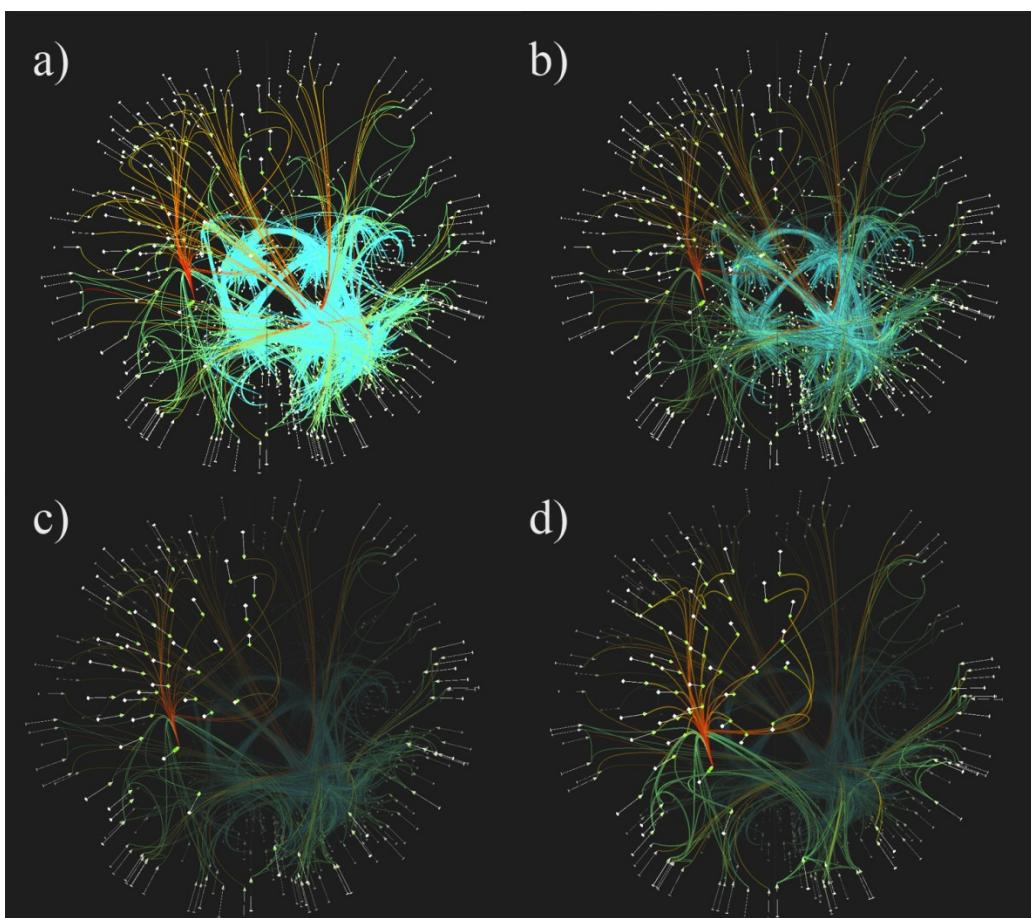


Obr. 42 – a),b) pôvodná vizualizácia, c) vyobrazenie hodnôt napäť vo vrcholoch, d) vysunutie vrcholov na základe hodnôt napäť.

Vysunutím vrcholov na základe hodnoty ich napäťa sme zväčšili priestor pre zobrazovanie kriviek. Na obrázku 42b, 42d môžeme pozorovať ako sa zmenila celková štruktúra vizualizácie. V statickom obrázku došlo k zlepšeniu vnímania zväzkov hrán pripájajúcich sa ku vrcholom v oblasti mimo centra záujmu. Zároveň je ale stážené určovanie vzájomných vzdialenosí vrcholov nachádzajúcich sa v rôznych výškach. Môžeme pozorovať, že tvar kriviek nie je veľmi zmenený, pretože jediné riadiace vrcholy krivky, ktoré zmenili svoju pozíciu sú pozície samotných vrcholov. Všetky riadiace vrcholy v hierarchii zostali na pôvodných miestach. Vysunutie riadiacich vrcholov do vyššej výšky nerieši problém, pretože

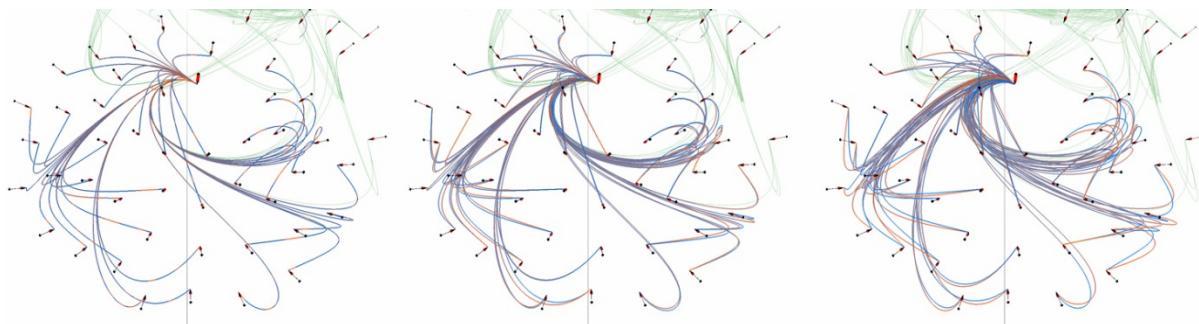
negatívne vplýva na tvar kriviek medzi vrcholmi, ktoré stále ležia na jej povrchu. Predmetom ďalšej práce tak môže byť, návrh modifikácie ovládania výšky riadiacich vrcholov tak, aby mohli byť vysúvané individuálne v lokálnych oblastiach, v závislosti od hodnôt napätí vrcholov v danej oblasti.

Demonštrácia účinnosti stereo zobrazovania sa nachádza na obrázkoch v prílohe č.2. Na obrázkoch je zobrazená komunikácia medzi 1000 emailovými adresami s počtom hrán 17388. Prvý obrázok zobrazuje vizualizáciu bez vysunutia vrcholov a riadiace vrcholy boli priblížené k stredu gule viac ako v druhom prípade, v ktorom už vrcholy sú vysunuté. Vzhľadom k spomínanému problému s vysúvaním riadiacich vrcholov, momentálne poskytuje prehľadnejšiu vizualizáciu štruktúra bez vysunutia vrcholov grafu. Získaná hĺbková informácia z anaglyfu sa výrazne podieľa na uvedomení 3D štruktúry. Priama interakcia s vizualizáciou v stereo zobrazení znásobuje tento efekt, vďaka ktorému dokážeme rozoznávať aj veľmi komplexné štruktúry s vysokým množstvom hrán. Samozrejme sa na tom výrazne podieľa aj technika zväzovanie hrán a jej nastavenie.



Obr. 43 – Pridávanie „depth cues“: a) žiadne, b) pridaná priehľadnosť, c) pridaný atmosférický vplyv, d) pridaná rôzna hĺbka čiar na základe vzdialenosť od kamery.

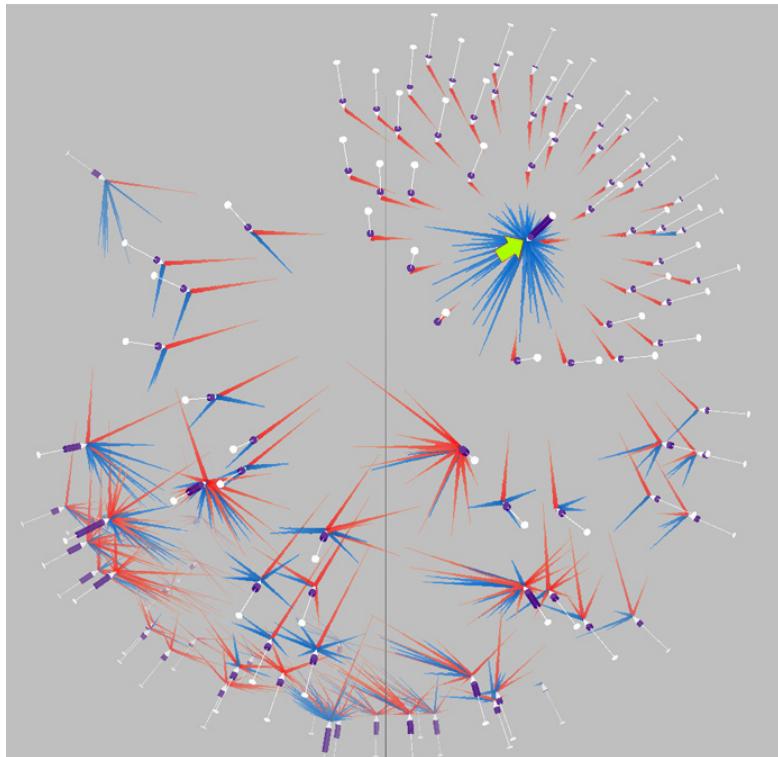
Získanie informácie o hĺbke však v aplikácii nezabezpečuje iba interakcia a stereo zobrazovanie. Veľmi dôležitú úlohu zohrávajú predovšetkým monokulárne statické “depth cues”, ktoré ovplyvňujú predovšetkým výsledok statickej vizualizácie. Vo všetkých demonštrovaných príkladoch boli prítomné, konkrétnie sme sa im však nevenovali. Na obrázku 43a, môžeme pozorovať ako vyzerá pôvodná vizualizácia pred aplikáciou akejkoľvek hĺbkovej informácie. Z vizualizácie nedokážeme povedať v akej hĺbke sa jednotlivé hrany nachádzajú, pretože celá vizualizácia pôsobí plocho. Zároveň sa veľmi husté zhluhy hrán zlievajú do jedného celku, v ktorom nie je možné identifikovať individuálne zväzky a hrany. Pridaním čiastočnej priehľadnosti hrán je tento problém odstránený (obr. 43b). Avšak až pridaním atmosférického vplyvu získame predstavu o hĺbkach jednotlivých hrán a vrcholov. Pozorujeme, že zhľuk modrých hrán sa nachádza na odvrátenej strane gule (obr. 43c). Záverečné posilnenie hĺbkovej informácie prináša pridanie hrúbky čiar, v závislosti od ich vzdialenosť od kamery (obr. 43d). Hrany na odvrátenej strane gule sú tenšie ako hrany v centre pozornosti nachádzajúce sa na privŕatenej strane gule. V danom príklade by bez nich bola statická vizualizácia pomerne bezcenná.



Obr. 44 – Efekt pri náhodnom posunutí riadiacich vrcholov.

Obrázok 44 demonštruje význam náhodného posunutia riadiacich vrcholov v prípade, ak sa medzi dvoma vrcholmi nachádzajú dve opačne orientované hrany. V ľavej časti obrázka na riadiace vrcholy nebolo aplikované žiadne posunutie, preto všetky dvojice hrán, medzi rovnakými vrcholmi, ležia v sebe. V strednej ukážke bolo aplikované iba veľmi malé posunutie, avšak vo väčšine prípadov už dokážeme rozoznať obidve hrany. V pravej časti obrázka je posunutie väčšie a každá z hrán je už samostatne odlišiteľná. Efekt zviazania je v tomto prípade čiastočne narušený, stále je však postačujúci. Je nutné zvoliť vhodnú veľkosť posunutia vzhladom od konkrétnej vizualizácie.

Posledný obrázok sa opäť venuje zobrazovaniu orientovaných hrán, tento krát sa však medzi každou dvojicou vrcholov nachádza maximálne jedna. Využili sme pritom rovnaké dátá, ktorými sme demonštrovali pružinový model (100 adries). Hrany v štruktúre sú neorientované, ale v príklade simulujeme situáciu, že ide o orientovaný vzťah, v ktorom jeden z vrcholov je odosielateľom správy a druhý prijímateľom. Hranu reprezentujeme priamou spojnicou, ktorej začiatok je ofarbený červenou a koniec modrou farbou. Hrubka čiary je na oboch koncoch zväčšená. Stredná časť čiary je zároveň úplne priečadná s nulovou hrubkou. Tým vytvoríme šípky, smerujúce z vrcholov v smere daných hrán. Červené šípky vychádzajú z vrcholov reprezentujúcich odosielateľov a modré prijímateľov. Aj keď v tomto prípade nevidíme celé hrany, dokážeme predpokladať smer a dĺžku hrán. Orientácia šípky napovedá o smere hrany, a jej dĺžka o veľkosti vzdialenosť medzi vrcholmi. Znížením vykreslovanej grafickej informácie sa v neprehľadných sietiach výrazne zníži chaos, najmä faktor kríženia hrán. Vďaka tomu dokážeme identifikovať smer, orientáciu a frekventovanosť hrán vo vrcholoch [21]. Na obrázku 45, je vo vizualizácii zelenou šípkou vyznačený najfrekventovanejší vrchol. V našom hypotetickom príklade, tak z obrázku 45 dokážeme vyvodiť, že najviac emailových správ priala adresa „2@enron.com“ a zároveň odoslala iba jedinú.



Obr. 45 – Zobrazovanie orientovaných hrán prerušenou čiarou.

Kapitola 5

5. Záver

Cieľom práce bolo navrhnuť techniku trojrozmernej interaktívnej vizualizácie grafových štruktúr, so zameraním nielen na zobrazovanie vzájomných vzťahov medzi entitami, ale aj atribútov entít samotných. Dosiahnuté výsledky boli demonštrované vo vlastnej aplikácii s implementáciou navrhnutých metód.

Návrh vizualizácie bol ovplyvnený vybranými technikami, predstavenými v prvej kapitole práce. Najväčší dôraz sme pritom kládli na prehľadnosť celkovej vizualizácie. Z tohto dôvodu rozloženie vrchov grafu obstaráva pružinový algoritmus modifikovaný pre sférickú geometriu, ktorý implicitne znižuje faktor kríženia hrán. Kríženie hrán predstavuje najčastejší a najzávažnejší problém v podobných vizualizačných technikách. Nám sa ho podarilo potlačiť využitím navrhнутej techniky zväzovania hrán, ktorá využíva hierarchiu riadiacich vrcholov, ktoré ovplyvňujú výsledný tvar kriviek. Spomínaná hierarchia nevyužíva priamo štruktúru grafu, ale rekurzívne rozdeľovanie priestoru do oblastí, preto je tento prístup zväzovania aplikovateľný aj pre cyklické grafové štruktúry. Rovnako technika nie je limitovaná iba pre sférickú geometriu, ale môže byť použiteľná aj v klasických 2D/3D vizualizáciach.

Okrem iného sme predstavili modifikáciu pružinového algoritmu vysúvaním vrcholov aj mimo povrchu gule, vzhľadom od tlaku, ktorý je na ne vyvíjaný. Jej výsledky sú priaznivé, avšak dosiahnutie lepších výsledkov by mala priniesť fyzikálne presná implementácia a spomínané lokálne vysúvanie riadiacich vrcholov hierarchie.

Celková vizualizácia ponúka dobré výsledky aj pri zobrazovaní komplexnejších štruktúr. Veľmi dobre si dokáže poradiť aj s vysokým počtom hrán. Na základe demonštrovaných výsledkov preto môžeme povedať, že sme splnili požadované ciele a priniesli obohatenie vedomostí v oblasti vizualizácie grafových štruktúr. Predstavené techniky zároveň ponúkajú priestor pre ich vylepšenie a podrobnejšie skúmanie, ktoré môžu byť predmetom ďalšej práce.

Zoznam použitej literatúry

- [1] O. Ore, *Theory of Graphs*, American Mathematical Society, Providence, 1962.
- [2] Ch. Chen, W. Härdle, A. Unwin, *Handbook of Data Visualization*, Springer, 2008.
- [3] R. Mazza, *Introduction to Information Visualization*, Springer, 2009.
- [4] Visual Complexity, [online][dátum:22.3.2011] Dostupné na internete: <http://www.visualcomplexity.com>
- [5] Protovis, [online][datum:22.3.2011] Dostupné na internete: <http://vis.stanford.edu/protovis/>
- [6] S.G. Kobourov, *Force-Directed Drawing Algorithms*, University of Arizona, CRC press, 2004.
- [7] P. Eades. *A heuristic for graph drawing*, Congressus Numerantium, 1984.
- [8] T. Fruchterman, E. Reingold. *Graph drawing by force-directed placement*, Software-practice and experience, vol. 21, November, 1991.
- [9] T. Kamada, S. Kawai, *An algorithm for drawing general undirected graphs*, Information Processing Letters 31, 1989.
- [10] prezentacia „Ellis, TVCG07“
- [11] D. Holten, J. J. van Wijk, *Force-Directed Edge Bundling for Graph Visualization*, Eurographics / IEEE-VGTC Symposium on Visualization, Volume 28, Number 3, 2009.
- [12] N. Wong, S. Carpendale, S. Greenberg, *EdgeLens: An interactive method for managing edge congestion in graphs*, 2003.
- [13] W.Cui, H.Zhou, H.Qu, P. Ch. Wong, X. Li, *Geometry-Based Edge Clustering for Graph Visualization*, IEEE Transactions on Visualization and Computer Graphics, Volume: 14 Issue:6, 2008.
- [14] D. Holten, *Hierarchical Edge Bundles: Visualization of Adjacency Relations in Hierarchical Data*, IEEE transactions on visualization and computer graphics, vol. 12, no. 5, September / október 2006.
- [15] P. Bourke, *Stereographics*, [online][dátum:22.3.2011] Dostupné na internete: <http://paulbourke.net/miscellaneous/stereographics/>

- [16] J. Lamping, R. Rao, P. Pirolli, *A focus+context technique based on hyperbolic geometry for visualizing large hierarchies*, In Proceedings of Computer Human Interaction, ACM, 1995.
- [17] D. I. Ostry, *Some Three-Dimensional Graph Drawing Algorithms*, Master thesis, The University of Newcastle, 1996.
- [18] S. G. Kobourov, K. Wampler, *Non-Euclidean Spring Embedders*, IEEE transactions on visualization and computer graphics, vol. 11, no. 6, november/december 2005.
- [19] W. S. Cleveland, R. McGill, *Graphical perception: Theory, experimentation, and application to the development of graphical methods*, Journal of the American Statistical Association, 1984.
- [20] W.H. Bardel, *Depth Cues For Information Design*, Carnegie Mellon University, Pittsburgh, Pennsylvania, May, 2001.
- [21] R. A. Becker, S. G. Eick, A. R. Wilks, *Visualizing Network Data*, IEEE Transactions on Visualization and Computer Graphics, Vol. 1, No. 1, March 1995.
- [22] B. Shneiderman, *Tree visualization with treemaps: a 2-d space-filling approach*, ACM Transactions on Graphics, vol. 11, Január 1992.
- [23] M. Balzer, O. Deussen, C. Lewerentz, *Voronoi treemaps for the visualization of software metrics*, SoftVis '05: Proceedings of the 2005 ACM symposium on Software visualization, page 165--172. ACM Press, (2005).
- [24] N. Henry, J.D. Fekete, M.J. McGuffin, *NodeTrix: a Hybrid Visualization of Social Networks*, IEEE Transactions on Visualization and Computer Graphics, Nov.-Dec. 2007.
- [25] J.B. Kruskal, M. Wish, *Multidimensional Scaling*, Sage University Paper series on Quantitative Application in the Social Sciences, 07-011. Beverly Hills and London: Sage Publications, 1978.
- [26] A. Quigley, *FADE*, [online][dátum:25.3.2011] Dostupné na internete:
<http://rp-www.cs.usyd.edu.au/~aquigley/3dfade/>
- [27] Simulated Annealing, [online][dátum:25.3.2011] Dostupné na internete:
<http://mathworld.wolfram.com/SimulatedAnnealing.html>

[28] Newton-Raphson metóda, [online][dátum:5.4.2011] Dostupné na internete:

<http://mathworld.wolfram.com/NewtonsMethod.html>

[29] G. Ellis, A. Dix, *The plot, the clutter, the sampling and its lens: occlusion measures for automatic clutter reduction*, AVI 06, Proceedings of the working conference on Advanced visual interfaces, ACM New York, 2006.

[30] R.A. Becker, W. S. Cleveland, *Brushing Scatterplots*, Technometrics, Vol. 29, 1987.

[31] A. Inselberg, B. Dimsdale, *Parallel coordinates: a tool for visualizing multi-dimensional geometry*, VIS '90, Proceedings of the 1st conference on Visualization '90, IEEE Computer Society Press Los Alamitos, 1990.

[32] R.L. Sollenberger, P. Milgram, *Effects of Stereoscopic and Rotational Displays in a Three-Dimensional Path-Tracing Task*, Human Factors: The Journal of the Human Factors and Ergonomics Society, Volume 35, Number 3, September 1993.

[33] P. Kapec, *Visualizing software artifacts using hypergraphs*, SCVG Presentation: 2010 Visualization.

[34] S.K. Card, J. Mackinlay, B. Schneiderman, *Readings in Information Visualization: Using Vision to Think*, Morgan Kaufmann Publishers Inc. San Francisco, 1999.

[35] Great circle, [online][dátum:25.3.2011] Dostupné na internete: http://en.wikipedia.org/wiki/Great_circle

[36] J. Abello, F. Ham, N. Krishnan, *ASK-GraphView: A Large Scale Graph Visualization System*, IEEE Transactions on Visualization and Computer Graphics, Sept.-Oct. 2006.

[37] C. Ware, *Information Visualization, Second Edition: Perception for Design*, Elsevier, 2004.

[38] R. Spence, *Information Visualization*, Addison Wesley, 1st edition, 2000.

Prílohy

Príloha č.1 - Vstupný súbor

Aplikácia využíva vlastný formát vstupných dát, ktoré sú uložené v textovom súbore, teda súbor s koncovkou .txt. Práca s týmto formátom je jednoduchá, čiže užívateľ by nemal mať problém transformovať dátový set do žiadanej podoby, ktorú naša aplikácia vyžaduje. Nasledujúca ukážka demonštruje dátový set, ktorého graf obsahuje 3 vrcholy, 3 hrany dvoch rôznych typov zároveň aj dĺžky a každý z vrcholov má 6 atribútov.

```
1 3  
2 2  
3 1 Friends 0  
4 2 Supervisor 1  
5 0 0 0  
6 1 0 2  
7 0 1 0  
8  
9 0.0 1.0 0.0  
10 1.0 0.0 0.5  
11 0.0 0.5 0.0  
12 6  
13 1 Name  
14 2 Male  
15 3 Age 0 100  
16 4 Height 50.0 220.0  
17 5 Face_image  
18 4 Weight 0.0 100.0  
19  
20 1 Peter 1 30 190.5 peter.bmp 85.4  
21 2 Jane 0 92 130.5 jane.bmp 62.2  
22 3 John 1 55 182.1 john.bmp 78.0
```

1 - Počet vrcholov.
2 - Počet rôznych typov hrán.
3, 4 - Definovanie názvov typov hrán a orientácie.
5, 6, 7 - Matica definujúca hrany.

8 - Prázdný riadok.
9, 10, 11 - Matica definujúca dĺžky hrán.

12 - Počet atribútov vrcholu.
13, 14, 15, 16, 17, 18 - Definícia typov atribútov v rovnakom poradí akom budú zapísané ich hodnoty.

18 - Prázdný riadok.
20, 21, 22 - Hodnoty atribútov jednotlivých vrcholov v usporiadanom poradí.

Všetky samostatné hodnoty v súbore musia byť spojité, teda bez medzier, pretože ich nimi oddelujeme medzi sebou. Preto vo viacslovných názvoch je potrebné medzeru nahradíť znakom „_“. V prvom riadku súboru sa nachádza informácia o počte vrcholov grafu. V druhom počet rôznych typov hrán v grafe. Každý typ hrany má vlastné nastavenia farby, hrúbky, prieľahnosti a animácie. Ak sú všetky hrany v grafe rovnakého typu, počet typov je 1. Pre každý typ hrany musí byť v samostatnom riadku definovaný názov a orientácia vo formáte číselné_označenie_typu_názov_typu_orientácia. Na číselné označenie typu sa používajú prirodzené čísla začínajúc od 1. Pomocou týchto čísel reprezentujeme typ konkrétnej hrany v matici susednosti. Názov typu je spojity text,

a orientácia určuje či daný typ reprezentuje orientovanú alebo neorientovanú hranu. Na základe toho zapíšeme hodnotu 1 (orientovaný) alebo 0 (neorientovaný).

Označme počet vrcholov n , potom v nasledujúcich n riadkoch sa nachádza zapísaná matica susednosti veľkosti $n \times n$, ktorá definuje medzi ktorými dvoma vrcholmi sa nachádza hrana a akého je typu. Ak sa medzi vrcholmi i a j nachádza iba jedna hrana, v matici zapíšeme číselnú hodnotu jej typu iba raz, buď na políčko $[i,j]$ alebo $[j,i]$. Ak táto hrana nie je orientovaná, je jedno ktoré políčko si zvolíme (nie však oba!). V prípade ak je orientovaná z vrcholu i do j je potrebné zapísať jej typ na políčko $[i,j]$. Ak vyplníme obe políčka, medzi i a j sa vytvoria dve hrany. Ak sa však medzi nimi nenachádza žiadna hrana zapíšeme na obe políčka nulu. Maximálne teda môžeme definovať dve hrany medzi každou dvojicou vrcholov. V našom príklade vstupného súboru existujú dva typy hrán. Medzi prvým a druhým vrcholom sa nachádza jedna neorientovaná hrana typu **Friends**. Medzi druhým a tretím vrcholom sa nachádzajú dve hrany, jedna neorientovaná typu **Friends**, druhá orientovaná začínajúca v druhom vrchole a smerujúca do tretieho, typu **Supervisor**. Pre lepšiu prehľadnosť súboru za maticou nasleduje jeden voľný riadok.

V nasledujúcich n riadkoch sa opäť nachádza ďalšia matica $n \times n$, ktorá tento krát reprezentuje vzdialenosť medzi vrcholmi spojenými hranou. Ak sa teda medzi i a j nachádza hrana/hrany, na políčku $[i,j]$ aj $[j,i]$ sa musí nachádzať údaj o optimálnej vzdialnosti týchto vrcholov v inom prípade musia byť obe políčka nulové. Hodnota musí byť rovnaká v oboch políčkach. Vzdialosť sa zadáva v normalizovanej forme s hodnotami v intervale $(0,1)$, pretože skutočná hodnota maximálnej vzdialnosti bude záležať od počtu vrcholov grafu. Môžeme si všimnúť, že v našom príklade je vzdialosť medzi prvým a druhým vrcholom maximálna (1.0), a medzi druhým a tretím o polovicu menšia (0.5).

V ďalšom riadku už začína definícia atribútov konkrétnie sa v ňom nachádza hodnota, ktorá označuje koľko atribútov má vrchol grafu. Každý z nich je potom v samostatnom riadku definovaný vo formáte `typ_názov(_dolná_hranica,_horná_hranica)`. Usporiadanie v akom sú jednotlivé atribúty definované musí byť totožný s poradím v akom sú zapísané ich hodnoty pri vrcholoch.

Celkovo aplikácia spracováva 5 rôznych typov atribútov:

- Textový reťazec (*string*) – `typ = 1`
- Boolova premenná (*bool*) – `typ = 2`
- Celé číslo (*integer*) – `typ = 3`
- Desatinné číslo (*float*) – `typ = 4`
- Cesta k bitmapovému obrázku (*uri, string*) – `typ = 5`

Pri atribútoch typu 3 a 4 musíme definovať aj dolnú a hornú hranicu intervalu, v ktorom chceme hodnoty škálovať. Všetky hodnoty väčšie ako horná hranica, budú škálované podľa hodnoty hornej hranice. Obdobne platí aj pre hodnoty menšie ako dolná hranica. Toto umožňuje zachovať pôvodné hodnoty atribútov a zároveň kontrolovať interval v ktorom sa snažíme pozorovať podobnosti/rozdiely. Po definícii všetkých atribútov nasleduje prázdný riadok.

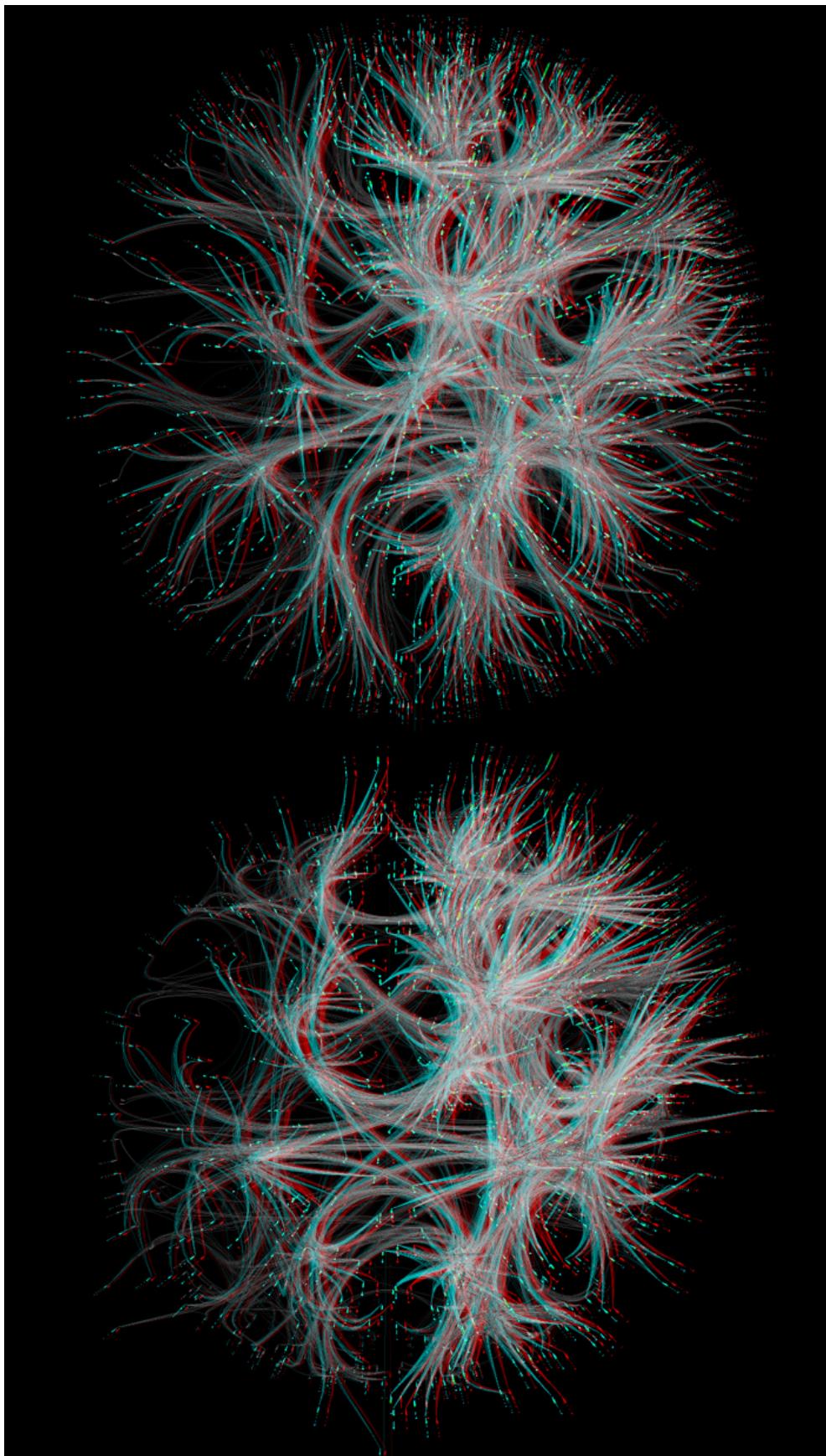
Poslednými údajmi, ktoré sa v súbore musia nachádzať sú hodnoty atribútov pre každý vrchol v samostatnom riadku. Vrcholy sú usporiadane od prvého po posledný tak aby sedeli ich index s definíciou matice susednosti. Na začiatku každého riadku sa nachádza index vrcholu, začínajúc hodnotou 1 až po n . Za indexom sa už nachádzajú konkrétné hodnoty atribútov oddelené medzi sebou medzerami a usporiadane v presne rovnakom poradí v akom boli definované atribúty.

Hodnoty atribútov musia zodpovedať ich typu:

- `typ = 1` – akceptuje spojity textový reťazec, zložený z písmen, čísel a podčiarkovníkov.
- `typ = 2` – akceptuje iba hodnotu 1 (pravda) alebo 0 (nepravda).
- `typ = 3` – akceptuje celé čísla.
- `typ = 4` – akceptuje celé aj desatinné čísla písané s bodkou.

`typ = 5` – akceptuje textový reťazec reprezentujúci cestu k obrázku vzhľadom k priečinku v ktorom sa program nachádza.

Príloha č.2 - Anaglyf



Príloha č.3 – Menu aplikácie

Main

- Load new file
- Background Color: [Color Box] Show Color: [Color Box]
- Stereo view: Enable Anaglyph Focal Point: Near Sphere surface Far
- Field of View: Degrees: 45
- Fog: Enable Near Start End
- Global display switches: Show Nodes, Show edges

Hierarchy

- Glyph: Display Glyph Bounding Box Color: White, Black Number of floors: 3, Width: [Slider], Number of sides: 4, Height: [Slider]
- Edit floor: vyska (Floor: 1, Attribute: jeMuz, Color: Green), vek, jeMuz
- Caption: Display caption, Font size: [Slider], Color: [Color Box], Opacity: [Slider]
- Image: Display Image, Size: [Slider]

Spring Model

- Show polyhedra, Control points: Levels: [Slider], Color: [Color Box]
- Type: Icosahedron, Octahedron (selected)
- Height if control points: Level: ALL, Height: [Slider]
- Extra control points (poles and 4 equator points): Enable/Disable, Height: [Slider]
- Randomize Position: No, Strength of random: [Slider], Yes
- Start, Stop, Next Iteration, New layout
- Layout Attributes: Max Spring Length: 10%, 100%, 300%, Attractive force: Lower, Higher, Repulsive force: [Slider], Damping: 70%, 95%
- Use Stress: No, Yes - Display stress, Yes - Displace nodes
- Number of spline segments: 50, Connect at: Bottom, Top
- Edge attributes: Global, According to edge type
- Edge parameters: Edge type: relation, Spline type: Bézier, Adjust, Project spline on sphere
- Spline gradient: Gradient bar with color controls
- Gradient control point attributes: Color: [Color Box], delete, Opacity: [Slider], Line width: [Slider]
- Spline animation: Animate color, Animate line width, Speed: Low, High

Príloha č.4 – DVD – ROM

Obsahuje:

- Aplikáciu
- Zdrojový kód aplikácie
- Vstupné dátá
- Elektronickú verziu diplomovej práce