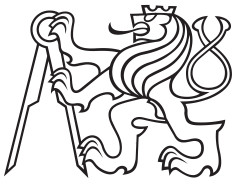
Bakalářská práce



České
vysoké
učení technické
v Praze

F3

Fakulta elektrotechnická
Katedra počítačové grafiky a interakce

Kreslení grafů a metriky estetičnosti

Kristýna Kořenská

Vedoucí: Ing. Matěj Dostál, Ph.D.
Obor: Otevřená informatika
Studijní program: Počítačové hry a grafika
Květen 2023

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kořenská** Jméno: **Kristýna** Osobní číslo: **491937**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra počítačové grafiky a interakce**

Studijní program: **Otevřená informatika**

Specializace: **Počítačové hry a grafika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Kreslení grafů a metriky estetičnosti

Název bakalářské práce anglicky:

Graph drawing and aesthetics metrics

Pokyny pro vypracování:

Seznamte se spektrálními vlastnostmi matic grafů a s využitím daných vlastností ke kreslení (geometrické representaci) grafů [3].
Proveďte rešerši algoritmů na kreslení grafů pomocí minimalisace energie representace daného grafu a metrik sloužících k měření estetičnosti nakreslení grafu [2].
Popište Eadesův algoritmus silově řízeného kreslení grafu [1].
Stanovte kritéria pro výběr vhodných metrik estetičnosti a vyberte metriky k implementaci.
Navrhněte a naprogramujte nástroj na kreslení grafu způsoby popsanými v teoretické části.
Proveďte experimenty s měřením estetičnosti výsledného nakreslení dle použité metody kreslení na vhodně zvolených příkladech grafů.
Porovnejte implementované metriky s názorem alespoň deseti účastníků a alespoň deseti příkladech různých typů grafů.

Seznam doporučené literatury:

[1] P. Eades, A Heuristic for Graph Drawing, Congressus Numerantium, 42 (11): 149–160 (1984)
[2] H. Purchase, Metrics for Graph Drawing Aesthetics, Journal of Visual Languages & Computing, 13 (5), 2002
[3] D. Spielman, Spectral and Algebraic Graph Theory, dostupné online

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Matěj Dostál, Ph.D. katedra matematiky FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **12.02.2023** Termín odevzdání bakalářské práce: **26.05.2023**

Platnost zadání bakalářské práce: **22.09.2024**

Ing. Matěj Dostál, Ph.D. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Mgr. Petr Páta, Ph.D. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Studentka bere na vědomí, že je povinna vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

. | Datum převzetí zadání

Podpis studentky

Poděkování

Chtěla bych poděkovat mému vedoucímu, panu Ing. Matěji Dostálovi, Ph.D. za konzultace, ochotu a pomoc při vypracování této práce. Moc bych chtěla také poděkovat svému příteli za podporu, také svému kamarádovi Fandovi za pomoc s korekcí textu a za velkou pomoc a podporu při studiích. Také všem svým kamarádům a rodině.

Prohlášení

Prohlašuji, že jsem tuto práci vypracovala samostatně a že jsem uvedla všechny použité zdroje a to v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze, 20. května 2023

Abstrakt

Tato práce se zabývá kreslením grafů a měřením výsledného vykreslení pomocí metrik estetičnosti. Nejprve seznámením se se třemi vykreslovacími algoritmy, poté se samotnými metrikami. Práce obsahuje také implementaci, za pomoci grafické knihovny OpenGL, jak těchto třech vykreslovacích algoritmů, tak i vybraných metrik. Na závěr obsahuje porovnání vykreslení stejných grafů těmito třemi algoritmy. A to jak podle rozdílných hodnot metrik, tak i porovnání podle názoru minimálně deseti pozorovatelů.

Klíčová slova: grafy, kreslení, teorie grafů, metriky estetičnosti kreslení grafů

Vedoucí: Ing. Matěj Dostál, Ph.D.
Katedra matematiky

Abstract

This work is about graph drawing and measuring rendered graph using aesthetic metrics. First by getting to know three different rendering algorithms, then the metrics themselves. The work also includes the implementation, with the help of the OpenGL graphics library, of these three rendering algorithms and selected metrics and also it´s description. Finally, it contains a comparison between implemented algorithms by rendering the same graph(s), according to the different values of metrics, as well as a comparison according to the opinion of at least ten observers.

Keywords: graph, drawing, graph theory, metrics for graph drawing aesthetics

Title translation: Graph drawing and aestetics metrics

Obsah

Obrázky

ctuthesis t1606152353

Kapitola 1

Úvod

Kreslení grafů je odvětví, jenž kombinuje matematiku, konkrétně teorii grafů, a počítačovou vědu. Zabývá se reprezentací a následnou vizualizací grafů za pomoci grafických knihoven a aplikací. Výsledkem je dvourozměrné zobrazení grafu, který může vycházet z rozličných dat, kupříkladu z analýzy sociálních vztahů, bioinformatiky a mnoha dalších [1].
V této práci se nejprve seznámíme se základními pojmy z matematického odvětví teorie grafů, následně se podíváme na některé vykreslovací metody, konkrétně na algoritmy, jenž vykreslují graf pomocí minimalizace energie. Také se seznámíme se spektrálními vlastnostmi matic grafů a jejich použití pro vykreslení. Představíme si a uvedeme také metody měření estetičnosti kreslení grafu a vybereme pár vhodných k implementaci. Nakonec se podíváme na samotnou implementaci, jejímž cílem bylo implementovat již zmíněné algoritmy na vykreslování a i vhodné metriky daných vykreslení. Na závěr si uvedeme deset příkladů grafů, které jsme za pomoci algoritmů vykreslili, a porovnáme různá vykreslení s názory alespoň deseti účastníků.

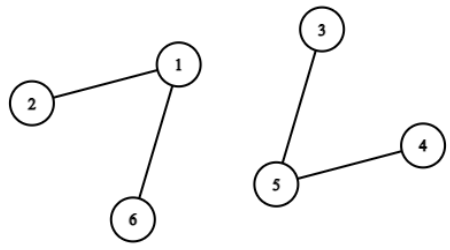
Kapitola 2

Základy z teorie grafů a matematická reprezentace grafů

Na začátek bychom se měli podrobněji seznámit s pojmy a definicemi, které budeme dále v textu využívat.
V matematice můžeme graf chápat jako strukturu, která zobrazuje nějakou množinu objektů a vztahy mezi nimi. Jednotlivé prvky množiny jsou znázorněny jako vrcholy grafu, vztahy mezi nimi hranami. Nyní si uvedeme matematickou definici grafu a připomeneme si nějaké základní pojmy a definice z teorie grafů. Společně s nimi si uvedeme i pár příkladů. Téměř všechny definice jsou kombinací literatury z [2].

Definice 2.1. Graf G, nazývaný též jednoduchý graf či také obyčejný graf je uspořádaná dvojice $G = (V, E)$, kde V je neprázdná množina vrcholů a E je množina hran - množina (některých) dvouprvkových podmnožin množiny V.

Příklad 2.2. Mějme graf $G = (V, E)$ s vrcholy $V = \{1, 2, 3, 4, 5, 6\}$ a hranami mezi nimi vedoucími $E = \{\{1, 2\}, \{1, 6\}, \{3, 5\}, \{4, 5\}\}$. Tento graf je znázorněn na obrázku .



Obrázek 2.1: Příklad grafu.

Definice 2.3. Mějme graf $H = (V_h, E_h)$ a graf G=(V_g, E_g). Graf G je podgrafem grafu H, jestliže splňuje následující podmínky:

1. $V_g \subseteq V_h$.
2. $H_g \subseteq H_h$.
3. Hrany grafu G mají oba dva vrcholy v G.

Definice 2.4. Nechť je dán graf $G = (V, E)$ s n vrcholy a m hranami. Maticí sousednosti pak nazýváme takovou matici $M = (a_{i,j})$ s rozměry $n \times n$, jenž splňuje následující:

$$a_{i,j} = \begin{cases} 1 & \text{, když } \{i,j\} \in E \\ 0 & \text{, když } \{i,j\} \notin E \end{cases}$$

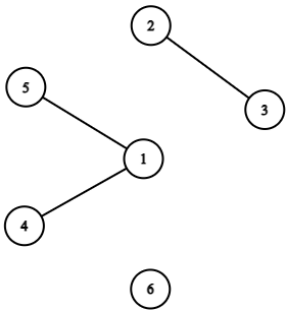
Definice 2.5. Je dán neorientovaný graf. Cestou v grafu nazýváme posloupnost vrcholů a hran $P = (v_1, e_1 ... v_{k-1}, e_{k-1}, v_k)$ takovou, pro kterou platí $e_i = \{v_i, v_{i+1}\}$ a zároveň platí, že $v_i \neq v_j$ pro $i \neq j$.

Jedná se tedy o posloupnost vrcholů takovou, pro níž platí, že v grafu existuje hrana z jednoho daného vrcholu do jeho následníka a zároveň se žádné dva vrcholy a ani hrany neopakují.

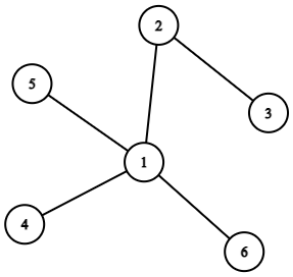
Definice 2.6. Graf (neorientovaný i orientovaný) nazýváme souvislý, jestliže pro každou dvojici jeho vrcholů u a v existuje cesta z u do v.

Graf je nesouvislý, pokud tomu tak není. Můžeme rovněž říci, že souvislý graf má vždy pouze jednu jedinou komponentu a nesouvislý graf jich má, či je složen z více. Komponentou grafu rozumíme maximální souvislý podgraf nějakého grafu G.

Příklad 2.7. Mějme dva grafy – graf $H = (V_h, E_h)$ a graf $G = (V_g, E_g)$, přičemž graf H je souvislý a G nesouvislý s třemi komponentami.



Obrázek 2.2: Příklad nesouvislého grafu G s třemi komponentami.



Obrázek 2.3: Příklad souvislého graf H, který má jen jednu komponentu.

Definice 2.8. Nakreslením grafu $G = (V, E)$ v rovině rozumíme přiřazení, ve kterém každému vrcholu v $\in$ V přiřadíme bod $b(v) \in \mathbb{R}^2$ a každé hraně $e \in E$ přiřadíme oblouk $o(e)$ se dvěma koncovými body $b(u)$ a $b(v)$, přičemž body u a v jsou vrcholy, mezi nimiž se hrana nachází. Zároveň předpokládáme, že vrcholové zobrazení je prosté a také to, že žádný z bodů $b(v)$ není nekoncovým bodem příslušného oblouku $o(e)$.

Definice 2.9. Ohodnoceným grafem (G, w) nazýváme graf G spolu s nějakou reálnou funkcí $w : E(G) \to (0, \infty)$, která přiřadí hraně grafu G reálné číslo $w(e)$. Toto číslo se poté nazývá se nazývá ohodnocení hrany nebo její váha (anglicky „weight“).

Definice 2.10. Mějme neorientovaný graf G s n vrcholy. Matice $D \in R^{n,n}$ s velikostí $n \times n$ grafu G se rovná

$$D_{i,j} = \begin{cases} degree(v_i), & \text{když } i = j \\ 0 & \text{jinak} \end{cases}$$

Přičemž $deg(v_i)$ je stupeň vrcholu v, tj. počet hran, které do daného vrcholu v vedou [3].

Definice 2.11. Mějme matici sousednosti grafu G značenou M a diagonální matici stupňů vrcholů D odpovídající stejnému grafu G. Laplaceova matice je čtvercová matice značená L, která se poté rovná rozdílu matic D a M. To znamená

$$L = D - M$$

[3].

Vidíme, že Laplaceova matice má na diagonále počet sousedních vrcholů daného vrcholu a na řádku a sloupci -1 tam, kde je daný vrchol spojený hranou s jiným vrcholem. Pokud bychom uvažovali Laplaceovu matici váženého grafu, tak změníme matici D a M tak, že u D budou na diagonále součty vah hran, které vedou do vrcholu a v M budou namísto „1“ na příslušné pozici váhy hran.

Kapitola 3

Spektrální vlastnosti matic grafů a kreslení grafu pomocí těchto vlastností

V této kapitole si řekneme něco ke kreslení grafu pomocí vlastních čísel a vlastních vektorů Laplaceovy matice. Nejprve se seznámíme obecně se spektrálními vlastnostmi Laplaceovy matice a pak se podíváme na samotné vykreslování. Je důležité zmínit, že v našem případě počítáme s tím, že je graf souvislý. Jelikož pokud by byl nesouvislý, nebyla by násobnost vlastních čísel rovných nule jedna, nýbrž j, podle počtu komponent. Také je důležité, že uvažujeme jen nevážené grafy.

3.1 Spektrální vlastnosti Laplaceovy matice

Na začátek si uvedeme, co je to spektrum matice. Spektrum čtvercové matice je množina všech jejích vlastních čísel. Laplaceova matice je dle definice symetrická, singulární a součet každého ze sloupců či řádků je roven nule [3]. Symetrická je z toho důvodu, že vzniká ze symetrické matice M a D. Jelikož D vypadá tak, že má čísla jen na diagonále a jinak nuly, je výsledná matice L symetrická. Součet sloupců a řádků je nula, protože uvažujeme neorientované grafy, tudíž se stupeň vrcholu v_i musí rovnat počtu hran, které z něho vychází. Toto platí jen pro nevážené grafy, které uvažujeme. To, že je singulární, vyplývá z Frobeniovy věty. Jelikož je matice reálná a symetrická, má díky tomu několik vlastností [3].

Tvrzení 3.1. *Vlastní čísla Laplaceovy matice jsou reálná a vlastní vektory jsou ortogonální.*

Tvrzení 3.2. *Jelikož je Laplaceova matice singulární, je rovno jedno vlastní číslo λ nule a vlastní vektor příslušný tomuto vlastnímu číslu je* $(1, \ldots, 1)^T$.

Další důležitou vlastností je, že je Laplaceova matice pozitivně semidefinitní, tudíž jsou všechna vlastní čísla nezáporná. Můžeme také usoudit, že právě nejmenší vlastní číslo λ_1 bude rovno nule [3].
Po souhrnu základních vlastností z vět a definic se můžeme podívat na použití oněch vlastností k vykreslení grafu.

3.2 Kreslení grafu pomocí vlastních vektorů Laplaceovy matice grafu

Než se dostaneme k samotnému vykreslování, je vhodné si ještě uvést Laplaceovu kvadratickou formu spojenou s grafem. Laplaceova matice grafu G s označením L_G je navržena tak, aby zachycovala Laplaceovu kvadratickou formu:

$$x^T L_G x = \sum_{a,b \in E} w_{a,b}(x(a) - x(b))^2 \tag{3.1}$$

Kde a,b je hrana mezi vrcholy, $w_{a,b}$ je její váha, $x \in R^V$ a $x(a)$ je a–tý element vektoru x, neboli element x odpovídající vrcholu a.
Nyní se můžeme podívat na samotný způsob vykreslování grafu pomocí vlastních čísel a vlastních vektorů Laplaceovy matice grafu. Budeme vycházet z myšlenky K. M. Halla z roku 1970 [4]. Nejprve se podíváme na jednorozměrné zobrazení – budeme se snažit vykreslit graf na přímku. Mějme vektor $x \in R^V$, který nám reprezentuje přiřazení reálného čísla každému vrcholu grafu. Zároveň chceme, aby byly sousední vrcholy blízko u sebe. Hall navrhl, abychom vybrali x takové, které minimalizuje již zmíněnou kvadratickou formu tj.

$$x^T L_G x = \sum_{a,b \in E} (x(a) - x(b))^2 \tag{3.2}$$

Můžeme si všimnout, že neuvažujeme váhy hran, jelikož jsme se rozhodli zabývat se jen neváženými grafy.
Bohužel ale nemůžeme uvažovat všechny reálné hodnoty, jelikož některé z nich mohou způsobit problémy, jenž vedou k degeneraci řešení. Jedním z problémů může být to, že se všechna řešení, to znamená přiřazení reálného čísla vrcholu, namapují například do 0. Právě kvůli tomuto problému si zavedeme podmínku, že norma x bude 1

$$\sum_{a \in V} x(a)^2 = \|x\|^2 = 1 \tag{3.3}$$

Tímto eliminujeme namapování do 0, jelikož norma nulového vektoru nebude nikdy jedna. Dalším problémem je to, že se všechny vrcholy zobrazí do $\frac{1}{\sqrt{n}}$, kde n je V. Kvůli tomuto si zavedeme i druhou podmínku:

$$\sum_{a \in V} x(a) = 1^T x = 0 \tag{3.4}$$

Což znamená, že součet prvků vektoru x musí být roven vždy nule, což v případě namapování do $\frac{1}{\sqrt{n}}$ nastat nemůže, jelikož součet prvků vektoru nedá 0. Tyto dvě podmínky dohromady nám částečně zabraňují degradaci řešení. Řešení, které splňuje podmínky a minimalizuje výraz $x^T L_G x$, podle textu poté odpovídá jednotkovému vlastnímu vektoru Laplaceovy matice ψ_2, který je příslušný nejmenšímu nenulovému vlastnímu číslu λ_2. Nejmenší vlastní číslo, tedy 0, respektive vlastní vektor využít nemůžeme, jelikož příslušný vlastní vektor nesplňuje podmínky (to vidíme dle spektrálních vlastností Laplaceovy matice výše).
Pro naše účely ale nestačí nalézt řešení a vykreslení v jedné dimenzi. Oním řešením dvoudimenzionálního grafu je

$$\sum_{(a,b) \in E} \| \begin{bmatrix} x(a) \\ y(a) \end{bmatrix} - \begin{bmatrix} x(b) \\ y(b) \end{bmatrix} \|^2 \tag{3.5}$$

Vidíme, že stačilo přiřadit každému vrcholu dvě reálná čísla místo jednoho. Výraz můžeme dále přepsat na

$$\sum_{a,b \in E} (x(a) - x(b))^2 + (y(a) - y(b))^2 = x^T L_G x + y^T L_G y \tag{3.6}$$

Opět musíme uvažovat podmínky (3.3) a (3.4), abychom částečně omezili degeneraci řešení.
První podmínkou je opět $\|x\|^2 = 1$ a $\|y\|^2 = 1$ a druhou $1^T x = 0$ a $1^T y = 0$. Zde se ale vyskytl nový problém a to ten, že x a y se budou rovnat. Toto by způsobilo, že se graf vykreslí na přímku (diagonálu). Hall tedy navrhl, aby x bylo ortogonální na y. Ve dvou dimenzích vidíme jako řešení vlastní vektor ψ_3 odpovídající λ_3, jelikož je kolmý na ψ_2 a zároveň splňuje podmínky [4]. Proč volíme zrovna λ_3 vychází z Courantovy–Fisherovy věty, která vysvětluje, jak minimalizovat hodnotu kvadratické formy (3.1) v závislosti na vlastních číslech symetrické matice, v našem případě L.
Výsledkem tohoto vykreslení mohou být hezky vypadající grafy. Toto ale platí pouze pro určitou množinu grafů. Pro mnoho grafů totiž není tento typ vykreslování vhodný [4]. Na samotné příklady vykreslení se podíváme později v kapitole „Výsledky“. Limitace této metody jsou zmíněny například v knize Spectral and Algebraic Graph Theory [4].

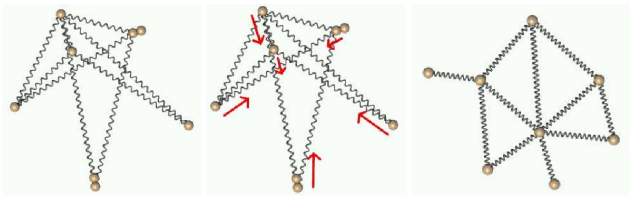
Kapitola 4

Algoritmy silově řízeného kreslení grafů obecně a Eadesův algoritmus

V této kapitole se seznámíme s tím, co to jsou a jak vypadají obecně algoritmy silově řízeného kreslení grafů. Uvedeme si také jejich výhody a nevýhody. Dále se seznámíme s Eadesovým algoritmem. Uvedeme si jak zmíněný algoritmus, tak i vzorce, ze kterých vychází.

4.1 Algoritmy silově řízeného kreslení grafů

Účelem těchto algoritmů je umístit vrcholy a hrany dvojrozměrného či trojrozměrného grafu tak, aby měly všechny hrany nejlépe stejnou, či alespoň velmi podobnou délku a aby docházelo k co možná nejmenšímu počtu překřížení hran. K docílení tohoto rozpoložení se přiřadí vrcholům a hranám síly, které na ně navzájem působí. Každý vrchol, popř. hrana, se poté snaží dostat na pozici takovou, aby síly, působící na graf, byly minimální.



Obrázek 4.1: Vizualizace funkčnosti algoritmů silově řízeného kreslení grafů [15].

Typicky jsou využívány, pro simulaci působících sil a simulaci následného přitahování a odpuzování vrcholů, fyzikální zákony, například Hookeův zákon nebo Coulombův zákon. Toto ovšem nemusí být pravidlem.
Dále se podíváme na výhody těchto algoritmů. První z výhod je dobrá kvalita výsledků. Tím rozumíme to, že u grafů malých a středních velikostí (pro velké grafy se obecně tento typ algoritmů nehodí), dosahujeme stejně dlouhých hran, minimalizaci překřížení, uniformního rozložení vrcholů a symetrie, což je přesně to, co je chtěné. Za malé a střední grafy považujeme grafy takové, které mají maximálně zhruba 500 vrcholů. Další výhodou je flexibilita, jelikož lze tyto algoritmy poměrně snadno upravovat a rozšiřovat tak, aby vykreslené grafy splňovaly i další estetická kritéria. Tato „volnost“ z nich dělá nejuniverzálnější třídu algoritmů pro kreslení grafů. Dále jsou tyto algoritmy celkem intuitivní, jelikož častokrát vychází z fyzikálních zákonů, které se vztahují na prosté objekty, například pružinky v našem případě. Dá se tedy dobře odhadnout a předpovídat jejich chování a také je to činí většinou celkem snadno pochopitelnými. Od tohoto se odvíjí částečně i další výhoda, kterou je relativně snadná implementace ve srovnání s jinými typy algoritmů pro vykreslování grafů. Rovněž se považuje za výhodu interaktivnost takto vykreslených grafů. Je snadné pozorovat, jak se z původního grafu vykreslí jiný, díky již zmíněným vlastnostem na pohled hezký, esteticky vypadající graf. Právě kvůli této vlastnosti jsou ideálními adepty například na on–line nástroje na vykreslování grafů. Jednou z posledních výhod, kterou si zde uvedeme jsou silné teoretické základy.
Jelikož většinou není nikdy nic úplně dokonalé, uvedeme si i pár nevýhod. První z nich je celkem velká časová náročnost výpočtu. Typická časová složitost těchto algoritmů obecně je $O(n^3)$, kde n značí počet vrcholů vstupního grafu. Je to kvůli tomu, že počet iterací je odhadován jako lineární a v každé iteraci je třeba navštívit všechny páry vrcholů a vypočítat jejich odpudivé síly. Jelikož jsou ale tyto odpudivé síly pro vrcholy lokální, je zde možnost vykreslovaný graf rozdělit tak, aby byly brány například pouze sousední vrcholy. Tudíž můžeme časovou náročnost výpočtu pro některá vykreslení snížit. Díky optimalizacím a úpravám tedy můžeme vykreslit i větší grafy. Jako druhou nevýhodu si uvedeme kolikrát špatný výběr lokálních minim. Z předchozího popisu těchto algoritmů víme, že výsledný vykreslený graf má minimalizované síly, působící mezi vrcholy. Tato minimalizace je ale častokrát pouze lokální minimum, jenž může být ale mnohem horší, než je obecné globální minimum, což poté může vést k nízké kvalitě vykresleného grafu. Zároveň tento problém narůstá ve chvíli, když zvýšíme počet vrcholů grafu. Tento problém můžeme částečně vyřešit například kombinovaným použitím více algoritmů, jelikož u hodně algoritmů z této skupiny záleží na vstupních datech, respektive základním nastaveným rozpoložení vrcholů. Jakožto příklad si můžeme uvést Kamada—Kawai algoritmus, kterým nejprve rychle vygenerujeme základní rozpoložení a poté Fruchterman–Reingoldův algoritmus, kterým poupravíme pozice sousedních vrcholů. Dalším řešením na eliminaci tohoto problému a nalezení globálního minima je víceúrovňový přístup (multilevel approach) [5].

4.2 Eadesův algoritmus

Eadesův algoritmus (algorithm of Eades) pochází z roku 1984. Řadí se mezi výše zmíněné algoritmy silově řízeného kreslení grafů. Algoritmus funguje tak, že nahradí vrcholy pomyslnými kroužky či úchytkami a každou hranu pružinkou. Graf se vždy nachází v nějakém počátečním stavu. To znamená, že všechny vrcholy mají nastavené počáteční souřadnice a jsou definovány hrany mezi nimi. Pružinky, respektive jejich síly na sebe působící, poté „posouvají" vrcholy, tak, aby došlo k minimalizaci sil a rovnovážnému stavu celého systému. Síly pružinek, působící na vrcholy, jsou definované jako $c_1 * log(\frac{d}{c_2})$, přičemž d je délka pružinky a c_1, c_2 jsou konstanty [6]. Jelikož se ukázalo, že podle původní myšlenky Hookeova zákona, ze kterého Eadesův algoritmus vychází, je pružinka (lineární) moc silná, pokud jsou vrcholy daleko od sebe, nahradil se původní lineární výpočet logaritmem, který tento problém řeší. Dále je potřeba vypočítat sílu, kterou se nesousedící vrcholy navzájem odpuzují. Ta se vypočítá jako $\frac{c_3}{d^2}$, kde c_3 je konstanta a d je vzdálenost mezi vrcholy [6]. Pro představu si nyní uvedeme pseudoalgoritmus, jenž vychází ze článku Force-Directed Drawing Algorithms a popisuje námi použitý algoritmus:

Data: Graph G
place vertices of G in random locations;
i=0;
while $i<M$ **do**
    calculate the force on each vertex;
    move the vertex c4 $\star$ (force on vertex);
    i++;
    draw graph on CRT or plotter;
end

Algorithm 1: Algorithm SPRING

Je také důležité zmínit, že algoritmus se hodí pro grafy, jenž mají maximálně 30 vrcholů. Co se konstant týče, v originálním textu byly použity hodnoty $c_1 = 2$, $c_2 = 1$, $c_3 = 1$, $c_4 = 0.1$ a $M = 100$ [6]. Toto ale nemusí býti pravidlem, konstanty můžeme zvolit i jiné.
Jak už jsme zmínili u algoritmů silově řízeného kreslení grafů obecně, je výsledkem tohoto algoritmu, dle pohledu pozorovatele, na pohled esteticky hezký graf, jelikož délky všech hran jsou většinou stejné, nebo alespoň velmi podobné a rozpoložení vrcholů působí symetricky, což je pro lidské oko příjemné na pohled [5].

Kapitola 5

Harmonické funkce, pružinkový systém a řešení Laplaceových lineárních rovnic

V této kapitole si uvedeme teorii k dalšímu z implementovaných algoritmů, jenž také vykresluje graf pomocí minimalizace sil. Nejprve se seznámíme s harmonickými funkcemi, poté si představíme základní myšlenky algoritmu a nakonec se podíváme na řešení systému Laplaceových lineárních rovnic.

5.1 Harmonické funkce

Uvažujme souvislý vážený graf $G = (V, E, w)$ s vrcholy $v \in V$ a hranami $e \in E$. Symbol w značí váhy hran.
Dále uvažujme neprázdnou množinu takzvaných mezních vrcholů $B \subseteq V$ (anglicky Boundary vertices). Mějme také druhou sadu vrcholů značenou S, pro kterou platí, že $S = V \setminus B$.
Funkce $x : V \to \mathbb{R}$ je harmonická ve vrcholu a právě tehdy, když se hodnota funkce x ve vrcholu a rovná váženému průměru hodnot jeho sousedů. To znamená:

$$x(a) = \frac{1}{d_a} \sum_{b \sim a} w_{a,b} x(b) \tag{5.1}$$

Zde d_a značí stupeň vrcholu a pro vážený graf (to znamená součet vah jeho hran, jejichž je a součástí), w je váha hrany a $b \sim a$ značí hranu mezi vrcholy. Pokud je funkce harmonická pro všechna $a \in S$, říkáme, že je funkce harmonická na S [4].

5.2 Pružinkový systém

Nyní se seznámíme s hlavní myšlenkou algoritmu. Základní myšlenka tohoto algoritmu je podobná již zmíněnému Eadesovu algoritmu, přesto je výpočet zcela odlišný. Představme si, podobně jako u Eadesova algoritmu, že každá hrana e vizualizovaného grafu $G = (V, E)$ je pružinka, jenž je uchycena mezi vrcholy a a b, kde a a b jsou koncové vrcholy hrany. Všechny pružinky jsou spojeny dohromady ve vrcholech, takto vytvářející celistvou strukturu. Nyní si vybereme množinu mezních vrcholů $B \subseteq V$ a každý mezní vrchol z B zafixujeme pevně na místě, respektive zafixujeme jeho souřadnice. Poté budeme sledovat, co se stane se zbylými vrcholy, které by se, silami na ně působícími, měly posunout do místa, jenž minimalizuje energii na ně působící. Toto chování a síly popisuje Hookův zákon, ze kterého budeme vycházet.
Na začátek si řekneme, že každá pružinka je ideální pružinou a její pružinová konstanta (anglicky spring constant) je 1. Pružinová konstanta reprezentuje tvrdost pružiny nebo elastického materiálu. Pokud je G vážený graf, pružinovou konstantu reprezentujeme váhou odpovídající hrany.
Hookeův zákon nám říká, že síla, která působí ve vrcholu a, je ve směru b a je také úměrná vzdálenosti mezi a a b. Řekněme, že pozice každého vrcholu a je $x(a)$. Pak síla, kterou pružinka působí mezi vrcholy a a b na a je:

$$x(b) - x(a) \tag{5.2}$$

V rovnovážném stavu musí všechny nezafixované vrcholy sečíst síly na ně působící na nulu, takže:

$$\sum_{b \sim a} (x(b) - x(a)) = 0 \tag{5.3}$$

Úpravami dostáváme:

$$\sum_{b \sim a} x(b) = d_a x(a) \tag{5.4}$$

$$x(a) = \frac{1}{d_a} \sum_{b \sim a} x(b)$$

Z toho vidíme, že v rovnovážném stavu musí pozice každého nezafixovaného vrcholu $a \in S$ odpovídat průměru hodnot pozic jeho sousedů. Pokud je námi uvažovaný graf vážený, platí, že každý nezafixovaný vrchol se rovná váženému průměru hodnot jeho sousedů, to znamená:

$$x(a) = \frac{1}{d_a} \sum_{b \sim a} w_{a,b} x(b)$$

Z toho vidíme, že x je harmonická na $V \setminus B$, tedy na S [4].

5.3 Laplaceovy lineární rovnice

Teď si ukážeme, že předchozí rovnice pro každý vrchol mají řešení pro každý souvislý graf G a neprázdnou množinu B. Začneme tím, že si vynásobíme rovnici (5.1) hodnotou d_a. Díky tomu dostaneme:

$$d_a x(a) = \sum_{b \sim a} w_{a,b} x(b) \tag{5.5}$$

Nyní převedeme $\sum_{b\sim a} w_{a,b}x(b)$ na (pro nás levou) stranu rovnice k $x(a)$ tak, aby byla na druhé straně 0. Díky tomu nyní máme:

$$d_a x(a) - \sum_{b \sim a} w_{a,b} x(b) = 0 \tag{5.6}$$

Vidíme, že rovnice odpovídá řádku Laplaceovy matice odpovídající vrcholu a. Naše řešení by mělo tedy odpovídat soustavě rovnic, které dostaneme z Laplaceovy matice, respektive podmatice, indexované vrcholy $V \setminus B$.
Nyní se přesuneme k samotnému výpočtu. Souřadnice každého vrcholu a z B zafixujeme na $f(a)$. Snažíme se totiž (v našem případě) o to, najít zobrazení $x : V \to \mathbb{R}^2$. Toto zobrazení má předpoklad, že některé souřadnice vrcholů, konkrétně B, jsou již pevně zafixovány na místě. Zobrazení $f : B \to \mathbb{R}^2$ přiřadí všem vrcholům a z B ony fixní souřadnice, $f(a)$ značí přiřazení souřadnic vrcholu a. Na základě toho můžeme nyní přepsat rovnici pro každé a z $V \setminus B$ na:

$$d_a x(a) - \sum_{b \notin B : (a,b \in E)} w_{a,b} x(b) = \sum_{b \in B : (a,b \in E)} w_{a,b} f(b) \tag{5.7}$$

Pro připomenutí si množinu vrcholů, které nejsou mezními, označíme jako S, přičemž $S = V \setminus B$. Při pohledu na rovnici (5.7) nyní vidíme, že si ji můžeme přepsat jako:

$$L(S,S)x(S) = r \tag{5.8}$$

Přičemž r je:

$$r = M(S,:)f$$

S tím, že $L(S,S)$ je podmatice Laplaceovy matice velikosti $S \times S$ s řádky a sloupci indexovanými S, $x(S)$ je vektor s elementy vektoru x na indexech S, $M(S,:)$ je matice sousednosti velikosti $S \times V$, s tím, že označení „:“ značí všechny sloupce původní matice. Můžeme zapsat nyní podmínku, že prvky B jsou zafixovány na f pomocí:

$$x(B) = f(B)$$

[4].
Nyní vidíme, že abychom dosáhli vykreslení vrcholů na příslušné pozice, stačí vyřešit soustavu lineárních rovnic (5.8).

Kapitola 6

Metriky pro měření estetičnosti kreslení grafů

V této kapitole si nejprve obecně zavedeme metriky pro měření estetičnosti kreslení grafů, poté si uvedeme příklady jednotlivých konkrétních metrik. Také se podíváme na metriky vhodné k implementaci do našeho programu.
Mějme graf $G = (V, E)$, který má n vrcholů a m hran. Pro naše účely uvažujme graf souvislý s minimálně jednou hranou. Graf G můžeme vykreslit či vyrenderovat na 2D rovinu, kde každý z vrcholů z V má dvojici souřadnic x a y, které odpovídají umístění vrcholu na rovině. U dvojice souřadnic x a y, přiřazené každému z vrcholů také předpokládáme, že žádné dvě kombinace x a y nejsou stejné – zobrazené vrcholy se nepřekrývají. K vykreslení můžeme použít různé algoritmy na vykreslování a rozvržení grafu (algoritmus vždy přiřadí určitému vrcholu jeho souřadnice). Z vykresleného grafu pak můžeme zkoumat „estetičnost“ jeho vykreslení – to znamená jak „hezky“ je graf vykreslen vzhledem ke zvoleným metrikám. Použití metrik nekončí u určování estetické kvality kresby grafu, ale hodí se například i na definování nákladové funkce (cost function) genetických algoritmů či pro programy simulovaného žíhání (simulated annealing programs). Jelikož se algoritmy na vykreslování liší, jsou vrcholy častokrát umístěny na jiných pozicích a tudíž se bude lišit i hodnota metrik pro graf [7].
Ještě než se dostaneme dále, dovolila bych si uvést drobnou poznámku ke genetickým algoritmům a programům simulovaného žíhání, abychom se s pojmy lépe seznámili a představili si je.
Genetický algoritmus se řadí mezi takzvané evoluční algoritmy, které jsou inspirovány přirozeným výběrem živočišných druhů [8]. Používá se pro řešení složitých optimalizačních úloh, jenž nelze řešit metodami kterými jsou například gradientní metoda nebo lineární programování [9].
Simulované žíhání slouží k aproximování globálního optima dané funkce. Je

vhodné pro funkce, jenž obsahují hodně lokálních minim a pro které by špatně fungovalo řešení pomocí gradientní metody kupříkladu [10].
Nyní se podíváme na konkrétní metriky, jimiž jsou překřížení (crossing) hran, ohyb (bends), symetrie (symmetry), minimální úhel (minimum angle), ortogonalita (orthogonality) a vzestupný tok (upward flow).

6.1 Crossing neboli překřížení

První metrikou, se kterou se seznámíme, je překřížení. Překřížení, značené $\aleph_c$, pro graf G je založeno na c, jenž symbolizuje součet počtu překřížení v grafu. Samotné překřížení je definované jako místo, respektive bod, v němž se dvě hrany a a b protínají.
Jelikož má hodnota metriky vracet číslo v intervalu $\langle 0, 1\rangle$, musíme číslo vyškálovat podle maximálního počtu všech možných překřížení – respektive vydělit počtem oněch všech možných překřížení. K tomu je zapotřebí nejprve vypočítat úplně všechna překřížení, která mohou nastat. Tuto hodnotu značí c_{all} a je definováno jako:

$$c_{all} = \sum_{i=1}^{m'}(i-1) = \frac{m'(m'-1)}{2} \tag{6.1}$$

Přičemž m' jsou hrany grafu G. Uvažujeme tedy, že každou hranu překříží úplně každá jiná hrana. Je ale možné, že některé z hran je nemožné vykreslit tak, aby se protínaly. Příkladem mohou být dvě hrany vedoucí ze stejného vrcholu. Tyto nevykreslitelné hrany, $c_{impossible}$, se vypočítají následovně:

$$c_{impossible} = \frac{1}{2}\sum_{j=1}^{n'} \mathsf{degree}(u_j)\mathsf{degree}(u_j) - 1 \tag{6.2}$$

Celková suma všech možných překřížení, značená c_{mx}, odpovídá rozdílu těchto dvou hodnot, jelikož chceme nevykreslitelné hrany eliminovat.
Tudíž $c_{mx} = c_{all} - c_{impossible}$. Touto hodnotou c_{mx} nyní budeme chtít dělit samotný součet překřížení c. Jelikož ale může nastat, že hodnota c_{mx} bude nula (v případě, že se $c_{all} = c_{impossible}$) a nulou dělit nelze, rozdělíme výpočet na dva případy: $c_{mx} > 0$ a $c_{mx} = 0$ [7].
Výsledná hodnota metriky se tedy rovná:

$$\aleph_c = 1 - \begin{cases} \frac{c}{c_{mx}} & \text{když } c_{mx} > 0. \\ 0 & \text{jinak} \end{cases} \tag{6.3}$$

Můžeme si tedy všimnout, že čím je číslo blíže 1, tím méně obsahuje graf překřížení [7].

6.2 Ohyby (Bends)

Ohybem hrany myslíme takovou hranu, jenž se nejeví jako rovná linka spojující dva vrcholy, nýbrž je v nějaké části ohnuta do jiného směru. Počet ohybů se značí b a je vypočítán pomocí následujícího vzorce:

$$b = n' - n = m' - m \tag{6.4}$$

Kde n' značí počet vrcholů grafu poté, co převedeme ohyby grafu na vrcholy, n je počet vrcholů původního grafu G, m' je počet hran a segmentů, které vzniknou ohybem hrany a m je počet hran původního grafu G. Oproti překřížení nemůžeme, abychom dostali opět číslo z intervalu 0 a 1, dělit všemi možnými ohyby, jelikož jich může existovat nekonečně mnoho. Vydělíme tedy b celkovým součtem všech částí jednotlivých hran. Díky tomu dostáváme, že

$$b_{avg} = \frac{m' - m}{m'} \tag{6.5}$$

Pokud $b_{avg} = 0$, graf neobsahuje žádné ohyby. V opačném případě se b_{avg} blíží k 1 (1 být nikdy nemůže, jelikož nemůžeme mít nekonečný počet ohybů), ohybů existuje mnoho.
Výsledná hodnota metriky se rovná:

$$\aleph_b = 1 - b_{avg} \tag{6.6}$$

a definuje „bezohybovost“ grafu, to znamená, že čím je výsledné číslo blíže k 1, tím má graf méně ohybů [7].

6.3 Minimální úhel

Metriku minimálního úhlu definujeme jako průměrnou odchylku úhlů sousedních hran, vycházejících z jednoho vrcholu, od ideálního minimálního úhlu. Odchylku značíme d a vypočítáme jí následovně:

$$d = \frac{1}{n} \sum_{i=1}^{n} \left| \frac{\vartheta_i - \theta_{imin}}{\vartheta_i} \right| \tag{6.7}$$

Parametr ϑ_i značí ideální úhel u i–tého vrcholu. Ideálním úhlem rozumíme minimální (maximální) úhel $\vartheta_i = \frac{360}{\mathsf{degree}(v_i)}$ a θ_{imin} je minimální (skutečný naměřený) úhel mezi přilehlými hranami, které vychází z i–tého vrcholu.
Samotná metrika je definována jako:

$$\aleph_m = 1 - d \tag{6.8}$$

Metrika $\aleph_m$ je z intervalu $\langle 0, 1 \rangle$ [7].

6.4 Symetrie

Výpočet metriky symetrie bude opět vracet číslo z intervalu $\langle 0, 1 \rangle$. Mohlo by nás totiž napadnout, že jen proložíme graf osou a budeme zkoumat jeho souměrnost, jenž vyprodukuje 0 (pokud je graf nesouměrný) či 1, pokud je souměrný. Algoritmus na výpočte metriky symetrie ale funguje lehce odlišně od této myšlenky. Celkový postup výpočtu je uveden níže.

1. Upravíme graf G na graf G' a to tak, že nahradíme ohyby v grafu vrcholy. To samé uděláme i s překříženími.
2. Inicializujeme proměnné TOTAL_SYM a TOTAL_AREA a nastavíme jejich hodnotu na 0.
3. Pro graf G' vygenerujeme množinu všech možných os (pro všechny možné páry vrcholů).
4. Pro každou osu A určíme, zda jsou podgrafy G', vzniklé rozdělením osou A, osově symetrické. Aby podgraf existoval, je potřeba, aby tam bylo minimálně tolik zrcadlených hran jako je proměnná TRESHOLD. Koncové vrcholy hran jsou zrcadleny a je zjišťováno, zda mají svůj zrcadlený odraz pomocí proměnné TOLERANCE, která udává maximální odchylku, při které je ještě zrcadlený vrchol odrazem.
5. Pro každý existující symetrický podgraf vypočítáme hodnotu jeho symetrie SUB_SYM v závislosti na tom, jestli jsou či nejsou odražené vrcholy stejného typu. To znamená, že pro každý pár zobrazených vrcholů zjistíme, jestli jsou původní nebo vznikly nahrazením ohybů či překřížení. Pokud jsou stejného typu, nastavíme proměnnou P nebo Q na 1, pokud ne, nastavíme hodnotu na proměnnou FRACTION, která je v rozmezí 0 a 1 a určuje, jak moc bereme v potaz to, že vrchol není původní ale vytvořen z ohybů či překřížení. Následně vypočítáme hodnotu celkové podsymetrie–TOTAL jako $P * Q$ a jelikož chceme průměrnou hodnotu symetrie pro každý pár hran, vydělíme proměnnou TOTAL počtem párů hran v podgrafu. Díky tomu nám vznikne hodnota „podsymetrie“ pro podgraf, značená SUB_SYM.
6. Dále vypočítáme pro každý podgraf jeho konvexní obálku a její plochu. Tuto hodnotu uložíme do proměnné SUB_AREA. Tuto hodnotu přičteme k celkové hodnotě plochy TOTAL_AREA a také přičteme k proměnné TOTAL_SYM násobek hodnot $SUB\_SYM * SUB\_AREA$.
7. Vypočítáme plochu konvexní obálky celého grafu WHOLE_AREA.
8. Abychom získali výslednou hodnotu symetrie, vezmeme všechny vypočítané a sečtené hodnoty symetrie všech symetrických podgrafů –

proměnnou TOTAL_SYM, kterou vydělíme maximem z velikostí celkové plochy grafu WHOLE_AREA a celkovou plochou všech podgrafů TOTAL_AREA. Toto nám vrátí vyškálovanou hodnotu symetrie pro graf G [7].

6.5 Ortogonalita

Ortogonalitu dělíme do dvou samostatně měřitelných sekcí. První, ortogonalita hran, měří hodnotu do jaké míry se hrany (pokud graf obsahuje ohyby, tak i části hran) odchýlily od pomyslné kartézské mřížky, respektive jejích vodorovných či svislých os. Druhá vychází z toho, jak moc vrcholy a místa ohybu (pokud graf opět obsahuje ohyby) využívají bodů mřížky [7].

6.5.1 Ortogonalita hran

Odchylku i–té hrany značíme δ_i a vypočítáme ji následovně

$$\delta_i = \frac{\min(\theta i, |90° - \theta i|, |180° - \theta i|)}{45°} \tag{6.9}$$

Přičemž θi reprezentuje pozitivní úhel od 0 do 180 stupňů mezi i–tou hranou a osou x mřížky. Samotná metrika se pak následně vypočítá vzorcem:

$$\aleph_m = 1 - \frac{1}{m'} \sum_{i=1}^{m'} \delta_i, \tag{6.10}$$

kde m' reprezentuje počet hran [7].

6.5.2 Ortogonalita vrcholů

Metrika ortogonality vrcholů vychází ze snahy zafixovat vrcholy a body ohybu grafu na bodech průniku svislých a vodorovných čar pomyslné mřížky. Zároveň se snažíme maximálně využít samotnou, v tomto případě také jednotkovou, mřížku [7].
Velikost buněk mřížky (v pixelech), ve kterých leží vrcholy grafu lze určit výpočtem největšího společného dělitele množiny vertikálních a horizontálních rozdílů pixelů mezi všemi geometricky přilehlými uzly.

Graf je nutno také posunout tak, že umístíme vrchol s nejmenšími souřadnicemi do počátku. Poté graf transformujeme transformační funkcí. Tato funkce rozdělí souřadnice vrcholů podle největšího společného dělitele a může se použít na určení jejich pozic na mřížce.
Ohraničující obdélník grafu (nejmenší obdélník, do kterého se transformovaný graf vejde), značený A, vypočítáme pak takto:

$$A = (w + 1)(h + 1) \tag{6.11}$$

Přičemž w a h jsou šířka a výška stran obdélníku.
Metriku ortogonality vrcholů pak následně definujeme jako rozsah využití pomyslné mřížky vrcholy a body ohybu transformovaného grafu.

$$\aleph_{no} = \frac{n'}{A} \tag{6.12}$$

Kde n' je počet vrcholů a ohybů grafu. Zároveň $n' < A$ a $0 < \aleph_{no} < 1$, jelikož žádné dva vrcholy nemají stejné souřadnice [7].

6.6 Vzestupný tok (Upward Flow)

Metrika vzestupného toku určuje poměr segmentů hran grafu, které mají konzistentní směr.
Předpokládáme, že graf je orientovaný a že žádoucí směr je většinou nahoru či dolů, vzhledem ke svislé ose (i když tomu tak být nemusí, samotná metrika nepředpokládá určitou orientaci). Výpočet metriky je roven:

$$\aleph_f = \frac{1}{m'} \sum_{i=1}^{m'} = \begin{cases} 1 & \text{když } \langle e_i \cdot \mathbf{1} \rangle > 0. \\ 0 & \text{jinak} \end{cases} \tag{6.13}$$

Kde m' jsou hrany grafu a $\mathbf{1}$ značí jednotkový vektor [7].

6.7 Výběr metrik vhodných k implementaci

Nyní se podíváme na výběr metrik vhodných k implementaci do programu. Na začátek bychom si měli zopakovat, jaké grafy uvažujeme a co je pro nás důležité při měření. Podle těchto kritérií pak vybereme vhodné metriky.
Grafy, které vykreslujeme, jsou souvislé, neorientované a neobsahují ohyby hran. Také uvažujeme jen menší nebo střední grafy, jelikož jsme uvedli, že algoritmy silově řízeného kreslení grafů nejsou vhodné pro grafy, které mají

více než 500 vrcholů. Speciálním případem je Eadesův algoritmus, který funguje nejlépe pro grafy, jenž mají vrcholů ještě méně a to maximálně 30. Zároveň nás bude nejspíše zajímat, jak moc symetrické grafy jsou a jestli došlo k minimalizaci překřížení. Tyto vlastnosti se totiž snaží cíleně měnit algoritmy silového řízení kreslení grafů a bylo by vhodné je porovnat s ostatními vykreslovacími algoritmy. Nyní už můžeme přistoupit k samotnému výběru.
První zmíněnou metrikou byl Crossing neboli překřížení. Jelikož dva ze tří vykreslovacích algoritmů, které budeme používat, pracují s minimalizací energie a uvažují hrany mezi vrcholy jako pružinky a vrcholy jako uchycení těchto pružinek, bylo by zajímavé pozorovat, jak toto ovlivňuje samotné překřížení hran, respektive pružinek. Zároveň jsme uvedli, že algoritmy silově řízeného kreslení grafů se snaží překřížení minimalizovat, tudíž bychom chtěli ověřit, jak moc se bude metrika lišit v porovnání s původním rozpoložením a s vykreslovacími algoritmy, jenž do této množiny nespadají. U algoritmu, jenž vykresluje graf pomocí vlastních vektorů, sice nedochází k cílené minimalizaci překřížení, ale metrika překřížení se bude jistě měnit a zároveň budeme moci tento algoritmus porovnat s ostatními. První kandidát, jevící se jako vhodný k implementaci je tedy Crossing.
Druhou v pořadí máme metriku Ohybu (Bends). Jelikož ale předpokládáme, že každá hrana z vrcholu a do vrcholu b je rovná linka, tudíž není nikde ohnuta do jiného směru a ani jeden z použitých algoritmů hrany nijak neohýbají, vycházela by metrika vždy 1 a to jak u původního grafu, tak po vykreslení jedním ze třech algoritmů. Tudíž se zdá být zbytečné metriku implementovat.
Třetí zmíněnou metrikou byl Minimální úhel. Jelikož všechny tři algoritmy mění pozici vrcholů a tudíž i „naklonění hran“, můžeme pozorovat změny úhlů mezi hranami, jenž vedou z jednoho společného vrcholu a jak metriku ovlivní každý z algoritmů. Dalším vhodným kandidátem by tedy mohla být metrika minimálního úhlu.
Další je symetrie. Symetrie by se mohla jevit jako vhodná metrika, jelikož jsme již zmínili na začátku, že by bylo vhodné jí zkoumat. Bohužel je ale zároveň tato metrika velmi časově náročná na výpočet, časová složitost této metriky je $O(n^7)$ v nejhorším případě a $O(n^5)$ v průměrném případě, což by mohlo dělat problémy u grafů s více vrcholy [7]. Jelikož jsme si ale také uvedli, že nebudeme uvažovat a testovat algoritmy na příliš velkých grafech, jeví se jako další vhodná metrika k implementaci i tato.
Předposlední metrikou byla ortogonalita hran a vrcholů. Ani jeden z algoritmů se nesnaží tuto metriku minimalizovat a pracovat s pomyslnou kartézskou mřížkou. S úhly a jejich změnou pracuje také již metrika minimálního úhlu, kterou jsme zvolili jako druhou vhodnou. Tuto metriku tudíž uvažovat a implementovat nebudeme.
Poslední metrikou byla metrika vzestupného toku. Jelikož grafy, které vykreslujeme, nejsou orientované a podle definice metriky musí být měřený graf orientovaný, je zbytečné tuto metriku měřit.
Vybranými a vhodnými metrikami na implementaci do našeho programu jsou tedy metrika překřížení, minimálního úhlu a symetrie.

Kapitola 7

Popis implementace

V této kapitole se podíváme na implementaci, bude popsáno vývojové prostředí a samotný proces implementace. Popíšeme si způsob, jakým je graf reprezentován a vykreslen, implementaci algoritmů a zvolených metrik.

7.1 Vývojové prostředí

Pro práci byla zvolena grafická knihovna OpenGL. Hlavním důvodem zvolení OpenGL byla předchozí zkušenost s touto knihovnou v předmětu Programování grafiky(PGR). Dále byly využity pro podporu vykreslování knihovny GLFW a GLAD. V neposlední řadě byla využita pro práci s maticemi, vektory, vlastními čísly a matematikou obecně knihovna Eigen.

7.2 Načítání grafu a jeho vykreslení

Graf je načítán z předem připraveného textového souboru, na jehož prvním řádku je definovaný celkový počet vrcholů grafu, na dalších n řádcích jsou potom zapsány samotné souřadnice vrcholů od -1 do 1 na ose x a od -1 do 1 na ose y. Poslední znak nám udává, jestli je vrchol „přibitý“ – drží pevně na svém místě a nepodléhá simulaci Eadesova algoritmu(1) či nikoliv(0). Posledních n řádků nám udává matici sousednosti, podle níž poznáme, jestli

je mezi dvěma zadanými vrcholy hrana. Samotný vrchol grafu je v kódu reprezentován jako struktura, jenž obsahuje x, y souřadnici (float) a přibití(bool). Struktura je, spolu s pomocnými funkcemi, definována v souboru „vertex.h“. Ke zobrazení vrcholů, respektive pro vykreslení vrcholů je potřeba vytvořit jejich meshe pomocí funkce createCircleMesh, o které se dozvíme více v podkapitole „Vytváření vrcholů a hran“.
Hrany jsou vykreslovány pomocí dvou bodů – v našem případě dvou vrcholů mezi kterými se hrana nachází. Implementována je rovněž simulace Eadesova algoritmu, vykreslování pomocí vlastních vektorů i vykreslování pomocí Laplaceových lineárních rovnic. Všem třem vykreslovacím metodám, přesněji řečeno popisu jejich implementace, budou věnovány samostatné podkapitoly.
Pro testování a zkoušení různých předem připravených typů grafů bylo vytvořeno deset textových souborů. Každý z nich obsahuje určitý typ grafu (například strom, úplný graf...). Tyto soubory je možno volit při spouštění programu přes příkazovou řádku tak, že jako první parametr zadáme název zvoleného souboru. Pokud žádný parametr nezadáme, je volen soubor „input.txt“. Pokud jako první parametr napíšeme písmeno H, zobrazí se krátká nápověda. Abychom mohli vybírat různé vykreslovací algoritmy, jimiž bude graf vykreslen, máme možnost zadat i druhý parametr. Pokud není zadán žádný druhý parametr, je graf vykreslen pomocí Eadesova algoritmu. Pokud jako druhý parametr zadáme písmeno E, bude graf vykreslen pomocí spektrálního rozkladu – vlastních vektorů. Při zadání parametru HB nebo HT budeme vykreslovat pomocí Laplaceových lineárních rovnic. Rozdíl mezi HB a HT je takový, že HB nám umožňuje vybírat mezní vrcholy na základě přibití (všechny přibité vrcholy jsou mezní) a HT vybírá set mezních vrcholů podle nastavení v kódu.
Vrcholy můžeme pomocí kliknutí a držení levého tlačítka myši posouvat. Po vykreslení grafu dojde k uložení vybraných metrik do souboru. Pokud máme zvolený Eadesův algoritmus a pohneme s vrcholy, dojde k uložení po každém ustálení grafu.
Také, pro lepší testování, byl vytvořen krátký Python script, který pro zadaný počet vrcholů a nastavitelný poměr počtu hran, vypíše náhodný graf ve tvaru textového souboru.

7.3 Vytváření vrcholů a hran

K vytváření vrcholů bylo zapotřebí vytvořit kruh, jenž bude reprezentovat každý z vrcholů V. Ten je tvořen z trojúhelníčků, jejichž vhodnou aproximací se nám bude jevit útvar jako kruh a trojúhelníčky nebudou viditelné. Každý z trojúhelníčků je tvořen třemi body – středem kruhu, předchozím a následujícím bodem, ležícím na obvodu kruhu. Aproximaci, přesněji řečeno počet trojúhelníčků, nám určuje parametr s názvem *steps*.
Hrany jsou vytvořeny vždy od středů dvou vrcholů mezi nimiž vedou. Na-

rozdíl od kruhu, dovoluje OpenGL vykreslit úsečku, která je přímo základní primitivum (GL_LINES).

7.4 Vykreslování pomocí vlastních vektorů

Funkce a samotný výpočet obsahuje třída *LaplacianEigen*. Nejprve bylo nutno vytvořit Laplaceovu matici. Ta se vypočítala pomocí matice sousednosti mínus matice stupňů vrcholů. Poté se za využití funkce z knihovny Eigen vypočítala vlastní čísla a vlastní vektory matice L. Nyní bylo potřeba vlastní čísla seřadit od nejmenšího po největší, jelikož chceme najít druhé a třetí nejmenší, respektive jejich vlastní vektory. K tomu slouží funkce *SortGetIndices*, která vrací indexy seřazených vlastních čísel od nejmenších po největších. To znamená, že na první pozici je index nejmenšího vlastního čísla, na druhé pozici druhého nejmenšího a tak dále. Díky tomuto lze vybrat dva vlastní vektory ψ_2 pro x a ψ_3 pro y, příslušné druhému a třetímu nejmenšímu vlastnímu číslu, které jsou řešením. Funkci jsme volali jen jednou a to mimo renderovací cyklus. Po spuštění programu již tedy rovnou vidíme vykreslený výsledek.
Kvůli tomu, že se vykreslené grafy jevily občas moc malé vzhledem k ploše, na níž byly vykreslovány, je celý graf, přesněji tedy souřadnice všech vrcholů x a y, vyškálován podle maxima ze souřadnic.

7.5 Eadesův algoritmus

V této podkapitole se blíže podíváme na implementaci Eadesova algoritmu. Jak již bylo zmíněno v kapitole výše, Eadesův algoritmus je jedním ze dvou implementovaných algoritmů, jenž pracuje s grafem jako se systémem pružinek (hrany) a jejich úchytem (vrcholy). Celý systém se snaží minimalizovat síly, tedy energii. Jednak tu, jenž působí pružinky na vrcholy a i tu, kterou působí nesousední vrcholy na sebe navzájem.
V našem případě jsme vrcholy umístili na předem definované pozice v textovém souboru a funkci na simulaci, v níž se vypočítaly síly, působící na každý vrchol, jsme volali v renderovacím cyklu. Tím jsme zajistili, že se po spuštění programu všechny vrcholy automaticky postupně posunou na správná místa dle simulace a graf se ustálí. Díky tomu také vidíme, na rozdíl od zbylých dvou vykreslovacích metod, postupné posouvání vrcholů na správné pozice. Při posunutí libovolného vrcholu se vrcholy opět posunou a graf se ustálí vzhledem k simulaci. Zároveň na vrcholy, které jsou „přibité“, simulace nemá vliv – neposouvají se, ale „působí“ konkrétní (dle vzorce vypočítanou) silou

na ostatní. Hodnoty konstant, které jsou použity ve výpočtu odpuzujících sil vrcholů spojených hranou, jenž se používají ve výpočtu, jsme volili nejčastěji následující $c_1 = 1$, $c_2 = 0.5$, $c_3 = 0.5$, $c_4 = 0.001$ nebo $c_4 = 0.01$. A to kvůli tomu, že po otestování různých hodnot (včetně těch z původního textu), se tyto jevily jako nejvhodnější.

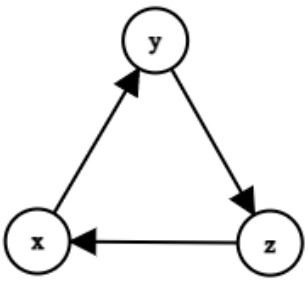
7.6 Vykreslování pomocí Laplaceových lineárních rovnic

Výpočet a pomocné funkce obsahuje třída *LaplacianNetwork*. V této třídě jsou rovněž nadefinované různé typy mezních vrcholů, přesněji řečeno jejich vybírání. Je totiž možnost vybírat mezní vrcholy buď podle toho, jaké jsou v načteném textovém souboru přibité vrcholy (když je vrchol přibitý, chová se jako mezní) nebo podle nějakého typu výběru. Toto záleží na volání funkce a jejích parametrech. Typy výběru jsou celkem čtyři: typ „first“ vybere prvních n vrcholů, které budou mezní, typ „lowestDegree“ vybere n vrcholů s nejmenším stupněm. Opakem je typ „highestDegree“, který naopak vybere ty s největším stupněm. Posledním typem je „random“, jenž vybere náhodných n vrcholů.
Následně je proveden samotný algoritmus. Ze vstupních dat je vytvořena matice D a matice sousednosti M, pomocí M a D je vypočtena Laplaceova matice, poté je vypočtena matice $L(S, S)$ a r pro x a pro y. Nakonec je za pomoci knihovny Eigen vyřešena soustava lineárních rovnic, ze které dostaneme výsledné souřadnice vrcholů x a y.

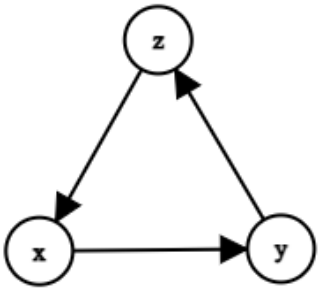
7.7 Překřížení

Překřížení je jednou z metrik estetičnosti kreslení grafu, o kterých jsme se něco dozvěděli již výše v kapitole 6 a kterou jsme spolu s dalšími dvěma vybrali jako vhodného kandidáta k implementaci. Šlo tedy o to naimplementovat měření počtu překřížení hran v grafu a výsledné číslo, jenž naměříme, uložit do souboru. Jelikož se může stát, že se u Eadesova algoritmu graf zcela neustálí a mírně se pohybuje, je nastaven treshold pohybu, při kterém se číslo vypíše a uloží také. Samotné zjištění, jestli se hrany kříží, vychází z předpokladu, že se dvě úsečky navzájem protínají pouze, když mají body (a_1, a_2 a b_1), (a_1, a_2 a b_2) jinou orientaci a zároveň (b_1, b_2 a a_1), (b_1, b_2 a a_2) jinou orientaci, přičemž body a_1 a a_2 jsou krajními body první úsečky a b_1 a b_2 krajními body druhé

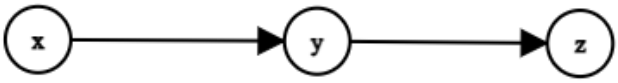
úsečky. Orientací rozumíme orientaci uspořádané trojice bodů x, y, z, která může být buď po směru hodinových ručiček nebo proti směru hodinových ručiček, či mohou body ležet všechny v jedné rovině. Pro lepší představu jsou všechny tři varianty zobrazeny na obrázcích níže [11].



Obrázek 7.1: Orientace po směru hodinových ručiček.



Obrázek 7.2: Orientace proti směru hodinových ručiček.



Obrázek 7.3: Všechny tři body leží v jedné rovině.

Zde je také nutno podotknout a zdůraznit, že tuto myšlenku nevymyslela autorka a není její původní, nýbrž je převzata a vhodně ozdrojována v kódu. Pro naše účely reprezentují první úsečku vždy dva vrcholy v_1 a v_2 spojené hranou a druhou úsečku jiné dva vrcholy u_1 a u_2, které jsou rovněž spojeny hranou. Uvažujeme také jen vrcholy takové, že z nich vychází dvě hrany do dvou různých vrcholů, to znamená v_1 a u_1 či v_2 a u_2 s totožnými souřadnicemi neuvažujeme.
Nevhodně zvolené vrcholy, tedy vrcholy mezi kterými hrana nemůže nikdy nastat, v kódu filtrujeme. Jak již bylo zmíněno v předchozí kapitole o metrikách

estetičnosti, výsledné číslo se pohybuje v rozmezí od 0 do 1, je tedy nutno ještě vydělit počet překřížení počtem všech možných překřížení. Jelikož v kódu již hrany filtrujeme a ukládáme jen použitelné, ze kterých pak vybíráme ty, co se kříží, stačí jen, když vydělíme počet překřížení počtem všech profiltrovaných možných hran a to odečteme od jedničky. Výsledné číslo je tedy uloženo do souboru, jak bylo již zmíněno výše.

7.8 Minimální úhel

Jako druhou metriku jsme k implementaci zvolili metriku minimálního úhlu. Nejprve jsme museli postupně zjistit pro každý vrchol i jeho sousední vrcholy, jenž s ním jsou spojeny hranou. Následně jsme vypočítali ideální úhel tak, že jsme vydělili 360 počtem sousedních vrcholů. Dále bylo zapotřebí zjistit, jaké úhly mezi sebou svírají sousední hrany. K tomu slouží funkce CalculateDegreeABC, jenž pro tři zadané vrcholy vypočítá pomocí kosinovy věty úhel Beta, tedy úhel, jenž svírají hrany u vrcholu i. Kosinova věta pro výpočet úhlů v obecném trojúhelníku zní:

$$cos_{(\beta)} = (a^2 + c^2 - b^2)/(2ac)$$

Přičemž v našem případě se β rovná úhlu u vrcholu i, strany a a c odpovídají hranám vycházejícím z úhlu i do dvou sousedních vrcholů a strana b je pomyslná protilehlá strana úhlu β.
Zároveň bylo tedy ještě potřeba, před vypočítáním úhlu, vypočítat délky hran. Ve třídě Vertex byla tedy přidána jednoduchá funkce na výpočet oněch délek. Zároveň jsme po výpočtu úhlu kontrolovali, jestli se nezměnil minimální úhel – jestli vypočtený není menší než stávající. Poté jsme podle vzorce v kapitole 6 vypočítali absolutní hodnotu rozdílu úhlů a nakonec i výslednou sumu, dělenou celkovým počtem uvažovaných vrcholů.

7.9 Symetrie

Poslední vybranou a naimplementovanou metrikou byla Symetrie. Pro lepší práci s grafem a pro tvorbu podgrafů byla vytvořena třída AdjacencyListGraph, jenž obsahuje množinu vrcholů grafu a mapu, v níž je klíčem vždy vrchol a hodnota klíče jsou všechny jeho sousední vrcholy. Třída také obsahuje funkce, které zjednodušují práci s grafem například přidání vrcholu, vytvoření podgrafu nebo zjištění, jestli je mezi vrcholy hrana. Samotný výpočet metriky

je naimplementován v souboru Metric.cpp, který obsahuje krom onoho výpočtu i funkce, sloužící k pomocným výpočtům, například výpočet konvexní obálky a plochy konvexní obálky grafu nebo bod průniku dvou úseček. Také byla v Metric.h vytvořena struktura Axis, která definuje osu v podgrafu. Výpočet probíhá podle postupu, uvedeného v kapitole 6.

Kapitola 8

Výsledky

Nyní se podíváme na výsledné snímky z programu, respektive na grafy, jenž se nám povedlo pomocí algoritmů vykreslit a hodnoty vybraných metrik pro každý vykreslený graf. Také jsme výsledné vykreslené grafy porovnávali s názory dvanácti pozorovatelů.

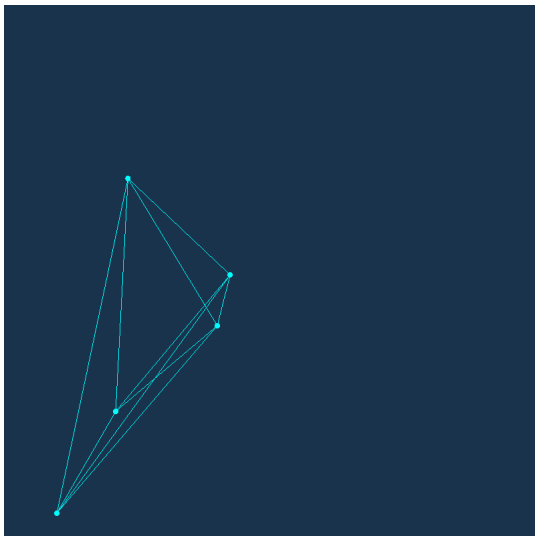
8.1 Vykreslené grafy

Program jsme testovali na deseti příkladech grafů, které jsme si vybrali pro otestování. Každý měl iniciální nastavení vrcholů a hran mezi vrcholy a každý graf jsme posupně zkoušeli vykreslit každým z algoritmů a to v pokaždé v novém běhu programu. Pro všechny vykreslené grafy se také vypočítaly metriky. U metriky symetrie byl ze základu nastaven TRESHOLD na 1 a TOLERANCE na 0,01. Teď přejděme k samotným ukázkám a snímkům z programu.

8.1.1 Graf č.1 – Úplný graf o 5-ti vrcholech

Prvním vykresleným grafem je úplný graf o 5-ti vrcholech. To znamená, že z každého vrcholu v vede hrana do všech ostatních vrcholů krom zmíněného

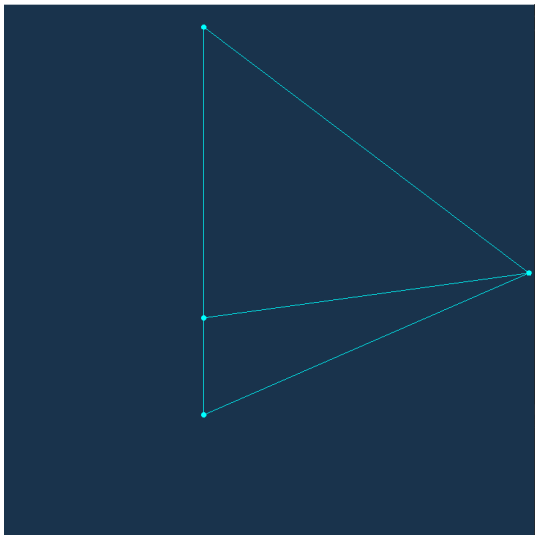
v. Hodnoty metrik vyšly 0,888889 pro překřížení, 0,0891569 pro metriku minimálního úhlu a 0 pro symetrii.



Obrázek 8.1: Iniciální nastavení grafu o 5-ti vrcholech.

Graf č.1 vykreslený pomocí vlastních vektorů

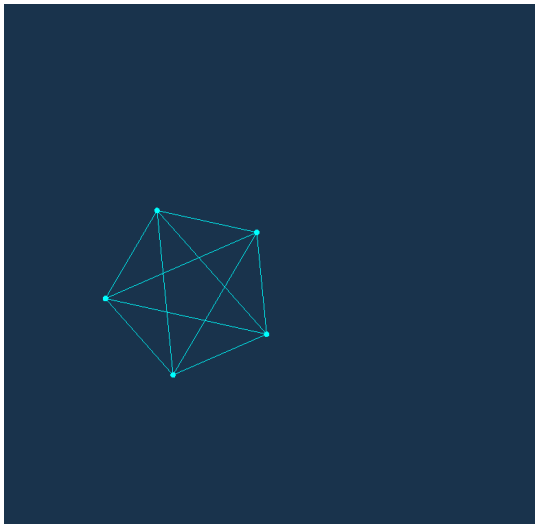
Vidíme, že graf vykreslený pomocí vlastních vektorů se nám nepovedlo vykreslit úplně šťastně. A to tak, že dva vrcholy mají stejné souřadnice a jsou přes sebe. Důvodem je to, že nám při spektrálním rozkladu Laplaceova grafu úplné matice vyjdou jen dvě rozdílná vlastní čísla. Přičemž prvním vlastním číslem je nejmenší vlastní číslo – 0 a zbylých $n-1$ vlastních čísel vyjde n, n odpovídá počtu vrcholů. Po dosazení do Laplaceovy matice, která vypadá tak, že má na diagonále $n-1$ a jinde 1 nám vyjde, pro tato stejná nenulová vlastní čísla, matice se samými jedničkami. Vidíme tedy, že řešením je nespočetně mnoho vlastních vektorů. A právě kvůli této násobnosti stejných vlastních čísel se vyskytují duplicitní souřadnice. Kvůli těmto duplicitám nebylo možno vypočítat metriky úplně přesně, jelikož vrcholy leží na sobě – metrika minimálního úhlu vyšla 0,365109, překřížení 1 a symetrie také 0. Jako řešení duplicit autorku napadlo, že by se mohly duplicitní hodnoty sjednotit a z grafu G by se mohl vytvořit graf G', který by obsahoval původní vrcholy v a nové vrcholy v', které vznikly spojením vrcholů, které leží na sobě. Poté by se vypočítaly hodnoty metrik pro nový graf G'. Toto řešení ovšem práce už nepokrývá.



Obrázek 8.2: Vykreslení pomocí vlastních vektorů pro graf o 5-ti vrcholech.

Graf č.1 vykreslený pomocí Eadesova algoritmu

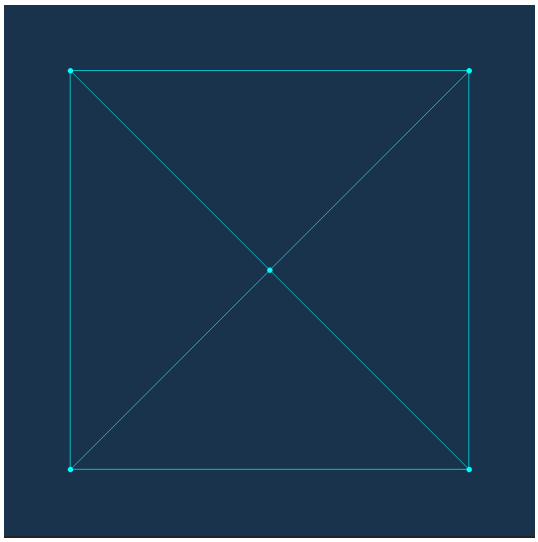
Graf se jevil jako symetrický, jak se dalo očekávat. Při vyšší hodnotě konstanty $c_4 = 0,01$ u vrcholů spojenými hranou se graf neustálil a otáčel se, jak mezi sebou síly působily. Bylo tedy nutné v tomto případě zvolit $c_4 = 0,001$. V tomto případě se graf ustálil. Žádný z vrcholů nebyl přibitý. Metrika překřížení se rovnala 0,66667, metrika minimálního úhlu zde byla 0.398398 a metrika symetrie 1.



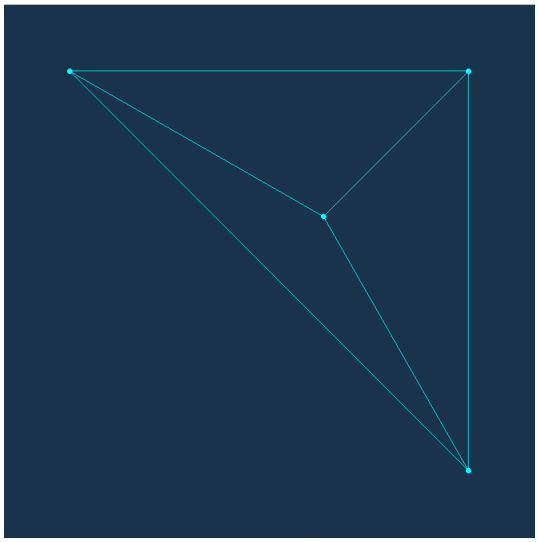
Obrázek 8.3: Eadesův algoritmus pro graf o 5-ti vrcholech.

Graf č.1 vykreslený pomocí Laplaceových lineárních rovnic

Zde jsme dosahovali nejlepších výsledků při zvolení prvních čtyř mezních vrcholů. S tím, že metrika překřížení zde vyšla 0,8 metrika nejmenšího úhlu 0,2. Metrika symetrie vyšla 1. Při zvolení menšího počtu se vrcholy překreslovaly přes sebe. Výsledky a porovnání můžeme vidět na obrázcích.



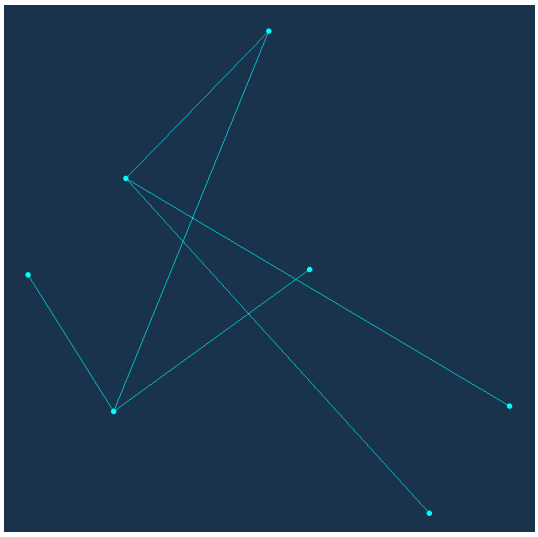
Obrázek 8.4: Graf se čtyřmi mezními vrcholy pro graf o 5-ti vrcholech.



Obrázek 8.5: Graf se třemi mezními vrcholy pro graf o 5-ti vrcholech.

8.1.2 Graf č.2–Strom

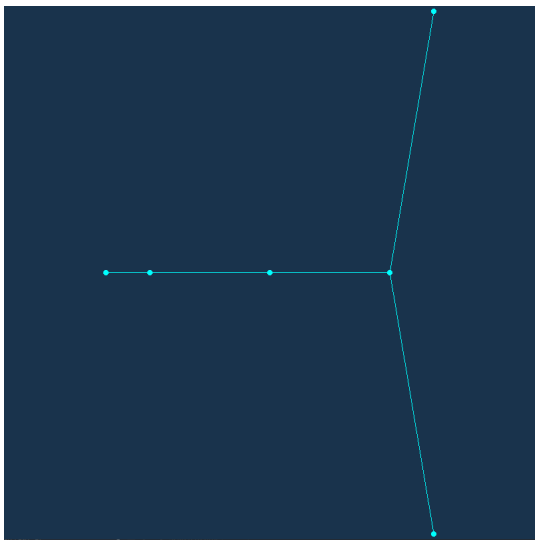
Dalším vykresleným grafem byl strom. Jako strom nazýváme graf, který v sobě neobsahuje kružnici. Zvolený graf měl 7 vrcholů a ze základu, bez vykreslovacích algoritmů, vycházely metriky 0,647178 pro minimální úhel, 0,5 pro překřížení a 0 pro symetrii.



Obrázek 8.6: Iniciální nastavení grafu stromu.

Graf č.2 vykreslený pomocí vlastních vektorů

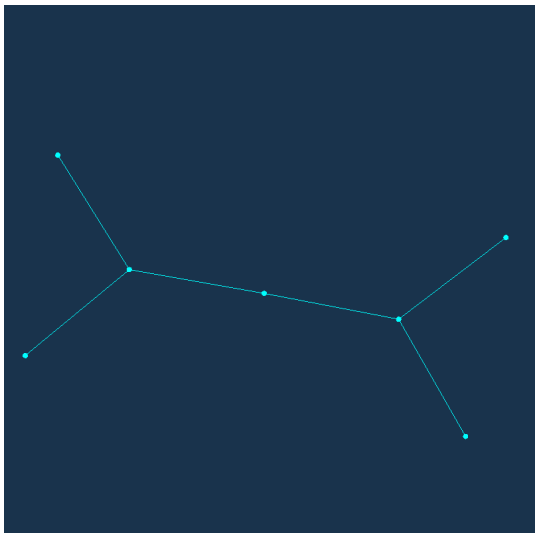
Na obrázku můžeme vidět vykreslení grafu stromu pomocí vlastních vektorů. Zde se také bohužel, jako v minulém případě, dva vrcholy překreslily přes sebe, konkrétně dva listy stromu, tudíž docházelo k nepřesnostem. Metrika překřížení byla naměřena 1, metrika minimálního úhlu 0.832832 a metrika symetrie kvůli překryvu 0.



Obrázek 8.7: Vykreslení grafu stromu pomocí vlastních vektorů.

Graf č.2 vykreslený pomocí Eadesova algoritmu

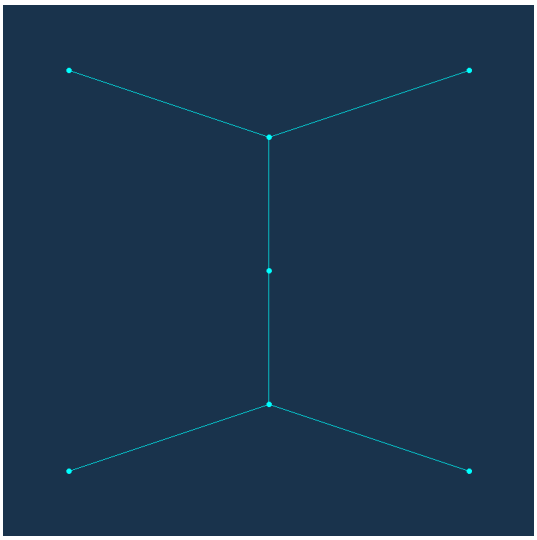
Opět byly všechny vrcholy nepřibité. Po otestování se jevila jako ideální hodnota konstanty $c_4 = 0,01$ mezi vrcholy propojenými hranou. Metriky vyšly 0,944188 pro minimální úhel, 1 pro překřížení a 1 pro symetrii. Zde bylo nutno snížit upravit hodnotu pro minimální pohyb, kdy se hodnoty uloží do souboru nebo lehce pohnout po zastavení simulace s kořenovým vrcholem. Jelikož se metriky zapisovaly moc brzo, což bylo nechtěné převážně u symetrie. Graf opět působí symetricky.



Obrázek 8.8: Vykreslení grafu stromu pomocí Eadesova algoritmu.

Graf č.2 vykreslený pomocí Laplaceových lineárních rovnic

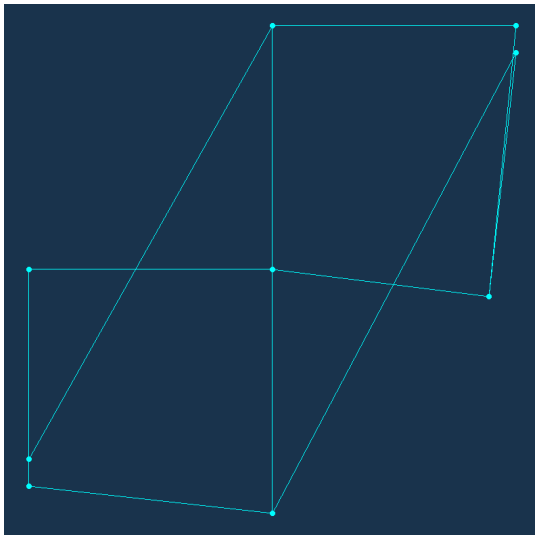
Tentokrát byly zvoleny mezní vrcholy čtyři listy stromu, to znamená čtyři poslední vrcholy v souboru. Při jiné volbě se vrcholy překrývaly. Výsledný graf působí opět symetricky a vypadá hodně podobně jako vykreslení pomocí Eadesova algoritmu. Jen zde byla naměřena vyšší metrika minimálního úhlu, což je vidět i na obrázku – vidíme větší rozpětí mezi listy. Metrika minimálního úhlu vyšla 0,972442, metrika překřížení byla opět 1 a metrika symetrie 1.



Obrázek 8.9: Graf stromu se čtyřmi mezními vrcholy.

8.1.3 Graf č.3–mřížka

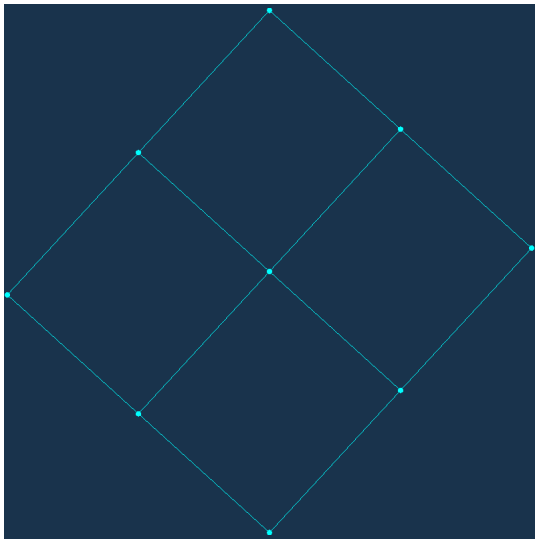
Graf, ve kterém ze čtyř „krajních vrcholů“ vedou dvě hrany, ze čtyř vrcholů mezi krajními vedou tři hrany a z prostředního čtyři hrany. Vrcholy jsme volili v podobném rozpoložení jako mělo být finální s menšími odchylkami. Hodnoty metrik původního grafu byly 0,029889 pro minimální úhel, 0,9090909 pro překřížení a 0 pro symetrii.



Obrázek 8.10: Iniciální nastavení grafu mřížky.

Graf č.3 vykreslený pomocí vlastních vektorů

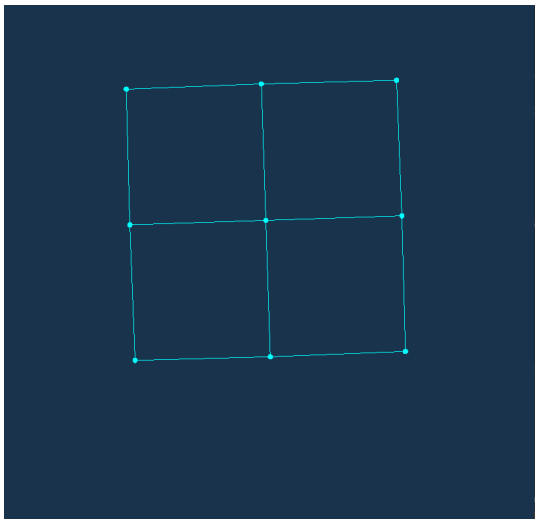
Pro tento graf byla naměřena metrika překřížení 1, metrika minimálního úhlu 0,664409 a metrika symetrie 1.



Obrázek 8.11: Vykreslení grafu mřížky pomocí vlastních vektorů

Graf č.3 vykreslený pomocí Eadesova algoritmu

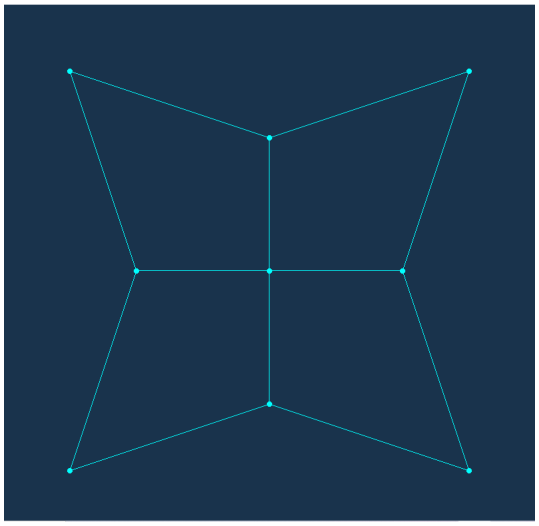
Podobně jako vykreslení pomocí vlastních vektorů vypadalo i vykreslení pomocí Eadesova algoritmu. Graf také vypadal jako symetrická mřížka. Konstanta c_4 byla zvolena jako $c_4 = 0,001$. Metriky, jak se dalo očekávat kvůli podobnosti s vykreslením pomocí vlastních vektorů, vyšly následovně – 1 pro metriku překřížení, 0,638181 pro minimální úhel a 1 pro symetrii.



Obrázek 8.12: Vykreslení grafu mřížky pomocí Eadesova algoritmu.

Graf č.3 vykreslený pomocí Laplaceových lineárních rovnic

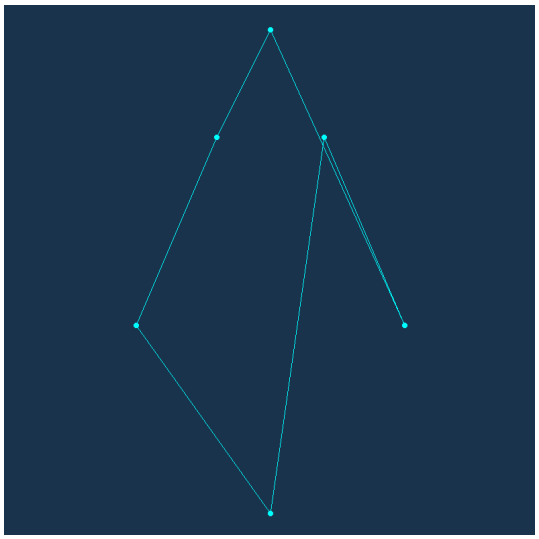
Jako mezní vrcholy byly zvoleny vrcholy 1, 3, 7 a 9, což odpovídalo okrajovým vrcholům pomyslného čtyřúhelníku. Vykreslení se trochu lišilo oproti dvěma předchozím vykreslovacím algoritmům, nicméně výsledek zůstává i tak symetrický. Metriky vyšly tentokrát 0,643907 pro minimální úhel, 1 pro překřížení a 1 pro symetrii.



Obrázek 8.13: Graf mřížky, vykreslený pomocí Laplaceových lineárních rovnic.

8.1.4 Graf č.4–Hexagon

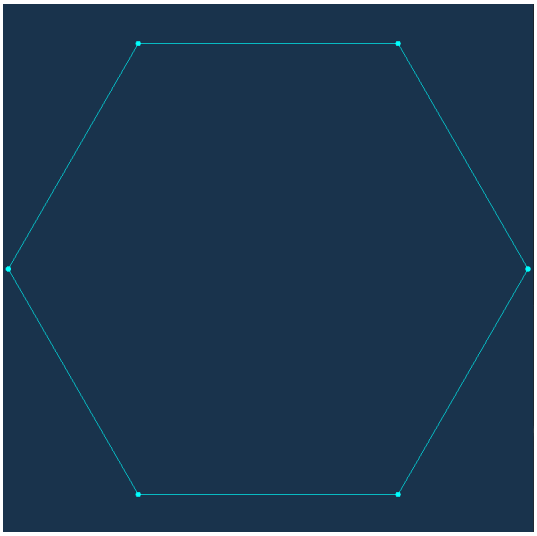
Graf kružnice se šesti vrcholy, kde každý vrchol má stupeň 2. Opět jsme volili podobné rozpoložení ve tvaru hexagonu, ale ne finální, u vrcholů se vyskytovaly odchylky. Hodnoty metrik iniciálního grafu byly 0,0393654 pro metriku minimálního úhlu, 0,888889 pro překřížení a 1 pro symetrii.



Obrázek 8.14: Iniciální nastavení grafu hexagonu.

Graf č.4 vykreslený pomocí vlastních vektorů

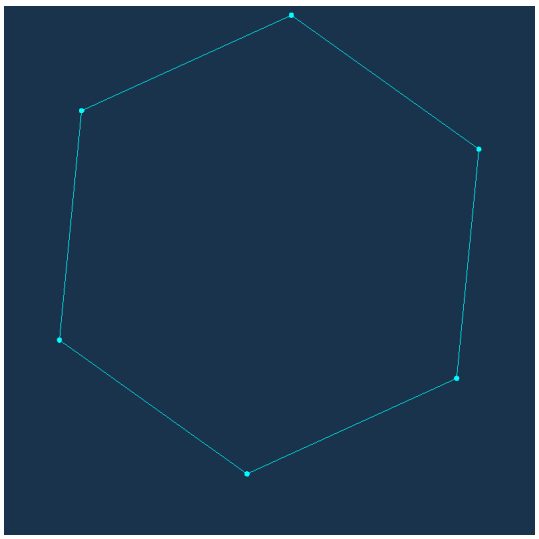
Graf byl vykreslen jako pravidelný hexagon. Metrika překřížení byla naměřena 1, metrika minimálního úhlu 0.666667 a metrika symetrie 1.



Obrázek 8.15: Vykreslení grafu hexagonu pomocí vlastních vektorů.

Graf č.4 vykreslený pomocí Eadesova algoritmu

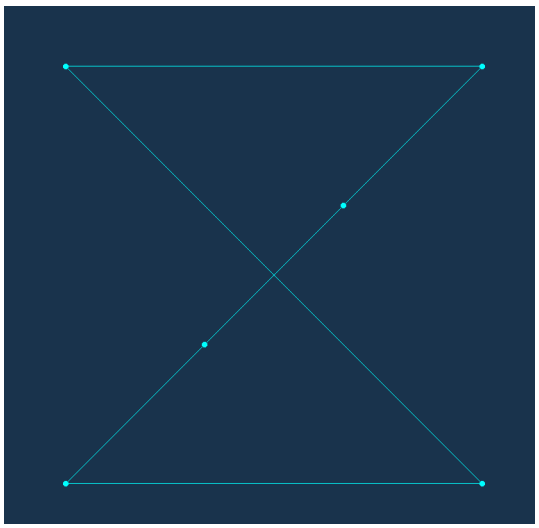
Stejně jako u předchozího grafu se vykreslení pomocí Eadesova aloritmu moc nelišilo od předchozího. Byl také vykreslen pravidelný hexagon. Hodnoty metrik byly úplně stejné jako u vykreslení pomocí vlastních čísel. Tudíž pro překřížení vyšla hodnota 1, pro minimální úhel 0.666667 a pro symetrii 1.



Obrázek 8.16: Vykreslení grafu hexagonu pomocí Eadesova algoritmu.

Graf č.4 vykreslený pomocí Laplaceových lineárních rovnic

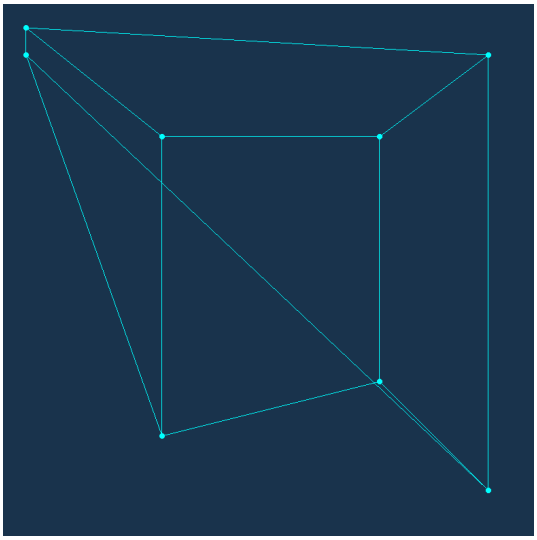
Pro toto vykreslení byly voleny jako mezní vrcholy první čtyři vrcholy. Hodnoty metrik vyšly 0,499924 pro metriku minimálního úhlu, 0,888889 pro překřížení a 1 pro symetrii.



Obrázek 8.17: Graf hexagonu se čtyřmi mezními vrcholy.

8.1.5 Graf č.5-krychle

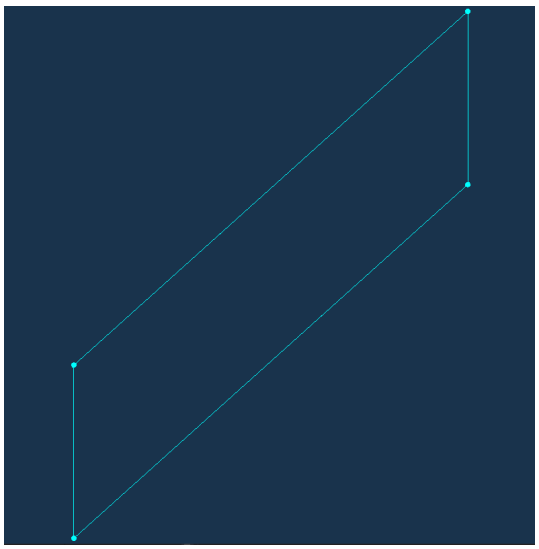
Graf, jehož umístění vrcholů bychom si mohli představit jako umístění vrcholů na krychli. Hranami byly spojeny vrcholy, které mezi sebou mají hranu jako jsou hrany krychle. Metriky u tohoto původního grafu vyšly 0,419266 pro minimální úhel, 0,952381 pro překřížení a 0 pro symetrii.



Obrázek 8.18: Iniciální nastavení grafu krychle.

Graf č.5 vykreslený pomocí vlastních vektorů

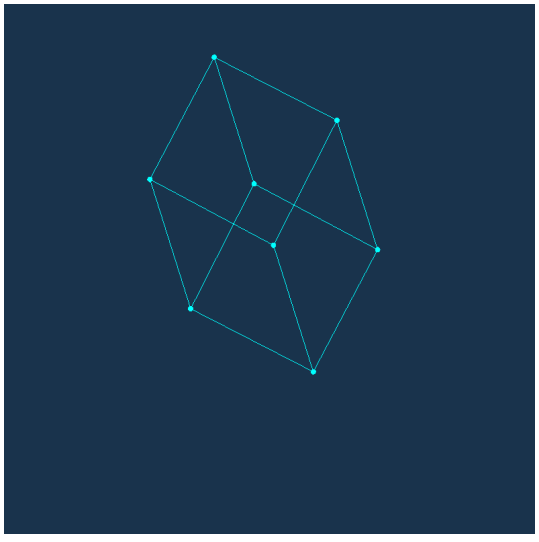
Graf se nám v tomto případě nepovedlo vykreslit, vrcholy se opět překrývaly. Výsledek vidíme na obrázku.



Obrázek 8.19: Vykreslení grafu krychle pomocí vlastních vektorů.

Graf č.5 vykreslený pomocí Eadesova algoritmu

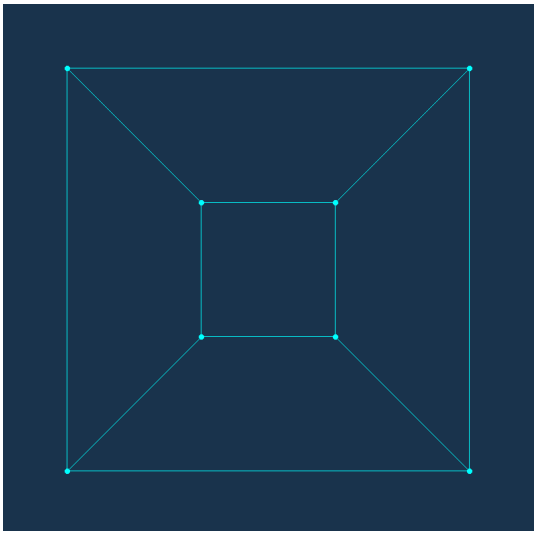
Výsledné vykreslení připomínalo krychli/kvádr na který se kouká pozorovatel šikmo ze shora. Hodnoty metrik vyšly 0,4440832 pro minimální úhel, 0,952921 pro překřížení a oproti původnímu se graf více zesymetričtil, takže hodnota symetrie vyšla 0,97737.



Obrázek 8.20: Vykreslení grafu krychle pomocí Eadesova algoritmu.

Graf č.5 vykreslený pomocí Laplaceových lineárních rovnic

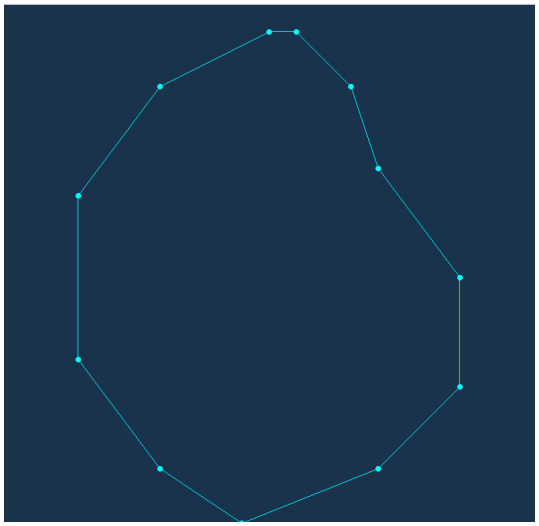
Graf vypadal jako pohled na krychli/kvádr zepředu. Jevil se symetricky, došlo k odstranění překřížení. To je znatelné i na hodnotách metrik, které vyšly 0,543297 pro minimální úhel, 1 pro překřížení a 1 pro symetrii.



Obrázek 8.21: Graf krychle, vykreslený pomocí Laplaceových lineárních rovnic.

8.1.6 Graf č.6 – kružnice o 12 vrcholech

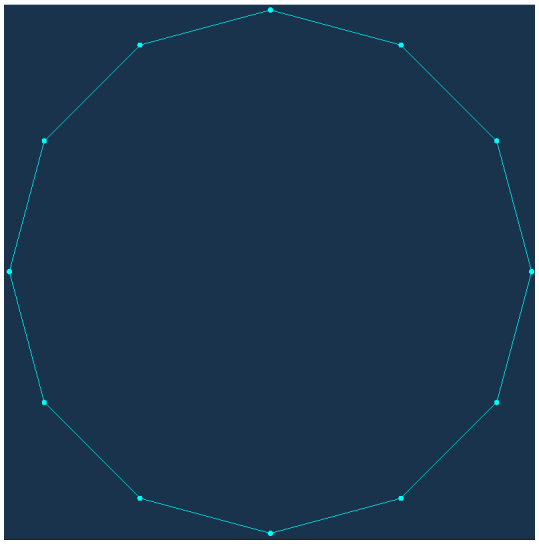
Graf kružnice se dvanácti vrcholy, kde každý vrchol má stupeň 2, podobně jako tomu bylo u hexagonu. Metriky u původního vykreslení vyšly následovně – 0,816264 minimální úhel, 1 překřížení a 0,330578 symetrie.



Obrázek 8.22: Iniciální nastavení grafu o 12-ti vrcholech.

Graf č.6 vykreslený pomocí vlastních vektorů

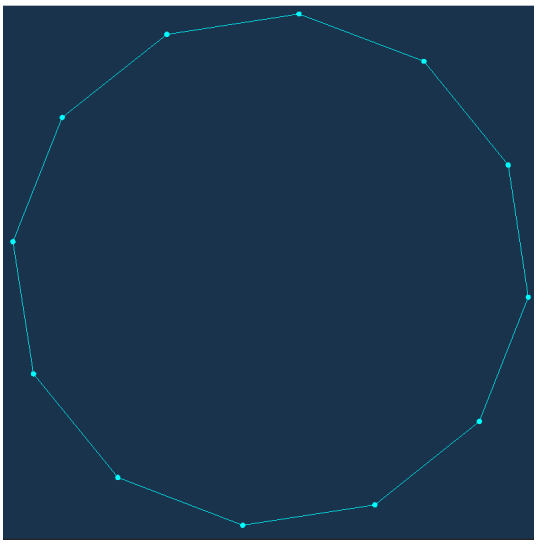
Graf byl vykreslen jako pravidelný útvar. Metrika překřížení byla naměřena 1, metrika minimálního úhlu 0.833333 a metrika symetrie 1.



Obrázek 8.23: Vykreslení grafu o 12-vrcholech pomocí vlastních vektorů.

Graf č.6 vykreslený pomocí Eadesova algoritmu

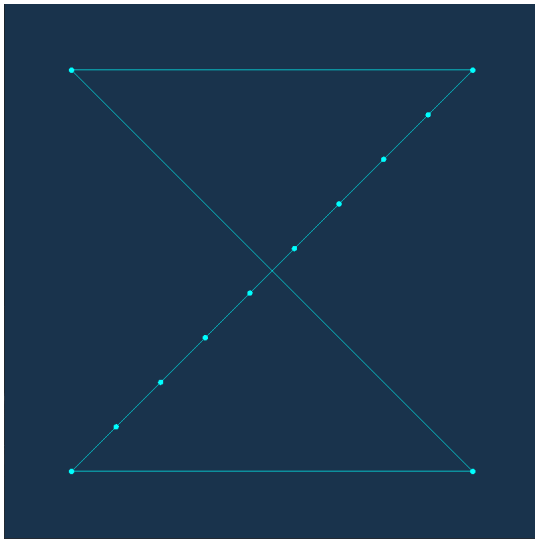
Opět, podobně jako u hexagonu, bylo vykreslení téměř totožné. Metriky vycházely úplně stejně jako u vykreslení pomocí vlastních vektorů.



Obrázek 8.24: Vykreslení grafu o 12-ti vrcholech pomocí Eadesova algoritmu.

Graf č.6 vykreslený pomocí Laplaceových lineárních rovnic

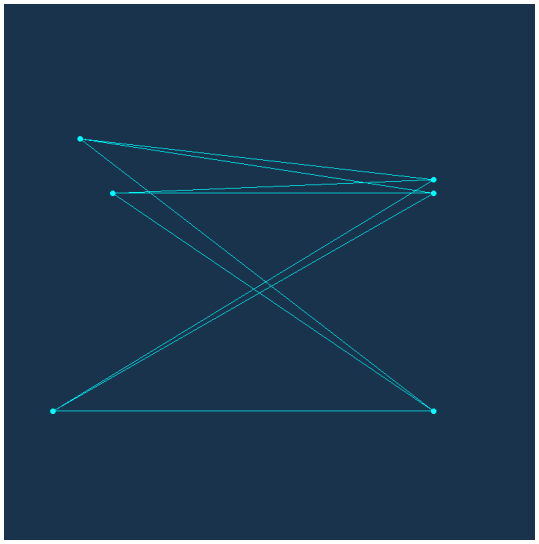
U tohoto grafu byly zkoušeny různé mezní vrcholy– první čtyři, první tři a poslední. Každé vykreslení vypadalo kvůli rozdílným mezním vrcholům jinak, ale většinou bylo vždy symetrické, což bylo vidět i u hodnoty symetrie, která zde dosahovala většinou hodnotu 1. Nakonec byly zvoleny jako mezní pro finální vykreslení první čtyři vrcholy. Toto vykreslení vidíme na obrázku. Hodnoty metrik byly pro toto vykreslení 0,333311 pro minimální úhel, 0,9814815 pro překřížení a 1 pro symetrii. Autorku zde napadlo, že by bylo vhodným vylepšením nejprve z iniciálního grafu dopočítat ideální kandidáty na mezní vrcholy a ty pak použít pro vykreslování. Tuto implementaci ale kód neobsahuje, jedná se zde pouze o nápad na vylepšení.



Obrázek 8.25: Graf o 12-ti vrcholech s prvními čtyřmi mezními vrcholy.

8.1.7 Graf č.7 – úplný bipartitní graf

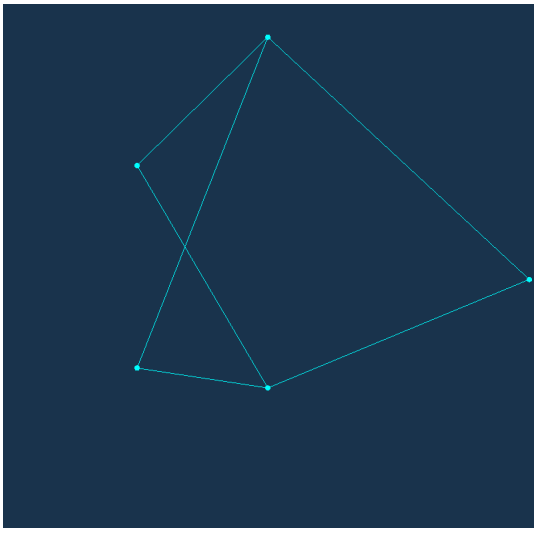
Dalším zvoleným grafem byl úplný bipartitní graf o šesti vrcholech. Základní vykreslený měl hodnoty metrik následující – 0,5 překřížení, 0,0379121 minimální úhel a 0 symetrie.



Obrázek 8.26: Iniciální nastavení úplného bipartitního grafu.

Graf č.7 vykreslený pomocí vlastních vektorů

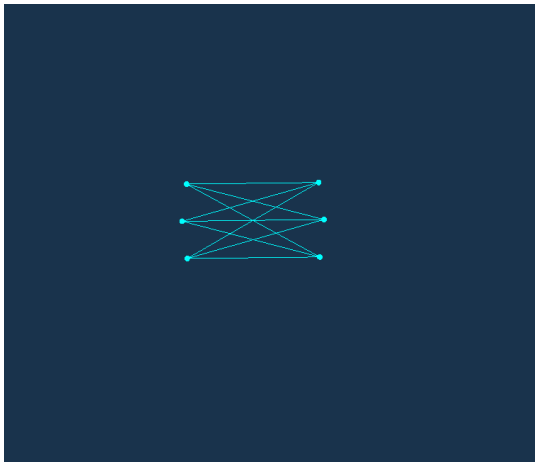
Vidíme, že zde se nám řešení, podobně jako u prvního grafu, moc nepovedlo– vrcholy jsou opět přes sebe. Můžeme tedy usoudit, že tento typ grafů není vhodné vykreslovat pomocí vlastních vektorů.



Obrázek 8.27: Vykreslení bipartitního grafu pomocí vlastních vektorů.

Graf č.7 vykreslený pomocí Eadesova algoritmu

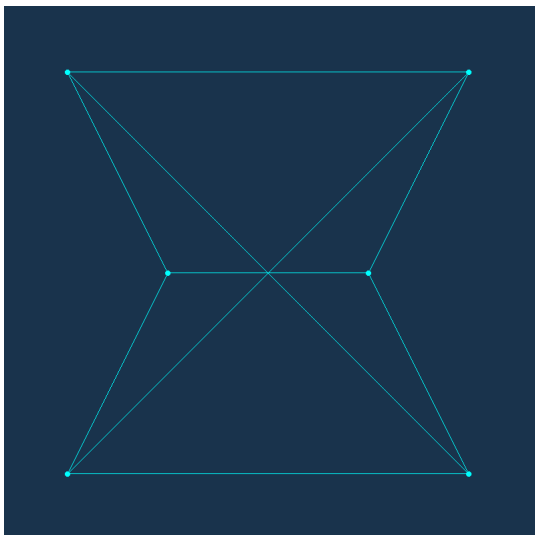
Zde vidíme vykreslení úplného bipartitního grafu pomocí Eadesova algoritmu. Graf působí symetricky a hodnoty metrik jsou – 0,5 překřížení, 0,12065 minimální úhel a 1 symetrie.



Obrázek 8.28: Vykreslení bipartitního grafu pomocí Eadesova algoritmu.

Graf č.7 vykreslený pomocí Laplaceových lineárních rovnic

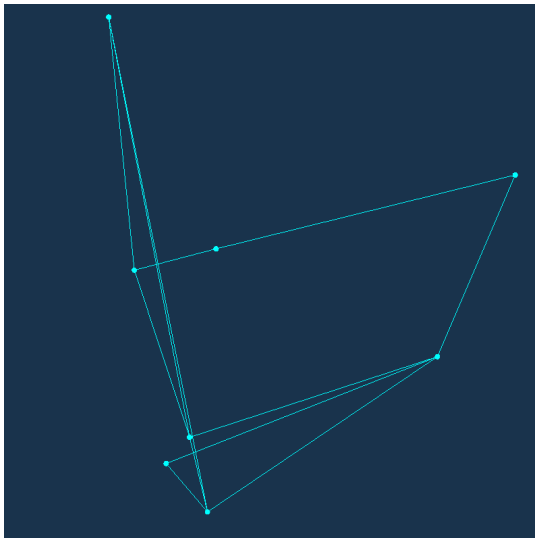
Jako mezní vrcholy jsme volili první čtyři. Výsledkem byl následující graf s metrikami – 0,426208 pro metriku minimálního úhlu, 0,833333 pro překřížení a 1 pro symetrii.



Obrázek 8.29: Úplný bipartitní graf, vykreslený pomocí Laplaceových lineárních rovnic.

8.1.8 Graf č.8 – malý náhodný graf

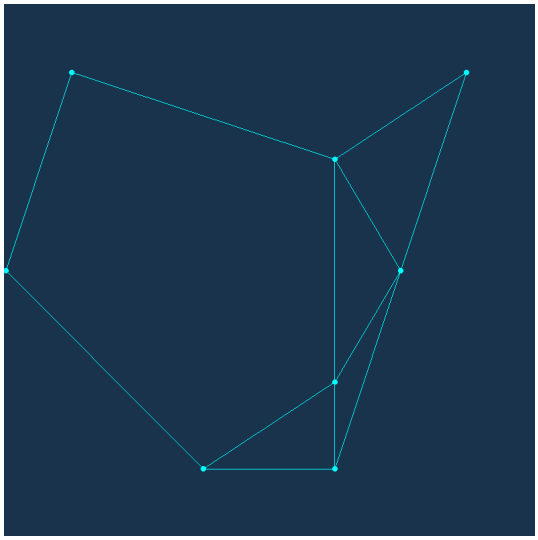
Zde jsme si nechali vygenerovat graf o 8 vrcholech. Vrcholy měly náhodné iniciální rozpoložení a hrany mezi sebou. Původní metriky vycházely 0,313572 pro minimální úhel, 0,871795 pro překřížení a 0 pro symetrii.



Obrázek 8.30: Iniciální nastavení vygenerovaného náhodného grafu.

Graf č.8 vykreslený pomocí vlastních vektorů

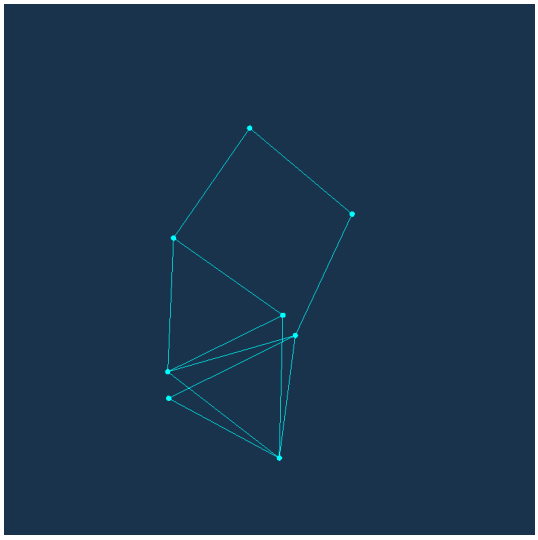
Metriky vycházely 1 pro překřížení, 0,326098 pro minimální úhel a 0 pro symetrii.



Obrázek 8.31: Vykreslení náhodného menšího grafu pomocí vlastních vektorů.

Graf č.8 vykreslený pomocí Eadesova algoritmu

Takto vypadal vykreslený náhodný graf pomocí Eadesova algoritmu. Jelikož základní iniciální nastavení bylo hodně odlišné od ideální pozice vrcholů a mezi vrcholy nepůsobily tak moc velké síly, vypadalo vykreslení jako první obrázek. Po zmenšení konstanty c4 na 0,0001, mezi vrcholy spojenými hranou, se graf ustálil do, na pohled příjemnější, podoby, jak si můžeme všimnout na obrázku dva. Metriky pro druhé vykreslení vycházely 1 pro překřížení, 0,53882 pro minimální úhel a 0 pro symetrii. Jelikož graf působil skoro symetricky, zkusili jsme zvýšit mírně toleranci vzdálenosti zrcadlených vrcholů. Při zvýšení hodnoty TOLERANCE na 0.02 vyšla symetrie 0,367929 a při hodnotě 0.05 vycházela metrika symetrie 1.

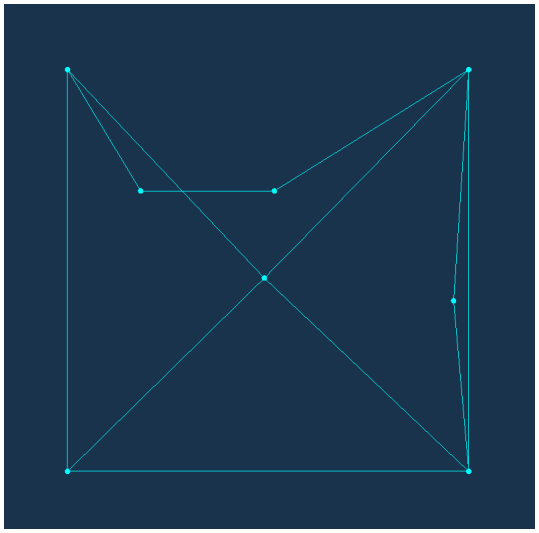


Obrázek 8.32: Vykreslení malého náhodného grafu pomocí Eadesova algoritmu s hodnotou konstanty 0,01.

Obrázek 8.33: Vykreslení malého náhodného grafu pomocí Eadesova algoritmu s hodnotou konstanty 0,0001.

Graf č.8 vykreslený pomocí Laplaceových lineárních rovnic

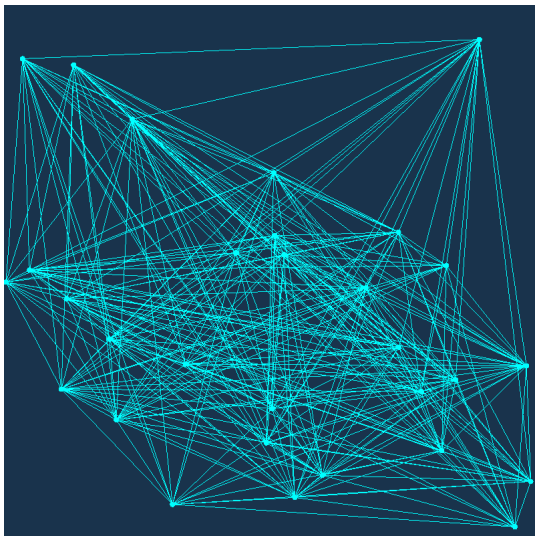
Jako mezní vrcholy jsme volili první čtyři. Výsledkem byl následující graf s metrikami – 0,499178 pro metriku minimálního úhlu, 0,974359 pro překřížení a 1 pro symetrii s TRESHOLDEM 1 a 0 pro symetrii s TRESHOLDEM 2.



Obrázek 8.34: Náhodný menší graf, vykreslený pomocí Laplaceových lineárních rovnic..

8.1.9 Graf č.9 – větší náhodný graf

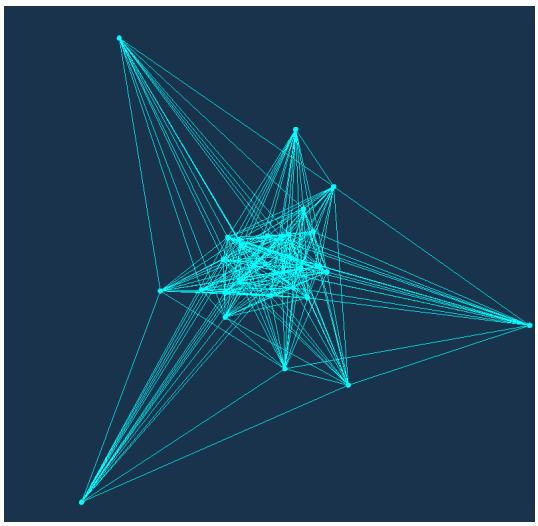
Nechali jsme si vygenerovat graf o 30 vrcholech. Vrcholy měly náhodné iniciální rozpoložení a hrany mezi sebou. Hodnoty metrik původního rozpoložení byly 0,223502 pro minimální úhel, 0,9709616 pro překřížení a 0,770636 pro symetrii.



Obrázek 8.35: Iniciální nastavení většího náhodně vykresleného grafu.

Graf č.9 vykreslený pomocí vlastních vektorů

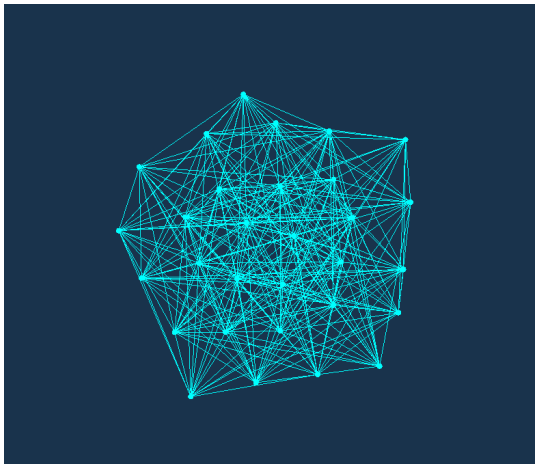
Metriky vycházely 0,828643 pro překřížení, 0,0144397 pro minimální úhel a 0,00105 pro symetrii.



Obrázek 8.36: Vykreslení většího náhodně generovaného grafu pomocí vlastních vektorů.

Graf č.9 vykreslený pomocí Eadesova algoritmu

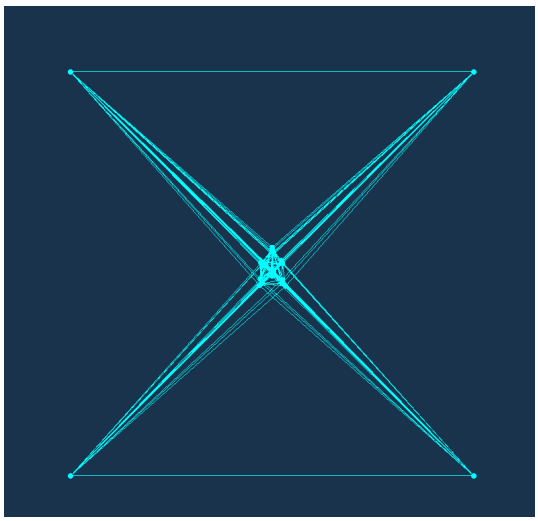
U tohoto grafu trvalo o hodně déle než se ustálil oproti původním menším grafům. Když k ustálení došlo, vyšly metriky následovně – 0,0505561 pro minimální úhel, 0,828643 pro překřížení a 0,745916 pro symetrii.



Obrázek 8.37: Vykreslení náhodného většího grafu pomocí Eadesova algoritmu.

Graf č.9 vykreslený pomocí Laplaceových lineárních rovnic

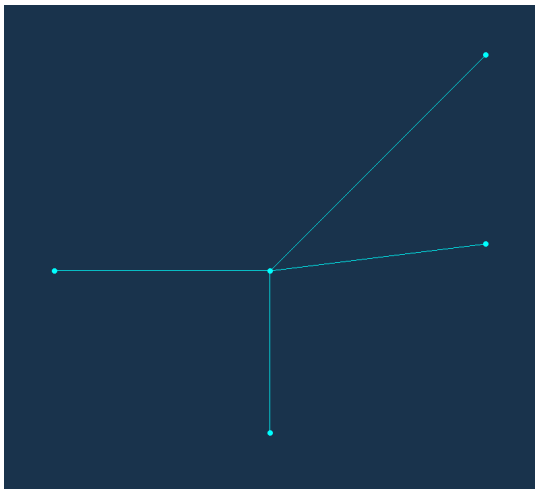
Jako mezní vrcholy jsme volili opět první čtyři, poté také prvních 11. Pro graf s prvními čtyřmi vyšly metriky takto – 0,01308 pro minimální úhel, 0,847944 pro překřížení a 0,767234 pro symetrii. Pro 11 mezních vrcholů vyšla metrika minimálního úhlu 0,0185542, překřížení 0,853567 a symetrie se také lehce zlepšila a vyšla 0,793556.



Obrázek 8.38: Náhodný větší graf se čtyřmi mezními vrcholy.

8.1.10 Graf č.10 – Hvězda

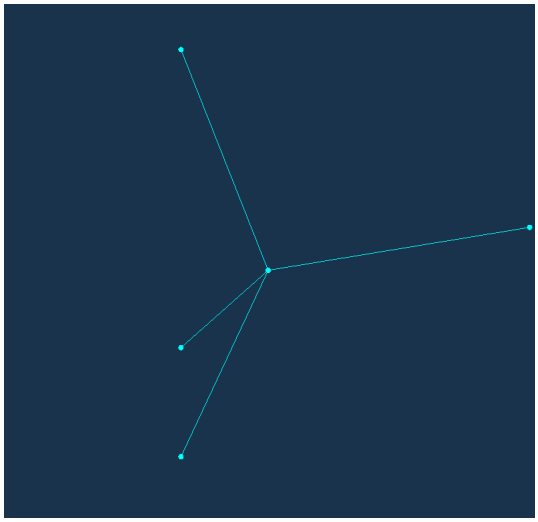
Graf o pěti vrcholech, který má jeden vrchol, který má za sousední vrcholy všechny ostatní to znamená n-1 sousedů. Zbylé vrcholy mají sousední vrchol právě jeden. Metriky pro počáteční nastavení vycházely 0,884167 pro minimální úhel, 1 pro překřížení a 0 pro symetrii.



Obrázek 8.39: Iniciální nastavení grafu hvězdy.

Graf č.10 vykreslený pomocí vlastních vektorů

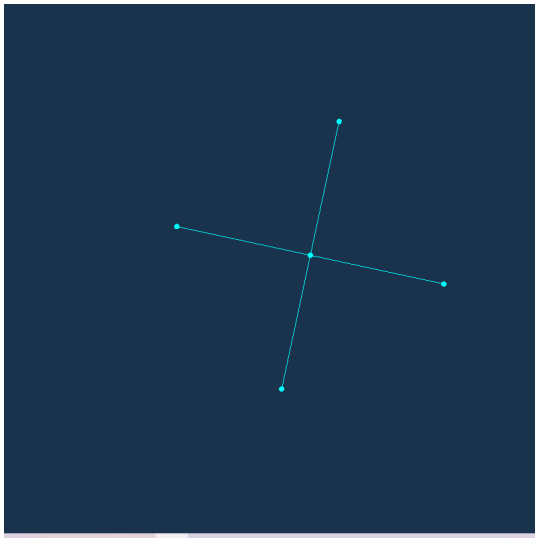
Pro tento typ vykreslení vycházely metriky 0,85792 pro minimální úhel, 1 pro překřížení a 0 pro symetrii.



Obrázek 8.40: Vykreslení grafu hvězdy pomocí vlastních vektorů.

Graf č.10 vykreslený pomocí Eadesova algoritmu

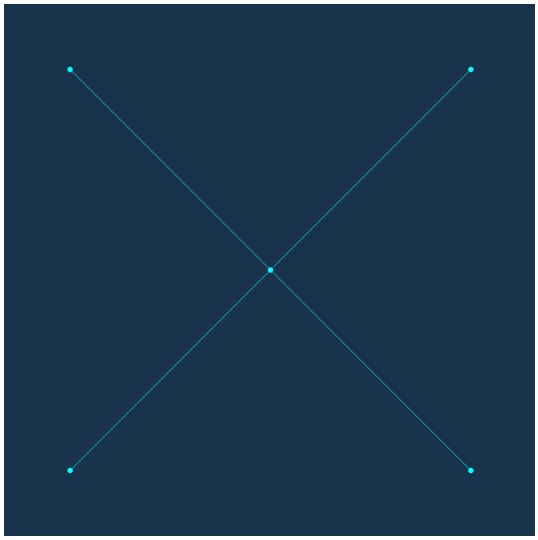
Metriky vycházely skoro stejné jako u vykreslení pomocí vlastních vektorů. Došlo jen k výraznému zlepšení metriky minimálního úhlu, která vyšla 0,9998.



Obrázek 8.41: Vykreslení grafu hvězdy pomocí Eadesova algoritmu.

Graf č.10 vykreslený pomocí Laplaceových lineárních rovnic

Výsledné vykreslení vypadalo velmi podobně jako vykreslení pomocí Eadesova algoritmu. Jako mezní vrcholy byly voleny čtyři vrcholy, které měly stupeň 1. Hodnoty metrik byly stejné jako u Eadesova algoritmu, jen se ještě nepatrně zlepšila metrika minimálního úhlu a to na 1.



Obrázek 8.42: Graf hvězdy se čtyřmi mezními vrcholy.

Tabulka, obsahující souhrn metrik pro každý typ grafu

Pro lepší přehlednost a souhrn hodnot naměřených metrik pro různá vykreslení grafů byla vytvořena tabulka, obsahující všechny hodnoty metrik. Na tabulku se můžeme podívat na obrázku pod tímto textem. Hodnoty metrik jsou pro větší přehlednost zaokrouhleny. Zároveň jsou světle zeleně vyznačeny nejvyšší hodnoty metrik pro každý vykreslený graf. Světle modře jsou označeny algoritmy vykreslení, které pro určitý graf mají největší součet nejvyšších hodnot metrik – vycházejí v součtu nejlépe, vzhledem k metrikám. Červeně označená políčka značí hodnoty metrik, které odpovídají grafu, vykresleného pomocí vlastních vektorů, jenž obsahuje vrcholy se stejnými souřadnicemi, neboli se překrývají. Červená políčka jsou neuvažována.

	Vykreslovací algoritmus	Metrika minimálního úhlu	Metrika překřížení	Metrika symetrie
Úplný graf o 5-ti vrcholech	Iniciální nastavení	0,08	0,89	0
	Vykreslení pomocí vlastních vektorů	0,37	1	0
	Eadesův algoritmus	0,4	0,67	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,2	0,8	1
Graf strom	Iniciální nastavení	0,65	0,5	0
	Vykreslení pomocí vlastních vektorů	0,83	1	0
	Eadesův algoritmus	0,94	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,97	1	1
Graf mřížka	Iniciální nastavení	0,03	0,9	0
	Vykreslení pomocí vlastních vektorů	0,66	1	1
	Eadesův algoritmus	0,64	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,64	1	1
Graf hexagon	Iniciální nastavení	0,04	0,89	1
	Vykreslení pomocí vlastních vektorů	0,67	1	1
	Eadesův algoritmus	0,67	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,5	0,89	1
Graf krychle	Iniciální nastavení	0,42	0,95	0
	Vykreslení pomocí vlastních vektorů	0,2	1	0,05
	Eadesův algoritmus	0,44	0,95	0,98
	Vykreslení pomocí Laplaceových lineárních rovnic	0,54	1	1
Graf kružnice o 12-ti vrcholech	Iniciální nastavení	0,82	1	0,33
	Vykreslení pomocí vlastních vektorů	0,83	1	1
	Eadesův algoritmus	0,83	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,33	0,98	1
Graf úplný bipartitní	Iniciální nastavení	0,04	0,5	0
	Vykreslení pomocí vlastních vektorů	0,17	0,83	0
	Eadesův algoritmus	0,12	0,5	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,43	0,83	1
Graf náhodný o 8-mi vrcholech	Iniciální nastavení	0,31	0,87	0
	Vykreslení pomocí vlastních vektorů	0,33	1	0
	Eadesův algoritmus	0,54	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,5	0,97	0
Graf náhodný o 30-ti vrcholech	Iniciální nastavení	0,2	0,97	0,77
	Vykreslení pomocí vlastních vektorů	0,014	0,83	0,001
	Eadesův algoritmus	0,05	0,83	0,75
	Vykreslení pomocí Laplaceových lineárních rovnic	0,01	0,85	0,77
Graf hvězda	Iniciální nastavení	0,89	1	0
	Vykreslení pomocí vlastních vektorů	0,86	1	0
	Eadesův algoritmus	0,86	1	1
	Vykreslení pomocí Laplaceových lineárních rovnic	0,86	1	1

Obrázek 8.43: Tabulka celkového přehledu metrik pro vykreslení.

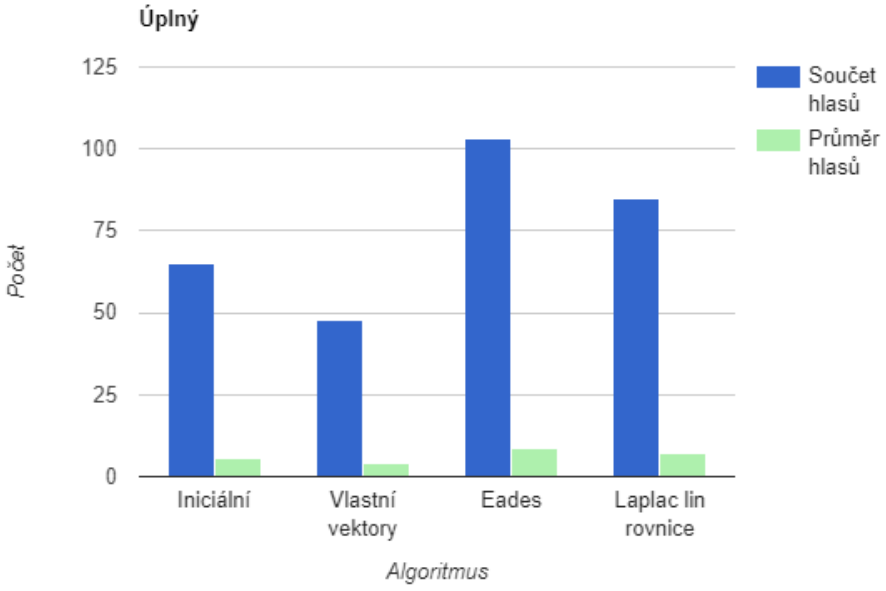
8.2 Názory pozorovatelů

Každému z dvanácti pozorovatelů byly postupně ve formuláři ukázány čtyři obrázky stejného grafu – iniciální nastavení, vykreslení pomocí vlastních vektorů, vykreslení pomocí Eadesova algoritmu a vykreslení pomocí Laplaceových lineárních rovnic pro každý typ grafu. Každý z obrázků byl označen typem grafu a pořadovým číslem, kupříkladu „Graf strom 1.“. Každý z respondentů měl ohodnotit obrázek číslem od 1 do 10 (s tím, že číslo deset bylo nejvyšší ohodnocení) a to podle toho, jak moc se mu graf vzhledově líbí. Tento typ testování byl zvolený kvůli tomu, abychom prozkoumali, jestli respondenti budou volit jakožto hezčí grafy ty, které mají vyšší hodnoty metrik. Ze se-

sbíraných grafů byly vytvořeny sloupcové grafy, které ukazují celkový součet ohodnocení pro dané vykreslení (modrá) a průměrné bodové ohodnocení pro dané vykreslení (zelená). Pro vykreslení grafů byl využit on–line nástroj z webové stránky.
Výsledky si můžeme prohlédnout níže.

Výsledky úplného grafu

U vykreslení úplného grafu získal nejvíce bodů graf vykreslený Eadesovým algoritmem. Celkem získal 103 bodů a průměr hlasů byl 8,6. Pokud se podíváme do tabulky metrik, jevil se také jako jeden ze dvou nejlépe vykreslených grafů (druhým byl graf vykreslený pomocí Laplaceových lineárních rovnic) vzhledem k metrikám–byl nejsymetričtější a nejlépe mu vyšla metrika minimálního úhlu.

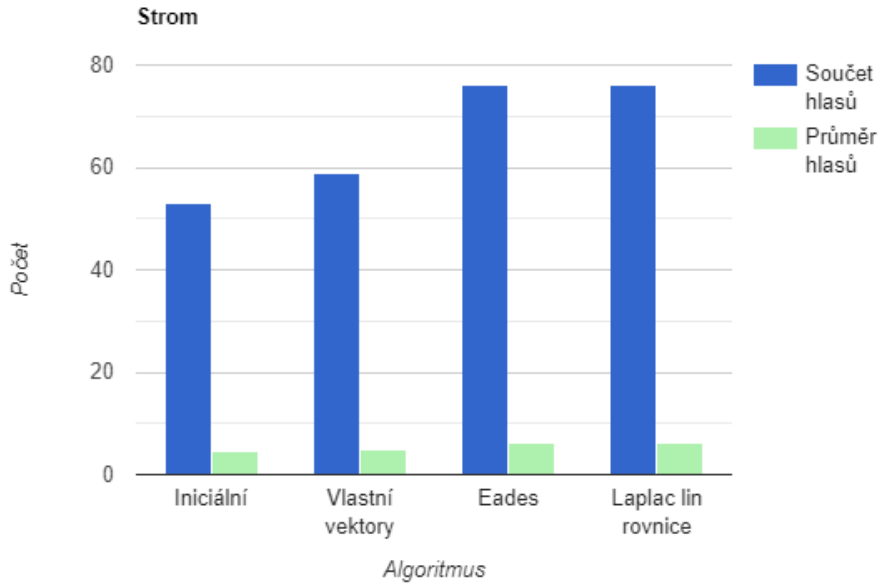


Obrázek 8.44: Graf ohodnocení pro úplný graf.

Výsledky grafu stromu

U tohoto typu grafu získaly nejvíce bodů – 76 a průměr 6,3 vykreslení pomocí Eadesova algoritmu a vykreslený pomocí Laplaceových lineárních rovnic.

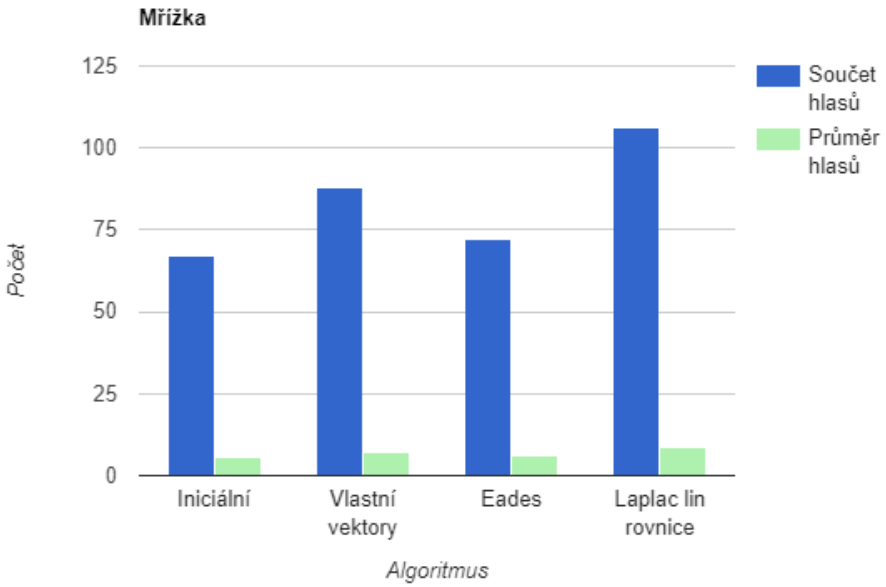
Podle tabulky metrik se jevil nejlépe vykreslený, vzhledem k metrikám graf, který byl vykreslen pomocí Laplaceových lineárních rovnic, jenž dosahoval nejlepšího vykreslení vzhledem ke všem třem metrikám. Eadesův algoritmus na tom byl vzhledem k metrikám jen o maličko hůře, o 0,03 měl menší metriku minimálního úhlu.



Obrázek 8.45: Ohodnocení pro graf stromu.

Výsledky grafu mřížky

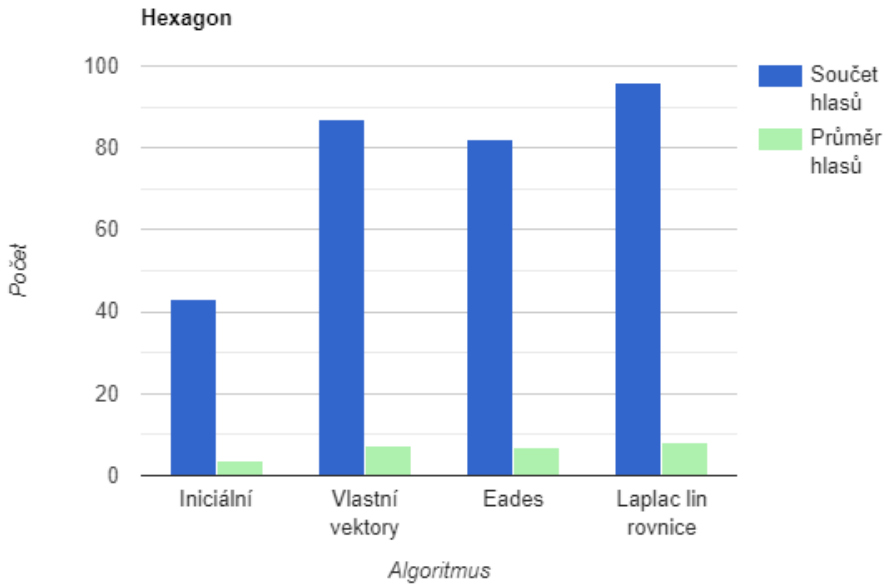
Co se bodového ohodnocení týče, v tomto případě dostal nejvíce bodů graf vykreslený pomocí Laplaceových lineárních rovnic. A to 106 v celkovém součtu a průměrně 8,83. Toto vykreslení dosahovalo v tabulce naměřených metrik nejvyšších hodnot u symetrie a překřížení. Podle tabulky byl ale „nejestetičtěji“ vykreslen graf pomocí vlastních vektorů,který skončil na druhém místě a který byl o malinko lepší u metriky minimálního úhlu. Hodnoty se ale tak moc nelišily.



Obrázek 8.46: Ohodnocení pro graf mřížky.

Výsledky grafu hexagonu

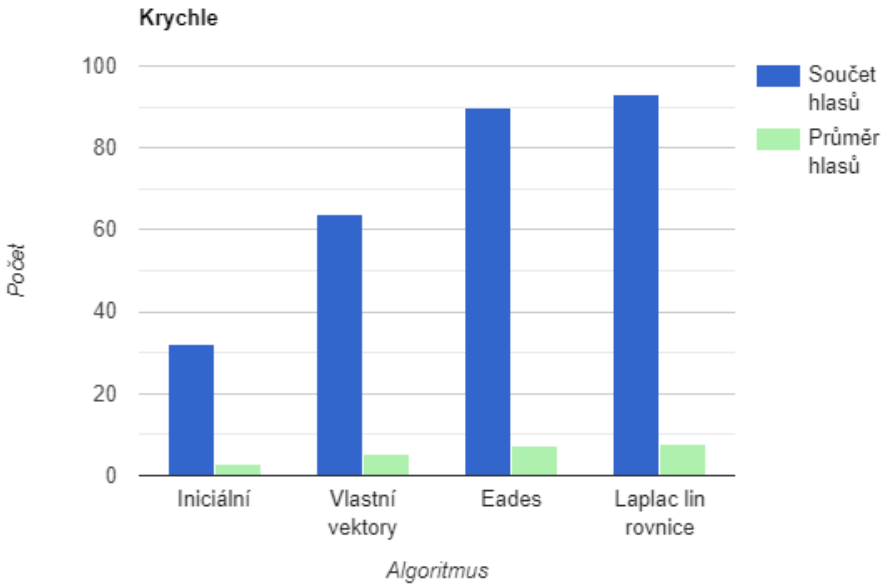
U grafu hexagonu dosáhl nejvíce bodů opět graf vykreslený pomocí Laplaceových lineárních rovnic. Celkem získal 96 bodů a průměr 8. V porovnání s tabulkou na tom nebylo toto vykreslení nejlépe, nejlépe si vedly vykreslení pomocí Eadesova algoritmu a pomocí vlastních vektorů.



Obrázek 8.47: Ohodnocení grafu hexagonu.

Výsledky grafu krychle

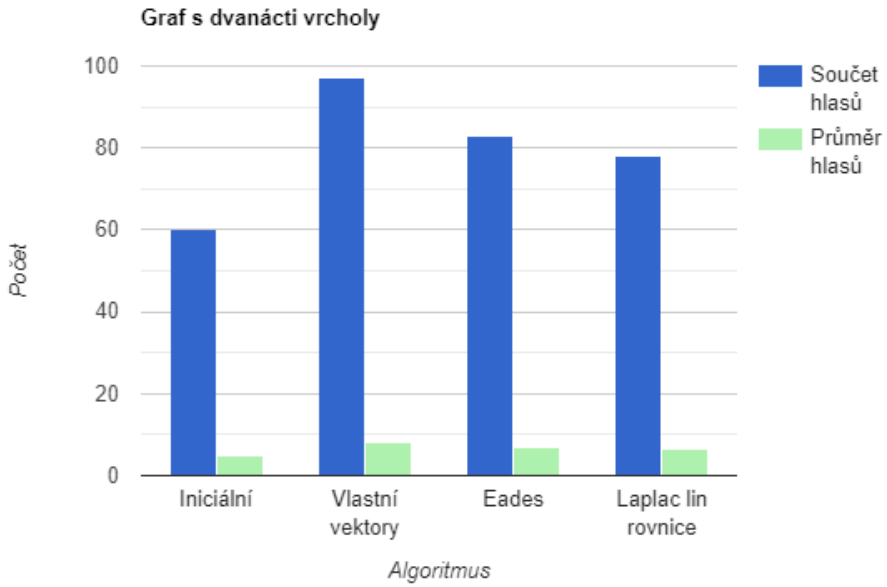
Zde si nejlépe vedlo vykreslení pomocí Laplaceových lineárních rovnic a to s celkem 93 body a průměrem 7,75. Toto vykreslení si vedlo nejlépe i vzhledem k metrikám, ve všech třech metrikách vycházely pro toto vykreslení nejlepší hodnoty.



Obrázek 8.48: Ohodnocení grafu krychle.

Výsledky grafu kružnice o 12–ti vrcholech

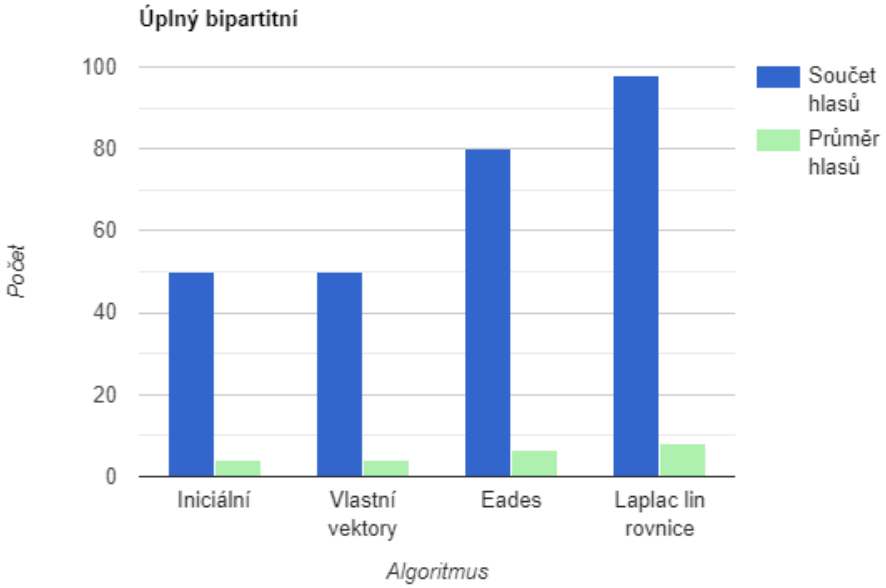
U tohoto grafu si nejlépe vedlo vykreslení pomocí vlastních vektorů se skórem 97 celkového součtu a 8,08 průměru. Při pohledu do tabulky má graf, vykreslený pomocí tohoto algoritmu, nejlepší hodnoty metrik a to spolu s Eadesovým algoritmem, který je zde hned druhý. Autorka se zde domnívá, že tomu tak je z toho důvodu, že graf vykreslený Eadesovým algoritmem má sice téměř totožný tvar, ale není umístěn uprostřed obrázku, nýbrž lehce posunutý a orotovaný do jedné strany. To může pro lidské oko působit méně symetricky.



Obrázek 8.49: Ohodnocení grafu kružnice s dvanácti vrcholy.

Výsledky úplného bipartitního grafu

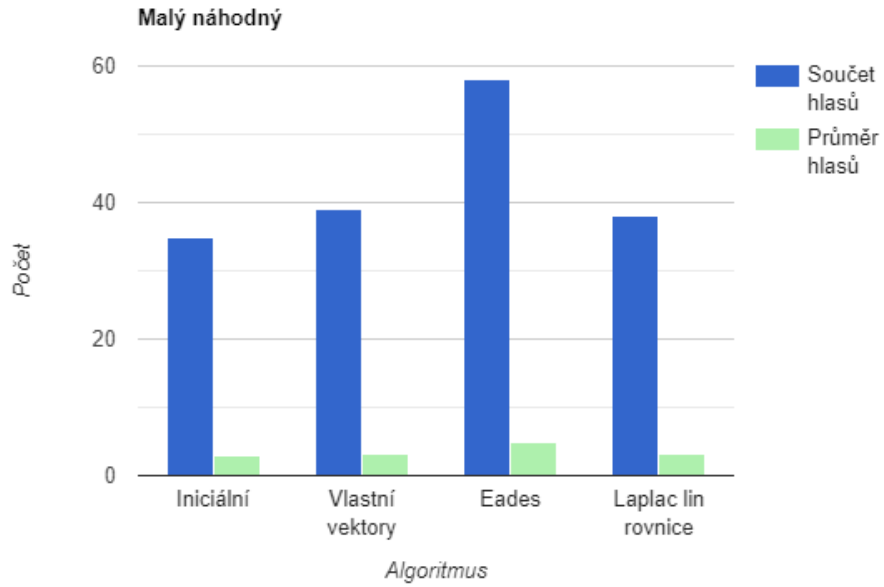
Zde si vedlo nejlépe vykreslení pomocí Laplaceových lineárních rovnic, s s celkovým součtem 98 a průměrem 8,17. Hodnoty u všech třech metrik vyšly pro tento graf a toto vykreslení také nejlépe.



Obrázek 8.50: Ohodnocení úplného bipartitního grafu.

Výsledky malého náhodného grafu o 8–mi vrcholech

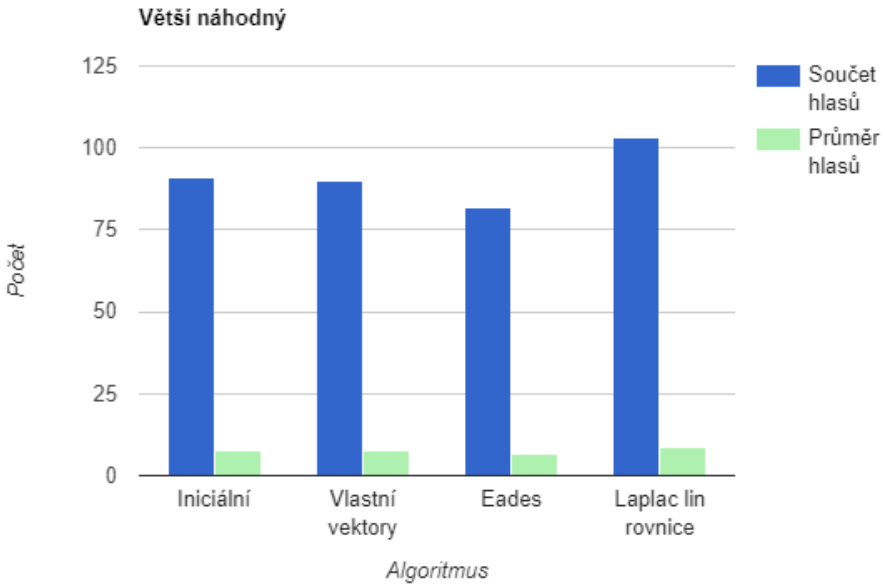
Nejvíce bodů mělo pro tento graf vykreslení Eadesovým algoritmem a to 58 bodů celkem a 4,83 v průměru. Podle metrik má také toto vykreslení nejvyšší hodnoty a to u všech třech metrik.



Obrázek 8.51: Ohodnocení náhodného malého grafu.

Výsledky většího náhodného grafu o 30–ti vrcholech

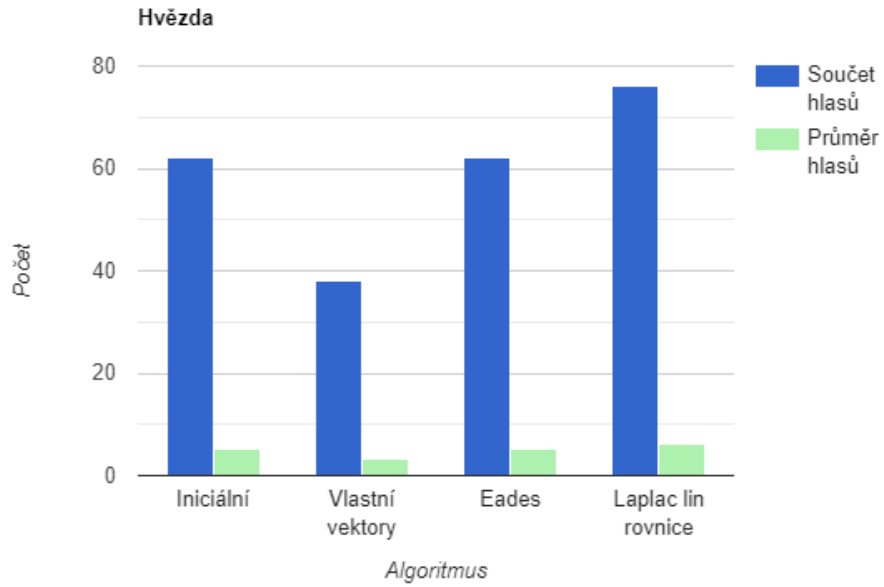
Nejlépe si zde vedlo vykreslené pomocí Laplaceových lineárních rovnic. Celkem získal takto vykreslený graf 103 bodů a 8,58 v průměru. Podle tabulky naměřených metrik ale nejlepší hodnoty vycházely u iniciálního nastavení. Naměřená hodnota symetrie ale byla stejná.



Obrázek 8.52: Ohodnocení většího náhodného grafu.

Výsledky grafu hvězdy

U tohoto posledního grafu vyšlo nejlépe také vykreslení pomocí Laplaceových lineárních rovnic. Celkový součet bodů činil 76, průměr byl 6,33. Po pohledu do tabulky vycházely metriky pro tento graf jako jedny z nejvyšších, jako nejvyšší vyšla hodnota u překřížení a symetrie.



Obrázek 8.53: Ohodnocení grafu hvězdy.

8.2.1 Souhrn výsledků

Po prozkoumání hodnocený pozorovatelů si můžeme všimnout, že ve většině případů vyšel „nejlépe“ ohodnocený graf, který měl zároveň nejlepší hodnoty metrik. Vycházely také nejlépe ty grafy, které působily nejvíce symetricky. Nejlépe hodnocená byla vykreslení pomocí Laplaceových lineárních rovnic, pro různé typy grafů měl nejvyšší ohodnocení celkem sedmkrát. Nejhůře bylo hodnoceno iniciální nastavení. Vykreslení pomocí Laplaceových lineárních rovnic má také, při pohledu do tabulky, nejlépe vycházející hodnoty metrik. Vychází téměř vždy symetricky a ve většině případů dochází k minimalizaci překřížení. Metrika minimálního úhlu pro tento typ vykreslení a příklady grafů vycházela ne už tak dobře, většinou ale došlo ke zlepšení oproti iniciálnímu nastavení.
Co se týká samotného vykreslování, každý z algoritmů měl svá pro a proti. Autorka pozorováním a různým vykreslováním zjistila, že vykreslování pomocí vlastních vektorů nefunguje nejlépe pro určité typy grafů, například pro úplné grafy. To, že vykreslování má své limitace a nehodí se pro určité typy grafů bylo zmíněno již v kapitole 3. Naopak u určitých typů grafů, například kružnice, tento algoritmus vytváří na pohled velmi hezká vykreslení, jenž dosahují i vysokých hodnot, vzhledem k metrikám. Také bylo u tohoto grafu výhodou, že vykreslení nezáleželo na iniciálním rozpoložení vrcholů (na rozdíl od Eadesova algoritmu).

Co se týče vykreslování pomocí Eadesova algoritmu, autorka pozorovala, že celkem dost záleželo na iniciálním rozpoložení vrcholů. Nejlepších výsledků algoritmus dosahoval, když bylo iniciální nastavení hodně podobné finální pozici vrcholů po simulaci. Také, konkrétně pro graf s 30-ti vrcholy, trvalo dlouho, než se graf ustálil a kvalita vykreslení byla lepší spíše pro menší grafy, což bylo zmíněno už u popisu Eadesova algoritmu. Pro menší grafy ale tento algoritmus vytvářel hezká, symetrická vykreslení. U některých typů grafů, které měly nastavené větší hodnoty působících sil, docházelo také k neúplnému ustálení–graf se pohyboval a „rotoval“.
Při vykreslování grafu pomocí Laplaceových lineárních rovnic záleželo na zvolení mezních vrcholů. Při „dobré volbě“ působily vykreslené grafy hezkým, symetrickým dojmem. Pokud byly ale zvoleny vrcholy ne úplně nejlépe, nevedlo to k hezkým výsledkům. Nápad na možné řešení tohoto problému byl nastíněn výše.

Kapitola 9

Závěr

Povedlo se nám seznámit se se všemi třemi vykreslovacími algoritmy a zároveň vytvořit program na vykreslování grafů pomocí těchto algoritmů. Seznámili jsme se jak s teoretickým podkladem, ze kterého algoritmy vychází, tak i s výhodami a nevýhodami těchto algoritmů a také s případným rozšířením a řešením nějakých těchto nevýhod v kódu. Dále jsme se seznámili s metrikami, sloužících k měření estetičnosti nakreslených grafů. Z těchto metrik jsme vybrali podle kritérií vhodné metriky k implementaci do programu a všechny tři vybrané se podařilo naimplementovat. Na závěr jsme na vybraných příkladech grafů ozkoušeli různé vykreslovací algoritmy. U výsledných nakreslených grafů byly zapsány a porovnány jejich metriky. Nakonec jsme výsledky porovnali s názory dvanácti pozorovatelů, kteří grafy hodnotili na základě jejich estetičnosti. Tato pozorování jsme poté srovnali s metrikami, konkrétně tak, že jsme hodnotili, jestli „nejhezčí“ grafy, to jest grafy s největším počtem bodů měly nejlépe vycházející hodnoty metrik.

Kapitola 10

Citace

1. Wikipedia [online]. [cit. 2.1.2023]. Dostupný na WWW: Graph drawing – Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/Graph_drawing

2. Kombinace literatury:
JIROVSKÝ, Lukáš. Základní pojmy/Podgraf [online]. [cit. 4.1.2023] Dostupný na WWW: https://teorie-grafu.cz/zakladni-pojmy/podgraf.php,
DEMEL, Jiří. Grafy a jejich aplikace [online]. [cit. 15.1.2023]. Dostupný na WWW: https://kix.fsv.cvut.cz/~demel/grafy/gr.pdf
AUTOR NEUVEDEN. Úvod do teorie grafů [online]. [cit. 15.1.2023]. Dostupný na WWW: https://www.fd.cvut.cz/personal/xfabera/BIVS/ALG_II/prednasky/prednaska4/grafy.pdf
DEMLOVÁ, Marie. Grafy [online]. [cit. 15.1.2023]. Dostupný na WWW: https://math.fel.cvut.cz/en/people/demlova/lgr/p-lgr812.pdf
PETR, Kovář. Úvod do teorie grafů [online]. [cit. 3.1.2023]. Dostupný na WWW: https://homel.vsb.cz/~kov16/files/uvod_do_teorie_grafu.pdf

3. KHOL, Daniel. Laplaceova matice a její aplikace [online]. [cit. 20.3.2023]. Dostupný na WWW: https://dspace.cvut.cz/handle/10467/103871

4. SPIELMAN, Daniel A. Spectral and Algebraic Graph Theory. Yale University, 2019 [online]. [cit. 2.3.2023]. Dostupný na WWW: http://cs-www.cs.yale.edu/homes/spielman/sagt/sagt.pdf

5. AUTOR NEUVEDEN. Force-directed graf drawing [online]. [cit. 7.5.2023]. Dostupný na WWW: https://en.wikipedia.org/wiki/Force-directed_graph_drawing

6. KOBOUROV, Stephen G.. Force-Directed Drawing Algorithms [online]. [cit. 3.1.2023]. Dostupný na WWW: https://cs.brown.edu/people/rtamassi/gdhandbook/chapters/force-directed.pdf

7. PURCHASE, Helen C. (2002). Metrics for graph Drawing Aesthetics. Journal of Visual Languages Computing, 13(5), 501-516.

8. MITCHELL, Melanie. An Introduction to Genetic Algorithms. Cambridge: First MIT Press, 1998, ISBN 0262133164.

9. LUNER, Petr. Jemný úvod do genetických algoritmů [online]. [cit. 3.1.2023]. Dostupný na WWW: https://cgg.mff.cuni.cz/~pepca/prg022/luner.html

10. LIANG, Frank. Optimization Techniques — Simulated Annealing [online]. [cit. 3.1.2023]. Dostupný na WWW: https://towardsdatascience.com/optimization-techniques-simulated-annealing-d6a4785a1de7

11. AUTOR NEUVEDEN. How to check if two given line segments intersect? [online]. [cit. 3.1.2023]. Dostupný na WWW: https://www.geeksforgeeks.org/check-if-two-given-line-segments-intersect/

12. Pro vykreslení obrázků k vysvětlení výpočtu orientace úseček byl použit nástroj https://csacademy.com/app/graph_editor/

13. Příklady grafů z kapitoly 2 byly vykresleny pomocí nástroje z https://csacademy.com/app/graph_editor/

14. Grafy v kapitole výsledky byly vykresleny pomocí nástroje z https://www.justfreetools.com/cs

15. Obrázek: P. Gajer and S. G. Kobourov. GRIP: Graph dRawing with Intelligent Placement. Journal of Graph Algorithms and Applications, 6(3):203–224, 2002