



MATEMATICKO-FYZIKÁLNÍ FAKULTA Univerzita Karlova

BAKALÁŘSKÁ PRÁCE

Zuzana Šimečková

Varianty Eberhardovy věty

Informatický ústav Univerzity Karlovy

Vedoucí bakalářské práce: doc. Mgr. Robert Šámal, Ph.D.

Studijní program: Informatika

Studijní obor: IOI

Praha 2018

Prohlašuji, že jsem tuto bakalářskou práci vypracovala samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V dne

Podpis autora

Děkuji doc. Mgr. Robertu Šámalovi, Ph.D., vedoucímu práce, za ochotu, trpělivost, odborné rady a čas, které mi věnoval. Za výběr tématu a připomínky při řešení i sepisování práce.

Název práce: Varianty Eberhardovy věty

Autor: Zuzana Šimečková

Ústav: Informatický ústav Univerzity Karlovy

Vedoucí bakalářské práce: doc. Mgr. Robert Šámal, Ph.D., Informatický ústav Univerzity Karlovy

Abstrakt: Pro konkrétní nakreslení rovinného grafu definujme posloupnost $(p_k) = (p_3, p_4, \dots)$ počtů k-hranných stěn – k-úhelníků. Důsledkem Eulerova vzorce o rovinných grafech pro kubické grafy splňuje p vztah $\sum_{k \geq 3} (6 - k)p_k = 12$. Je celkem přirozené ptát se, jak vypadají p , pro která existuje odpovídající graf. Eberhard ukázal, že pokud p splňuje výše uvedenou rovnost, pak existuje rovinný kubický graf, který odpovídá p až na počet šestiúhelníků. DeVos a kol. dokázali obdobu věty, kde je povoleno k p přidat pětiúhelníky a sedmiúhelníky. V této práci na jejich výsledky navazujeme, využijeme jejich důkazové strategie a díky navrženému programu najdeme stavební bloky, které autorům k zobecnění věty chyběly. Výsledkem práce je následující věta: pro každou dvojici $r, s \in \mathbb{N}$ splňující $s < 6 < r < 14$, s, r nesoudělné, platí následující věta: pro každou posloupnost p nezáporných celých čísel splňující $\sum_{k \geq 3} (6 - k)p_k = 12$ existuje nekonečně mnoho kubických rovinných grafů, které p odpovídají až na r -úhelníky a s -úhelníky.

Klíčová slova: rovinné grafy kreslení grafů

Title: Eberhard-Like Theorems

Author: Zuzana Šimečková

Institute: Computer Science Institute of Charles University

Supervisor: doc. Mgr. Robert Šámal, Ph.D., Computer Science Institute of Charles University

Abstract: Define sequence $(p_k) = (p_3, p_4, \dots)$ as numbers of k -sized faces – k-gons – of an embedding of a planar graph. A corollary to Euler's formula for planar graphs states that for cubic graphs $\sum_{k \geq 3} (6 - k)p_k = 12$ holds. Naturally, this leads us to explore the nature of p for which a corresponding cubic planar graph exists. Eberhard proved that if p satisfies the equality above then a cubic planar graph that corresponds to p except for the number of hexagons, exists. DeVos et al. show similar theorem, but instead of hexagon, both pentagons and heptagons can be added. In this thesis, we extend their result by using their proof strategy and designing a program to find graphs needed in such proof. We were able to prove that for every pair $r, s \in \mathbb{N}$ where $s < 6 < r < 14$ and r, s are coprime the following theorem holds: for each sequence of nonnegative integers satisfying $\sum_{k \geq 3} (6 - k)p_k = 12$ there are infinitely many cubic planar graphs corresponding to p except for the number of both r -gons and s -gons.

Keywords: planar graphs graph drawing

Obsah

Úvod	2
1 Pojmy a definice	3
2 Strategie důkazu	5
2.1 Triarky a operace s nimi	5
2.2 Důkaz za předpokladu existence pomocných grafů	7
3 Řešítko	9
3.1 Algoritmus	9
3.2 Program a výstupy	13
4 Kreslení	14
5 Výsledky	17
Závěr	19
Seznam použité literatury	20
A Přílohy	21
A.1 Program	21
A.2 Získané výsledky	21

Úvod

Zkusíme-li upravovat Eulerův vzorec pro souvislé rovinné grafy $|V| - |E| + |S| = 2$, můžeme pro kubické grafy dospět do tvaru

$$\sum_{k \geq 3} (6 - k)p_k = 12, \quad (1)$$

kde p_k značí počet k -hranných stěn grafu. Výraz (1) je nutnou podmínkou existence rovinného kubického grafu s daným počtem stěn. Pozoruhodně, počet šesti-hranných stěn v této podmínce nehraje žádnou roli. Toho si všiml Eberhard [1], který v roce 1891 formuloval a dokázal, že pokud můžeme volit p_6 , dokážeme najít kubický graf zadáný počtem ostatních stěn, který je rovinný.

Věta 1 (Eberhardova věta). *Pro každou posloupnost $(p_k \mid 3 \leq k \neq 6)$ kladných celých čísel, splňující (1) existuje taková hodnota p_6 , že existuje rovinný 3-regulární 3-souvislý graf, který má právě p_k k -hranných stěn pro každé $k \in \mathbb{N}, k \geq 3$.*

Na větu navázali další a dnes je známá celá řada jejích obměn. Fisher [2] dokázal silnější verzi Eberhardovy věty, kde počet potřebných 6ti-hranných stěn shora omezil počtem ostatních stěn chtěného grafu. Grünbaum [3] představuje vlastní stručnější důkaz Eberhardovy věty a shrnuje výsledky podobného typu.

DeVos a kol. [4] představuje obdobu Eberhardovy věty, kde místo p_6 můžeme volit p_5 a p_7 . Článek také ukazuje, že strategie důkazu, kterou autoři použili, by mohla jít uplatnit i pro další velikosti stěn, které mohou splnit (1).

V tomto textu nejprve zavedeme potřebnou terminologii a představíme zmiňovanou strategii důkazu, která pro dokončení potřebuje najít pomocné grafy. Cílem práce bylo tyto grafy získat. V závěrečných kapitolách navrhнемe program, který bude potřebné grafy hledat, a nakonec představíme, pro které velikosti stěn jsme grafy získali a tedy dokončili důkaz dalších variant Eberhardovy věty.

1. Pojmy a definice

Pokud čtenář není seznámen se základy teorie grafů, doporučujeme začít třeba knihou Kapitoly z diskrétní matematiky [5]. Před zadefinováním základní terminologie pro tuto práci dokažme nejprve, jak z Eulerova vzorce získáme (1). Doplňmě navíc, že – pokud není řečeno jinak – uvažujeme pouze souvislé grafy a to ve všech větách i důkazech.

Důkaz. Pro každý kubický graf $G = (V, E)$ platí $3|V| = 2|E|$. Obecně platí, že počet stěn s můžeme přepsat následovně $s = \sum_{k \geq 3} p_k$ a pro rovinné grafy platí i $\sum_{k \geq 3} k \cdot p_k = 2|E|$, protože pokud pro každou stěnu započteme každou její hranu, získáme právě $2|E|$. Úpravou a dosazením do vzorce získáme požadovaný výraz:

$$\begin{aligned} |V| - |E| + |S| &= 2 \\ -\frac{1}{3}|E| + s &= 2 \\ -\frac{1}{6} \sum_{k \geq 3} k \cdot p_k + \sum_{k \geq 3} p_k &= 2 \\ \sum_{k \geq 3} (6 - k)p_k &= 12 \end{aligned}$$

□

Zavedme nyní základní pojmy. Stěně rovinného grafu, která se skládá z k hran, budeme říkat jednoduše **k -úhelník**. Pokud máme posloupnost $(p_k) = (p_3, p_4, \dots)$, která bude představovat počty k -úhelníků v grafu, definujme $\mathbf{P} = \{k \mid p_k \neq 0\}$ jako její **množinu stěn**.

Klasifikujme posloupnosti $(p_k) = (p_3, p_4, \dots)$ podle jejich vlastností.

Definice 1 (Neutrální posloupnost). *Posloupnost $(p_k) = (p_3, p_4, \dots)$ nezáporných celých čísel je neutrální, pokud $\sum_{k \geq 3} (6 - k)p_k = 0$.*

Definice 2 (Přípustná posloupnost). *Posloupnost $(p_k) = (p_3, p_4, \dots)$ nezáporných celých čísel je přípustná, pokud $\sum_{k \geq 3} (6 - k)p_k = 6$.*

Definice 3 (Realizovatelná posloupnost). *Posloupnost $(p_k) = (p_3, p_4, \dots)$ nezáporných celých čísel je realizovatelná, pokud existuje konečný rovinný kubický graf, který má právě p_k k -úhelníků.*

Všimněme si, že původní Eberhardova věta, a i mnoho jejích variant, byla formulována pro 3-souvislé grafy, tedy přesněji pro jednoduché konvexní 3-polytopy. Bijekci mezi těmito dvěma strukturami dokázal až o několik desítek let později Steinitz [6].

Pro ujištění pojmu uvedme následující jednoduché vlastnosti neutrálních, přípustných a realizovatelných posloupností, které využijeme později.

Pozorování 2. *Pro každou neutrální (či přípustnou nebo realizovatelnou) posloupnost $p = (p_3, p_4, \dots)$ existuje $h \in \mathbb{N}$, že $\forall k \in \mathbb{N}, k > h$ platí $p_k = 0$.*

Důkaz. Neutrální posloupnost p splňuje $\sum_{k \geq 3} (6 - k)p_k = 0$, což můžeme upravit na $\sum_{3 \leq k \leq 5} (6 - k)p_k = \sum_{k \geq 5} (k - 6)p_k$. Všechny sčítance na obou stranách výrazu jsou kladné. Hodnota levého součtu tří sčítanců je konečná. Pravý součet má tedy také konečnou hodnotu a navíc mají všechny jeho nenulové sčítance hodnotu alespoň 1. Nenulových hodnot v posloupnosti je tedy pouze konečně. \square

Pozorování 3. Pro každou q neutrální posloupnost $\exists a,b \in Q : a < 6 \wedge b > 6$.

Součet dvou neutrálních posloupností je neutrální posloupnost. Součet neutrální a přípustné posloupnosti je přípustná posloupnost.

Pro každou přípustnou posloupnost $(p_k) = (p_3, p_4, \dots)$ existuje neutrální posloupnost q , že $p + qn$ je realizovatelná pro nějaké $n \in \mathbb{N}$.

Důkaz. Pro ukázání první vlastnosti postačí, když si uvědomíme, že chceme $\sum_{k \geq 3} (6 - k)p_k = 0$, kde pro $k < 6$ je sčítaný člen kladný, zato pro $k > 6$ je sčítaný člen záporný. Druhá vlastnost plyne z definice a distributivity násobení. Třetí tvrzení pro je slabší verze Věty 1. \square

K poslední vlastnosti se nabízí otázka, jak lze q omezit, aby pořád $p + qn$ byla realizovatelná. Eberhard ukázal, že stačí $q = (0,0,0,q_6 = 1,0,\dots)$, tedy přidat šestiúhelníky. DeVos a kol. ukázal že i omezení na $q = (0,0,1,0,1,0,\dots)$ funguje. V této práci ukážeme, že posloupností s pouze dvěma nenulovými hodnotami, které v tomto směru vyhovují, existuje výrazně více.

2. Strategie důkazu

Představme hypotézu, kterou se snažíme ověřit.

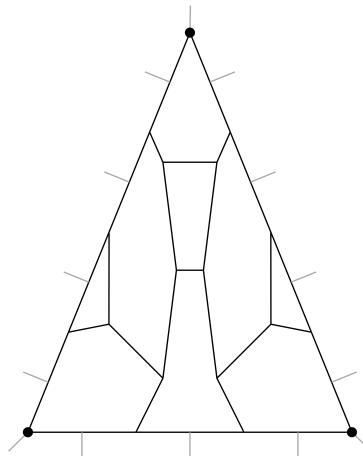
Hypotéza 4. *Mějme přípustnou posloupnost $p = (p_k | 3 \leq k \neq 6)$ a neutrální posloupnost $q = (q_k | 3 \leq k \neq 6)$, pak existuje takové přirozené n , že $p + nq$ je realizovatelná.*

V článku [4] autoři naznačují strategii důkazu, za předpokladu, že existují nějaké pomocné grafy. V další kapitole představíme způsob, jak takové grafy hledat. Ted se zaměříme na důkaz samotný, respektive nejprve představíme graf, který v důkazu pomáhal již Eberhardovi.

2.1 Triarky a operace s nimi

Definice 4 (Triark). *Triark je takový rovinný graf T , že vrcholy jeho vnější stěny tvoří cyklus C , každý vnitřní vrchol (tj. vrcholy $T - C$) má v T stupeň právě 3 a v C jsou tři navzájem různé vrcholy x, y, z stupně 2 – **rohy**, že vrcholy každé ze tří cest v C , které vzniknou odstraněním rohů z C , mají střídavě stupeň 2 a 3, počínaje i konče stupněm 2.*

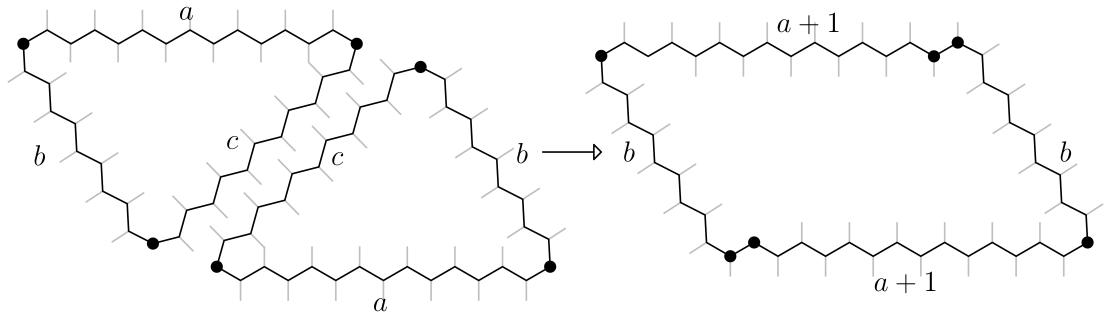
Strana triarku je každá z výše zmíněných cest v C , ke které na oba konce připojíme i příslušný roh. **Délka strany** triarku odpovídá počtu jejích vnitřních vrcholů stupně 2 v T . O triarku se stranami délky a, b, c mluvíme jako o **(a, b, c)-triarku**. Poznamenejme, že na pořadí stran v názvu nezáleží (odpovídají rotacím a zrcadlení). Později využijeme ještě dalšího značení. **M -triark** má vnitřní strany pouze velikostí z množiny M . **(a, b, c)- M -triark** je zároveň (a, b, c) -triark a M -triark. Tato značení využíváme i pro další typy grafů a jejich význam je analogický. U triarku přejímáme některé termíny používané pro trojúhelníky a jejich význam bude vždy intuitivní.



Obrázek 2.1: $(4,4,3)\{4,7\}$ -triark.

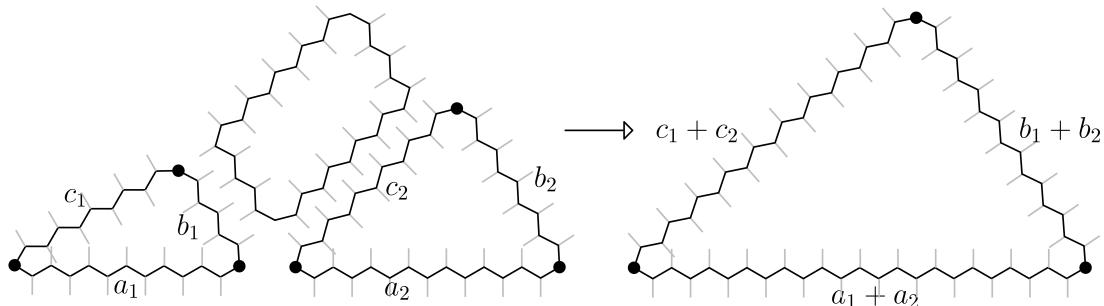
Zmiňme velmi užitečnou vlastnost triarků: pokud máme dva triarky, oba mající stranu stejné délky, můžeme je za tuto stranu slepit jako na Obrázku 2.2

a získáme opět graf, jehož vnitřní vrcholy mají stupeň 3. (Při slepovaní se ztožní vždy vrchol stupně 2 s vrcholem stupně 3.) Pokud slepíme triarky tak, že protější strany výsledného grafu budou stejně dlouhé (tedy speciálně při slepení dvou stejných triarků), budeme podle jeho tvaru mluvit o **rovnoběžníku**. Rovnoběžníky jdou navíc slepovat podobně jako triarky a můžeme tím každý již existující rovnoběžník zvětšit na libovolný rozměr, který je násobkem jeho původních rozměrů.



Obrázek 2.2: Spojení dvou triarků za vzniku rovnoběžníku.

Mohli bychom ale chtít spojovat triarky tak, aby výsledkem byl opět triark. Mějme (a_1, b_1, c_1) -triark a (a_2, b_2, c_2) -triark a vhodný rovnoběžník. Slepáním, jako na Obrázku 2.3, vznikne $(a_1 + a_2, b_1 + b_2, c_1 + c_2)$ -triark.



Obrázek 2.3: Spojení triarků spolu s rovnoběžníkem za vzniku triarku.

A závěrem budeme chtít spojit dva triarky tak, aby výsledkem byl kubický graf (tedy aby nevznikly žádné nové stěny s vrcholy stupně dva). Graf, který má tuto funkci označíme za **prstenec**. Pro lepší představu si prstenec představujme jako pláště trojbokého hranolu, triarky jako dolní a boční podstavu. Na všech podstavných hranách dojde ke splynutí vrcholů triarků s vrcholy prstence a vznikne graf nakreslitelný na hranol (tedy tedy i na sféru), takže výsledkem je rovinný graf. Zadefinujme prstenec formálně. Prstenec je souvislý kubický rovinný graf, ve kterém můžeme vyznačit dva disjunktní cykly tak, že stupně vrcholů každého cyklu jsou opačné než stupně příslušných triarků, které chceme spojit. Kde opačné znamená záměna dvoj- a třívaznosti vrcholů. Pro případné dokreslení můžeme použít Obrázek 3.3, na kterém je speciální druh prstence.

2.2 Důkaz za předpokladu existence pomocných grafů

Převedme hypotézu ve větu.

Věta 5 (DeVos a kol. [4] Sekce 4). *Mějme přípustnou posloupnost $p = (p_k | 3 \leq k \neq 6)$ a neutrální posloupnost $q = (q_k | 3 \leq k \neq 6)$ a následující grafy pro nějaké přirozené k :*

- (i) $(k, k, k)Q$ -triark;
- (ii) $(k, k, k-1)Q$ -triark;
- (iii) $\forall p_l \neq 0$ rovnoramenný $Q \cup \{l\}$ -triark, délky jehož stejných stran jsou dělitelné k a který obsahuje právě jednu stěnu velikosti l , této stěně říkejme **jádro triarku**;
- (iv) $\forall t \in \mathbb{N}, t > 1$ Q -prstenec, který dokáže spojit dva (t, t, t) -triarky.

Počet stěn Q v každém z grafů (i) – (iv) navíc musí zachovat poměr z q . Pak existuje takové přirozené n , že $p + nq$ je realizovatelná.

Myšlenka důkazu pak není příliš složitá: každou stěnu ze zadанé posloupnosti p zabalíme do triarku (iii), připravíme si pomocné lepicí a zkrášlující prvky (ii), díky kterým získáme jediný velký rovnostranný triark. K němu zkonstruujeme ještě jeden se stejně dlouhými stranami (i) a pomocným prstencem je spojíme v kýžený graf (iv). Všechny tyto pomocné objekty jsou totiž rovinné a (kromě jader) jen ze stěn, které jsou v zadáné neutrální posloupnosti. Lepicí operace zachovávají kubičnost uvnitř grafu a prstenec pak spojí dva triarky v hledaný kubický graf.

Důkaz. Nejprve slepíme dva $(k, k, k-1)$ -triarky za stěnu délky k a získáme rovnoběžník se všemi stranami délky k . A díky slepování můžeme získat i libovolný rovnoběžník o rozměrech mk, lk pro m, l přirozená. Poté postupně spojujeme jednotlivé jádrové triarky za pomoci příslušného lichoběžníku, dokud nezískáme jediný triark, který obsahuje všechny stěny z P .

Zkusme vzniklý triark upravit na rovnostranný. Při slepování triarků se velikosti výsledných stran rovnají součtem původním. Pokud tedy slepujeme s $(k, k, k-1)$ -triarkem, zmenšíme vždy tu stěnu, na kterou připadne rozdíl $k-1$, vůči ostatním. Takže pokud budeme opakovat lepení výsledného triarku, v každém kroku se součet rozdílů mezi stranami zmenší a tedy nutně získáme rovnostranný triark. Modifikujme ho stejnou operací, aby zůstal rovnostranný, ale navíc délka jeho stran byla násobkem k a označme výsledný triark T_1 .

Slepováním (k, k, k) -triarků z (i) spolu s (k, k) rovnoběžníky získáme druhý triark T_2 o stejném rozměru jako T_1 . Kdybychom celý problém řešili na toru, stačilo by T_1 a T_2 spojit do rovnoběžníku a sjednotit odpovídající strany. Na kouli místo toho použijeme prstenec z (iv). Tím získáme graf, který je kubický, rovinný a obsahuje požadované stěny.

□

Všimněme si, že větu jde jednoduše zesílit: nejen že za splnění předpokladů existuje nějaké $n \in \mathbb{N}$, pro které je $p + nq$ realizovatelná; existuje jich libovolně mnoho. V důkazu stačí před spojením T_1 a T_2 prstencem nejprve oba triarky slepit s dalšími instancemi T_2 tak, aby vznikly nové, stejně velké rovnostranné triarky T_1^* a T_2^* . Tímto způsobem lze libovolně zvětšit n . T_1^* a T_2^* pak opět spojíme prstencem.

3. Řešítko

Abychom mohli dokončit důkaz některých instancí Hypotézy 4, potřebujeme získat požadované stavební bloky. Nabízí se naprogramovat řešítko, které bude umět alespoň některé typy hledaných grafů najít. Hlavním cílem této práce bylo takový program připravit a pomocí něj získat lepší představu o potenciálu uvedené strategie důkazu.

3.1 Algoritmus

Program na vstupu očekává zadání vnější stěny: každý vrchol je zastoupen jedním bitem, který určuje, zda má být ve výsledném grafu stupně 2 nebo 3. Navíc očekává seznam velikostí stěn, které má využít. Na výstupu informuje, zda se mu daný graf podařilo najít (říkejme **vyplnit**), a umožní jej exportovat.

Postup hledání původně imitoval lidské pokusy o řešení problému:

1. nakreslit si vnější stěnu,
2. vybrat si dvojici vrcholů, kterým ještě chybí soused, a spojit je řetízkem vhodné délky (aby nově vzniklá oblast byla stěna a měla velikost z neutrální posloupnosti)
3. krok 2 opakovat, dokud je místo na papíře.
4. Když místo dojde, překreslit si nejvnitřnější, zatím neuzavřenou stěnu (budem mluvit o **hranici**). Ta se stane „vnější stěnou“ na novém papíře a pokračovat od kroku 1.
5. V situaci, kdy nelze dál nic spojit, nebo je jasné, že graf nemůže vyhovovat parametrům, se vrátit podle uvážení zpět.

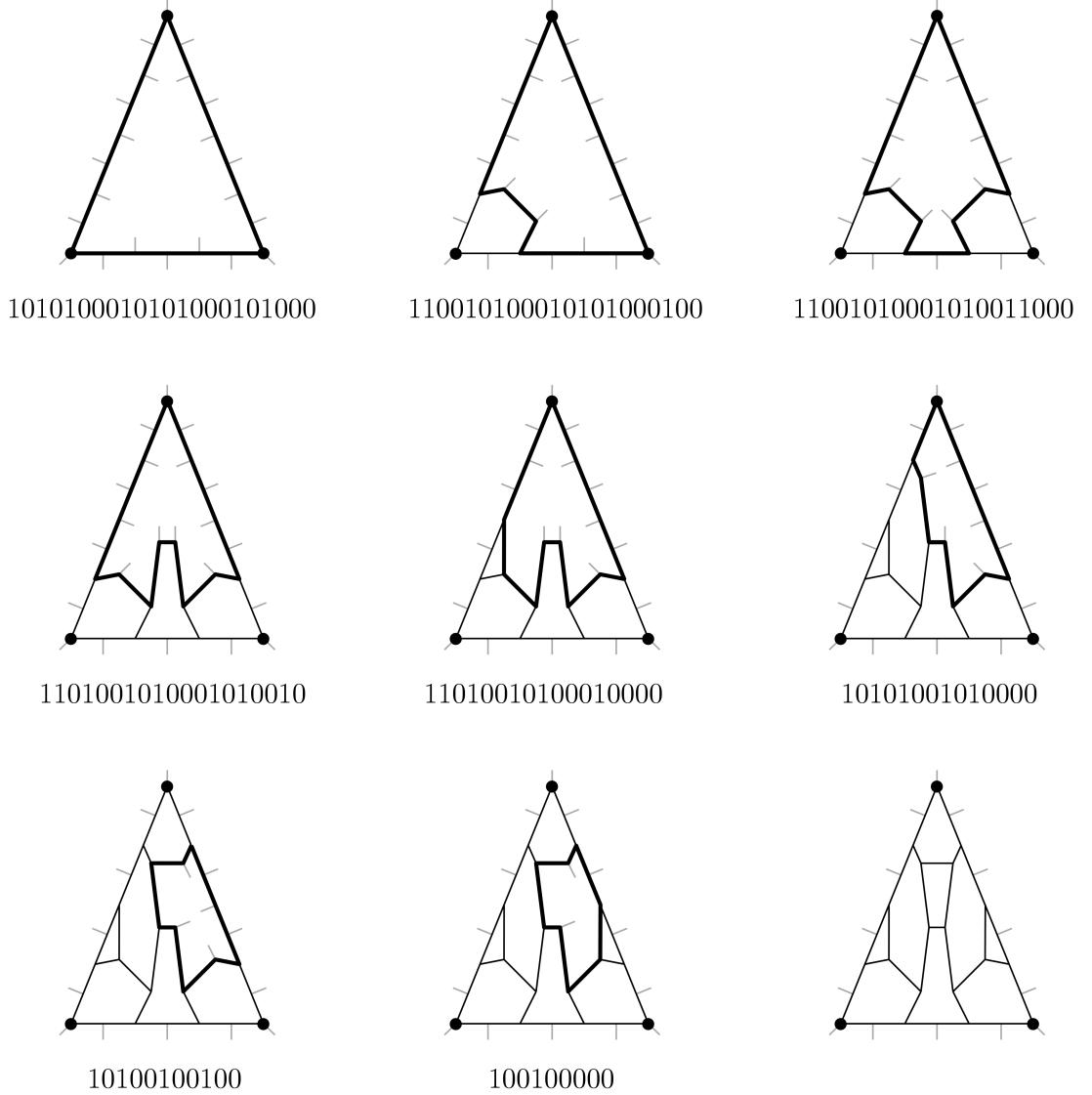
Část běhu algoritmu je znázorněna na Obrázku 3.1, nejsou v něm zaznamenány „slepé uličky“.

Kdybychom chtěli znát jen „ano/ne“ odpověď, jestli graf existuje, nebylo by vůbec třeba si pamatovat celý rozpracovaný graf, stačilo by pracovat s hranicemi, které navíc stačí reprezentovat jako binární číslo. Výsledkem by pak mohla být jen posloupnost hranic, kterými se prošlo před uzavřením grafu, nebo samotné „ano/ne“. Překvapivě obtížné je pak z této posloupnosti nestrojově získat skutečný graf, proto program nabízí i možnost graf dodatečně rekonstruovat podle prošlých stavů.

V tento okamžik je jasné, že problém je vlastně prohledávání v binárních řetězcích (které reprezentují hranice). Je proto vhodné zmínit, podle jakého kritéria se program rozhoduje, kterým směrem hledat dále. Implementace vždy upřednostňuje ke zpracování již nalezený řetězec nejmenší délky, a pro něj najde všechny další sousedy. Pro názornost přikládáme i pseudokód Vyplň. Vzhledem k návrhu lze algoritmus implementovat vícevláknově – každou nezpracovanou hranici lze zpracovávat v jiném vlákně, stačí zajistit bezpečnost sdílených datových struktur.

Aby toto prohledávání fungovalo dobře, je třeba trochu zkomplikovat reprezentaci hranice. Hlavním požadavkem bude identita mezi reprezentací (jediným

binárním řetězcem) a všemi hranicemi (tedy cykly, na kterých vyznačené vrcholy ještě vyžadují dalšího souseda), které jsou pro algoritmus izomorfní – tedy všechny rotace a převrácení hranice.



Obrázek 3.1: Možný postup vyplnění $(4,4,3)\{4,7\}$ -triarku. Hodnota pod grafem vždy odpovídá reprezentaci aktuální, tučně zvýrazněné, hranice.

Definice 5 (Hranice a její reprezentace). *Dvojici cyklus C a množina $I \subseteq V(C)$ nazveme hranicí H . Množina I jsou právě ty vrcholy, které ve výsledném vyplnění musí mít dalšího souseda. Pokud $I = \emptyset$, mluvíme o hranici přímo jako o stěně a nedefinujeme pro ni reprezentaci.*

Definujme funkci $f(H, (u, v))$ jako charakteristický vektor množiny I následně: každý vrchol H popišme buď znakem 1 (jako I v „in“ podle orientace pomyslené hrany) nebo 0 (jako 0 v „out“). Hodnota $f(H, (u, v))$, kde $H = (C, I)$, $u, v \in V(C)$ a $\{u, v\} \in E(C)$, se pak získá zaznamenáním popisků vrcholů H , začínaje vrcholem u , pokračujíce vrcholem v a dále po hranách C .

Reprezentací hranice je hodnota $\max_{\{u, v\} \in E(C)} f(H, (u, v))$.

Objasněme pojem opět jednoduchým pozorováním a Obrázkem 3.1 – posloupností hranic s jejich reprezentacemi, která řeší $(4,4,3)\{4,7\}$ -triark.

Pozorování 6. *Při výpočtu reprezentace podle definice hledáme maximum přes $2|V(C)|$ hodnot a reprezentace každé hranice začíná znakem 1.*

Důkaz. Každý vrchol v C má dva sousedy, tedy v (multi)množině, ze které hledáme řetězce je $2|V(C)|$ (ne nutně různých) hodnot. Odpovídá to představě, že zkoušíme začít v každém vrcholu a čteme v obou směrech. Vzhledem k tomu, že $I \neq \emptyset$, je v každém přečteném řetězci alespoň jedna jednička. Hledání maxima zajistí, že se upřednostní řetězec, který jedničkou začíná (ten je dostupný, protože se začínalo číst i z vrcholů, které jsou v I).

□

```

procedure VYPLŇ(reprezentace vnější hranice vnějšíHraniceStdRep, velikosti povolených
stran stěny, limit)
    nevyřešenéHranice  $\leftarrow$  vnějšíHraniceStdRep
    přechody  $\leftarrow$   $\emptyset$ 
    while přechody.Počet  $\leq$  limit do
        hranice  $\leftarrow$  nevyřešené.Nejmenší
        rotace  $\leftarrow$  RotaceHraniceZačínajícíInVrcholem(hranice)
        for all h in rotace do
            for all s in stěny do
                nováHraniceStdRep  $\leftarrow$  ZkusSpojitPrvníDvaInVrcholy(h, s)
                if nováHraniceStdRep.MáHodnotu then
                    nevyřešenéHranice.Přidej(nováHraniceStdRep)
                    přechody.Přidej(nováHraniceStdRep, hranice)
                if JeStěna(nováHraniceStdRep, stěny) then
                    vrat Cesta(přechody, nováHraniceStdRep, vnějšíHraniceStdRep)
            oznam nenalezení
    end procedure

```

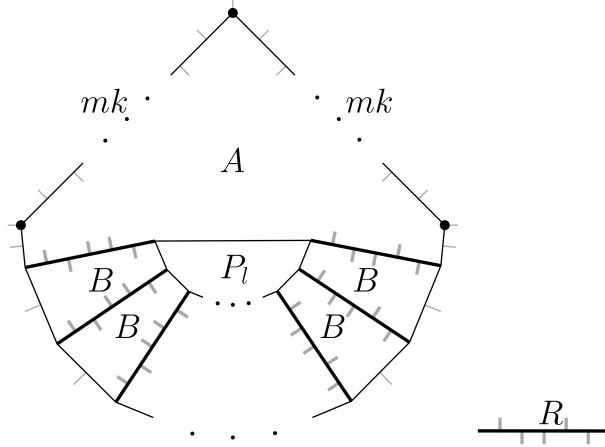
Pro jistotu poznamenejme, že pokud program hledaný graf nenašel, může, ale nemusí to znamenat, že neexistuje.

Podle předchozí kapitoly pro dokončení důkazu pro konkrétní dvojici p a q a nějaké přirozené k potřebujeme tyto čtyři typy grafů:

- (i) $(k, k, k)Q$ -triark;
- (ii) $(k, k, k-1)Q$ -triark;
- (iii) $(mk, mk, x)Q \cup \{l\}$ -triark, kde je právě jedna stěna velikosti l ;
- (iv) prstenec, který dokáže spojit dva stejně velké, rovnostranné triarky.

Pro dané k získáme grafy (i) a (ii) z řešítka hned. Pro zbylé je nutné pomocí si konstrukcí, která se ukázala jako úspěšná pro některé neutrální sekvence. O výsledcích získaných z programu píšeme v Kapitole 5.

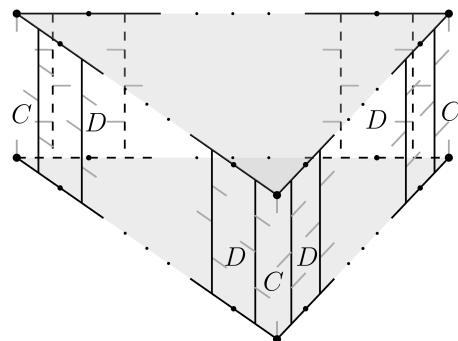
Na graf (iii) se neumíme zeptat přímo, protože potřebujeme v grafu mít právě jednu stěnu délky p_l . Spojme proto stěnu s vnější stěnou triarku ručně a ptejme se na výplň vzniklých oblastí A a B . Ke spojení použijeme l kopií řetízku R ,



Obrázek 3.2: Konstrukce grafu typu (iii) pomocí řetízků R , konkrétně $(mk, mk, l+1)$ -triarku s l -úhelníkem P_l jako jádro.

každý napojíme na jeden z vrcholů jádra a druhé konce spojíme s „in“ vrcholy základny triarku. V řetízku navíc fixujeme, ve které straně od něj budou mít které jeho vrcholy třetího souseda (znázorněno šedě v Obrázku 3.2). Konstrukce je obecná pro všechny přípustné hodnoty l , tedy není závislá na volbě p . Nevíme ani o důvodu, který by implikoval nemožnost jejího využití pro libovolné q .

Podobnou konstrukci tvoříme i pro graf typu (iv). V tomto případě za pomocí řetízků spojujeme odpovídající vrcholy rovnostranných, stejně velkých triarků T_1 a T_2 . Vzniknou dva typy oblastí – C při rozích triarku a D mezi odpovídajícími kusy stran triarků. Přesný nákres je na Obrázku 3.3. Poznamenejme, že pokud jde graf nakreslit na povrch hranolu, je rovinný.



Obrázek 3.3: Konstrukce grafu typu (iv), slepované triarky jsou vyznačené šedě, prstenec odpovídá plásti.

Tyto konstrukce jsou v řešítku implementovány, pro ukázku přikládáme pseudokód Dokaž. Výslednému programu stačí zadat seznam velikostí stěn, které mohou tvořit neutrální posloupnost. Pokud pro daný seznam stěn existuje více neutrálních posloupností, využije program pro každý pomocný graf libovolnou z nich.

```

procedure DOKAŽ(velikosti povolených stran stěny, limit)
    řetízky                                ▷ Předdefinovaná množina řetízků
    velikostiStran                      ▷ Předdefinovaná množina velikostí stran

    iGraf  $\leftarrow$  velikostiStran.Kde(k  $\Rightarrow$  JdeVyplnit(HraniceTriarku(k, k, k), stěny, limit))
    iiGraf  $\leftarrow$  iGraf.Kde(k  $\Rightarrow$  JdeVyplnit(HraniceTriarku(k, k, k-1), stěny, limit))
    řetízkyA  $\leftarrow$  řetízky.Kde(r  $\Rightarrow$  iiGraf.Existuje(k  $\Rightarrow$  JdeVyplnit(HraniceA(r, k), stěny, limit))
    iiiGraf  $\leftarrow$  řetízkyA.Kde(r  $\Rightarrow$  JdeVyplnit(HraniceB(r), stěny, limit))
    řetízkyC  $\leftarrow$  řetízky.Kde(r  $\Rightarrow$  JdeVyplnit(HraniceC(r), stěny, limit))
    ivGraf  $\leftarrow$  řetízkyC.Kde(r  $\Rightarrow$  JdeVyplnit(HraniceD(r), stěny, limit))

    vrať iiiGraf.Počet  $\neq 0$  AND ivGraf.Počet  $\neq 0$ 
end procedure

```

3.2 Program a výstupy

Implementaci výše popsaného algoritmu k práci příkládáme jako spustitelný program (pro Windows) i zdrojové kódy, Příloha A.1. Jde o standardní konzolovou aplikaci, včetně nápovědy pro použití v podobě **-h** argumentu při spuštění. Například Příloha A.2 včetně dat pro Tabulku 5.2 lze vygenerovat argumenty **-r -l 50000 -ExportAsAll**.

Zastavme se u typů souborů, které program může generovat:

Graf vypíše nalezený graf jako seznam hran.

Hranice vypíše posloupnost hranic, podle kterých je graf sestaven, příkladem může být Obrázek 3.1.

Stěny vypíše vnitřní stěny grafu, které odpovídají chodu algoritmu a které jinak nemusí být dány jednoznačně.

Graphviz – neato vypíše graf v jazyku DOT, navíc rozmístí vrcholy vnější hranice grafu na kružnici.

Tuttoovo kreslení vypíše skript pro SageMath, který graf vykreslí podle barycentrické metody, je v něm připravená funkce pro změnu vah vrcholů.

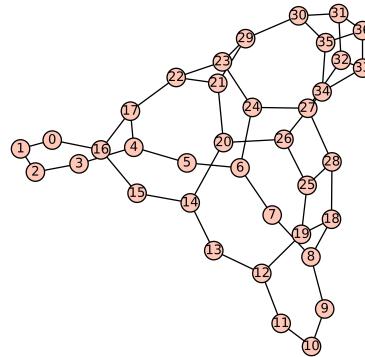
4. Kreslení

Přirozenou snahou pro studování nalezených grafů, a pochopení, proč některé nelze najít, je hledat jejich zobrazení, které je pro člověka dostatečně čitelné. Překvapivě, i přes rovinnost grafů a celkem dobré znalosti jejich struktury není jednoduché hezké rovinné nakreslení najít.

Zamysleme se, jak graf bude vypadat. Vrcholy, které mají ležet na vnější stěně, rozmístěme po kružnici a pak – podle jednotlivých hranic, které graf řeší – vždy vrcholy nově uzavřené stěny nakresleme na soustřednou, menší kružnici tak, aby spojnice žádného bodu hranice se středem kružnic neprotínala jiný bod hranice. Tímto způsobem určitě získáme rovinné nakreslení (ale hrany mohou být libovolné křivky), protože v každém kroku je možné spojit libovolné dva vrcholy, které mohou být v řešení zrovna spojovány, tak, abychom zachovali požadovanou vlastnost tvaru hranice.

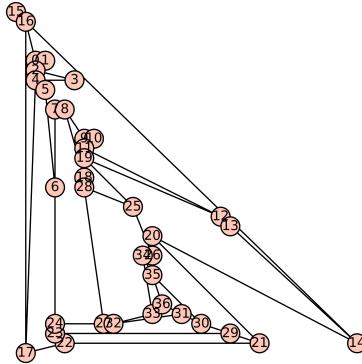
Problémem takového nakreslení je počet soustředných kružnic, které bychom potřebovali, který odpovídá počtu stěn grafu. Množství potřebných kružnic by šlo celkem jednoduše snížit. Nově přidávané vrcholy nakreslíme vždy na největší kružnici, na které v příslušné výseči ještě žádný vrchol neleží. Dalším problémem by bylo rozložení vrcholů do výsečí. Představme si třeba zadání, ve kterém značný podíl tvoří souvislá posloupnost „out“ vrcholů. Pokud ve výpočtu dojde k uzavření stěny, která tyto vrcholy obsahuje, až na závěr, bude výseč, ve které leží, jinak zcela prázdná.

Další možnosti jsou běžně dostupné programy či funkce na kreslení grafů. Posouzení jejich kvality na náhodném z malých nalezených grafů necháváme na čtenáři.

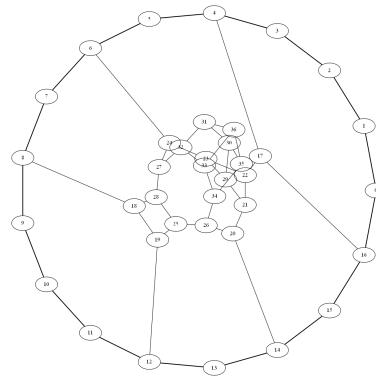


Obrázek 4.1: Automatické nakreslení SageMath [7]. Všimněme si, pokud bychom si zkusili představit, že cyklus vrcholů $0, 1, \dots, 16$ je ona vnější kružnice z vylepšeného prvního popsaného kreslení a navíc nechtě je základnou kužele a všechny další kružnice nechtě jsou vždy v tomto kuželu výš (tak, aby mohly ležet na plášti), pak (s množstvím fantazie) se koukáme na podobné prostorové nakreslení, jen z různých úhlů. Při nakreslení většího grafu je tato myšlenka viditelnější.

Nejuspokojivější nalezenou možností je Tuttův (barycentrický) algoritmus. Jak název napovídá, jde o umisťování vrcholů do „těžiště“. Nejprve je třeba rozdělit vrcholy do dvou skupin: pevné a volné. Pevné vrcholy jsou rozestaveny, aby tvořily konvexní n -úhelník. Pozice volných vrcholů se pak dopočítá jako vážený průměr sousedních vrcholů, tedy stačí řešit soustavu lineárních rovnic.



Obrázek 4.2: Rovinné nakreslení SageMath.



Obrázek 4.3: Graphviz [8] – Neato s předdefinovanými pozicemi vnější stěny.

Aby mohl algoritmus dobře fungovat, je nutné, aby graf byl 3-souvislý (že jde i o postačující podmítku ukazuje článek [9]). Pokud by nebyl 3-souvislý, pak vrcholy komponenty, která by po odebrání dvou vrcholů byla oddělena od zbytku grafu a neobsahovala by pevné vrcholy, budou ležet v jedné přímce.

Pozorování 7. *Mějme graf $G = (V, E)$ a seznam jeho stěn F , oboje získané algoritmem z Kapitoly 3. Pro každou stěnu $f \in F$ vytvořme nový vrchol v_f a bud V^* množina těchto vrcholů. Pak graf $G^* = (V \cup V^*, E \cup \{ \{u, v\} \mid v_f \in V^*, u \in f, f \in F \})$ je 3-souvislý.*

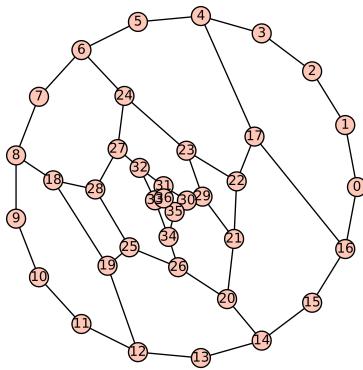
Důkaz. Pro spor nechť jsou vrcholy $u, v \in V \cup V^*$ takové, že po jejich odebrání se graf G^* rozpadne na více komponent. Ze způsobu, kterým graf vzniká, víme, že je 2-souvislý, a proto ani jeden z u, v není z V^* . Označme vrcholy a, b , mezi kterými po odebrání u, v nevede cesta, nechť P je cesta, která je spojuje v G a bez újmy na obecnosti díky 2-souvislosti nechť $u \in P$ a $v \notin P$. Pak vrcholy těsně před a po u v P sdílí v G nějakou stěnu f (protože v G mají vrcholy stupeň nejvýše 3) a tedy lze P upravit vyměněním u za v_f , aby spojovala a a b v grafu po odebrání u, v , což je spor s volbou a a b .

□

Pro převedení grafu na 3-souvislý stačí tedy do každé stěny vložit nový vrchol a spojit ho se všemi vrcholy dané stěny. Aby nakreslení odpovídalo představě, nemůžeme přidat vrchol pro vnější stěnu. I po této modifikaci zůstane nový graf

3-souvislý, v důkazu by stačilo uvážit speciálně případ, kdy vrchol u leží na vnější stěně a tedy nelze použít k nahradě v cestě vrchol vnější stěny. Pokud je v G u dvojvazný, použijme vrchol příslušné vnitřní stěny. Pokud je třívazný, je třeba rozlišit jestli jeho sousední vrcholy v cestě jsou oba na vnější hraně. Pokud ano, obejdeme u pomocí vrcholů vnitřních stěn a vnitřního souseda u . Pokud ne, stačí použít odpovídající vnitřní stěnu.

Pak zbývá zvolit pevné vrcholy – v našem případě volíme vrcholy původní vnější stěny, které rozmístíme na kružnici a podle konkrétního grafu je pak možné nastavit váhy jednotlivých vrcholů. Obecně se pro dostatečně malé grafy (do 40-ti vrcholů) osvědčila lineární závislost váhy na pořadí přidání vrcholu do grafu, nejdříve přidaný je nejtěžší. Skript, ve kterém lze Tuttovo nakreslení spočítat je jedním z výstupních formátů přiložené aplikace, více v Sekci 3.2.



Obrázek 4.4: Tuttovo nakreslení.

5. Výsledky

Díky programu popsanému v předchozí kapitole bylo možné zkusit dokončit důkaz Věty 4 pro některé neutrální posloupnosti q (na volbě posloupnosti p nezáleží, protože konstrukce pro jádrový triark nezávisí na velikosti jádra).

Vzhledem k výpočetním omezením programu jsme zkusili dokončit důkaz věty pouze pro takové neutrální posloupnosti q , že $Q = \{r, s\}$ a navíc $r < s < 18$. Seznam posloupností, pro které program nalezl potřebné grafy je v Tabulce 5.1. Pokud bychom se omezili na rozsah $r < s < 14$, pak program grafy nalezl právě tehdy, když r a s jsou nesoudělná čísla.

$r \setminus s$	7	8	9	10	11	12	13	14	15	16	17
3	•	•		•	•		•	•		•	•
4	•		•		•		•		•		
5	•	•	•		•	•	•				

Tabulka 5.1: Úplný výčet dvojic stran, pro které se podařilo dokončit důkaz Věty 4.

Formulujme výsledek do věty.

Věta 8. *Pro každou neutrální posloupnost q , která má nenulové hodnoty pouze pro q_r a q_s , kde r, s je vyznačená dvojice z Tabulky 5.1, platí následující: mějme přípustnou posloupnost $p = (p_k | 3 \leq k \neq 6)$, pak existuje nekonečně takových přirozených n , že $p + nq$ je realizovatelná.*

Všechny potřebné grafy pro doložení důkazu jsou v Příloze A.2.

$s \setminus r$	3	4	5
7	(i): 3 (ii): 3 AB : 101010 CD : 101010	(i): 3 (ii): 3 AB : 101010 CD : 101010	(i): 3 (ii): 3 AB : 101010 CD : 1010
8	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3,5,6,9 (ii): AB : CD :	(i): 3 (ii): 3 AB : 1010 CD : 1010
9	(i): 3,4,8 (ii): AB : CD :	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3 (ii): 3 AB : 1010 CD : 1010
10	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3,4,5,6,7,8,9,10 (ii): AB : CD : 1010	(i): 4,5,9,10 (ii): AB : CD :
11	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3 (ii): 3 AB : 1010 CD : 1010
12	(i): 3,4,5,6,7,8,9,10 (ii): AB : CD :	(i): 5,6 (ii): AB : CD :	(i): 3 (ii): 3 AB : 1010 CD : 1010
13	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3 (ii): 3 AB : 1010 CD : 1010
14	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3,4,5,6,7,8,9,10 (ii): AB : CD : 1010	(i): (ii): AB : CD :
15	(i): 3,4,7,8 (ii): AB : CD :	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 9,10 (ii): AB : CD :
16	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): 3,5,6,8,9 (ii): AB : CD :	(i): (ii): AB : CD :
17	(i): 3 (ii): 3 AB : 1010 CD : 1010	(i): (ii): AB : CD :	(i): (ii): AB : CD :

Tabulka 5.2: Přehled nalezených grafů. Ohraničené buňky kódují výsledek pro danou dvojici r, s . Pokud jsou grafy nalezené, je buňka šedá. První řádek buňky: k pro nalezené (k, k, k) -triarky; druhý: k pro nalezené (k, k, k) - a zároveň $(k, k, k - 1)$ -triarky; třetí: řetízek pro konstrukci grafu (iii), kde mk odpovídá poslední hodnotě v druhém řádku; čtvrtý: řetízek pro graf (iv).

Závěr

V práci se podařilo dokončit důkaz několika instancí Hypotézy 4, konkrétně pro neutrální posloupnosti, které mají jen dvě nenulové hodnoty, velikosti jejích stěn jsou nesoudělné a dostatečně malé. Výčet všech dvojic těchto hodnot je v Tabulce 5.1.

Kromě tohoto teoretického výsledku poskytujeme implementaci algoritmů z Kapitoly 3, který může sloužit při hledání 3-regulárních grafů, speciálně i s konkrétně zadánými velikostmi stěn.

Na základě získaných výsledků bychom v budoucnu rádi zjistili, jestli Hypotéza 4 platí i pro nějaké posloupnosti, jejichž množina stěn má soudělné prvky. Také bychom rádi ověřili, že pokud jsou nesoudělné, platí hypotéza vždy.

Vedle hlavního tématu práce bychom rádi získali vhodnější algoritmy pro kreslení roviných grafů do lidsky dobře čitelné podoby.

Seznam použité literatury

- [1] V. Eberhard. Zur Morphologie der Polyeder. *Leipzig: B. G. Teubner, 1891*, 1891.
- [2] J. C. Fisher. An existence theorem for simple convex polyhedra. *Discrete Math.* 7, pages 75–97, 1974.
- [3] B. Grünbaum. *Convex Polytopes*. Second Edition. Springer-Verlag, New York, 2003.
- [4] Matt DeVos, Agelos Georgakopoulos, Bojan Mohar, and Robert Šámal. An Eberhard-like theorem for pentagons and heptagons. *Discrete Comput. Geom.*, 44:931–945, 2010.
- [5] J. Matoušek and J. Nešetřil. *Kapitoly z diskrétní matematiky*. Karolinum, Praha, 2009.
- [6] E. Steinitz. Polyeder und Raumeinteilungen. *Encyclopädie der mathematischen Wissenschaften, Band 3*, pages 1–139, 1922.
- [7] The Sage Developers. *SageMath, the Sage Mathematics Software System (Version 8.1)*, 2017. <http://www.sagemath.org>.
- [8] The GraphViz Developers. *Graphviz (Version 2.38)*. <http://www.graphviz.org/>.
- [9] W. T. Tutte. How to Draw a Graph. *Proceedings of the London Mathematical Society*, s3-13:743–767, 1963.

A. Přílohy

Přiložený disk obsahuje zdrojové kódy aplikace, spustitelnou aplikaci a výsledky, které jsme získali.

A.1 Program

Přikládáme zdrojové kódy v C# (.NET Framework 4.6.1) v podobě kompletního projektu pro Visual Studio 2015. A také zkompilovaný program, který zpřístupňuje uživatelům Windows aplikaci bez nutné komplikace.

A.2 Získané výsledky

Přikládáme grafy, které dokazují pravdivost hlavního výsledku, Věty 8. Pro každou dvojici stěn r, s , která je v Tabulce 5.1 označena, je ve složce **grafy** přiložena složka s velikostmi stěn v názvu. V ní jsou shrnuté parametry nalezených grafů a podadresáře s jednotlivými grafy, ve všech formátech popsaných v Sekci 3.2.