Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

# Farbenia grafov s obmedzeniami do vzdialenosti dva

Diplomová práca

2018
Bc. Jaroslav Petrucha

Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

# Farbenia grafov s obmedzeniami do vzdialenosti dva

Diplomová práca

| | |
|---|---|
| Študijný program: | Informatika |
| Študijný odbor: | Informatika |
| Školiace pracovisko: | Katedra informatiky |
| Školiteľ: | RNDr. Michal Forišek, PhD. |

Bratislava, 2018
Bc. Jaroslav Petrucha


Univerzita Komenského v Bratislave
Fakulta matematiky, fyziky a informatiky

# ZADANIE ZÁVEREČNEJ PRÁCE

**Meno a priezvisko študenta:** Bc. Jaroslav Petrucha
**Študijný program:** informatika (Jednoodborové štúdium, magisterský II. st., denná forma)
**Študijný odbor:** informatika
**Typ záverečnej práce:** diplomová
**Jazyk záverečnej práce:** slovenský
**Sekundárny jazyk:** anglický

**Názov:** Farbenia grafov s obmedzeniami do vzdialenosti dva
*Graph colorings with constraints reaching up to distance two*

**Anotácia:**

**Cieľ:** Práca by mala nadviazať na obhájené diplomové práce A. Dresslerovej a M. Anderleho a pokračovať vo výskume vybraných algoritmických problémov súvisacich s farbeniami grafov v situáciách, kedy sú predpísané minimálne rozdiely nielen na susedných vrcholoch ale taktiež vo vzdialenosti dva. Súčasťou práce by mal byť pokus využiť výsledky o špeciálnych typoch grafov dokázané v skorších prácach na zlepšenie algoritmu na L(2,1) farbenie všeobecných grafov.

**Vedúci:** RNDr. Michal Forišek, PhD.
**Katedra:** FMFI.KI - Katedra informatiky
**Vedúci katedry:** prof. RNDr. Martin Škoviera, PhD.

**Dátum zadania:** 15.12.2016

**Dátum schválenia:** 15.12.2016 prof. RNDr. Rastislav Kráľovič, PhD.
garant študijného programu

.....
študent

.....
vedúci práce

**Poďakovanie:** Ďakujem môjmu školiteľovi, RNDr. Michalovi Foriškovi, PhD., za všetku pomoc pri tvorbe tejto práce, či už išlo o cenné nápady k problémom, podporu pri výskume, alebo rady k tomu, ako úspešne a načas napísať textovú podobu práce.

Ďakujem mojej rodine, ktorá ma počas celého štúdia podporovala. Ďakujem všetkým kamarátom z Trojstenu. Ste úžasní a bez Vás by bol Matfyz chudobnejší.

V neposlednom rade ďakujem škole, jej učiteľom, docentom a profesorom, za výborných päť rokov štúdia a veľa príležitostí rásť, najmä v smeroch, ktorými by som sa sám nevydal.

# Abstrakt

$L(2,1)$-farbenie grafu je očíslovanie jeho vrcholov prirodzenými číslami tak, že čísla susedných vrcholov sa líšia aspoň o 2 a čísla vrcholov vo vzdialenosti 2 sa líšia aspoň o 1. Rozpätie $L(2,1)$ farbenia je najväčšie číslo, ktoré používa. Minimálne rozpätie $L(2,1)$-farbenia grafu $G$ označujeme $\lambda(G)$. Najlepší doterajší algoritmus na hľadanie $\lambda(G)$ na všeobecných grafoch má časovú zložitosť $O^*(2.6488^n)$.

V práci popisujeme metódy rozdelenia problému na menšie časti, pomocou ktorých vytvárame efektívnejšie algoritmy pre hľadanie $\lambda(G)$. Vytvoríme algoritmus na hľadanie rozpätia $L(2,1)$-farbenia na planárnych grafoch v čase $O^*(2.2^{n+o(n)})$ a v čase $O^*(2.613^n)$ na grafoch s malým vrcholovým separátorom. Nakoniec popisujeme postup generovania všetkých neoznačených minimálne 2-hranovo súvislých grafov. Experimentálne overujeme, že spomedzi 2-hranovo súvislých grafov majú najväčší počet kombinatorickej štruktúry, ktorá sa nazýva vlastný pár, práve kružnice.

**Kľúčové slová:** $L(2,1)$-farbenie, planárny graf, bezmostový graf, exponenciálny algoritmus

# Abstract

$L(2,1)$-colouring of a graph is an assignment of natural numbers to its vertices such that adjacent vertices differ by at least 2 and vertices sharing a common neighbour differ by at least 1. The span of an $L(2,1)$-colouring is the largest number assigned to a vertex. The minimum span of any $L(2,1)$-colouring of graph is denoted by $\lambda(G)$. The best current algorithm for determining $\lambda(G)$ for general graphs has time complexity of $O^*(2.6488^n)$.

We describe methods for splitting the problem into independent smaller parts and use them to develop faster algorithms for determining $\lambda(G)$. We showcase our method for the planar graphs, for which we create an $O^*(2.2^{n+o(n)})$ algorithm. We create an algorithm for graphs with small cuts having time complexity $O^*(2.613^n)$. Last, we develop an algorithm for generating all the minimally 2-edge-connected graphs. Using this, we experimentally verify that cycles have the most proper pairs from the class of 2-edge-connected graphs.

**Keywords:** $L(2,1)$-colouring, planar graph, edge-free graph, exponential-time algorithm

# Obsah

**Úvod** **1**

**1 Základné pojmy a predošlé výsledky** **3**

1.1 Základné pojmy . . . . . 3

1.2 Predošlé výsledky . . . . . 4

1.3 Polynomiálne farbenie špeciálnych grafov . . . . . 5

1.3.1 Chang-Kuo algoritmus . . . . . 5

1.3.2 Zlepšenia Chang-Kuo algoritmu . . . . . 6

1.3.3 Cyklové stromy a vonkajšie planárne grafy . . . . . 7

1.4 Farbenie všeobecných grafov . . . . . 7

1.4.1 Algoritmus Junosza-Szaniawski . . . . . 8

1.4.2 Analýza časovej zložitosti algoritmu Junosza-Szaniawski . . . . 10

1.4.3 Horný odhad hodnoty $pp(n)$ . . . . . 11

1.4.4 Dolné odhady hodnoty $pp(n)$ . . . . . 12

**2 Delenie problému $L(2,1)$-farbenia** **14**

2.1 Teoretické základy rozdelenia problému . . . . . 14

2.1.1 Rozdelenie na hranových separátoroch . . . . . 16

2.2 Mostová veta . . . . . 18

2.2.1 Pseudoblokovo-mostový strom . . . . . 20

2.2.2 Algoritmus na rozdelenie problému . . . . . 22

**3 Rozdelenie pri čiastočných farbeniach 25**

3.1 Rozdelenie na vrcholových separátoroch . . . . . 26

3.2 Farbenie planárnych grafov . . . . . 28

3.2.1 Separátor s veľkým okolím . . . . . 29

3.2.2 Separátor s malým okolím . . . . . 31

3.2.3 Vylepšenie a poznámky . . . . . 33

3.3 Farbenie vyvážene rozdeliteľných grafov . . . . . 33

3.3.1 Horný odhad počtu prechodov . . . . . 35

3.3.2 Časová zložitosť algoritmu . . . . . 39

**4 Experiment na 2-hranovo súvislých grafoch 40**

4.1 Konštrukcia 2-hranovo súvislých grafov . . . . . 40

4.2 Redukcia počtu grafov . . . . . 42

4.2.1 Najväčší cyklus v grafe . . . . . 43

4.2.2 Minimálne 2-hranovo súvislý graf . . . . . 44

4.2.3 Usporiadané pridávanie ciest a cyklov . . . . . 44

4.2.4 Najväčší cyklus v grafe znova . . . . . 45

4.2.5 Symetria prvého cyklu . . . . . 46

4.3 Počítanie vlastných párov . . . . . 46

4.3.1 Vlastné páry na cestách a kružniciach . . . . . 47

4.4 Výsledky experimentu . . . . . 48

**Záver** **51**

# Zoznam obrázkov

1.1 Prevesenie listového vrcholu . . . . . 11

1.2 Trieda grafov s vysokým $pp(n)$ . . . . . 13

2.1 Chlpaté 2-hranovo súvislé grafy . . . . . 19

4.1 2-hranovo súvislé grafy s najviac vlastnými pármi . . . . . 49

# Úvod

Farbenie grafov je známy a pomerne dobre preštudovaný ťažký problém na grafoch. Jedným zo zovšeobecnení farbenia grafu je $L(2,1)$-farbenie grafu, v ktorom vrcholom priradzujeme prirodzené čísla. Požadujeme pri tom, aby susedné vrcholy mali priradené čísla s rozdielom aspoň 2 a vrcholy so spoločným susedom mali priradené rôzne čísla.

Jedným z dôvodov, prečo je $L(2,1)$-farbeniam venovaná pozornosť, je ich uplatnenie pri prideľovaní frekvenčných pásiem vysielacím staniciam. Kvôli interferencii elektromagnetických vysielačov a prijímačov na podobných frekvenciách totiž musia veľmi blízke vysielače dostať veľmi rozdielne pásmo, ale viac vzdialeným vysielačom stačí menší rozdiel.

Dôležitou charakteristikou $L(2,1)$-farbenia je jeho rozpätie – najväčšie číslo, ktoré je priradené niektorému vrcholu. Väčšina problémov ohľadom $L(2,1)$-farbení sa týka práve tohto rozpätia. Jedným zo základných problémov je pre daný graf a danú hodnotu $k$ povedať, či existuje $L(2,1)$-farbenie tohto grafu s rozpätím $k$.

Problémy $L(2,1)$-farbenia formulujeme v jazyku prirodzených čísel. Ukazuje sa, že toto zjednodušenie z reálneho sveta je postačujúce. Už v prvých prácach venujúcich sa tejto problematike bola dokázaná ekvivalencia problémov definovaných na prirodzených číslach so zovšeobecnením na reálne čísla [7].

Keďže ide o problém príbuzný farbeniam grafov, je očakávateľné, že bude NP-ťažký, čo sa podarilo dokázať v tom istom článku. Taktiež sa postupom času dokázala zložitosť problému aj na niektorých obmedzených triedach grafov, napr. planárnych, bipartitných, na grafoch s priemerom 2 [10].

Pre niektoré triedy grafov boli objavené algoritmy, ktoré hľadajú minimálne rozpätie $L(2,1)$-farbenia v polynomiálnom čase. Prvým výsledkom bol polynomiálny algoritmus na stromoch [3]. Ďalšiu triedu grafov s polynomiálnym algoritmom tvoria cyklové stromy [6] – grafy, v ktorých sú všetky kružnice vrcholovo disjunktné.

Druhou oblasťou skúmania sú algoritmy, ktoré riešia problém na všeobecných grafoch. Snahou je vytvoriť algoritmus, ktorý rieši problém s najlepšou časovou zložitosťou. Keďže ide o NP-ťažký problém, skúmajú sa riešenia s exponenciálnou časovou zožitosťou. Pre zjednodušenie zápisu sa používa $O^*$-notácia, ktorá zanedbáva polynomiálne faktory v zložitosti.

Najrýchlejším algoritmom na hľadanie minimálneho rozsahu $L(2,1)$-farbenia grafu je algoritmus od Junosza-Szaniawskeho a kol., ktorý dosahuje časovú zložitosť na grafe s $n$ vrcholmi $O^*(2.6488^n)$ [12]. Časová zložitosť tohto algoritmu je úzko spätá s kombinatorickou štruktúrou zvanou vlastný pár – časová zložitosť algoritmu je zhruba lineárne závislá od počtu vlastných párov v grafe.

Na druhú stranu platí, že najviac vlastných párov majú stromy – grafy, na ktorých vieme problém riešiť v polynomiálnom čase. V tejto práci sa budeme venovať využitiu metódy rozdeľuj a panuj na konštrukciu rýchlejšieho algoritmu.

V druhej kapitole sa pozrieme na delenie problému na nezávislé časti s použitím hranového separátora. Podrobnejšie sa pozrieme na grafy s mostami a pre dostatočne veľké $k$ ukážeme, ako zredukovať problém existencie $L(2,1)$-farbenia s rozpätím $k$ v ľubovoľnom grafe na problém na triede grafov, v ktorých odstránením ľubovoľného mostu vznikne izolovaný vrchol. Z rýchleho riešenia na tejto špeciálnej triede grafov potom bude vyplývať rýchle riešenie pre všeobecné grafy.

V 3. kapitole ukážeme algoritmus, ktorý nadviaže na prácu Junosza-Szaniawekeho a kol. Pomocou delenia problému skonštruujeme algoritmus pre planárne grafy s časovou zložitosťou $O^*(2.2^{n+o(n)})$ a algoritmus pre dobre rozdeliteľné grafy s časovou zložitosťou $O^*(2.613^n)$.

Vo 4. kapitole odprezentujeme experiment na triede 2-hranovo súvislých grafov. Ukážeme, že pri grafoch do 20 vrcholov majú najväčší počet vlastných párov práve kružnice. Zaujímavým výsledkom tejto časti je aj postup generovania všetkých neoznačených 2-hranovo súvislých grafov (s opakovaním).

Implementácia algoritmov popísaných v 4. kapitole bude prístupná na stránke `https://people.ksp.sk/~sameth/dipl_experiment/` .

# Kapitola 1

# Základné pojmy a predošlé výsledky

V tejto kapitole si zavedieme pojmy a označenia, ktoré budeme používať vo zvyšku práce. Taktiež si podrobne popíšeme predošlé výsledky o $L(2,1)$-farbeniach grafov.

## 1.1 Základné pojmy

Pokiaľ nebude povedané inak, grafy spomínané v tejto práci budú neorientované, súvislé a jednoduché, bez slučiek a násobných hrán. Množinu vrcholov budeme označovať $V(G)$ a množinu hrán $E(G)$. Vzdialenosť dvoch vrcholov $u$ a $v$ v grafe $G$ budeme označovať $d_G(u,v)$. Pokiaľ bude z kontextu jasné, o ktorý graf sa jedná, budeme vzdialenosť vrcholov $u, v$ značiť $d(u,v)$. Symbolom $n$ budeme označovať počet vrcholov grafu $G$.

Pre ľubovoľný vrchol $v$ budeme výrazom $N(v)$ značiť množinu vrcholov, ktoré sú hranou spojené s $v$. Pod značením $N[v]$ budeme rozumieť množinu $N(v) \cup \{v\}$. Toto značenie rozšírime aj na množiny vrcholov. Pre množinu vrcholov $S$ budeme výrazom $N[S]$ značiť množinu $\bigcup_{v \in S} N[v]$ a výrazom $N(S)$ označíme množinu $N[S] - S$.

Ako prvé si poriadne zadefinujeme $L(2,1)$-farbenie grafu a problémy s ním súvisiace.

**Definícia 1.** *$L(2,1)$-farbenie grafu $G$ je zobrazenie $\omega : V(G) \to \mathbb{N}$, ktoré spĺňa nasledujúce podmienky:*

- $\forall u, v \in V(G) : d(u,v) = 1 \Rightarrow |\omega(u) - \omega(v)| \geq 2$,
- $\forall u, v \in V(G) : d(u,v) = 2 \Rightarrow \omega(u) \neq \omega(v)$.

Pri obyčajných vrcholových, či hranových farbeniach grafov skúmame, aký najmenší počet farieb potrebujeme pre platné ofarbenie daného grafu. Podobne si vieme pri $L(2,1)$-farbeniach všímať, aké najväčšie číslo používajú. Toto číslo budeme nazývať *rozpätím* $L(2,1)$*-farbenia*. Najmenšie možné rozpätie $L(2,1)$-farbenia grafu $G$ budeme nazývať $L(2,1)$*-farbiace číslo grafu* $G$ a označovať $\lambda(G)$.

Kvôli prehľadnosti budeme miestami používať pojem $k$-$L(2,1)$-farbenie, pod ktorý budeme mať na mysli $L(2,1)$-farbenie s rozsahom $k$. Graf $G$ budeme volať $k$-zafarbiteľný, ak $\lambda(G) \leq k$. Všimnime si, že $L(2,1)$-farbenie s rozpätím $k$ môže používať až $k+1$ rôznych hodnôt.

**Definícia 2.** *Pod problémom* $L(2,1)$*-farbenia budeme rozumieť nasledujúci rozhodovací problém. Inštanciou je graf* $G$ *a prirodzené číslo* $k$*. Inštancia je riešiteľná práve vtedy, ak existuje* $L(2,1)$*-farbenie grafu* $G$ *s rozsahom* $k$*.*

Podobne by sme mohli zadefinovať problém rozpätia $L(2,1)$-farbenia, v ktorom by sme pre daný graf hľadali jeho farbiace číslo. Pre každý graf existuje triviálne ofarbenie používajúce párne čísla od 0 po $2n-2$, z hľadiska výpočtovej zložitosti sú teda tieto problémy takmer totožné.

V práci budeme často používať pojem *čiastočného* $L(2,1)$*-farbenia*, preto si ho poriadne zadefinujeme.

**Definícia 3.** *Nech* $G$ *je graf, nech* $S$ *je podmnožina vrcholov grafu* $G$*. Pod pojmom čiastočné* $L(2,1)$*-farbenie množiny* $S$ *v grafe* $G$ *budeme rozumieť zobrazenie* $\omega : S \to \mathbb{N}$*, ktoré spĺňa nasledujúce podmienky:*

- $\forall u, v \in S : d_G(u,v) = 1 \Rightarrow |\omega(u) - \omega(v)| \geq 2$,
- $\forall u, v \in S : d_G(u,v) = 2 \Rightarrow \omega(u) \neq \omega(v)$.

*Pojmom čiastočné* $L(2,1)$*-farbenie grafu* $G$ *budeme označovať čiastočné farbenie niektorej podmnožiny jeho vrcholov.*

## 1.2 Predošlé výsledky

Veľkou oblasťou skúmania $L(2,1)$-farbení boli dolné a horné ohraničenia na hodnotu $\lambda(G)$. Vhodnou charakteristikou sa ukázal byť maximálny stupeň grafu. Pre všeobecné grafy bolo najprv dokázané horné ohraničenie $\lambda(G) \leq \Delta^2 + 2\Delta$ pomocou jednoduchého

pažravého priradenia [7], ktoré bolo neskôr vylepšené na $\lambda(G) \leq \Delta^2 + \Delta$ s použitím šikovnejšieho priradzovania [3].

Pre špeciálne triedy grafov sa podarilo dokázať silnejšie ohraničenia. Napríklad pre grafy s priemerom 2 platí a je tesný odhad $\lambda(G) \leq \Delta^2$ [7]. Ďalej pre chordálne grafy platí odhad $\lambda(G) \leq \frac{(\Delta+3)^2}{4}$ [7].

Medzi ďalšie triedy grafov, na ktorých sú dokázané tesnejšie obmedzenia na $L(2,1)$-farbiace číslo, patria aj stromy s obmedzením $\Delta + 1 \leq \lambda(G) \leq \Delta + 2$ [7], kaktusy s obmedzením $\Delta + 1 \leq \lambda(G) \leq \Delta + 3$ [6] a vonkajšie planárne grafy s obmedzením $\Delta + 1 \leq \lambda(G) \leq \Delta + 8$ [2].

Griggs a Yeh taktiež vyslovili hypotézu, že v každom grafe platí $\lambda(G) \leq \Delta^2$, ktorá dodnes nebola vyriešená.

Ďalej sa pozrieme na zopár tried grafov, na ktorých vieme riešiť problém $L(2,1)$-farbenia v polynomiálnom čase a popíšeme si zodpovedajúce algoritmy.

## 1.3 Polynomiálne farbenie špeciálnych grafov

Prvou triedou grafov, pre ktoré bol objavený polynomiálny algoritmus na riešenie problému $L(2,1)$-farbenia, sú stromy [3]. Tento algoritmus si podrobne popíšeme.

### 1.3.1 Chang-Kuo algoritmus

Vstupom pre algoritmus je strom $T$ zakorenený v listovom vrchole so synom $r$ a číslo $k$. Tento algoritmus zistí, či existuje $L(2,1)$-farbenie grafu $T$ s rozpätím $k$.

Pre ľubovoľný vrchol $v$ zadefinujeme $T(v)$ ako podstrom stromu $T$ zakorenený vo vrchole $v$. Taktiež zadefinujeme $T'(v')$ ako $T(v)$ s novým vrcholom $v'$, ktorý je spojený iba s $v$, čiže

$$G\left(V(T(v)) \cup \{v'\}, E(T(v)) \cup \{(v', v)\}\right)$$

kde $v'$ je nový vrchol nevyskytujúci sa v $T(v)$.

Základnou myšlienkou algoritmu je konštruovať množiny $S(T(v))$ definované nasledovne:

$S(T(v)) = \{(a,b)|$ existuje $L(2,1)$ farbenie $f$ grafu $T'(v')$ také, že $f(v) = a$ a $f(v') = b\}$

Pre ľubovoľný listový vrchol $v_l$ ľahko vidíme, že

$$S(T(v_l)) = \{(a,b)|a \leq k \wedge b \leq k \wedge |a-b| \geq 2\}$$

Pre vnútorný vrchol $v$ so synmi $v_1, v_2, \ldots v_q$ vieme pre ľubovoľné $a, b, |a-b| \geq 2$ overiť, či $(a,b) \in S(T(v))$ nasledovne:

1. Zostrojíme bipartitný graf $G_{a,b}(v)$, ktorého jednou partíciou budú vrcholy $v_1 \ldots v_q$ a druhou partíciou hodnoty množiny $\{0, 1, \ldots k\}$ a ktorého množinu hrán zostrojíme nasledovne:

$$E(G_{a,b}) = \{(v_i, x)|x \neq b \wedge (a,x) \in S(T(v_i))\}$$

2. Nájdeme najväčšie párenie na grafe $G_{a,b}$.

3. Ak nájdené párenie má veľkosť $q$, potom $(a,b) \in S(T(v))$, inak $(a,b) \notin S(T(v))$.

Do pozornosti dávame skutočnosť, že nájdené párenie zodpovedá priradeniu hodnôt vrcholom $v_1, \ldots v_q$, pri ktorom je platné ohodnotenie vrcholov $v$ a $v'$ hodnotami $a$ a $b$.

Nakoniec $L(2,1)$-farbenie stromu $T$ s rozpätím $k$ existuje práve vtedy, keď je množina $S(T(r))$ neprázdna.

Časová zložitosť tohto algoritmu je $O(n \cdot k^{4.5})$ [3]. Neskôr sa objavili mierne upravené algoritmy s lepšou časovou zložitosťou.

### 1.3.2 Zlepšenia Chang-Kuo algoritmu

Prvý zlepšujúci algoritmus urýchľuje konštrukciu párení tak, že najprv nájde základné párenie bez obmedzení spôsobených priradením hodnoty $b$ vrcholu $v'$.

Pre overenie každej dvojice $(a,b)$ vo vrchole $v$ teda stačí nájsť jedno základné párenie pre každé $a$. Priradenie farby $b$ vrcholu $v'$ môže odstrániť nanajvýš jednu hranu z párenia, tú, ktorá spája nejaký vrchol s hodnotou $b$.

Vďaka tomu vieme overiť existenciu párenia v Chang-Kuo algoritme hľadaním nanajvýš jednej zlepšujúcej cesty. Ďalšie zlepšenie vyplýva z využitia predspracovania stromu popísaného už v článku s pôvodným algoritmom. Vďaka nemu sa dá vstupný strom zredukovať tak, aby každý listový vrchol bol hranou spojený s vrcholom maximálneho stupňa v grafe [3].

Dôslednou analýzou zložitosti dostaneme celkovú časovú zložitosť vylepšeného algoritmu $O(n^{1.75})$ [8].

Pri poslednom zlepšení algoritmu riešiaceho problém $L(2,1)$-farbenia autori zavádzajú pojem kompatibility ohodnotenia vrcholov, vďaka ktorému dokážu viaceré farby vrcholu $v$ spracovať naraz. Časová zložitosť algoritmu bude lineárna od veľkosti vstupného stromu [9].

### 1.3.3 Cyklové stromy a vonkajšie planárne grafy

Ďalšími triedami grafov s polynomiálnymi algoritmami sú cyklové stromy [6] a vonkajšie planárne grafy [13].

**Definícia 4.** *Cyklový strom je graf, v ktorom každý vrchol leží na najviac jednej kružnici.*

**Definícia 5.** *Vonkajší planárny graf je planárny graf, pre ktorý existuje rovinné nakreslenie, v ktorom všetky vrcholy ležia vo vonkajšej oblasti.*

Oba algoritmy fungujú principiálne podobne ako algoritmus pre stromy v tom, že lokálne pre nejakú množinu vrcholov zisťujú, aké všetky kombinácie hodnôt im môžu byť priradené na základe rovnakej informácie pre blízke okolité body, ktorých v týchto triedach grafov nie je príliš veľa.

## 1.4 Farbenie všeobecných grafov

Paralelne k výskumu špeciálnych tried grafov, na ktorých sa dá riešiť problém $L(2,1)$-farbenia v polynomiálnom čase, sa skúmajú algoritmy schopné riešiť tento problém na všeobecných grafoch. Keďže ide o problém, ktorého exponenciálne riešenia majú pomerne veľký základ, pre časovú zložitosť budeme používať $O^*$-notáciu, pri ktorej sa neuvádzajú ani násobné polynomiálne faktory.

S prvým významným riešením tohto problému prišiel D. Kráľ, ktorý riešil všeobecnejší problém priradzovania kanálov (channel assignment problem). Pre problém $L(2,1)$-farbenia z neho jednoduchou úpravou vstupu vznikol algoritmus s časovou zložitosťou $O^*(4^n)$ [14].

S lepšími algoritmami prišli Havet a kol. Prezentujú algoritmus pre problém 4-$L(2,1)$-farbenia s časovou zložitosťou $O^*(1.3009^n)$ a algoritmus pre riešenie problému

$L(2,1)$-farbenia s časovou zložitosťou $O^*(3.8730^n)$ [11].

Algoritmy na riešenie všeobecného problému sa postupom času vylepšovali spolu s približovaním horných a dolných odhadov ich časových zložitostí. Aktuálne najlepším algoritmom pre všeobecné grafy je algoritmus od Junosza-Szaniawskeho a kol., ktorého časová zložitosť je $O^*(2.6488^n)$ [12]. Tento algoritmus budeme ďalej nazývať ako algoritmus Junosza-Szaniawski.

### 1.4.1 Algoritmus Junosza-Szaniawski

Na začiatok si zadefinujeme dôležité pojmy 2-*pakovania* a vlastných párov, ktoré budeme viackrát potrebovať pri popise algoritmu.

**Definícia 6.** *Podmnožinu $S$ vrcholovej množiny grafu $G$ nazývame* 2-pakovaním *grafu $G$, pokiaľ každé dva vrcholy v $S$ majú vzdialenosť aspoň 3 v grafe $G$.*

**Definícia 7.** *Usporiadanú dvojicu $(S, X)$ podmnožín vrcholovej množiny grafu $G$ nazývame* vlastný pár *grafu $G$, ak $S \cap X = \emptyset$ a $S$ je 2-pakovaním grafu $G$.*

*Počet všetkých vlastných párov grafu $G$ budeme označovať $pp(G)$. Najväčšiu hodnotu $pp(G)$ spomedzi všetkých $k$-vrcholových* súvislých *grafov budeme označovať $pp(k)$.*

**Poznámka 8.** *Počet vlastných párov grafu $G$ vieme vypočítať nasledovne:*

$$pp(G) = \sum_{\substack{S \in \mathcal{P}(V_G) \\ S \text{ je 2-pakovanie}}} 2^{n-|S|}$$

Základnou myšlienkou algoritmu je vytvárať všetky čiastočné $L(2,1)$-farbenia s obmedzeným rozpätím. Tieto čiastočné farbenia rozdelíme do tabuliek $T_0, T_1, \ldots T_{2n-2}$ a budeme kódovať pomocou $n$-znakových reťazcov nad abecedou $\{0, \bar{0}, 1, \bar{1}\}$. Tabuľka $T_k$ bude obsahovať reťazec $a$ práve vtedy, keď existuje čiastočné $L(2,1)$-farbenie $f$ s rozpätím $k$ a s nasledujúcimi vlastnosťami:

1. $a_i = 0$ ak je vrchol $v_i$ neohodnotený farbením $f$ a *neexistuje* vrchol susediaci s $v_i$ ofarbený farbou $k$,
2. $a_i = \bar{0}$ ak je vrchol $v_i$ neohodnotený farbením $f$ a *existuje* vrchol susediaci s $v_i$ ofarbený farbou $k$,
3. $a_i = 1$ ak $f(i) < k$,
4. $a_i = \bar{1}$ ak $f(i) = k$.

Tabuľku $T_i$ vieme vypočítať z tabuľky $T_{i-1}$ tak, že vezmeme každý reťazec z $T_{i-1}$, každé 2-pakovanie grafu $P$ a vrcholom obsiahnutým v $P$ priradíme farbu $i$. Takéto priradenie farieb vieme pekne reprezentovať ako operáciu po jednotlivých znakoch na reťazcoch.

Najprv si zadefinujeme čiastočné zobrazenie $\oplus : \{0, \bar{0}, 1, \bar{1}\} \times \{0, 1\} \to \{0, 1, \bar{1}\}$ nasledujúcou tabuľkou:

| $\oplus$ | 0 | $\bar{0}$ | 1 | $\bar{1}$ |
|---|---|---|---|---|
| 0 | 0 | 0 | 1 | 1 |
| 1 | $\bar{1}$ | – | – | – |

Znakom – označujeme nedefinovanú hodnotu.

Ďalej zobrazenie prirodzene rozšírime na reťazce a množiny reťazcov:

$$a_1a_2\ldots a_n \oplus b_1b_2\ldots b_n = \begin{cases} (a_1 \oplus b_1)(a_2 \oplus b_2)\ldots & \text{ak je } a_i \oplus b_i \text{ definované} \\ \ldots(a_n \oplus b_n) & \text{pre každé } i \in \{0,\ldots n\} \\ \text{nedefinované} & \text{inak} \end{cases} \tag{1.1}$$

$$A \oplus B = \{a \oplus b | a \in A \wedge b \in B \wedge a \oplus b \text{ je definované}\}$$

Pri počítaní $T_i$ teda najprv vypočítame $T_{i-1} \oplus P$, kde $P$ je množina všetkých 2-pakovaní, kde každé 2-pakovanie prirodzene reprezentujeme ako Booleovský reťazec dĺžky $n$. Nakoniec zmeníme 0 na $\bar{0}$ pre všetky vrcholy, ktoré susedia s nejakým vrcholom farby $i$.

Posledným stavebným kameňom algoritmu je spôsob, ako rýchlo počítať $A \oplus B$. Principiálne sa bude tento postup podobať na Strassenov algoritmus. Graf $G$ si rozdelíme na niekoľko podgrafov $G_1, G_2, \ldots G_q$ tak, aby s vopred zvoleným $k$ platilo:

1. $\forall v \in V(G)\ \exists j : v \in V(G_j)$
2. $\forall j \in \{1, 2, \ldots q-1\} : |V(G_j)| \geq k$
3. $\forall j \in \{1, 2, \ldots q\} : |V(G_j)| \leq 2k$
4. $\sum_{j=1}^{q} |V(G_j)| \leq n(1 + \frac{1}{k})$

Pre množinu reťazcov $A$ a reťazec $w$ zadefinujeme množinu suffixov $w$ v $A$ ako $A_w = \{v \mid wv \in A\}$. S takýmto označením vieme potom výpočet $A \oplus B$ rozdeliť na dve

časti nasledovne:

$$A \oplus B = \bigcup_{\substack{u\in\{0,\bar{0},1,\bar{1}\}^p \\ v\in\{0,1\}^p \\ [u\oplus v \text{ je definované}]}} (u \oplus v)(A_u \oplus B_v) = \bigcup_{\substack{v\in\{0,1\}^p \\ w\in\{0,1,\bar{1}\}^p}} w \left[ \left( \bigcup_{\substack{u\in\{0,\bar{0},1,\bar{1}\}^p \\ [u\oplus v=w]}} A_u \right) \oplus B_v \right]$$

Z definície zobrazenia $\oplus$ na reťazcoch vidíme, že ak je pre dané $v$ a $w$ prázdna množina

$$\bigcup_{\substack{u\in\{0,\bar{0},1,\bar{1}\}^p \\ [u\oplus v=w]}} A_u,$$

výpočet sa pre túto dvojicu môže ukončiť.

Z predpisu zobrazenia $\oplus$ vidíme, že $v_i = 1$ práve vtedy, keď $w_i = \bar{1}$. Pre dané $v$ preto môže existovať nanajvýš $2^{p-\|v\|}$, kde $\|v\|$ označuje počet jednotkových znakov vo $v$. Počet dvojíc $v$ a $w$, pre ktoré existuje nejaké platné $u$ teda vieme zhora ohraničiť hodnotou

$$\sum_{v\in\{0,1\}} 2^{p-\|v\|}.$$

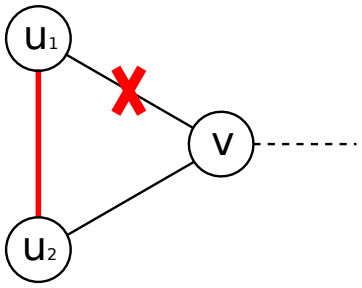
Keďže v algoritme vykonávame operáciu $T_i \oplus P$, kde $P$ sú reťazce zodpovedajúce 2-pakovaniam grafu $G$ a grafy $G_1, G_2, \ldots G_q$, podľa ktorých budeme operáciu $T_i \oplus P$ deliť na menšie časti, sú súvislé, vieme túto sumu zhora ohraničiť na $pp(p)$ 8.

Keďže súčet počtov vrcholov v grafoch $G_1 \ldots G_q$ je $n' \leq n(1 + \frac{1}{k})$, kde $n' \geq n$, v reťazcoch z výpočtu $T_i \oplus P$ budeme mať pre niektoré vrcholy viacero nezávislých pozícií kódujúcich ich stav ofarbenia. Všetky reťazce, v ktorých bude niektorý vrchol zároveň ofarbený aj neofarbený, z množiny odstránime. Keďže reťazce sú polynomiálne dlhé v závislosti od $n$, túto kontrolu vieme pre každý reťazec vykonať v polynomiálnom čase.

### 1.4.2 Analýza časovej zložitosti algoritmu Junosza-Szaniawski

Pre odhad časovej zložitosti najprv zhora odhadneme veľkosť tabuľky $T_i$. Vrcholy s hodnotou $\bar{1}$ musia tvoriť 2-pakovanie grafu $G$. Pre akýkoľvek výber vrcholov s hodnotou $\bar{1}$ majú zvyšné vrcholy iba dve možnosti, 0 alebo 1. Všetkých reťazcov dĺžky $n$ teda môže byť nanajvýš $pp(n)$. Podobný odhad vieme spraviť aj pre počet reťazcov dĺžky $n''$, ktoré sa vyskytnú v nejakom rekurzívnom volaní pri počítaní operácie $\oplus$.

V každom kroku rekurzívneho výpočtu operácie $\oplus$ na množinách sa pripraví nanajvýš $pp(k')$ možných prefixov výsledných reťazcov. Pre každý z nich sa spustí rekurzívny




Obr. 1.1: Znázornenie operácie prevesenie listového vrchola na iný list, s ktorým majú spoločného suseda. Táto operácia nezmenšuje počet vlastných párov stromu.

výpočet na dvojici množín reťazcov, kde dĺžka reťazcov je $n' - k'$. Časová zložitosť výpočtu teda zodpovedá nasledujúcej rekurencii:

$$t(n') = O(n \cdot pp(n') + pp(k')t(n' - k')); k \leq k' \leq 2k$$

Riešením tejto rekurencie sa ukáže byť funkcia $O(nn' \cdot pp(n')$. Pripomíname, že hodnota $n'$ je zhora ohraničená ako $n(1 + \frac{1}{k})$. Voľbou dostatočne veľkej konštanty $k$ dostaneme časovú zložitosť dostatočne blízku $O^*(pp(n))$. Pre dokončenie odhadu časovej zložitosti nakoniec potrebujeme nejaké ohraničenie pre počet vlastných párov v $n$-vrcholovom grafe.

### 1.4.3 Horný odhad hodnoty $pp(n)$

Pre odhad hodnoty $pp(n)$ najprv zmenšíme množinu uvažovaných grafov nasledujúcimi dvoma pomocnými pozorovaniami.

**Lema 9.** *Nech $e$ je hrana súvislého grafu $G$ taká, že graf $G - e$ je súvislý. Potom $pp(G) \leq pp(G - e)$.*

*Dôkaz.* Nech $S$ je 2-pakovaním grafu $G$. Odstránením hrany $e$ z grafu $G$ nezmenšíme vzdialenosť žiadnej dvojice vrcholov v $G$, čiže nezmenšíme vzdialenosť žiadnej dvojice vrcholov v $S$. Množina $S$ je teda zároveň aj 2-pakovaním grafu $G - e$. Z vyjadrenia $pp(g)$ pomocou 2-pakovaní 8 vyplýva dokazovaná nerovnosť. □

Keďže najmenším súvislým grafom je strom, ďalej sa pri hornom odhade hodnoty $pp(k)$ obmedzíme práve na stromy.

**Lema 10.** *Nech $T$ je strom, nech $u_1$ a $u_2$ sú listové vrcholy so spoločným susedom $v$. Potom $pp(T) \leq pp(T \cup (u_1, u_2) - (u_1, v))$. Táto grafová operácia je znázornená na obr. 1.1.*

*Dôkaz.* Podobne ako v predošlej leme ukážeme, že každé 2-pakovanie $S$ stromu $T$ je zároveň 2-pakovaním v $T \cup (u_1, u_2) - (u_1, v)$. Nahliadneme, že zatiaľčo $d(u_1, u_2)$ klesne, vzdialenosť $u_1$ od akéhokoľvek iného vrcholu takouto výmenou neklesne. Zároveň sa nezmení vzdialenosť žiadnej inej dvojice vrcholov. Keďže $S$ nemôže obsahovať zároveň vrcholy $u_1$ aj $u_2$, vzdialenosť akejkoľvek dvojice vrcholov v $S$ touto operáciou neklesne, preto každá dvojica vrcholov v $S$ stále bude mať vzdialenosť aspoň 3. □

Vďaka tejto leme sa pri dokazovaní horného ohraničenia počtu vlastných párov môžeme obmedziť na stromy, ktorých žiadne dva listy nemajú spoločného suseda, okrem triviálneho prípadu stromu s tromi vrcholmi.

Nech $T$ je strom s aspoň štyrmi vrcholmi, ktorého žiadne listy nemajú spoločný vrchol. Pozrieme sa na najdlhšiu cestu $P$ v $T$. Koncový vrchol cesty označíme $v$, jeho suseda označíme $u$ a suseda $u$ iného, ako $v$ označíme $c$. Keďže žiaden vrchol nemá za susedov dva listy, $c$ je jednoznačne určený. Ďalej sa v dôkaze horného ohraničenia rozoberá niekoľko prípadov podľa okolia vrchola $c$. Spomedzi všetkých si ukážeme jeden, ostatné sa dajú vyriešiť podobným spôsobom.

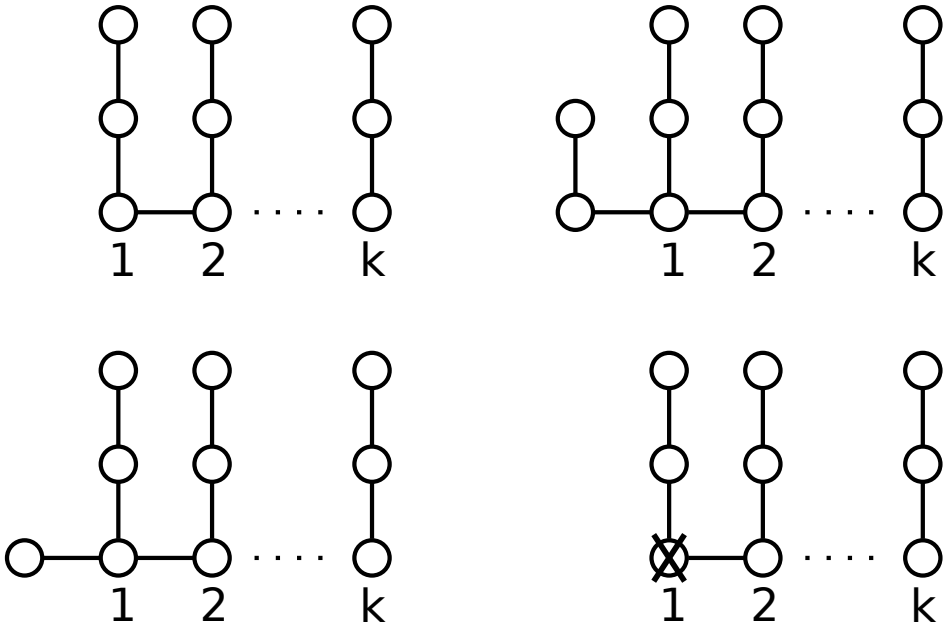
Ak $\delta(c) = 2$, všetky vlastné páry $(S, X)$ rozdelíme na tie, v ktorých $v$ patrí do $S$ a tie, kde $v$ nepatrí do $S$. Vlastných párov, kde $v$ nepatrí do $S$ je dvakrát toľko ako v grafe $T - \{v\}$, podľa toho, či $v$ patrí do $X$. Tento počet vieme zhora odhadnúť na $pp(n-1)$. Ak $v$ patrí do $S$, tak $u$ ani $c$ nemôžu patriť do $S$. Vlastných párov v tomto prípade bude štornásobok počtu vlastných párov v grafe $T - \{v, u, c\}$, čo vieme zhora odhadnúť ako $pp(n-3)$.

Dostávame teda nasledujúcu nerovnosť: $pp(n) \leq 2pp(n-1)+4pp(n-3)$. Vyriešením nerovností pre všetky prípady dostaneme horné ohraničenie $pp(n) \leq 2\tau^n$[12], kde $\tau \approx 2.6488$ je kladný koreň rovnice

$$\tau^5 = 16\tau + 88.$$

### 1.4.4 Dolné odhady hodnoty $pp(n)$

Prvé dolné odhady hodnoty $pp(n)$ využívali grafy znázornené na obr. 1.2. Pomocou rekurzívneho vyjadrenia počtu vlastných párov na týchto špeciálnych grafoch vzniklo dolné ohraničenie $pp(n) = \Omega(2.6117^n)$.




Obr. 1.2: Trieda grafov s $pp(n) = \Theta(2.6117^n)$

Analýzou zakorenených stromov s priemerom 6 sa dá dolný odhad zvýšiť na $pp(n) = \Omega(2.6313^n)$[1]. Dôkladnejšou analýzou zakorenených stromov, ktorých všetky podstromy sú rovnaké, sa dá odhad vylepšiť až na $pp(n) = \Omega(2.6391^n)$ [1].

# Kapitola 2

# Delenie problému $L(2,1)$-farbenia

Časová zložitosť algoritmu Junosza-Szaniawski je principiálne veľká na grafoch, v ktorých existuje veľa rôznych čiastočných $L(2,1)$ ofarbení. Ako sme videli v predošlej kapitole, spomedzi súvislých grafov majú najviac čiastočných ofarbení, resp. vlastných párov, stromy. Na dostatočne jednoduchých triedach grafov však existujú polynomiálne algoritmy, ktoré riešia problém $L(2,1)$-farbenia.

Intuitívne platí, že pri jednoduchých grafoch vieme problém rozdeliť na viacero menších, menšie problémy vyriešiť nezávisle a nakoniec ich riešenia pospájať. V tejto kapitole sa pozrieme na spôsob, akým vieme túto intuíciu aplikovať. Nakoniec ukážeme, ako redukovať problém $L(2,1)$-farbenia na triedu chlpatých 2-hranovo súvislých grafov, okrem prípadov, keď sa pýtame na príliš malý rozsah farbenia.

Aj keď v práci rozoberáme problém $L(2,1)$-farbenia, postupy z tejto kapitoly sa principiálne dajú aplikovať aj na všeobecnejšie $L(p,q)$-farbenia.

## 2.1 Teoretické základy rozdelenia problému

Ako sa ďalej v práci ukáže, aby sme vedeli problém dobre rozdeliť, potrebujeme riešiť o trochu všeobecnejší problém: Niektoré vrcholy môžu mať predpísanú svoju farbu.

**Definícia 11.** *Rozhodovací problém* doplnenia $L(2,1)$-farbenia *definujeme nasledovne: Inštanciou je graf $G$, prirodzené číslo $k$ a čiastočné $L(2,1)$-farbenie $f$ množiny $S \subseteq V(G)$ v grafe $G$, kde rozsah $f$ je nanajvýš $k$.*

*Inštancia je riešiteľná práve vtedy, keď existuje $L(2,1)$ farbenie $f'$ grafu $G$ s rozsahom $k$, pre ktoré platí: $\forall v \in S : f'(v) = f(v)$.*

Ľubovoľné riešenie problému doplnenia $L(2,1)$-farbenia vieme prerobiť na riešenie problému $L(2,1)$-farbenia, ak za čiastočné farbenie $f$ vezmeme prázdne farbenie. Mnohé riešenia problému $L(2,1)$-farbenia sa dajú prerobiť na riešenia problému doplnenia $L(2,1)$-farbenia bez výrazného zhoršenia časovej zložitosti. Ukážeme si to na príklade algoritmu Junosza-Szaniawski.

Najprv si pripomenieme, ako funguje algoritmus Junosza-Szaniawski. Tento algoritmus postupne počíta tabuľky $T_0, T_1, \ldots T_{2n}$. Tabuľka $T_i$ obsahuje reprezentáciu všetkých čiastočných $L(2,1)$-farbení grafu s rozsahom $i$. Pre každé čiastočné farbenie v tabuľke $T_i$ vieme povedať zoznam vrcholov, ktoré používajú farbu $i$.

Úprava algoritmu teda bude spočívať v tom, že po vypočítaní tabuľky $T_i$ z nej odstránime všetky čiastočné farbenia, ktoré nejakému vrcholu $v \in S$ priradili inú hodnotu, ako $f(v)$.

Pozor si však treba dať na počet tabuliek, ktoré potrebujeme vypočítať. Keďže niektoré vrcholy majú predpísanú farbu, nemusí existovať triviálne farbenie s rozsahom $2n$. Ďalej však dokážeme, že všetky vrcholy mimo množiny $S$ vieme triviálne ofarbiť farbami veľkosti nanajvýš $3n$. Keďže časová zložitosť počítania tabuľku $T_i$ z tabuľky $T_{i-1}$ je $O^*(2.6488^n)$ a tento výpočet spravíme najviac $(3n)$-krát, časová zložitosť tohto algoritmu bude $O^*(2.6488^n)$.

**Lema 12.** *Nech $f$ je čiastočné $L(2,1)$-farbenie množiny $S$ v grafe $G$. Potom existuje $L(2,1)$-farbenie $f'$ grafu $G$, ktoré dopĺňa $f$ a pre ktoré platí $\forall v \in V(G) - S : f(v) < 3n$.*

*Dôkaz.* Základná myšlienka dôkazu je nasledovná. Vezmeme množinu farieb, ktoré sú priradené vrcholom v $S$ a označíme ju $F$. Ak nájdeme množinu farieb $F'$, ktorá má veľkosť (aspoň) $|V(G) - S|$, každá dvojica farieb v nej má rozdiel aspoň 2 a každá dvojica farieb $(c_1, c_2) \in F \times F'$ má rozdiel aspoň 2, tak neofarbeným vrcholom môžeme ľubovoľne priradiť rôzne farby z $F'$ a dostneme $L(2,1)$-farbenie.

V každej trojici čísel $(3x, 3x+1, 3x+2)$, kde $0 \leq x < n$, buď existuje vrchol v množine $S$, ktorý má priradenú jednu z týchto farieb, alebo sa farba $3x+1$ môže nachádzať v množine $F'$. Ak má množina $S$ presne $m$ vrcholov, nájdeme takto $n - m$ farieb, ktoré môžeme priradiť vrcholom vo $V(G) - S$. □

**Dôsledok 13.** *Nech $(G, k, f)$ je inštancia problému doplnenia $L(2,1)$-farbenia. Ak $k \geq 3n$, tak je inštancia riešiteľná.*

### 2.1.1 Rozdelenie na hranových separátoroch

Prvý typ "jednoduchosti" grafu, ktorú vieme využiť pri hľadaní rýchlejšieho algoritmu, zodpovedá hranovej súvislosti grafu. Vezmime si nejaký hranový separátor $S_E$ grafu $G$. Všetkým vrcholom, ktoré sú incidentné s niektorou hranou separátora, zafixujeme nejaké farby. Ďalej si vezmime komponenty súvislosti grafu $G - S_E$.

Ak v každom komponente súvislosti nájdeme (nezávisle) nejaké $L(2,1)$-farbenie konzistentné s ohodnotením separátora, spojením farbení pre každý podgraf dostaneme $L(2,1)$-farbenie celého grafu $G$. Toto tvrdenie si formálne zhrnieme v nasledujúcej leme.

**Lema 14.** *Nech $S_E \subseteq E(G)$ je hranový separátor grafu $G$, nech $S_V \subseteq V(G)$ je množina všetkých vrcholov, ktoré sú incidentné s niektorou hranou v $S_E$, nech $G_1, G_2, \cdots G_k$ sú komponenty súvislosti grafu $G - S_E$.*

*Nech $f_S : S_V \to \mathbb{N}$ je čiastočné $L(2,1)$-farbenie množiny $S_V$ v grafe $G$, nech $f_1, f_2, \ldots, f_k$ sú čiastočné $L(2,1)$-farbenia množín $V(G_1) \cup S_V$, $V(G_2) \cup S_V$, $\ldots, V(G_k) \cup S_V$ v grafe $G$ a nech platí*

$$\forall i \in 1 \ldots k,\ \forall v \in S_V : f_i(v) = f_S(v).$$

*Potom zobrazenie $\omega : V(G) \to \mathbb{N}$ definované nasledovne:*

$$\omega(v) = \begin{cases} f_S(v), & v \in S_V \\ f_i(v), & v \in V(G_i) - S_V \end{cases}$$

*tvorí $L(2,1)$-farbenie grafu $G$.*

*Dôkaz.* Zobrazenie $\omega$ je dobre definované, lebo každý vrchol patrí do práve jedného komponentu grafu $G - S_E$, zároveň každému vrcholu v $G$ priradí hodnotu. Potrebujeme teda overiť podmienky $L(2,1)$-farbenia pre každú dvojicu vrcholov $u, v \in V(G)$. Ďalej rozoberieme niekoľko prípadov podľa príslušnosti $u, v$:

**Oba vrcholy patria do množiny $S_V$:** Z definície $\omega$ platí $\omega(u) = f_S(u), \omega(v) = f_S(v)$. Keďže $f_S$ je čiastočné $L(2,1)$-farbenie množiny $S_V$, vrcholy $u, v$ musia spĺňať podmienky $L(2,1)$-farbenia.

**Vrchol $u$ patrí do $S_V$, vrchol $v$ do $V(G_i) - S_V$ pre niektoré $i$:** Z definície $\omega$ platí $\omega(u) = f_S(u) = f_i(u)$ a $\omega(v) = f_i(v)$. Keďže $f_i$ je čiastočné $L(2,1)$-farbenie, vrcholy $u, v$ musia spĺňať podmienky $L(2,1)$-farbenia.

**Oba vrcholy patria do toho istého komponentu $G_i$:** Z definície $\omega$ platí $\omega(u) = f_i(u), \omega(v) = f_i(v)$. Keďže $f_i$ je čiastočné $L(2,1)$-farbenie, vrcholy $u, v$ musia spĺňať podmienky $L(2,1)$-farbenia.

**Vrchol $u$ patrí do komponentu $G_i$, vrchol $v$ do komponentu $G_j$, $i \neq j$:** Keďže vrcholy patria do rôznych komponentov grafu $G - S_E$, každá cesta medzi $u, v$ obsahuje aspoň jednu hranu z $S_E$. Špeciálne to platí pre každú najkratšiu cestu medzi $u$ a $v$. Keďže $u, v \notin S_V$, každá najkratšia cesta medzi $u$ a $v$ musí pozostávať z neprázdnej cesty z $u$ do niektorého vrchola v $S_V$, hrany v $S_E$ a z neprázdnej cesty z vrchola v $S_V$ do vrchola $v$. To znamená, že vrcholy $u$ a $v$ sú vzdialené aspoň 3, teda spĺňajú podmienky $L(2,1)$-farbenia.

Každá dvojica vrcholov spĺňa obmedzenia $L(2,1)$-farbenia, a preto $\omega$ je $L(2,1)$-farbením grafu $G$. □

**Dôsledok 15.** *Nech $e \in E(G)$ je most grafu $G$, nech $u, v$ sú vrcholy hrany $e$, nech graf $G - e$ pozostáva z dvoch komponentov $G_1$ a $G_2$ veľkosti aspoň 2. Potom je graf $G$ $k$-$L(2,1)$-zafarbiteľný práve vtedy, ak existujú hodnoty $a, b$ také, že $0 \leq a, b \leq k, |a - b| \geq 2$ a existujú $k$-$L(2,1)$-farbenia $f_1, f_2$ grafov $G_1 \cup \{v, u, e\}$ a $G_2 \cup \{v, u, e\}$, pre ktoré platí $f_1(u) = f_2(u) = a \wedge f_1(v) = f_2(v) = b$.*

*Dôkaz.* $[\Rightarrow]$ Ak je graf $G$ $k$-$L(2,1)$-zafarbiteľný, existuje jeho $k$-$L(2,1)$-farbenie $f$. Zobrazenia $f_1, f_2$ môžeme definovať ako zúženie zobrazenia $f$ na príslušné množiny vrcholov. Hodnoty $a, b$ určíme ako $f(u)$, resp. $f(v)$.

$[\Leftarrow]$ Tvrdenie je inštanciou predošlej lemy: $S_E = \{e\}$, $S_V = \{u, v\}$, $f_S(u) = a$, $f_S(v) = b$. Keďže $f_1$ je $k$-$L(2,1)$-farbenie grafu $G_1 \cup \{v, u, e\}$, ide o čiastočné $L(2,1)$-farbenie v jeho nadgrafe $G$. Inak povedané, ide o čiastočné $L(2,1)$-farbenie množiny $V(G_1) \cup \{u, v\}$. Analogicky je $f_2$ čiastočné farbenie množiny $V(G_2) \cup \{u, v\}$. Obe farbenia používajú hodnoty medzi 0 a $k$ vrátane, čiže ich spojením vznikne tiež $k$-$L(2,1)$-farbenie. □

Tento dôsledok nám vlastne dáva spôsob, ktorým vieme rozdeliť rozhodovanie problému $L(2,1)$-farbenia grafu $G$ na dve nezávislé časti v prípade, že $G$ obsahuje *netriviálny most* – most, ktorého odstránením dostaneme dva komponenty súvislosti s aspoň dvomi vrcholmi.

Pre každé možné ohodnotenie vrcholov mosta vyskúšame nezávisle ofarbiť oba komponenty. K obom komponentom potrebujeme pripojiť aj most a jeho druhý koniec. Za

cenu skúšania zhruba $k^2$ možností ofarbenia mostových vrcholov dostaneme dva nezávislé problémy.

Intuitívne platí, že rozdelenie na dva podproblémy najviac zlepší časovú zložitosť vtedy, keď majú oba komponenty zhruba rovnakú veľkosť. Ak bude jeden komponent príliš malý, môžeme si takýmto rozdelením časovú zložitosť zhoršiť: Za zanedbateľné zmenšenie problému zaplatíme tým, že ho budeme musieť riešiť viackrát.

Rýchly algoritmus pre problém $L(2,1)$-farbenia môžeme skonštruovať napríklad z dvoch menších algoritmov. Jeden z nich bude riešiť problém $L(2,1)$-farbenia na grafoch bez triviálnych mostov. Druhý bude redukovať problém $L(2,1)$-farbenia na všeobecných grafoch na niekoľko inštancií problému na grafoch bez triviálnych mostov. V ďalšej časti ukážeme druhý typ algoritmu a odhadneme jeho časovú zložitosť v závislosti od časovej zložitosti prvého typu algoritmu.

## 2.2 Mostová veta

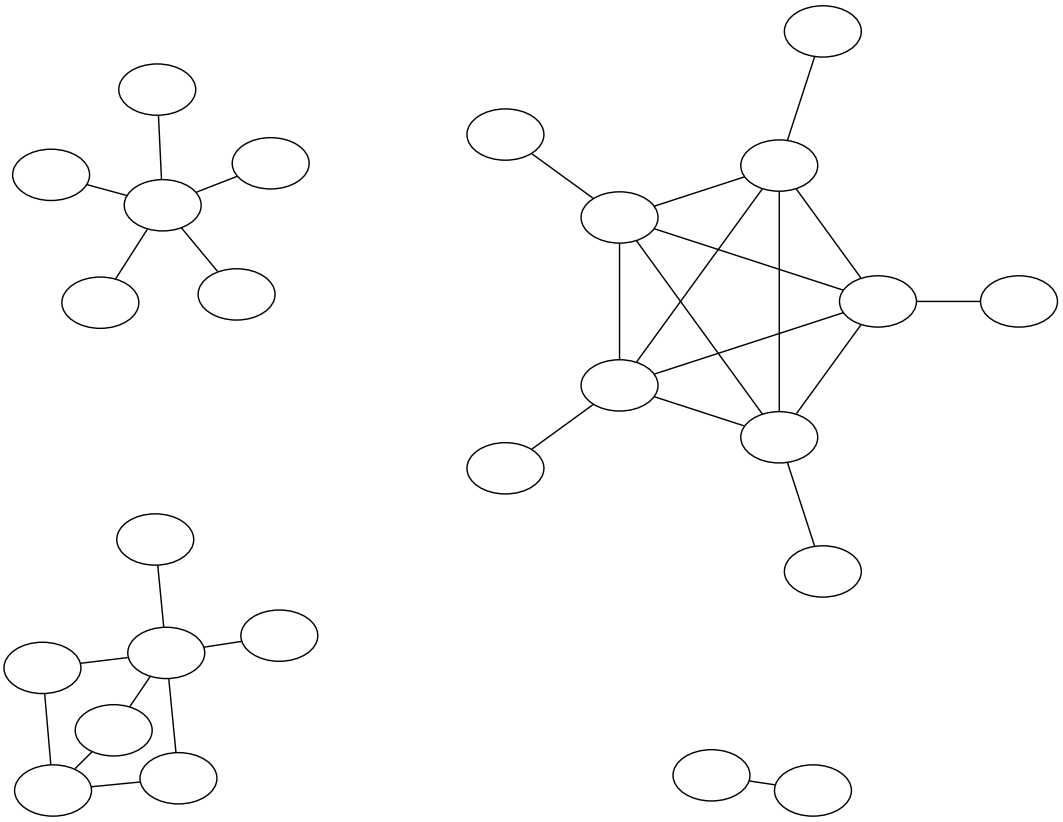
Na začiatok si zadefinujeme triedu grafov, ktorú môžeme opakovaným delením na menšie podproblémy dostať. Základnou triedou grafov, ktoré nevieme rozdeliť, sú súvislé bezmostové grafy. Tieto grafy sa inak nazývajú 2-*hranovo súvislé grafy.*

Trieda grafov, ktoré nevieme podľa predošlého dôsledku rozdeliť, obsahuje aj grafy, v ktorých každý most spája jediný vrchol so zvyškom grafu. Niekoľko takýchto grafov je znázornených na obrázku 2.1.

**Definícia 16.** *Graf $G$ budeme nazývať* chlpatý 2-hranovo súvislý graf*, ak je súvislý a pre každú mostovú hranu $e \in E(G)$ platí, že jeden z komponentov súvislosti grafu $G - e$ obsahuje jeden vrchol.*

Pripomeňme si, že každý graf, bez ohľadu na jeho štruktúru, má triviálne $L(2,1)$-farbenie s rozsahom $2n-2$, kde každému vrcholu priradíme rôzne párne číslo. Pokiaľ pri rozdelení grafu na dva komponenty ostane jeden z nich dostatočne malý v porovnaní s hodnotou $k$, pre každé ohodnotenie mostových vrcholov vieme nájsť jeho triviálne $L(2,1)$-farbenie s rozsahom $k$. Tento fakt zachytáva nasledujúce tvrdenie.

**Lema 17.** *Nech $e$ je mostová hrana grafu $G$, nech $u_1, u_2$ sú vrcholy incidentné s $e$, nech $G_1, G_2$ sú komponenty grafu $G - e$, nech vrchol $u_1$ leží v komponente $G_1$ a vrchol $u_2$ v komponente $G_2$ a nech $n_1$ je počet vrcholov komponentu $G_1$. Ak $2n_1 \leq k$, tak $k$-$L(2,1)$-farbenie grafu $G$ existuje práve vtedy, ak existuje $k$-$L(2,1)$-farbenie grafu $G_2 \cup \{e, u_1\}$.*



Obr. 2.1: Ukážka štyroch chlpatých 2-hranovo súvislých grafov.

*Dôkaz.* Implikácia $\boxed{\Rightarrow}$ triviálne platí, dokazovať budeme iba $\boxed{\Leftarrow}$.

Ukážeme, že ľubovoľné $k$-$L(2,1)$-farbenie $f$ grafu $G_2 \cup \{e, u_1\}$ vieme doplniť na $k$-$L(2,1)$-farbenie grafu $G$ s tým, že použijeme iba hodnoty z množiny $\{0, 1, \ldots, 2n_1\}$. Presnejšie, dokážeme existenciu množiny farieb $F$ obsahujúcu $f(u_1)$, neobsahujúcu $f(u_2)$, v ktorej rozdiel každej dvojice prvkov bude aspoň 2 a ktorej veľkosť bude aspoň $n_1$.

Graf $G_1$ má $n_1$ vrcholov, vrchol $u_1$ z neho už má priradenú niektorú farbu z $F$. Keď vrcholom grafu $G_1 - u_1$ ľubovoľne priradíme rôzne hodnoty z $F - \{f(u_1)\}$, dostaneme $k$-$L(2,1)$-farbenie grafu $G$.

Rozoberieme niekoľko prípadov:

$\boxed{f(u_1) \not\equiv f(u_2) \pmod 2}$: Ak je $f(u_1)$ párne, vezmeme $F = \{0, 2, 4, \ldots, 2n_1\}$. Ak je $f(u_1)$ nepárne, vezmeme $F = \{1, 3, 5, \ldots, 2n_1 - 1\}$. V oboch prípadoch majú všetky čísla v $F$ inú paritu, ako $f(u_2)$, čiže $f(u_2) \notin F$. Zároveň je veľkosť $F$ aspoň $n_1$.

$\boxed{f(u_1) \equiv f(u_2) \pmod 2 \wedge f(u_1) < f(u_2)}$: Ak je $f(u_2)$ párne, použijeme množinu $F = \{0, 2, \ldots, f(u_1), f(u_1)+3, f(u_1)+5, \ldots, 2n_1-1\}$. Všetky čísla v $F$, ktoré sú väčšie,

ako $f(u_2)$, majú inú paritu než $f(u_2)$.

Podobne pre nepárne $f(u_2)$ použijeme množinu $F = \{1, 3, \ldots, f(u_1), f(u_1)+3, f(u_1)+5, \ldots, 2n\}$.

$\boxed{f(u_1) \equiv f(u_2) \pmod 2 \wedge f(u_1) > f(u_2)}$ : Analogicky k predošlému prípadu, pre párne $f(u_2)$ použijeme množinu $F = \{1, 3, \ldots, f(u_1)-3, f(u_1), f(u_1)+2, \ldots, 2n_1\}$ a pre nepárne $f(u_2)$ množinu $\{0, 2, \ldots, f(u_1)-3, f(u_1), f(u_1)+2, \ldots, 2n_1-1\}$. □

Pokiaľ overujeme existenciu $L(2,1)$-farbenia s rozsahom $k$ pre dostatočne veľké $k$ a v grafe nájdeme most, ktorého odstránením bude prvý komponent malý a druhý veľký, podľa tejto lemy môžeme malý komponent odignorovať.

Keď v grafe objavíme most, ktorý delí graf na dve podobne veľké časti, môžeme si dovoliť obe vyriešiť pomocou algoritmu Junosza-Szaniawski. Najväčší problém teda budú spôsobovať grafy, v ktorých bude veľa mostov oddeľovať stredne veľké podgrafy od zvyšku.

### 2.2.1 Pseudoblokovo-mostový strom

Pri skúmaní grafov vzhľadom na bloky sa uplatnil pojem blokovo-artikulačného stromu. Podobný pojem vzhľadom na chlpaté 2-hranovo súvislé grafy nám pomôže pri konštrukcii algoritmu a zdôvodnení jeho zložitosti.

**Definícia 18.** *Nech $G$ je súvislý graf, nech $E$ je množina všetkých netriviálnych mostov v $G$, nech $K_1, K_2 \ldots K_m$ sú komponenty súvislosti grafu $G - E$. Pod pojmom* pseudoblokovo-mostový strom grafu $G$ *budeme rozumieť graf s ohodnotenými vrholmi $v_1, v_2, \ldots, v_m$, kde vrchol $v_i$ má priradenú hodnotu $|V(K_i)|$ a vrcholy $v_i$ a $v_j$ sú spojené hranou práve vtedy, keď existuje hrana $e \in E$, ktorá prepája nejaký vrchol v komponente $K_i$ s nejakým vrcholom v komponente $K_j$.*

Z daného grafu $G$ s $n$ vrcholmi a $m$ hranami vieme zostrojiť jeho pseudoblokovo-mostový graf v čase $O(n+m) = O(n^2)$. Pre všetky hrany grafu zistíme, či sú mostové pomocou Tarjanovho algoritmu [16]. Pre ľubovoľný most $e$ vieme v konštantnom čase overiť, či je netriviálny: Pre oba koncové vrcholy $e$ skontrolujeme, či sú incidentné aj s inou hranou, ako $e$. Keď poznáme netriviálne mosty, vhodným prehľadaním grafu, ktoré nebude môcť prejsť cez netriviálny most, vieme zistiť aj ohodnotenie vrcholov v pseudoblokovo-mostovom grafe v čase $O(n+m)$.

Medzi všeobecne známe vlastnosti stromov patrí aj existencia vyváženého vrcholového separátora: V každom strome $T$ existuje vrchol $v$ taký, že každý komponent grafu

$T - v$ má nanajvýš polovicu vrcholov z $T$. Podobné tvrdenie platí aj pre ohodnotené stromy:

**Lema 19.** *Nech $T$ je $n$-vrcholový strom s kladným ohodnotením vrcholov, nech $s$ je súčet hodnotení jeho vrcholov. Potom existuje vrchol $v \in V(T)$ taký, že súčet hodnotení vrcholov v každom komponente grafu $T - v$ je nanajvýš $\frac{s}{2}$.*

*Dôkaz.* Dôkaz sporom. Pre každý vrchol $v \in V(T)$ existuje komponent grafu $T - v$, ktorý má súčet ohodnotení viac ako $\frac{s}{2}$. Pre každý vrchol môže byť takýto komponent nanajvýš jeden, lebo dva komponenty so súčtom hodnotení viac ako $\frac{s}{2}$ sú v spore so súčtom hodnotení v $T$.

Pre každý vrchol $v$ teda existuje práve jeden komponent $K$ súvislosti grafu $T - v$, ktorý má súčet hodnotení viac ako $\frac{s}{2}$. Vrchol $v$ je hranou spojený s práve jedným vrcholom v každom komponente $T-v$, čiže je spojený s presne identifikovaným vrcholom $u$ v komponente $K$. Na hranu $(uv)$ nakreslíme šípku od $v$ k $u$. Tento postup vykonáme pre každý vrchol v grafe $T$, čiže $n$-krát.

Hrán v $n$-vrcholovom strome je $n-1$, teda aspoň na jednu hranu sme nakreslili dve šípky. Nech táto hrana spája vrcholy $u$ a $v$. Komponent grafu $T - u$, ktorý obsahuje vrchol $v$, označíme $K_v$ a podobne komponent grafu $T - v$ obsahujúci vrchol $u$ označíme $K_v$. Kedže hrana $(uv)$ dostala dve šípky, komponenty $K_u$ aj $K_v$ musia mať súčet ohodnotení viac ako $\frac{s}{2}$. Komponenty $K_u$ a $K_v$ však majú disjunktné množiny vrcholov, čo je v spore s faktom, že súčet hodnotení vrcholov v $T$ je $s$. □

Z dôkazu predošlej lemy nám taktiež vyplýva algoritmus, ktorým vieme v polynomiálnom čase nájsť takýto vrchol $v$. Najprv v čase $O(n)$ spočítame súčet hodnotení vrcholov. Potom začneme v nejakom vrchole $u$, spočítame súčet hodnotení v každom komponente grafu $G-u$ (v lineárnom čase). Ak má niektorý komponent priveľký súčet, nájdeme v ňom vrchol, ktorý susedí s $u$ a proces opakujeme. V opačnom prípade sme našli vrchol $v$ spĺňajúci lemu.

Pri tomto procese nemôžeme navštíviť žiaden vrchol viackrát, ináč by sme dostali buď spor s acyklickosťou stromu $T$, alebo by sme dostali podobný spor, ako na konci dôkazu. Algoritmus na hľadanie vrcholu $v$ sa dá dokonca spraviť v lineárnom čase od počtu vrcholov $T$, ale nám stačí ľubovoľný polynomiálny čas.

Nakoniec budeme potrebovať ešte dva stavebné algoritmy k našej redukcii. Jedným z nich je viackrát spomínaný algoritmus Junosza-Szaniawski, ktorý vie riešiť problém doplnenia $L(2,1)$-farbenia v ľubovoľnom súvislom grafe v čase $O^*(2.6488^n)$ [12]. Dru-

hým z nich je algoritmus od Haveta a kol., ktorý vie riešiť problém $L(2,1)$-farbenia pre rozsah $k \leq 5$ v čase $O^*(2.5^n)$ [11].

### 2.2.2 Algoritmus na rozdelenie problému

Nakoniec ukážeme, ako môžeme ľubovoľný algoritmus $\mathcal{B}$, ktorý rieši problém doplnenia $L(2,1)$-farbenia na chlpatých 2-hranovo súvislých grafoch, prerobiť na algoritmus $\mathcal{A}$, ktorý rieši problém $L(2,1)$-farbenia na všeobecných grafoch. Ak je časová zložitosť $\mathcal{B}$ $O^*(\alpha^n)$ pre nejaké $\alpha \geq 2.5$, časová zložitosť algoritmu $\mathcal{A}$ bude $O^*(\alpha^n)$ s výnimkou prípadov, keď $k$ bude z rozsahu $6 \ldots 11$. Do pozornosti dávame fakt, že $\mathcal{A}$ a $\mathcal{B}$ riešia mierne odlišné problémy.

Algoritmus $\mathcal{A}$ dostane na vstupe súvislý graf $G$ a číslo $k$ a akceptuje práve vtedy, keď existuje $L(2,1)$-farbenie grafu $G$ s rozsahom $k$. Popis algoritmu:

1. Ak $k \geq 2n$, akceptuj. $O^*(1)$
2. Ak $k \leq 5$, spusti a vráť výsledok algoritmu od Haveta a kol. $O^*(2.5^n)$
3. Nájdi všetky netriviálne mosty grafu $G$. $O^*(1)$
4. Ak $G$ nemá netriviálne mosty, spusti a vráť výsledok algoritmu $\mathcal{B}$. $O^*(\alpha^n)$
5. Vytvor pseudoblokovo-mostový strom $T$ grafu $G$. $O^*(1)$
6. Kým $T$ obsahuje listový vrchol $l$ s hodnotou nanajvýš $\frac{k}{2}$, odstráň z $T$ vrchol $l$ a zvýš rodičovi $l$ hodnotu o 1. V grafe $G$ odstráň podgraf zodpovedajúci vrcholu $l$ okrem hrany a vrcholu, ktorý ho pripájajú ku zvyšku grafu (Lema 17). $O^*(1)$
7. Ak je $G$ po predošlom kroku chlpatý 2-hranovo súvislý graf, spusti algoritmus $\mathcal{B}$. $O^*(\alpha^n)$
8. Nájdi vrchol $v$ grafu $T$, ktorý spĺňa lemu o rovnovážnom rozdelení stromu (Lema 19). $O^*(1)$
9. Pre každý komponent $K_i$ grafu $T - v$ spočítaj súčet jeho hodnôt $s_i$. $O^*(1)$
10. Ak pre niektorý komponent $K_i$ je hodnota $s_i$ väčšia ako $\frac{n}{10}$, nájdi hranu $e'$, ktorá pripája $K_i$ k $v$ v grafe $T$ a zodpovedajúci most $e = (uw)$ v grafe $G$. Pre každé ohodnotenie vrcholov $u, w$ spusti upravený algoritmus Junosza-Szaniawski na komponentoch grafu $G - e$. Akceptuj, ak pre niektoré ohodnotenie vrcholov $u, w$ algoritmus Junosza-Szaniawski akceptoval na oboch komponentoch grafu $G - e$ zároveň. V opačnom prípade zamietni. $O^*(k^2 \cdot 2.6488^{\frac{9n}{10}}) = O^*(2.5^n)$

11. Pre každý z komponentov $K_i$ nájdi hranu $e_i'$, ktorá prepája vrchol $v$ a vrchol v komponente $K_i$ a jej zodpovedajúcu hranu $e_i$ v grafe $G$. Pre komponent $K_i$ zostroj $G_i$ ako podgraf $G$, ktorý zodpovedá grafu $K_i$ v $T$ s hranou $e_i$.

12. Pre každý z grafov $G_i$ a každé ohodnotenie vrcholov $u_i, w_i$ hrany $e_i$ spusti prispôsobený algoritmus Junosza-Szaniawski. Zoznam ohodnotení vrcholov $u_i, w_i$, pre ktoré algoritmus akceptoval, označíme $Z_i$. $O^*(k^2 n \cdot 2.6488^{\frac{n}{10}})$

13. Skonštruuj graf $G_v$ ako podgraf $G$, ktorý zodpovedá vrcholu $v$ v $T$, spolu so všetkými hranami $e_i$. Pre každú hranu $e_i$ a každý spôsob ofarbenia hrany $e_i$ zo zoznamu $Z_i$ spusti algoritmus $\mathcal{B}$ na $G_v$. Ak aspoň jeden z behov akceptoval, akceptuj. V opačnom prípade zamietni.

Ďalej odvodíme časovú zložitosť algoritmu $\mathcal{A}$. Poriadnejšie si vysvetlíme časovú zložitosť krokov $10, 12$ a $13$.

Pri kroku 10 sme našli hranu, ktorej odstránením dostaneme dva komponenty grafu, z ktorých každý má nanajvýš $\frac{9n}{10}$ vrcholov. Ohodnotení vrcholov hrany je nanajvýš $(k+1)^2 = O(n^2)$, čiže algoritmus Junosza-Szaniawski pustíme nanajvýš $O(n^2)$-krát.

Ku kroku 12 sa dostaneme iba v prípade, že sme v kroku 10 nenašli komponent, ktorý by mal aspoň $\frac{n}{10}$ vrcholov. Zároveň vieme počet komponentov zhora odhadnúť na $n$. Pre každý z komponentov, ktorý má nanajvýš $\frac{n}{10}$ vrcholov teda spustíme $(k+1)^2 = O(n^2)$-krát algoritmus Junosza-Szaniawski, čím dostaneme dohromady časovú zložitosť $O^*(1.2^n)$.

Pre odhad časovej zložitosti kroku 13 najprv odhadneme, koľko rôznych (usporiadaných) dvojíc farieb môžeme priradiť dvom susedným vrcholom $u, w$. Pokiaľ vrchol $u$ ofarbíme farbou $x$, vrchol $w$ môže dostať ľubovoľnú farbu okrem $x, x-1$ a $x+1$. To znamená, že ak vrchol $u$ ofarbíme farbou 0 alebo $k$, ostane nám pre vrchol $w$ presne $k+1-2$ možností. Ak vrchol $u$ ofarbíme ktoroukoľvek zo zvyšných $k-1$ možností, pre vrchol $w$ nám vždy ostane $k+1-3$ možností. Úpravou dostaneme, že počet ofarbení týchto vrcholov je $2(k-1) + (k-1)(k-2) = k(k-1)$.

Každý z komponentov $K_i$ má súčet hodnôt ostro viac ako $\frac{k}{2}$, ináč by bol v kroku 6 odstránený. Keďže hodnoty v grafe $T$ sú prirodzené čísla, môžeme túto nerovnosť upraviť: Každý z komponentov $K_i$ má súčet hodnôt aspoň $\lfloor \frac{k}{2} \rfloor + 1$. Symbolom $p$ označíme počet komponentov grafu $T - v$ a symbolom $q$ označíme počet vrcholov grafu $G_v$, na ktorom budeme v kroku 13 spúšťať algoritmus $\mathcal{B}$.

Platí nerovnosť $q \leq n - p \cdot \lfloor \frac{k}{2} \rfloor$, lebo z každého z $p$ komponentov si ponecháme iba jeden vrchol. Pre každý z $p$ komponentov teda vyskúšame všetkých $k(k-1)$ ofarbení

mostu, ktorý ho pripája. Pre každú z týchto $(k(k-1))^p$ možností spustíme algoritmus $\mathcal{B}$ s časovou zložitosťou $O^*(\alpha^q)$. Celková časová zložitosť teda bude $O^*\left((k^2-k)^p\alpha^q\right) = O^*\left((k^2-k)^p\alpha^{n-p\cdot\lfloor\frac{k}{2}\rfloor}\right)$.

Ďalej ukážeme, že pre $k \geq 12$ platí nerovnosť $(k^2-k)^{\frac{1}{\lfloor\frac{k}{2}\rfloor}} \leq 2.5$. Najprv spravíme niekoľko odhadov: $(k^2-k)^{\frac{1}{\lfloor\frac{k}{2}\rfloor}} \leq k^{\frac{2}{\lfloor\frac{k}{2}\rfloor}} \leq k^{\frac{2}{\frac{k}{2}-1}} \leq k^{\frac{2}{\frac{k}{3}}} = k^{\frac{6}{k}} = \exp\left(\frac{6\ln k}{k}\right)$.

Funkcia $\frac{6\ln k}{k}$ je klesajúca pre $k \geq e$ a hodnota $\exp\left(\frac{6\ln k}{k}\right)$ je v bode $k = 20$ je zhruba 2.46. Pre $k$ v rozsahu $12\ldots19$ uvedieme hodnoty funkcie $(k^2-k)^{\frac{1}{\lfloor\frac{k}{2}\rfloor}}$:

| $k=12$ | $k=13$ | $k=14$ | $k=15$ | $k=16$ | $k=17$ | $k=18$ | $k=19$ |
|---|---|---|---|---|---|---|---|
| 2.25647 | 2.32018 | 2.10314 | 2.14657 | 1.98393 | 2.01521 | 1.88882 | 1.91231 |

Nakoniec s využitím nerovnosti $(k^2-k)^{\frac{1}{\lfloor\frac{k}{2}\rfloor}} \leq 2.5$ a $\alpha \geq 2.5$ upravíme časovú zložitosť pre $k \geq 12$:

$$O^*\left((k^2-k)^p\alpha^{n-p\cdot\lfloor\frac{k}{2}\rfloor}\right) = O^*\left(\alpha^n\cdot\frac{(k^2-k)^p}{\alpha^{p\cdot\lfloor\frac{k}{2}\rfloor}}\right) = O^*\left(\alpha^n\cdot\left(\frac{(k^2-k)^{\frac{1}{\lfloor\frac{k}{2}\rfloor}}}{\alpha}\right)^{p\cdot\lfloor\frac{k}{2}\rfloor}\right) =$$

$$= O^*(\alpha^n)$$

Ukázali sme teda spôsob, ako takmer ľubovoľnú inštanciu problému $L(2,1)$-farbenia vyriešiť pomocou algoritmu pre problém doplnenia $L(2,1)$-farbenia na chlpatom 2-hranovo súvislom grafe. Ostalo niekoľko hodnôt $k$, pre ktoré by náš algoritmus nedosahoval dobrú časovú zložitosť. Grafy, ktoré by dosahovali veľmi zlú časovú zložitosť, sú však pomerne špeciálne. Na ich vyriešenie by však bol potrebný oveľa dôkladnejší rozbor.

# Kapitola 3

# Rozdelenie pri čiastočných farbeniach

Pri všetkých významnejších algoritmoch na hľadanie farbiaceho čísla všeobecného grafu $G$ sa uplatnila kompaktnejšia reprezentácia množiny čiastočných farbení. V tejto kapitole sa pozrieme na to, ako sa dá kompaktná reprezentácia použiť pri delení problému a ukážeme si algoritmy na rozdeliteľných grafoch, ktoré budú dosahovať lepšiu časovú zložitosť, ako algoritmus Junosza-Szaniawski.

Ako prvú si uvedieme definíciu našej kompaktnej reprezentácie čiastočných ofarbení.

**Definícia 20.** *Nech $G$ je graf s vrcholmi usporiadanými v postupnosti $v_1, v_2, \ldots, v_n$. Nech $f$ je čiastočné $L(2,1)$-farbenie množiny $S$ v grafe $G$ s rozsahom $k$. Pod pojmom* charakteristika rádu $k$ zobrazenia $f$ *budeme rozumieť reťazec $r \in \{0, 1, \bar{1}\}^n$, ktorý spĺňa:*

$$r_i = 0 \Leftrightarrow v_i \notin S$$
$$r_i = 1 \Leftrightarrow v_i \in S \wedge f(v_i) \neq k$$
$$r_i = \bar{1} \Leftrightarrow v_i \in S \wedge f(v_i) = k$$

*Ďalej pod pojmom* rozšírená charakteristika rádu $k$ zobrazenia $f$ *budeme rozumieť reťazec $s \in \{0, \bar{0}, 1, \bar{1}\}^n$, ktorý spĺňa*

$$s_i = 0 \Leftrightarrow v_i \notin S \wedge (\forall j \in \{1 \ldots n\} : v_j \in N(v_i) \Rightarrow s_j \neq \bar{1}$$
$$s_i = \bar{0} \Leftrightarrow v_i \notin S \wedge (\exists j \in \{1 \ldots n\} : v_j \in N(v_i) \wedge s_j = \bar{1}$$
$$s_i = 1 \Leftrightarrow v_i \in S \wedge f(v_i) \neq k$$
$$s_i = \bar{1} \Leftrightarrow v_i \in S \wedge f(v_i) = k$$

*Charakteristiky čiastočných farbení budeme ďalej v texte označovať symbolom $r$.*

Podobne vieme zadefinovať charakteristiku (resp. rozšírenú charakteristiku) množiny $C$ vo farbení $f$ ako podpostupnosť charakteristiky (resp. rozšírenej charakteristiky) zobrazenia $f$, ktorá obsahuje iba znaky zodpovedajúce vrcholom v $C$.

Z algoritmu Junosza-Szaniawski môžeme vidieť, že na reprezentáciu množiny čiastočných farbení s obmedzeným rozsahom nám stačí ukladať množinu charakteristík fixného rádu. Keď nepotrebujeme pre každý ofarbený vrchol ukladať jeho presnú farbu, môžeme si dovoliť reprezentovať aj vrcholové separátory a použiť ich na rozdelenie problému. O presnom spôsobe a aplikácií takéhoto rozdelenia si povieme v ďalšej časti.

Predtým si ale stručne pripomenieme lemu o hranových separátoroch z predošlej kapitoly (14). Táto lema hovorí, že existencia $L(2,1)$-farbenia grafu $G$ vyplýva z existencie $L(2,1)$-farbení komponentov grafu $G - E$ pre hranový oddeľovač $E$. Tieto farbenia musia navyše farbiť všetky vrcholy z hranového oddeľovača a musia sa zhodovať v ich farbách.

Podobné tvrdenie platí aj pre čiastočné farbenia. V tomto prípade je podmienkou, že všetky čiastočné farbenia farbia tú istú podmnožinu vrcholov z hranového oddeľovača a navyše im priradzujú rovnakú farbu. Keďže ide o takmer identické tvrdenie k leme o hranových separátoroch (14), nebudeme jeho presné znenie uvádzať.

## 3.1 Rozdelenie na vrcholových separátoroch

Základnou myšlienkou rozdelenia na vrcholových separátoroch je, že ak máme zafixované čiastočné farbenie separátora a všetkých jeho susedov, farbenie zvyšných vrcholov v rôznych komponentoch je nezávislé.

**Lema 21.** *Nech $S_V$ je vrcholový separátor grafu $G$, nech $G_1, G_2, \dots G_k$ sú komponenty grafu $G - S_V$ a nech $H_1, H_2, \dots, H_k$ sú podgrafy $G$ indukované množinami $V(G_1) \cup S_V,\ V(G_2) \cup S_V,\ \dots,\ V(G_k) \cup S_V$.*

*Nech $f$ je čiastočné $L(2,1)$-farbenie nejakej podmnožiny $S \subseteq N[S_V]$ v grafe $G$ a nech $f_1, f_2, \dots f_k$ sú čiastočné farbenia množín $S_1, S_2, \dots, S_k$ v grafoch $H_1, H_2, \dots, H_k$ (v tomto poradí), ktoré spĺňajú podmienku*

$$\forall i \in \{1 \dots k\},\ \forall v \in V(H_i) \cap N[S_V] : (v \in S_i \Leftrightarrow v \in S) \wedge (v \in S_i \Rightarrow f_i(v) = f(v)).$$

*Potom zobrazenie $\omega : \bigcup_{i=1}^{k} S_i \to \mathbb{N}$ definované nasledovne:*

$$\omega(v) = \begin{cases} f(v), & v \in S \\ f_i(v), & v \in S_i - S \end{cases}$$

*tvorí čiastočné farbenie grafu $G$.*

*Dôkaz.* Dôkaz je veľmi podobný dôkazu lemy o hranových separátoroch 14 v predošlej kapitole. Všetky vrcholy ofarbené vrámci jedného grafu $H_i$ nemôžu porušiť podmienku čiastočného $L(2,1)$-farbenia. Podobne pre dva vrcholy v $S$.

Každá dvojica vrcholov, kde jeden patrí do $S_i - S$ a druhý do $S - S_i$ má vzdialenosť aspoň 3, lebo cesta medzi nimi musí obsahovať aspoň jeden úsek, ktorý tvorí hrana z $S_i - S$ do $N(S)$, hrana z $N(S)$ do $S$ a hrana z $S$ do iného vrchola $N(S)$.

Nakoniec každá dvojica vrcholov, kde jeden patrí do $S_i - S$ a druhý do $S_j - S$, kde $i \neq j$, musí mať tiež vzdialenosť aspoň 3. □

Okolie jedného vrchola môže byť pomerne veľké, napríklad vo hviezdach v ňom môžu byť všetky zvyšné vrcholy. Pri reprezentácii čiastočného farbenia cez jeho charakteristiku nám to však nevadí. Rôznych charakteristík, ktoré môžu mať čiastočné farbenia malého separátora a jeho okolia, je malý. Horný odhad pre počet rôznych charakteristík si zhrnieme v nasledujúcej leme.

**Lema 22.** *Nech $S$ je vrcholový separátor grafu $G$, nech $q = |S|$ a $p = |N[S]|$, nech $k \in \mathbb{N}$ je ľubovoľné číslo, nech $F$ je množina všetkých čiastočných farbení s rozsahom $k$, ktoré ofarbujú nejakú podmnožinu $N[S]$, nech $R$ je množina všetkých charakteristík rádu $k$ farbení v $F$. Potom $|R| \leq 2^p \cdot (p+1)^q$.*

*Dôkaz.* Najprv zhora odhadneme počet rôznych množín vrcholov, ktoré majú priradenú farbu $k$. Pre každý vrchol $v \in S$ platí, že v jeho okolí $N[v]$ je nanajvýš jeden vrchol, ktorý má farbu $k$. V opačnom prípade by totiž mali dva vrcholy vo vzdialenosti nanajvýš 2 priradenú tú istú farbu. Keďže celkový počet vrcholov v $N[S]$ je $p$, pre každý z $q$ vrcholov separátora môžeme vybrať nanajvýš jednu z $p$ možností, alebo nevybrať žiadny vrchol. Preto množín vrcholov, ktoré majú priradenú farbu $k$, je nanajvýš $(p+1)^q$.

Ďalej pre každú množinu vrcholou s farbou $k$ ostáva nanajvýš $p$ vrcholov, každý z nich nezávisle patrí, alebo nepatrí do množiny ofarbených vrcholov. Pre každú možnosť množiny vrcholov s farbou $k$ máme teda nanajvýš $2^p$ možností, ako vyzerá množina ostatných ofarbených vrcholov. Charakteristík je teda nanajvýš $2^p(p+1)^q$. □

Rozdeľovanie pomocou vrcholového separátora budeme využívať na konštrukciu rýchlejších algoritmov pre problém $L(2,1)$-farbenia. Pre všetky možné charakteristiky separátora a okolia si budeme pamätať, aké charakteristiky môžu mať farbenia vo všetkých komponentoch.

Podobne bude prebiehať pridanie novej farby. Vyskúšame pridanie všetkých farieb do separátora a jeho okolia. Pre každú takúto možnosť potom pridáme novú farbu do každého komponentu. Ako to bude presne vyzerať s postupom a s časovou zložitosťou, si ukážeme na príklade planárnych grafov.

## 3.2 Farbenie planárnych grafov

Dôležitým výsledkom pre farbenie planárnych grafov je veta o separátoroch v planárnych grafoch [15], ktorú si uvedieme.

**Veta 23.** *Nech $G$ je planárny graf s $n$ vrcholmi. Potom sa dajú vrcholy grafu $G$ rozdeliť do troch množín $A, B, C$ tak, že neexistuje hrana medzi žiadnym vrcholom v $A$ a žiadnym vrcholom v $B$, množiny $A$ aj $B$ majú nanajvýš $\frac{n}{2}$ vrcholov a množina $C$ má nanajvýš $\frac{2\sqrt{2n}}{1-\sqrt{\frac{2}{3}}}$ vrcholov.*

**Poznámka 24.** *Hodnota výrazu $\frac{2\sqrt{2}}{1-\sqrt{\frac{2}{3}}}$ je menšia ako 16, preto budeme kvôli prehľadnosti veľkosť množiny $C$ zhora odhadovať ako $16\sqrt{n}$.*

Ďalej budeme vytvárať algoritmus, ktorý dokáže pre danú množinu charakteristík čiastočných $L(2,1)$-farbení s rozsahom $k$ vypočítať množinu charakteristík čiastočných $L(2,1)$-farbení s rozsahom $k+1$. Túto funkciu nad množinami charakteristík budeme ďalej označovať $\oplus$. Pre množinu charakteristík $R$ zadefinujeme množinu suffixov reťazca $w$ ako $R_w = \{u | wu \in R\}$. Budeme potrebovať ešte jeden pojem:

**Definícia 25.** *Nech $G$ je graf. Dvojicu charakteristík $(r_1, r_2)$ budeme nazývať* prechod rádu $k$, *ak existujú čiastočné $L(2,1)$-farbenia $f_1, f_2$, ktoré spĺňajú nasledujúce podmienky:*

1. *$f_1$ má rozpätie $k$, $f_2$ má rozpätie $k+1$,*
2. *$r_1$ je charakteristika rádu $k$ farbenia $f_1$ a $r_2$ je charakteristika rádu $k+1$ farbenia $f_2$,*
3. *$\forall i \leq k : \{v \mid f_1(v) = i\} = \{v \mid f_2(v) = i\}$.*

*Dvojicu charakteristík* $(s_1, s_2)$ *množiny* $S \subseteq V(G)$ *budeme nazývať* prechod rádu $k$ nad množinou $S$, *ak sa* $s_1$ *a* $s_2$ *dajú doplniť na prechod grafu* $G$.

*Počet všetkých prechodov v* $G$ *označíme* $pr(G)$. *Podobne počet prechodov nad množinou* $S$ *v grafe* $G$ *označíme* $pr_S(G)$.

Vrcholy separátora a okolia (množinu $N[C]$, kde $C$ je množina z vety 23) označíme $S$, počet vrcholov v $S$ označíme $p$. Vrcholy grafu usporiadame tak, aby vrcholy $S$ boli na začiatku.

Základný postup pri počítaní $\oplus(R)$ s využitím separátorov bude nasledovný: Množinu (nerozšírených) charakteristík $R$ rozdelíme podľa ohodnotenia $S$. Pre každý prechod nad $S$ nezávisle vypočítame $\oplus$ na charakteristikách zvyšných vrcholov. Nakoniec tieto čiastočné výsledky pospájame a odstránime duplikáty. Formálny zápis výpočtu a jeho odvodenia bude vyzerať nasledovne:

$$\oplus(R) = \bigcup_{w \in R} \oplus(w) = \bigcup_{w \in \{0,1,\bar{1}\}^p} \oplus(R_w) = \bigcup_{u \in \{0,1,\bar{1}\}^p} u \cdot \left( \bigcup_{\substack{w \in \{0,1,\bar{1}\}^p \\ \text{t.ž. } u \in \oplus(w)}} \oplus(R_w) \right)$$

Riešenie problému $L(2,1)$ farbenia potom niekoľkokrát aplikuje tento algoritmus, až kým nebude v množine charakteristík $R$ taká, ktorá zodpovedá $L(2,1)$-farbeniu. Keďže každý graf má triviálne farbenie s rozsahom $2n-2$, tento algoritmus nám zaručene stačí spustiť najviac $(2n-2)$-krát.

Základom algoritmu bude nájsť separátor podľa vety o planárnom separátore 23. Tento krok stačí vykonať iba raz, preto si môžeme dovoliť jeho hľadanie tak, že prejdeme všetkých $2^n$ podmnožín vrcholov a pre každú v polynomiálnom čase overíme, či je dostatočne malá a či po jej odstránení bude mať každý komponent súvislosti nanajvýš $\frac{n}{2}$ vrcholov.

Ďalej teda predpokladáme, že poznáme vrcholový separátor $C$. Podľa toho, koľko vrcholov je v okolí separátora, čiže v $N[C]$, budeme robiť rôzne veci. Najprv rozoberieme jednoduchší prípad, kde $|N[C]| \geq \frac{3n}{4}$, potom rozoberieme prípad $|N[C]| < \frac{3n}{4}$.

### 3.2.1 Separátor s veľkým okolím

Popíšeme algoritmus, ktorý pre danú množinu charakteristík $R$ rádu $k$ spočíta novú množinu charakteristík rádu $k+1$. Základný postup sme už popísali, pre každú mož-

nosť prechodu separátora vyskúšame pridať novú farbu zvyšným vrcholom. Separátor označíme $C$, počet vrcholov v $N[C]$ označíme $p$, počet vrcholov $C$ označíme $q$.

Pre ľubovoľnú charakteristiku separátora a jeho okolia existuje najviac $(p+1)^q$ možností, ktoré vrcholy dostanú novú farbu $k+1$. Taktiež vieme všetky tieto možnosti generovať – pre každý vrchol v separátore vyberieme niektorý z vrcholov jeho uzavretého okolia, alebo nevyberieme v jeho okolí žiadny vrchol. Pre každý takýto výber následne overíme, či sme nevybrali dva vrcholy vo vzdialenosti menšej, ako 3, alebo či nejaký vybratý vrchol nesusedí s vrcholom farby $k$.

Pre každú možnosť starej a novej charakteristiky separátora ostane nejaka množina charakteristík zvyšných $n - p \leq \frac{n}{4}$ vrcholov, pre ktorú potrebujeme vypočítať nové charakteristiky. Tu si môžeme dovoliť extrémne neefektívne riešenie, kde pre každú starú charakteristiku $r$ a každú podmnožinu $U$ jej neofarbených vrcholov vyskúšame, či je povolené ofarbiť všetky vrcholy $U$ farbou $k+1$. Takýmto spôsobom budeme overovať nanajvýš $4^{n-p}$ možností – keby boli vrcholy mimo separátora nezávislé, pre každý máme iba 4 typy možností, v ktorých ho budeme skúšať: Neofarbený vrchol ostáva neofarbený, neofarbený vrchol farbíme $k+1$, vrchol ofarbený farbou $k$, vrchol ofarbený inou farbou.

Overenie jedného výberu je polynomiálne, lebo stačí pre každú dvojicu vrcholov vo vzdialenosti nanajvýš 2 skontrolovať, či by ich priradenie farieb neporušilo podmienky $L(2,1)$-farbenia.

Pre každú z nanajvýš $2^p(p+1)^q$ možností ofarbenia $S$ vyskúšame nanajvýš $(p+1)^q$ možností pridania novej farby do $C$. Pre každú takúto možnosť následne vyskúšame nanajvýš $4^{n-p}$ možností pridania novej farby do množiny charakteristík a pre každú možnosť vykonáme polynomiálne veľa práce.

Celková časová zložitosť teda bude $O^*(2^p(p+1)^{2q}4^{n-p})$. Z vety o separátore v planárnych grafoch 23 vieme, že veľkosť separátora, $q$, je nanajvýš $16\sqrt{n}$. Túto vetvu algoritmu spúšťame, keď $p \geq \frac{3n}{4}$. Funkcia $2^p4^{n-p}$ klesá s $p$ a teda jeho najväčšia možná hodnota pre minimálne $p = \frac{3n}{4}$ je $2^{\frac{5n}{4}}$. Výraz $(p+1)^{2q}$ zhora odhadneme na $(n+1)^{32\sqrt{n}} \leq (2n)^{32\sqrt{n}}$.

Dosadením horných odhadov dostaneme časovú zložitosť $O^*\left(2^{\frac{5n}{4}+32\sqrt{n}+32\lg n\sqrt{n}}\right) = O^*\left(2.38^{n+o(n)}\right)$.

### 3.2.2 Separátor s malým okolím

V prípade separátora s malým okolím budeme využívať rozdelenie pomocou hranového separátora. Nech $A, B, C$ sú množiny vrcholov spĺňajúce vetu o planárnom separátore 23. Bez ujmy na všeobecnosti má množina $C$ viac susedov v množine $A$, než v množine $B$. Keďže dokopy má množina $N[C]$ menej ako $\frac{3n}{4}$ vrcholov a množiny $A$ a $B$ sú disjunktné, množina $B \cap N[C]$ má nanajvýš polovicu z tohto počtu, čiže $\frac{3n}{8}$.

Pre konštrukciu algoritmu použijeme hranový separátor medzi množinami $B$ a $C$, ktorý obsahuje vrcholy $C \cup (B \cap N[C])$. Túto množinu vrcholov budeme ďalej označovať $S$. Množinu hrán medzi množinami $B$ a $C$ označíme $E$. Komponenty súvislosti grafu $G - E$ označíme $K_1, \ldots, K_l$. Počet vrcholov v $S$ označíme $p$, počet vrcholov v $C$ označíme $q$.

Z lemy o hranových separátoroch vieme, že pre zafixovanú charakteristiku vrcholov $S$ môžeme pre každý z komponentov $K_1 \ldots K_l$ ukladať množinu jeho charakteristík nezávisle. Množinu charakteristík celého grafu by sme vedeli zostrojiť ako karteziánsky súčin týchto dielčích charakteristík, zjednotený cez všetky charakteristiky množiny $S$. Explicitne to však robiť nebudeme.

Postup počítania funkcie $\oplus$ bude podobný, ako pri separátore s veľkým okolím. Pre každý prechod nad $S$ a každý komponent $K_i$ skonštruujeme z jeho pôvodnej množiny charakteristík $R_i$ novú množinu charakteristík $\oplus(R_i)$. Keďže každý komponent tvorí súvislý graf, môžeme na počítanie $\oplus(R_i)$ použiť postup z algoritmu Junosza-Szaniawski.

Pre počítanie operácie $\oplus$ podľa algoritmu Junosza-Szaniawski potrebujeme poznať rozšírené charakteristiky. Pre každý komponent a každú jeho (nerozšírenú) charakteristiku vieme vypočítať jej rozšírený ekvivalent v polynomiálnom čase: Pre každý vrchol s hodnotou $\bar{1}$ prečíslujeme všetkých jeho susedov s hodnotou 0 na hodnotu $\bar{0}$.

Pri počítaní operácie $\oplus$ na komponentoch $K_1, \ldots, K_l$ neberieme do úvahy celý separátor, vždy používame iba tú časť, ktorá sa nachádza v danom komponente. Môže teda nastať situácia, že prechod na celom separátore nebude konzistentný s niektorými starými, alebo s niektorými novými charakteristikami v komponente.

Toto vieme vyriešiť ľahko. Pre každý prechod separátora $S$ budeme pred vykonaním $\oplus$ na komponente $K_i$ filtrovať jeho charakteristiky, aby sme používali iba tie, pri ktorých môže daný prechod separátora nastať. Po vypočítaní $\oplus$ na komponente $K_i$ môžeme taktiež dostať nejaké charakteristiky, ktoré nie sú kompatibilné s prechodom separátora. Aj v tomto prípade stačí tieto charakteristiky odfiltrovať po vypočítaní operácie $\oplus$.

V krátkosti si popíšeme, ako bude fungovať filtrovanie pred počítaním operácie $\oplus$ v komponente $K_i$. Filtrovanie po výpočte $\oplus$ sa dá zostrojiť analogicky.

Zistiť, či je charakteristika kompatibilná s prechodom separátora, vieme v polynomiálnom čase. Pre každý vrchol $v$ v komponente $K_i$ a pre každý vrchol $u$ separátora skontrolujeme, či sú vzdialené aspoň 3. Ak nie sú, môžu byť vzdialené 2 alebo 1. Ak sú vzdialené 2, nesmie nastať situácia, že $v$ má charakteristiku $\bar{1}$ a $u$ má starú charakteristiku $\bar{1}$. Ak sú vzdialené 1, nesmie nastať situácia, že $v$ má charakteristiku $\bar{1}$ a $u$ má starú, alebo novú charakteristiku $\bar{1}$.

### Časová zložitosť

Všetky komponenty, ktoré obsahujú nejaký vrchol v $B$, majú nanajvýš $\frac{n}{2}$ vrcholov, lebo oddeľovač $E$ oddeľuje množinu $B$ a $C$. Množiny $A$ a $B$ sú oddelené podľa vety o separátore a množina $B$ má nanajvýš $\frac{n}{2}$ vrcholov.

Podobne, všetky komponenty, ktoré obsahujú nejaký vrchol v $A$, majú nanajvýš $\frac{n}{2} + 16\sqrt{n}$ vrcholov, lebo množiny $A$ a $C$ majú dohromady najviac $\frac{n}{2} + 16\sqrt{n}$ vrcholov a ich vrcholy nie sú spojené s vrcholmi v $B$.

Počet charakteristík $S$ odhadneme podobne, ako pre vrcholové separátory: Pre každý vrchol v $C$ máme nanajvýš $p + 1$ možností, ako vybrať nanajvýš jeden vrchol z jeho okolia. Výberov pre všetky vrcholy je teda nanajvýš $(p+1)^q$ a pre zvyšné vrcholy máme nanajvýš $2^p$ možností. Dohromady dostávame horný odhad $2^p(p+1)^q$ možností. Pre každú z týchto možností potom máme nanajvýš $(p + 1)^q$ možností, ktoré vrcholy dostanú farbu $k + 1$.

Nakoniec pre každú možnosť starej charakteristiky a množiny vrcholov s novou farbou budeme počítať na každom komponente operáciu $\oplus$ pomocou algoritmu z článku Junosza-Szaniawskeho a kol. [12], ktorá na komponente s veľkosťou $x$ pracuje v čase $O^*(2.65^x)$. Každý komponent, na ktorom ju budeme spúšťať, má nanajvýš $\frac{n}{2}+16\sqrt{n}$ vrcholov. Komponentov je nanajvýš $n$, teda budeme potrebovať nanajvýš $n$ nezávislých výpočtov. Tento faktor sa stratí v $O^*$ notácii.

Časová zložitosť tohto algoritmu teda bude

$$O^*\left(2^{\frac{3n}{8}} \cdot \left(\frac{3n}{8} + 16\sqrt{n} + 1\right)^{32\sqrt{n}} \cdot 2.65^{\frac{n}{2}+16\sqrt{n}}\right) = O^*\left(2.12^{n+o(n)}\right).$$

### 3.2.3 Vylepšenie a poznámky

Popísali sme si algoritmus, ktorý na základe veľkosti okolia separátora robil veľmi rôzne výpočty. Pri vysvetľovaní sme ako hranicu použili ľahko zapísateľnú, ale nie najoptimálnejšiu konštantu $\alpha = \frac{3}{4}$. Pre grafy, v ktorých má okolie separátora aspoň $\alpha n$ vrcholov sme používali algoritmus pre separátor s veľkým okolím, pre ostatné algoritmus pre separátor s malým okolím. Ďalej odhadneme lepšiu konštantu a časovú zložitosť, ktorá z jej použitia vyplýva.

Časová zložitosť algoritmu pre separátor s veľkým okolím je pre ľubovoľnú konštantu $\alpha \in (0, 1)$ v triede funkcií $O^*(2^{\alpha n+o(n)}4^{(1-\alpha)n}) = O^*(2^{n(2-\alpha)+o(n)})$.

Časová zložitosť algoritmu pre separátor s malým okolím je pre ľubovoľnú konštantu $\alpha \in (0, 1)$ v triede funkcií $O^*(2^{n\frac{\alpha}{2}+o(n)}2.65^{\frac{n}{2}+o(n)})$.

Zvyšovaním konštanty $\alpha$ teda zlepšujeme časovú zložitosť algoritmu pre separátor s malým okolím a zhoršujeme časovú zložitosť algoritmu pre separátor s veľkým okolím. Vyváženú časovú zložitosť dostaneme pre takú konštantu $\alpha$, kde $2-\alpha = \frac{\alpha}{2} + \frac{\lg(2.65)}{2}$. Vyriešením tejto rovnice dostávame hodnotu $\alpha \approx 0.865$, z ktorej vyplýva časová zložitosť $O^*(2.2^{n+o(n)})$ pre oba prípady.

Uvedený algoritmus sa okrem existencie vrcholového rozdeľovača veľkosti $O(\sqrt{n})$ nespoliehal na žiadnu inú vlastnosť planárnych grafov. Preto uvedený algoritmus funguje pre všetky triedy grafov, v ktorých existuje vrcholový separátor veľkosti $O(\sqrt{n})$. Toto tvrdenie vieme dokonca mierne zosilniť na triedy grafov, v ktorých existuje vrcholový separátor veľkosti $O(n^{1-\epsilon})$ pre ľubovoľnú hodnotu $\epsilon > 0$.

## 3.3 Farbenie vyvážene rozdeliteľných grafov

Na príkade planárnych grafov sme ukázali spôsob, ako využiť existenciu malého vrcholového separátora pre konštrukciu rýchlejšieho algoritmu na hľadanie farbiaceho čísla grafu. Už pri tejto konštrukcii sme potrebovali prejsť od vrcholového separátora k hranovému separátoru, ktorý má menej vrcholov a teda aj rôznych charakteristík, resp. prechodov. Ďalej túto myšlienku využijeme na konštrukciu algoritmu pre dobre rozdeliteľné grafy.

**Definícia 26.** *Nech $G$ je jednoduchý graf s $n$ vrcholmi, nech $A, B$ sú podmnožiny $V(G)$, nech $C = N(B)$ a $D = N(A)$. Dvojicu množín $(A, B)$ budeme volať* vyvážené rozdelenie grafu *$G$, ak platí:*

1. $A \cup B = V(G) \wedge A \cap B = \emptyset$
2. $|A| \leq \frac{2n}{3}$
3. $|B| \leq \frac{2n}{3}$
4. $|C \cup D| \leq \frac{n}{4}$

*Triedu grafov, v ktorých existuje vyvážené rozdelenie, budeme volať* vyvážene rozdeliteľné.

Pre ľubovoľný graf vieme overiť, či je vyvážene rozdeliteľný, v čase $O^*(2^n)$ a v prípade, že je vyvážene rozdeliteľný aj nájdeme jeho vyvážené rozdelenie. Pre každú z $2^n$ podmnožín vrcholov v polynomiálnom čase overíme, či môže byť množinou vo vyváženom rozdelení.

Pre ľubovoľnú podmnožinu $X \subseteq V(G)$ naozaj vieme v polynomiálnom čase overiť, či môže byť množinou vyváženého rozdelenia: Skontrolujeme, či má nanajvýš $\frac{2n}{3}$ a aspoň $\lceil \frac{n}{3} \rceil$ vrcholov. Ďalej pre každý vrchol v $X$ overíme, či nejaký z jeho susedov leží mimo $X$ a počet takýchto vrcholov označíme $c$. Podobne pre každý vrchol mimo $X$ overíme, či má suseda v $X$ a počet takýchto vrcholov označíme $d$. Ak platí $c + d \leq \frac{n}{4}$, tak dvojica $(X, V(G) - X)$ tvorí vyvážené rozdelenie grafu $G$.

Algoritmus pre hľadanie farbiaceho čísla vyvážene rozdeliteľných grafov budeme konštruovať podobne, ako algoritmus pre planárne grafy (s malým okolím separátora). Najprv si označíme objekty, s ktorými budeme pracovať. Graf budeme tradične označovať $G$, jeho počet vrcholov $n$. Množiny vyváženého rozdelenia $G$ označíme $A$ a $B$. Množinu $N(B)$ označíme $C$ a množinu $N(A)$ označíme $D$. Do pozornosti dávame, že platí $N(B) \subseteq A$ a $N(A) \subseteq B$. Separátor bude tvorený vrcholmi $C \cup D$ a zodpovedá hranovému separátoru $\{(u, v) \in E(G) \mid u \in C \wedge v \in D\}$, ktorý označíme $E$. Komponenty grafu $G - E$ budeme označovať $K_1 \ldots K_m$.

Základný postup bude nasledovný: Pre každú možnosť prechodu nad separátorom $C \cup D$ a každý komponent (nezávisle) vypočítame, aká množina charakteristík v ňom môže byť po pridaní novej farby. Pre tento výpočet budeme používať postup z algoritmu Junosza-Szaniawski.

Pre aplikovanie postupu z algoritmu Junosza-Szaniawski treba prerobiť charakteristiky na rozšírené charakteristiky. Rovnako treba pred spustením a po spustení algoritmu pre daný prechod separátora a daný komponent prefiltrovať množinu starých a nových charakteristík, aby sme odstránili tie, ktoré nie sú kompatibilné s prechodom separá-

tora. Tento problém vyriešime rovnako, ako v prípade planárnych grafov, v ktorých má separátor malé okolie.

Ďalej budeme odhadovať časovú zložitosť tohto algoritmu. Najzložitejšia časť odhadu sa bude týkať počtu rôznych prechodov v separátore, čiže počtu kombinácií starej a novej charakteristiky separátora.

### 3.3.1 Horný odhad počtu prechodov

Najprv dokážeme, že separátor vieme pokryť hviezdami. Potom odvodíme počet prechodov vo hviezde s $h$ vrcholmi a nakoniec zhora ohraničíme počet prechodov v našom separátore $C \cup D$.

**Definícia 27.** *Graf $G$ voláme* hviezda*, ak v ňom existuje vrchol $v$, ktorý je spojený hranou s každým iným vrcholom, a každá hrana v $G$ je incidentná s vrcholom $v$. Vrchol $v$ spĺňajúci tieto podmienky budeme volať* stred hviezdy*. Pre určený stred hviezdy budeme zvyšné vrcholy nazývať* ramená*.*

*Graf $G$ voláme* netriviálna hviezda*, ak má aspoň dva vrcholy a je hviezda.*

**Lema 28.** *Nech $G$ je súvislý graf s aspoň dvomi vrcholmi. Potom existuje množina jeho podgrafov $H_1, H_2, \ldots H_k$, ktorá spĺňa nasledujúce podmienky:*

1. $\forall i, j \in \{1 \ldots k\}, i \neq j : V(H_i) \cap V(H_j) = \emptyset$,
2. $\forall v \in V(G) \exists i \in \{1 \ldots k\} : v \in V(H_i)$,
3. $\forall i \in \{1 \ldots k\} : H_i$ *je netriviálna hviezda.*

*Množinu $H_1, H_2, \ldots, H_k$ budeme volať* pokrytie grafu $G$ hviezdami*.*

*Dôkaz.* Dokážeme indukciou vzhľadom na počet vrcholov grafu, $n$.

Báza indukcie, $n = 2$: Existuje iba jeden dvojvrcholový súvislý graf: obsahuje dva vrcholy prepojené hranou. Ľahko vidíme, že tento graf je hviezda a oba vrcholy môžu byť jej stredom.

Indukčný krok, $n \to n + 1$: V každom súvislom grafe $H$ existuje vrchol $u$ taký, že graf $H - u$ je súvislý. Nech $v$ je takýto vrchol v našom grafe $G$ a nech $H_1 \ldots H_k$ sú podgrafy $G - v$, ktoré spĺňajú podmienky lemy a ktorých existencia vyplýva z indukčného predpokladu. Keďže $G$ je súvislý graf s aspoň tromi vrcholmi, musí v ňom

existovať aspoň jedna hrana $e$ medzi vrcholom $v$ a nejakým iným vrcholom $u$. Bez ujmy na všeobecnosti platí $u \in V(H_1)$.

Ďalej rozoberieme niekoľko možností podľa toho, ako vyzerá hviezda $H_1$:

$H_1$ **má dva vrcholy:** V tomto prípade môže byť stredom hviezdy ľubovoľný z vrcholov $H_1$. Preto graf $H_1 \cup \{e, v\}$ je hviezda s vrcholom $u$. Vyhovujúcou množinou pre lemu je teda $H_1 \cup \{e, v\}, H_2, \ldots, H_k$.

$H_1$ **má aspoň tri vrcholy a** $u$ **je jeho stred:** Pokiaľ pripojíme k stredu hviezdy ďalší vrchol, dostaneme opäť hviezdu, preto $H_1 \cup \{e, v\}, H_2, \ldots H_k$ je množina hviezd spĺňajúca lemu.

$H_1$ **má aspoň tri vrcholy a** $u$ **nie je jeho stred:** Keď z hviezdy odoberieme vrchol, ktorý nie je jej stredom, dostaneme opäť hviezdu. Keďže $H_1$ má aspoň tri vrcholy, $H_1 - u$ je netriviálna hviezda. Zároveň graf $H$ tvorený vrcholmi $u, v$ a hranou $e$ je netriviálna hviezda. Preto množina hviezd $H_1 - u, H_2, \ldots, H_k, H$ je množina hviezd spĺňajúca podmienky lemy.

□

Ďalej budeme odvádzať vzťah pre počet možných prechodov medzi charakteristikami hviezd v závislosti od veľkosti hviezdy. Počet možných prechodov vieme zhora odhadnúť ako súčin medzi počtami pre jednotlivé hviezdy. Preto nás bude pre hviezdu s $h$ vrcholmi zaujímať hlavne $h$-ta odmnocnina z jeho počtu prechodov. Uvidíme, že najvyššiu $h$-tu odmocninu budú mať práve dvojvrcholové hviezdy. Najprv vypíšeme všetky prechody pre dvojvrcholové hviezdy a potom dokážeme všeobecný vzťah pre hviezdy s viacerými vrcholmi.

### Prechody nad hviezdami

Dvojvrcholová hviezda je tvorená dvomi vrcholmi, ktoré sú prepojené hranou. Charakteristiky budeme pre prehľadnosť vypisovať ako dvojice čísel. Ešte neofarbený vrchol (s charakteristikou 0), môže dostať novú farbu $k+1$ (charakteristika $\bar{1}$), ak jeho sused doteraz nemal farbu $k$ (charakteristika $\bar{1}$). Sused, ktorý mal farbu $k$ (charakteristika $\bar{1}$)

nebude mať najnovšiu farbu $k+1$, preto sa mu zmení charakteristika na 1.

$$(0,0) \to (0,0) \mid (0,\bar{1}) \mid (\bar{1},0)$$
$$(0,1) \to (0,1) \mid (\bar{1},1)$$
$$(0,\bar{1}) \to (0,1)$$
$$(1,0) \to (1,0) \mid (1,\bar{1})$$
$$(1,1) \to (1,1)$$
$$(1,\bar{1}) \to (1,1)$$
$$(\bar{1},0) \to (1,0)$$
$$(\bar{1},1) \to (1,1)$$

**Lema 29.** *Nech $H$ je netriviálna hviezda so stredom $u$ a s $x$ ramenami. Potom je počet prechodov medzi charakteristikami na $H$ presne $2^{x+1}\left(2+x+\frac{x(x-1)}{4}\right)$.*

*Dôkaz.* Rozoberieme všetky možnosti. Pokiaľ má vrchol $u$ charakteristiku $\bar{1}$, žiadne z ramien nemohlo mať charakteristiku $\bar{1}$ a tiež nemohlo dostať novú farbu. Preto z každej charakteristiky, v ktorej má vrchol $u$ hodnotu $\bar{1}$, existuje len jeden prechod. Pre každé z $x$ ramien máme dve nezávislé možnosti ich hodnoty v charakteristike: 1 a 0. Preto je takýchto možností $2^x$.

Pokiaľ má vrchol $u$ charakteristiku 1, rozoberieme 4 prípady podľa toho, či má niektoré rameno hodnotu v starej charakteristike $\bar{1}$ a či má niektoré rameno hodnotu v novej charakteristike $\bar{1}$. Keďže vzdialenosť každej dvojice ramien je 2, v žiadnej charakteristike nemôžu mať dve ramená priradenú zároveň hodnotu $\bar{1}$. Rozoberanie možností:

**Stará aj nová charakteristika obsahuje rameno s hodnotou $\bar{1}$:** Spomedzi ramien môžeme nezávisle a usporiadane vybrať dve, jedno rameno bude mať v starej charakteristike hodnotu $\bar{1}$ a jedno rameno v novej charakteristike. Takýchto výberov je $x(x-1)$. Pre zvyšných $x-2$ ramien máme $2^{x-2}$ možností. $\rightsquigarrow x(x-1)2^{x-2}$

**Stará charakteristika obsahuje rameno s hodnotou $\bar{1}$, nová nie:** Máme $x$ možností, ktoré rameno má charakteristiku $\bar{1}$. Pre zvyšných $x-1$ ramien máme $2^{x-1}$ možností. $\rightsquigarrow x2^{x-1}$

**Stará charakteristika neobsahuje, nová obsahuje rameno s hodnotou $\bar{1}$:** Rovnako ako v predošlom prípade máme $x$ možností pre výber význačného ramena a $2^{x-1}$ možností pre zvyšné. $\rightsquigarrow x2^{x-1}$

**Ani jedna charakteristika neobsahuje rameno s hodnotou $\bar{1}$:** Máme $2^x$ možností pre farby ramien. $\rightsquigarrow 2^x$.

Pokiaľ má vrchol $u$ charakteristiku 0, máme dve základné možnosti. Ak má po prechode tiež charakteristiku 0, máme presne rovnako veľa možností, ako keď mal vrchol $u$ starú charakteristiku 1. Ak má po prechode charakteristiku $\bar{1}$, máme rovnako veľa možností, ako keď mal vrchol $u$ starú charakteristiku $\bar{1}$.

Sčítaním a úpravou dostaneme:

$$2(2^x + x(x-1)2^{x-2} + x2^{x-1} + x2^{x-1} + 2^x) = 2^{x+1}\left(2 + x + \frac{x(x-1)}{4}\right).$$

□

**Dôsledok 30.** *Nech $H$ je netriviálna hviezda s $m$ vrcholmi a nech $p$ je jej celkový počet prechodov. Potom platí $\sqrt[m]{p} \leq \sqrt{12}$.*

*Dôkaz.* Podľa predošlej lemy má hviezda s $m$ vrcholmi presne $p = 2^m(2 + m - 1 + \frac{(m-1)(m-2)}{4})$. Pre $m \geq 2$ platí $2 + m - 1 + \frac{(m-1)(m-2)}{4} \leq m^2$. Zaujíma nás hodnota $\sqrt[m]{p}$, ktorej horný odhad je $\sqrt[m]{2^m m^2} = 2m^{\frac{2}{m}}$.

Funkcia $m^{\frac{2}{m}}$ je klesajúca pre $m \geq e$ a pre $m = 8$ je hodnota výrazu $2m^{\frac{2}{m}}$ menšia ako $\sqrt{12}$. Pre $m = 2$ je hodnota $p$ presne 12, čiže $\sqrt[m]{p} = \sqrt{12}$. Pre $2 \leq m \leq 7$ uvedieme tabuľku približných hodnôt pôvodnej funkcie $\sqrt[m]{2 + m - 1 + \frac{(m-1)(m-2)}{4}}$, čím zavŕšime dôkaz:

| $m =$ | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| $\sqrt[m]{p} =$ | 3.4641 | 3.30193 | 3.19344 | 3.10369 | 3.02617 | 2.95854 |

□

**Dôsledok 31.** *Ľubovoľný graf $G$ s $n$ vrcholmi a bez izolovaných vrcholov má nanajvýš $12^{\frac{n}{2}}$ prechodov.*

*Dôkaz.* Nech $K_1 \dots K_m$ sú komponenty grafu $G$. Pre každý z nich existuje pokrytie hviezdami, ktoré sa neprekrývajú. Keď zjednotíme pokrytie hviezdami pre každý komponent, dostaneme pokrytie hviezdami grafu $G$. Hviezdy v tomto pokrytí označíme $H_1 \dots H_k$ a ich počty vrcholov označíme $n_1 \dots n_k$ v tomto poradí.

Keď z grafu odstránime nejakú hranu, všetky pôvodné prechody ostanú platné. To znamená, že graf zložený z hviezd $H_1 \dots H_k$ má aspoň toľko prechodov, ako graf $G$. Každá dvojica rôznych hviezd je nezávislá, preto počet prechodov na grafe zloženom z hviezd $H_1 \dots H_k$ je presne $\prod_{i=1}^{k} pr(H_i)$. Nakoniec s použitím predošlého dôsledku 30 odvodíme tvrdenie:

$$pr(G) \leq \prod_{i=1}^{k} pr(H_i) \leq \prod_{i=1}^{k} \sqrt{12}^{n_i} = \sqrt{12}^{\left(\sum_{i=1}^{k} n_i\right)} = \sqrt{12}^{n} = 12^{\frac{n}{2}}.$$

□

### 3.3.2 Časová zložitosť algoritmu

Z definície množín $C$ a $D$ v algoritme musí platiť, že každý vrchol v $C$ má suseda v $D$ a taktiež každý sused v $D$ má suseda v $C$. To znamená, že podgraf grafu $G$ indukovaný množinou $C \cup D$ nemá izolované vrcholy a teda má nanajvýš $12^{\frac{c+d}{2}}$ prechodov. Teraz máme pripravené podklady pre horný odhad časovej zložitosti algoritmu.

Komponentov grafu $G - E$ je nanajvýš $n$ a každý z nich má nanajvýš $\frac{2n}{3}$ vrcholov. Pre každý z nanajvýš $12^{\frac{n}{8}}$ prechodov a pre každý z nanajvýš $n$ komponentov budeme počítať množinu charakteristík, ktorá vznikne pridaním novej farby v čase $O^*\left(2.65^{\frac{2n}{3}}\right)$. Pri každej kombinácii prechodu a komponentu dostaneme množinu nanajvýš $O^*\left(2.65^{\frac{2n}{3}}\right)$ charakteristík. Pre každú z nich v polynomiálnom čase overíme, či je konzistentná s prechodom separátora.

Časová zložitosť tohto algoritmu je teda zhora ohraničená na $O^*\left(2.65^{\frac{2n}{3}} \cdot 12^{\frac{n}{8}}\right) = O^*(2.613^n)$.

Podobne, ako pri planárnych grafoch, aj pri vyvážene rozdeliteľných grafoch je možné prispôsobovať konštanty. Ak by sme napríklad požadovali, že množiny $A$ a $B$ majú obe nanajvýš $\frac{n}{2}+o(n)$ vrcholov, a povolili by sme väčšie množiny $C$ a $D$, napr. aby dohromady mali nanajvýš $\frac{n}{3}$ vrcholov, dostali by sme algoritmus s časovou zložitosťou $O^*(2.464^{n+o(n)})$.

# Kapitola 4

# Experiment na 2-hranovo súvislých grafoch

V prvej kapitole sme sa venovali konštrukcii algoritmu, ktorý dokáže z efektívneho riešenia problému $L(2,1)$-farbenia na chlpatých 2-hranovo-súvislých grafoch vyskladať efektívne riešenie problému $L(2,1)$-farbenia na všeobecných grafoch.

Najefektívnejšie riešenie problému $L(2,1)$-farbenia má časovú zložitosť zhora ohraničenú počtom vlastných párov na súvislých grafoch. Pripomenieme, že vlastný pár grafu $G$ je dvojica $(S,X)$ podmnožín vrcholov grafu $G$, kde $S$ je 2-pakovanie a $S \cap X = \emptyset$. V tejto kapitole sa budeme venovať experimentálnemu odhadu počtu vlastných párov na 2-hranovo súvislých grafoch.

Cieľom experimentu bolo odhaliť, aký typ 2-hranovo súvislých grafov má najviac vlastných párov.

## 4.1 Konštrukcia 2-hranovo súvislých grafov

Najprv popíšeme a dokážeme spôsob, ktorým vieme skonštruovať všetky 2-hranovo súvislé grafy. Bude sa nápadne ponášať na konštrukciu 2-súvislých grafov. Každý 2-hranovo súvislý graf sa dá vytvoriť z kružnice pridávaním $H$-ciest do existujúceho grafu [5]. Tento postup vieme rozšíriť na 2-hranovo súvislé grafy, ak dovolíme okrem $H$-ciest pridávať aj $H$-kružnice.

**Definícia 32.** *Nech $H$ je graf. Cestu $P$ budeme nazývať $H$-cesta, ak prienik $P$ a $H$ tvoria práve koncové vrcholy $P$.*

*Kružnicu $C$ budeme nazývať $H$-kružnica, ak prienik $C$ a $H$ tvorí práve jeden vrchol.*

**Veta 33.** *Každý 2-hranovo súvislý graf $G$ sa dá vytvoriť z kružnice postupným pridávaním $H$-ciest a $H$-kružníc do grafov $H$, ktoré sme už skonštruovali.*

*Dôkaz.* Najprv dokážeme, že $G$ obsahuje kružnicu ako podgraf. Graf $G$ je 2-hranovo súvislý, preto pre ľubovoľnú hranu $(uv) = e \in E(G)$ platí, že graf $G - e$ je súvislý. Nájdeme cestu $P$ medzi vrcholmi $u$ a $v$ v grafe $G-e$. Cesta $P$ doplnená o hranu $e$ tvorí kružnicu.

Ďalej budeme postupovať sporom, graf $G$ sa nedá zostrojiť popísaným spôsobom. Graf $G$ obsahuje kružnicu a teda existuje nejaký podgraf grafu $G$, ktorý sa dá týmto spôsobom skonštruovať. Označme $H$ ako najväčší (vzhľadom na inklúziu) takýto podgraf.

Ak by existovala hrana $(uv) \in E(G) - E(H)$, ktorej koncové vrcholy $u, v$ ležia v $H$, tak $(uv)$ tvorí $H$-cestu a teda je v spore s maximálnosťou $H$. To znamená, že $H$ je indukovaný podgraf.

Graf $G$ je súvislý a teda v ňom musí existovať hrana medzi nejakou dvojicou vrcholov $u, v$, kde $u \in V(G) - V(H)$ a $v \in V(H)$. Z predpokladu, že $G$ je 2-hranovo súvislý vieme, že $G - (uv)$ je súvislý a teda v ňom existuje nejaká cesta $P$ medzi vrcholom $u$ a $v$. Nech $w$ je posledný vrchol na ceste $P$ z množiny $V(G) - V(H)$ a $P_w$ je časť cesty $P$ od vrchola $w$ do $v$. Cesta $P_w$ s hranou $(uv)$ tvorí $H$-cestu, ak $w \neq u$, alebo $H$-kružnicu, ak $w = u$.

To znamená, že pridaním cesty $P_w$ s hranou $(uv)$ do grafu $H$ dostaneme väčší graf, ktorý vieme skonštruovať popísaným postupom. To je však v spore s predpokladom, že $H$ je maximálny podgraf, ktorý vieme týmto postupom skonštruovať. □

Za zmienku stojí fakt, že dôkaz vety nám dáva spôsob, ako pre ľubovoľný graf určiť postupnosť operácií, ktorými ho vieme vytvoriť.

Množinu všetkých 2-hranovo súvislých grafov s daným počtom vrcholov teda vieme skonštruovať nasledovne: Najprv určíme, aký veľký je prvý cyklus a pre každú túto možnosť spustíme rekurzívnu procedúru, ktorá pridáva $H$-kružnice a $H$-cesty.

Rekurzívna procedúra dostane ako parameter nejaký 2-hranovo súvislý graf $H$ a počet vrcholov $k$, ktoré treba ešte do grafu pridať.

Pridávanie $H$-ciest bude fungovať nasledovne: Pre každý možný výber dvojice rôznych vrcholov $u, v$ a každú možnosť počtu nových vrcholov $p \leq k$ vytvorí graf $H'$. Graf

$H'$ vznikne z grafu $H$ pridaním cesty s $p$ novými vrcholmi medzi vrcholy $u, v$. Nakoniec rekurzívne spustí túto procedúru s parametrami $H'$ a $k - p$.

Pridávanie $H$-cyklov bude fungovať podobne: Pre každý vrchol $v$ a pre každý možný počet nových vrcholov $2 \leq p \leq k$ skonštruujeme graf $H'$. Graf $H'$ vznikne z grafu $H'$ pridaním cyklu s $p + 1$ vrcholmi, kde jeden z nich je $v$. Na graf $H'$ rekurzívne spustíme procedúru s parametrami $H'$ a $k - p$.

Takto popísaný algoritmus pre konštrukciu 2-hranovo súvislých grafov je nepraktický, lebo väčšinu 2-hranovo súvislých grafov vieme podľa predošlého postupu skonštruovať viackrát. Pri skúmaní počtu vlastných párov nás nezaujíma označenie vrcholov. Preto stačí, ak každý neoznačený 2-hranovo súvislý graf skonštruujeme aspoň raz.

Na reprezentáciu vrcholov používame čísla $0 \ldots n - 1$, na reprezentáciu hrán používame maticu susednosti.

V ďalšej časti si popíšeme viacero spôsobov, ktorými vieme zredukovať množstvo vygenerovaných grafov. Postupne tak načrtneme základný algoritmus a 5 zlepšení, ktoré nakoniec medzi sebou porovnáme.

## 4.2 Redukcia počtu grafov

Pripomeňme si viackrát využitý fakt: odobratím ľubovoľnej hrany z grafu sa jeho počet vlastných párov nezníži. Ak nás teda zaujímajú grafy s najväčším počtom vlastných párov, môžeme sa obmedziť na také, ktoré nemajú hrany navyše. Z pohľadu nášho algoritmu na vytváranie 2-hranovo súvislých grafov to znamená, že stačí pridávať iba také $H$-cesty, ktoré obsahujú aspoň jeden nový vrchol.

Ďalší jednoduchý spôsob, ako vieme zmenšiť počet vygenerovaných grafov je nasledovný: $H$-cestu medzi vrcholy $u, v$ pridáme iba vtedy, ak nejde o susedné vrcholy. Dôvod si zhrnieme v nasledujúcej leme.

**Lema 34.** *Nech $H$ je 2-hranovo súvislý graf, nech $u, v$ sú vrcholy v $H$ spojené hranou $e$. Nech graf $H'$ vznikne z $H$ pridaním cesty $P$ medzi vrcholy $u, v$. Potom $H' - e$ je 2-hranovo súvislý graf.*

*Dôkaz.* Dokážeme, že pre ľubovoľnú hranu $f \in E(H - e)$ platí, že graf $H' - \{e, f\}$ je súvislý. Graf $H$ je 2-hranovo súvislý, čiže $H - e$ je súvislý. Zároveň je $H - e$ podgrafom $H' - e$.

Pokiaľ vezmeme hranu $f$ z cesty $P$, tvrdenie platí, lebo pre každý nový vrchol v $P$ existujú dve hranovo disjunktné cesty do grafu $H - e$ a graf $H - e$ je súvislý.

Pre hrany v grafe $H - e$ použijeme Mengerovu vetu. Pre každú dvojicu vrcholov v 2-hranovo súvislom grafe existujú dve hranovo disjunktné cesty [5].

Vezmime ľubovoľné dva vrcholy $u, v$ a dve hranovo disjunktné cesty $P_1, P_2$ medzi nimi. Nanajvýš jedna z ciest obsahuje hranu $e$. Ak žiadna z nich neobsahuje hranu $e$, tak $P_1$ a $P_2$ je dvojica hranovo disjunktných ciest medzi $u, v$. Ak jedna z nich hranu obsahuje, bez ujmy na všeobecnosti je to $P_1$, tak túto hranu nahradíme cestou $P$.

Pre ľubovoľnú dvojicu vrcholov $u, v \in V(H)$ teda platí, že existuje dvojica hranovo disjunktných ciest medzi $u$ a $v$ v grafe $H' - e$. Ak pre súvislý graf platí, že $G - (xy)$ je nesúvislý, vrcholy $x$ a $y$ musia ležať v rôznych komponentoch súvislosti $G - (xy)$. Keďže ľubovoľná dvojica vrcholov vo $V(H)$ je spojená dvomi hranovo disjunktnými cestami v $H' - e$, hrana $(xy) \in E(H) - \{e\}$ patrí nanajvýš do jednej z týchto ciest. Vrcholy $x, y$ teda ležia v tom istom komponente súvislosti grafu $H' - \{e, (xy)\}$ a teda graf $H' - \{e, (xy)\}$ je súvislý.

Nech zvolíme ľubovoľnú hranu $e' \in E(H' - e)$, graf $H' - \{e, e'\}$ je súvislý. Graf $H' - e$ teda spĺňa definíciu hranovej 2-súvislosti. □

Obe tieto zlepšenia implementujeme už v základnom programe. Ďalej budeme redukovať symetrie na základe najväčšieho cyklu.

### 4.2.1 Najväčší cyklus v grafe

Prvý spôsob odstraňovania symetrií sa bude týkať najväčšieho cyklu v grafe. Ľubovoľný 2-hranovo súvislý graf $G$ môžeme našim postupom vypestovať tak, že najväčší cyklus v $G$ bude zodpovedať cyklu, ku ktorému pridávame $H$-cesty a $H$-kružnice. Našej funkcii teda pridáme číselný parameter $c$, ktorý bude popisovať najväčší cyklus, ktorý sa môže nachádzať v pestovanom grafe.

Pri pridávaní $H$-cyklu môžeme pridať nanajvýš $c-1$ nových vrcholov, ináč vyrobíme väčší cyklus, ako $c$. Pri pridávaní $H$-cesty medzi vrcholy $u, v$ môžeme pridať iba $c - l(u, v) - 1$ nových vrcholov, kde $l(u, v)$ označuje dĺžku ľubovoľnej cesty medzi $u, v$. Pri prvom zlepšení budeme ako $l(u, v)$ používať vzdialenosť $d(u, v)$, lebo sa dá rýchlo počítať.

V každom behu rekurzívnej funkcie teda pre každu dvojicu už existujúcich vrcholov

spočítame ich vzdialenosť Floyd-Warshallovym algoritmom, ktorý na grafe s $n$ vrcholmi pracuje v čase $O(n^3)$.

Ako poznámku pridáme, že v skutočnosti náš druhý algoritmus nebol tesný, lebo pri pridávaní $H$-cyklu sme mohli pridať až $c$ vrcholov a pri pridaní $H$-cesty medzi vrcholy až $c - d(u, v)$ vrcholov.

### 4.2.2 Minimálne 2-hranovo súvislý graf

V treťom algoritme naplno využijeme vlastnosť, že odstránenie hrany nezmenšuje počet vlastných párov. Zaujímať nás budú teda také grafy $H$, v ktorých pre každú hranu $e$ platí, že graf $H - e$ nie je 2-hranovo súvislý. Takéto grafy sa nazývajú *minimálne 2-hranovo súvislé*.

Ak niektorý 2-hranoco súvislý podgraf $G \subseteq H$ nie je minimálne 2-hranovo súvislý, tak ani graf $H$ nemôže byť 2-hranovo súvislý [4]. Ak teda v ktoromkoľvek volaní rekurzívnej funkcie dostaneme graf $H$, ktorý nie je minimálne 2-súvislý, nemá zmysel s ním naďalej pracovať.

Overovať, či je graf minimálne 2-hranovo súvislý, budeme nasledovne: Pre každú hranu $e \in E(H)$ pomocou Tarjanovho algoritmu overíme, či graf $H - e$ obsahuje most. Ak pre niektorú hranu graf $H - e$ neobsahuje most, tak $H$ nie je 2-hranovo súvislý. Časová zložitosť kontroly je $O(m^2)$.

Toto zlepšenie výrazne zredukovalo počet vytvorených grafov, ako uvidíme v tabuľke.

### 4.2.3 Usporiadané pridávanie ciest a cyklov

V ľubovoľnom postupe vytvárania 2-hranovo súvislého grafu môžeme operácie pridávania $H$-ciest a $H$-kružníc vhodne usporiadať. Využijeme, že v našej reprezentácii máme vrcholy očíslované číslami $0 \ldots n - 1$. Pridanie $H$-cesty medzi vrcholy s číslami $u, v$, kde $u > v$, vieme symbolicky reprezentovať ako dvojicu čísel $(u, v)$. Podobne pridanie $H$-cyklu pripojeného k vrcholu s číslom $u$ môžeme reprezentovať ako dvojicu čísel $(u, u)$.

Pri ľubovoľnom postupe vytvárania grafu môžeme dve operácie s reprezentáciami (v poradí) $X = (x_1, x_2)$ a $Y = (y_1, y_2)$ vymeniť, ak je $Y$ lexikograficky menšia ako $X$. Ak pri vykonávaní operácie $X$ graf obsahoval vrchol $x_1$, tak musel obsahovať aj menší

(rovný) vrchol $y_1$ a menší (rovný) vrchol $y_2$.

Z tohto pozorovania vyplýva náš štvrtý algoritmus: Do rekurzívnej funkcie, ktorá vytvára grafy, pridáme ďalší parameter: Najväčšiu doterajšiu reprezentáciu pridania $H$-cesty alebo $H$-kružnice. Pre dané volanie rekurzívnej funkcie budeme pridávať iba také $H$-cesty a $H$-kružnice, ktoré majú väčšiu alebo rovnakú reprezentáciu.

### 4.2.4 Najväčší cyklus v grafe znova

Ďalšie zlepšenie sa bude opäť týkať najväčšieho cyklu. Pre vopred stanovené $c$ môžeme medzi vrcholy pridať nanajvýš $c-l(u,v)-1$ nových vrcholov, kde $l(u,v)$ označuje dĺžku nejakej cesty. Pri tomto zlepšení budeme budeme $l(u,v)$ počítať ako dĺžku najdlhšej cesty medu $u$ a $v$.

Zisťovanie dĺžky najdlhšej cesty medzi dvomi vrcholmi je NP-ťažký problém. Popíšeme riešenie v čase $O(n^3 2^n)$, ktoré používa dynamické programovanie. Pre každý z $n$ vrcholov spočítame najdlhšiu cestu z neho do každého iného vrchola. Ďalej popíšeme, ako vyzerá hľadanie najdlhšej cesty z vrchola $u$ do ostatných.

Podproblém bude vyzerať nasledovne: Pre daný vrchol $v$ a nejakú podmnožinu dovolených vrcholov $S$, aká je najdlhšia cesta z $v$ do $u$, ak môžeme použiť iba vrcholy v $S$?

Pre $v = u$ je odpoveď 0. Pre $v \neq u$ sa pozrieme na všetkých susedov $w$ vrcholu $v$, ktorí sú v množine $S$. Pre každého z nich spočítame najdlhšiu cestu z vrcholu $w$ do vrcholu $u$, ktorá môže použiť iba vrcholy v $S-\{v\}$. Nech najväčšiu hodnotu nadobúdal nejaký vrchol $w_{max}$. Potom odpoveď pre $v, S$, je o jedna väčšia, ako odpoveď pre $w_{max}, S-\{v\}$. Najdlhšia cesta z vrcholu $x$ do $u$ potom zodpovedá riešeniu problému $x, 2^{V(G)}$.

Stavov je $n2^n$, pre každý z nich vyskúšame $O(n)$ možností. Celé riešenie s dynamickým programovaním budeme spúšťať $n$-krát. Preto je časová zložitosť $O(n^3 2^n)$.

Všetky doteraz popísané zlepšenia sme vedeli vypočítať v polynomiálnom čase od veľkosti grafu. Toto zlepšenie má však exponenciálnu časovú zložitosť od počtu vrcholov. Ukázalo sa, že ho nie je dobré aplikovať úplne v každom kroku výpočtu. Čas, ktorý strávime počítaním najdlhšej cesty v grafe, preváži čas, ktorý ušetríme orezaním niektorých vetiev výpočtu.

Ukázalo sa však, že ak budeme hľadať najdlhšiu cestu pre všetky grafy $H$, ktoré

potrebujú doplniť aspoň 3 nové vrcholy a pre zvyšné, veľké grafy, budeme používať najkratšie cesty, celkovo ušetríme zhruba polovicu času. Tento empirický fakt je, pochopiteľne, závislý od časovej náročnosti operácie, ktorú na grafoch vykonávame. Pre rôzne výpočty nad grafmi teda môže byť výhodnejšie počítať najdlhšie cesty na menších, alebo aj väčších grafoch.

### 4.2.5 Symetria prvého cyklu

Posledné vylepšenie sa bude týkať úplne prvej kružnice. Prvé pridanie $H$-cesty alebo $H$-kružnice sa musí vykonať na niektorom vrchole, resp. dvojici vrcholov z prvej kružnice. Nech je postupnosť pridávania $H$-ciest a $H$-cyklov ľubovoľná, môžeme ju otočiť tak, aby jeden z vrcholov kružnice, ku ktorej pridávame $H$-cestu alebo $H$-kružnicu, bol vrchol s číslom 0.

## 4.3 Počítanie vlastných párov

Na počítanie vlastných párov využijeme vzorec $pp(G) = \sum_{\substack{S \in \mathcal{P}(V_G) \\ S \text{ je 2-pakovanie}}} 2^{n-|S|}$. Pre každé $x$ vypočítame počet 2-pakovaní, ktoré majú presne $x$ vrcholov.

Najrv pre každý vrchol $v \in V(G)$ vypočítame zoznam vrcholov, ktoré sú od neho vzdialené nanajvýš 2. Tento zoznam pre vrchol $v$ označíme $N^2[v]$. Ďalej budeme generovať všetky 2-pakovania grafu $G$ rekurzívnou funkciou.

Naša rekurzívna funkcia bude mať dva parametre, číslo aktuálneho vrcholu $i$ a zoznam zakázaných vrcholov $S$. Naša funkcia sa vždy rekurzívne zavolá s parametrami $i+1$ a $S$, čo zodpovedá situácii, keď vrchol $i$ nepridáme do 2-pakovania. Ak vrchol $i$ nie je v ozname $S$, tak navyše rekurzívne spustíme funkciu s parametrom $i$ a $S \cup N^2[i]$.

Každé volanie rekurzívnej funkcie vedie k nejakému 2-pakovaniu. Pozrime sa, koľko operácií sme spravili pri postupnosti výpočtov, ktoré viedli k danému 2-pakovaniu. V každom z presne $n$ rekurzívnych volaní sme vykonali buď $O(1)$ operácií, ak sme daný vrchol nevzali do 2-pakovania, alebo $O(n)$ operácií, ak sme daný vrchol zobrali do 2-pakovania a pridávali jeho okolie do zoznamu $S$. Časová zložitosť hľadania všetkých 2-pakovaní grafu $G$ je teda $O(n^3 + n^2 P(G))$, kde $P(G)$ je počet 2-pakovaní v grafe $G$.

### 4.3.1 Vlastné páry na cestách a kružniciach

Pre dostatočne jednoduché triedy grafov vieme počet vlastných párov v závislosti od veľkosti grafu vyjadriť cez rekurentný vzťah. Najprv ukážeme, ako sa dá vypočítať počet vlastných párov na ceste s $n$ vrcholmi a následne pomocou neho vyjadríme počet vlastných párov na kružnici s $n$ vrcholmi. Z rekurentného vzťahu nakoniec odvodíme asymptotický odhad pre počet vlastných párov na cestách a kružniciach.

Počet vlastných párov cesty s $n$ vrcholmi označíme $p_n$. Pre $n > 3$ vieme rekurentný vzťah vyrobiť nasledovne: Pozrieme sa na koncový vrchol $v$ cesty $P$. Vlastné páry $(S, X)$ rozdelíme na tie, v ktorých vrchol $v$ patrí do 2-pakovania $S$ a tie, v ktorých nepatrí.

Všetky vlastné páry prvého druhu majú nasledovný tvar: Sused $u$ vrchola $v$ ani sused $w$ vrchola $u$ nemôžu patriť do 2-pakovania $S$ a môžu nezávisle od seba patriť do $X$. Každú z týchto štyroch možností môžeme nezávisle doplniť vlastnými pármi na kratšej ceste $P - \{u, v, w\}$. Vlastných párov, kde $v$ patrí do $S$, je teda $4p_{n-3}$.

Všetky vlastné páry druhého druhu rozdelíme na tie, kde $v$ patrí do množiny $X$ a tie, kde $v$ do množiny nepatrí. Každú z týchto možností možeme nezávisle doplniť ľubovoľným vlastným párom na ceste $P - v$. Vlastných párov, kde $v$ nepatrí do $S$, je teda $2p_{n-1}$.

Počet vlastných párov kružnice s $n$ vrcholmi označíme $c_n$. Pre $n > 5$ použijeme nasledovný odhad: Nech $v$ je ľubovoľný vrchol na kružnici $C$, $u_1$ a $u_2$ sú susedia $v$ a $w_1$ je sused $u_1$ rôzny od $v$. Postupovať budeme podobne, ako pri cestách. Všetky vlastné páry $(S, X)$ si rozdelíme na tri disjunktné typy. Prvý typ spĺňa $v \in S$, druhý typ spĺňa $u_1 \in S$ a tretí typ spĺňa $v \notin S \wedge u_1 \notin S$.

Pre cyklus s aspoň piatimi vrcholmi platí, že pre ľubovoľný vrchol existujú práve štyri ďalšie, ktoré sú od neho vo vzdialenosti nanajvýš 2. Pri prvom aj druhom type vlastných párov teda ostanú štyri vrcholy, ktoré nemôžu patriť do $S$ a môžu nezávisle od seba patriť, alebo nepatriť do $X$. Pre každú možnosť ostane cesta s $n - 5$ vrcholmi, ktorej koncové vrcholy sú vzdialené viac ako 2 a teda sú nezávislé. To znamená, že vlastných párov prvého aj druhého typu je $2 \cdot 2^4 \cdot p_{n-5}$.

Pre každý z vlastných párov tretieho typu môžemú vrcholy $v$ a $u_1$ nezávisle patriť, alebo nepatriť do množiny $X$. Vrcholy $w_1$ a $u_2$ sú vo vzdialenosti 3, a teda pre každú možnosť môžeme doplniť nezávisle $p_{n-2}$ vlastnými pármi cesty $C - \{v, u_1\}$.

Počet vlastných párov na cestách teda spĺňa rekurenciu $p_n = 2p_{n-1} + 4p_{n-3}$ pre $n \geq 4$ a počet vlastných párov na kružniciach spĺňa vzťah $c_n = 32p_{n-5} + 4p_{n-2}$.

Charakteristický polynóm rekurencie $p_n$ je $p(x) = x^3 - 2x^2 - 4$. Polynóm $p$ má tri rôzne korene, z ktorých najväčší má hodnotu zhruba 2.5944. Veľkosť $p_n$ teda je v triede funkcií $O(2.5944^n)$. Podobne hodnota $c_n$ je zhora ohraničená konštantným násobkom $p_n$, preto $c_n \in O(2.5944^n)$.

## 4.4 Výsledky experimentu

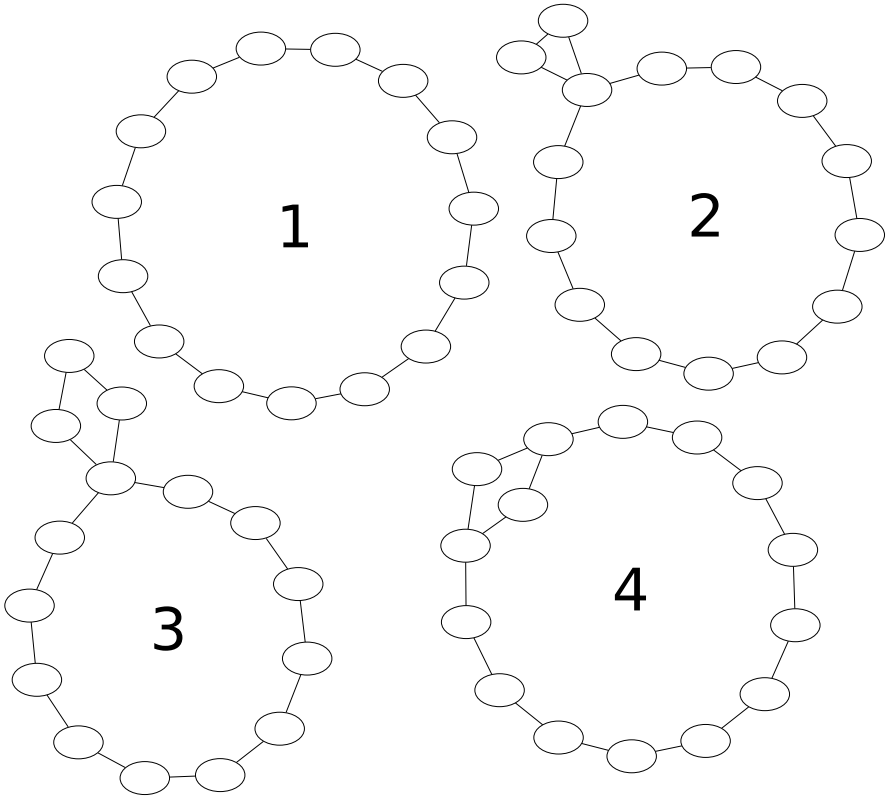
Otestovali sme grafy, ktoré majú nanajvýš 20 vrcholov. Pre každý počet vrcholov do 20 mala najviac vlastných párov kružnica. Štyri pätnásťvrcholové grafy s najväčším počtom vlastných párov sú znázornené na obrázku 4.1. Najväčší rozdiel v počte vlastných párov bol medzi kružnicou a druhým grafom v poradí, zhruba 9% z počtu vlastných párov kružnice. Rozdiel medzi druhým a tretím grafom v poradí tvoril zhruba 2% z počtu druhého grafu. U poslednej dvojice bol takýto rozdiel na úrovni 0.2%.

Ak by sme sa pozreli na ostatné veľkosti grafov, niekoľko grafov s najväčším počtom vlastných párov by tvorili analogické konštrukcie. Zdá sa teda, že najviac vlastných párov spomedzi 2-hranovo súvislých grafov majú kružnice, navyše s nezanedbateľným odstupom.

Ďalej uvedieme ešte tabuľku s počtami grafov, ktoré boli vygenerované postupne sa zlepšujúcimi algoritmami. Niektoré hodnoty nie sú vyplnené, lebo ich výpočet by trval príliš dlho.

| $n =$ | 1. algoritmus | 2. algoritmus | 3. algoritmus | 4. algoritmus | 5. algoritmus | 6. algoritmus |
|---|---|---|---|---|---|---|
| 5 | 8 | 8 | 8 | 6 | 6 | 3 |
| 6 | 78 | 78 | 30 | 12 | 12 | 5 |
| 7 | 1 256 | 1 213 | 153 | 53 | 53 | 21 |
| 8 | 30 392 | 28 227 | 751 | 148 | 143 | 53 |
| 9 | 1 041 728 | 922 767 | 4 611 | 597 | 526 | 200 |
| 10 | 48 272 012 | 40 511 279 | 29 943 | 2 077 | 1 638 | 601 |
| 11 | 2 915 065 368 | 2 310 399 982 | 219 390 | 8 434 | 5 989 | 2 228 |
| 12 | — | — | 1 738 810 | 32 980 | 20 176 | 7 372 |
| 13 | — | — | 15 188 850 | 137 670 | 74 175 | 27 132 |
| 14 | — | — | 145 265 192 | 574 765 | 263 028 | 94 977 |
| 15 | — | — | — | 2 479 948 | 971 402 | 348 777 |
| 16 | — | — | — | 10 829 224 | 3 542 688 | 1 257 944 |
| 17 | — | — | — | 48 283 823 | 13 173 428 | 4 642 477 |
| 18 | — | — | — | 218 323 671 | 48 852 235 | 17 043 272 |
| 19 | — | — | — | 1 003 214 946 | 183 037 711 | 63 331 223 |
| 20 | — | — | — | 4 674 538 111 | 686 891 813 | 235 416 186 |

Jedno z najvýraznejších zlepšení nastalo, keď sme začali priebežne kontrolovať, či sú




Obr. 4.1: Ukážka štyroch pätnásťvrcholových grafov, ktoré mali najviac vlastných párov.

naše už vytvorené grafy minimálne 2-hranovo súvislé. Ďalšie výrazné zlepšenie nastalo, keď sme vynútili usporiadané pridávanie $H$-ciest a $H$-cyklov.

Pre každý počet vrcholov do 20 bola grafom s najväčším počtom vlastných párov práve kružnica. Ďalšie skúmanie problému $L(2,1)$-farbenia sa teda môže zaoberať práve 2-hranovo súvislými grafmi a chlpatými 2-hranovo súvislými grafmi. Z nášho experimentu vyplývajú dva možné smery ďalšej práce na probléme $L(2,1)$-farbenia.

Jedným smerom by bola práca na dôkaze, že pre každý počet vrcholov má spomedzi 2-hranovo súvislých grafov najviac vlastných párov práve kružnica. Alternatívnym smerom skúania môže byť horný odhad maximálneho počtu vlastných párov na 2-hranovo súvislých grafov, ktorý by mohol byť pre $n$-vrcholový graf práve $O(2.5944^n)$.

# Záver

V našej práci sme sa venovali problému $L(2,1)$-farbenia. Popísali sme spôsob, ktorým môžeme problém $L(2,1)$-farbenia pre väčšinu prípadov zredukovať na problém $L(2,1)$-farbenia na chlpatých 2-hranovo súvislých grafoch. Taktiež sme ukázali konštrukciu algoritmu na planárnych grafoch s časovou zložitosťou $O^*(2.2^{n+o(n)})$ a konštrukciu algoritmu na vyvážene rozdeliteľných grafoch s časovou zložitosťou $O^*(2.613^n)$. Nakoniec sme ukázali algoritmus konštrukcie všetkých neoznačených minimálne 2-hranovo súvislých grafov.

Väčšina z predošlých efektívnych algoritmov riešiacich problém $L(2,1)$-farbenia využívala reprezentáciu čiastočných farbení pomocou ich charakteristík. Najväčší počet rôznych charakteristík, a teda aj najhoršiu časovú zložitosť, dosahovali dobre rozdeliteľné grafy. V našej práci sme využili rozdeliteľnosť na konštrukciu rýchlejšieho algoritmu pre problém $L(2,1)$-farbenia. Ďalší výskum problému $L(2,1)$-farbení a podobných problémov by sa teda mohol venovať grafom, ktoré nie sú vyvážene rozdeliteľné. Očakávame, že niektorý zo skorších algoritmov, napr. Junosza-Szaniawski, by sa dal prispôsobiť, aby na takýchto grafoch dosahoval lepšiu časovú zložitosť.

Ďalší možný smer výskumu je venovať sa $L(2,1)$-farbeniam 2-hranovo súvislých grafov, resp. chlpatých 2-hranovo súvislých grafov. Predpokladáme, že rýchly algoritmus pre problém $L(2,1)$-farbenia na tejto triede grafov by sa dal prispôsobiť, aby rýchlo riešil problém na všeobecných grafoch.

Nakoniec postup generovania neoznačených minimálne 2-hranovo súvislých grafov nám dáva možnosť ďalej študovať túto triedu grafov. Jedným z využití tohto postupu môže byť enumerácia neoznačených minimálne 2-hranovo súvislých grafov.

# Literatúra


[1] Michal Anderle. Analýza algoritmov pre L(2,1) farbenie grafov. Master's thesis, Univerzita Komenského v Bratislave, 2016.

[2] Hans L Bodlaender, Ton Kloks, Richard B Tan, and Jan Van Leeuwen. Approximations for $\lambda$-colorings of graphs. *The Computer Journal*, 47(2):193–204, 2004.

[3] Gerard J Chang and David Kuo. The L(2,1)-labeling problem on graphs. *SIAM Journal on Discrete Mathematics*, 9(2):309–316, 1996.

[4] Guy Chaty and Michel Chein. Minimally 2-edge connected graphs. *Journal of Graph Theory*, 3(1):15–22, 1979.

[5] Reinhard Diestel. *Graph theory*. Springer Publishing Company, Incorporated, 2017.

[6] Anna Dresslerová. L(2,1)-farbenia špeciálnych grafov. Master's thesis, Univerzita Komenského v Bratislave, 2016.

[7] Jerrold R Griggs and Roger K Yeh. Labelling graphs with a condition at distance 2. *SIAM Journal on Discrete Mathematics*, 5(4):586–595, 1992.

[8] Toru Hasunuma, Toshimasa Ishii, Hirotaka Ono, and Yushi Uno. An $O(n^{1.75})$ algorithm for L(2, 1)-labeling of trees. In *Scandinavian Workshop on Algorithm Theory*, pages 185–197. Springer, 2008.

[9] Toru Hasunuma, Toshimasa Ishii, Hirotaka Ono, and Yushi Uno. A linear time algorithm for L(2, 1)-labeling of trees. In *European Symposium on Algorithms*, pages 35–46. Springer, 2009.

[10] Toru Hasunuma, Toshimasa Ishii, Hirotaka Ono, and Yushi Uno. Algorithmic aspects of distance constrained labeling: a survey. *International Journal of Networking and Computing*, 4(2):251–259, 2014.

[11] Frédéric Havet, Martin Klazar, Jan Kratochvíl, Dieter Kratsch, and Mathieu Liedloff. Exact algorithms for L (2, 1)-labeling of graphs. *Algorithmica*, 59(2):169–194, 2011.

[12] Konstanty Junosza-Szaniawski, Jan Kratochvíl, Mathieu Liedloff, Peter Rossmanith, and Paweł Rzążewski. Fast exact algorithm for L (2, 1)-labeling of graphs. *Theoretical Computer Science*, 505:42–54, 2013.

[13] Angela Erika Koller. *The frequency assignment problem.* PhD thesis, Brunel University, School of Information Systems, Computing and Mathematics, 2004.

[14] Daniel Kráľ. An exact algorithm for the channel assignment problem. *Discrete Applied Mathematics*, 145(2):326–331, 2005.

[15] Richard J Lipton and Robert Endre Tarjan. A separator theorem for planar graphs. *SIAM Journal on Applied Mathematics*, 36(2):177–189, 1979.

[16] R Endre Tarjan. A note on finding the bridges of a graph. *Information Processing Letters*, 2(6):160–161, 1974.