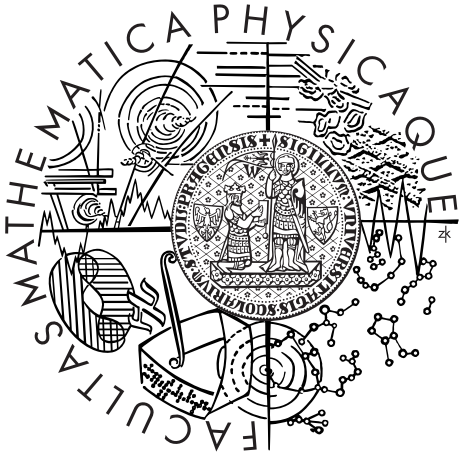
Univerzita Karlova v Praze
Matematicko-fyzikální fakulta

DIPLOMOVÁ PRÁCE



Lucie Mohelníková

Grupová souvislost grafů

Informatický ústav Univerzity Karlovy

Vedoucí diplomové práce: Mgr. Robert Šámal, Ph.D.
Studijní program: Informatika
Studijní obor: Diskrétní modely a algoritmy

Praha 2014

Na tomto místě bych ráda poděkovala Mgr. Robertu Šámalovi, Ph.D. za odborné vedení mé práce, za věnovaný čas a předané cenné zkušenosti. Dále bych chtěla poděkovat svým nejbližším za podporu při studiu a psaní této práce.

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

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

V Praze dne 14.7.2014

Název práce: Grupová souvislost grafů

Autor: Lucie Mohelníková

Katedra: Informatický ústav Univerzity Karlovy

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

Abstrakt: Zabývali jsme se grupovou souvislostí grafů, zejména pak $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislostí. Implementovali jsme v jazyce C++ test, zda je graf grupově souvislý a pomocí něho hledáme grafy, které jsou grupově souvislé v jedné ze zkoumaných grup a v druhé nikoliv. Zkoumali jsme grafy, které vzniknou podrozdělením hran několika speciálních grafů např. K_4 a krychle. Hlavním přínosem této práce je nalezení dvou grafů, které jsou $\mathbb{Z}_4$-souvislé a nejsou $\mathbb{Z}_2^2$-souvislé. Pomocí druhé nezávislé implementace testu na grupovou souvislost napsané v jazyce Prolog s využitím CSP jsme ověřili, že tyto grafy jsou $\mathbb{Z}_4$-souvislé. Analyticky jsme dokázali, že jeden z nalezených grafů není $\mathbb{Z}_2^2$-souvislý.

Klíčová slova: grupová souvislost, toky, grupa

Title: Group connectivity of graphs

Author: Lucie Mohelníková

Department: Computer Science Institute of Charles University

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

Abstract: The main topic of this thesis is the group connectivity of graphs, especially $\mathbb{Z}_2^2$- and $\mathbb{Z}_4$-connectivity. We implement a program in C++, which tests the group connectivity and we use it to find graphs which are group connected in one of these groups and aren't connected in the other. We consider graphs created by subdividing edges of specific graphs, like K_4 or cube. The main result of this work is finding two graphs which are $\mathbb{Z}_4$-connected and aren't $\mathbb{Z}_2^2$-connected. We also wrote a program in Prolog using CSP to check that these two graphs are $\mathbb{Z}_4$-connected. We proved analytically that one of these graphs isn't $\mathbb{Z}_2^2$-connected.

Keywords: group connectivity, flows, group

Obsah

1. Úvod

Souvislost grafu patří mezi základní pojmy teorie grafů. Tento pojem je úzce spjatý s problémem toků v grafech a nikde-nulových toků. W. Tutte [15] zavedl kolem roku 1955 pojem nikde-nulový tok (v angličtině nowhere-zero flow), který je populárním tématem mezi kombinatoriky na celém světě.

V 90. letech 20. století Jaeger, Linial, Payan a Tarsi [4] zavedli pojem grupová souvislost grafu, který je zobecněním klasické souvislosti. Definovali tento pojem s pomocí přebytků ve vrcholech. Motivací proč zkoumat grupovou souvislost je její vztah k nikde-nulovým tokům, neboť platí, že pro každý A-souvislý graf existuje nikde-nulový A-tok.

V článku [4] také formulovali otázku, zda $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislost jsou ekvivalentní. Tato otázka se stane pro naši práci stěžejní. Budeme se věnovat zkoumání grupové $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislosti několika speciálních tříd grafů. Cílem práce je implementovat program na testování grupové souvislosti, pomocí kterého nalezneme dva grafy, které jsou $\mathbb{Z}_4$- a nejsou $\mathbb{Z}_2^2$-souvislé. Jejich $\mathbb{Z}_4$-souvislost nezávisle ověříme programem napsaným v Prologu, $\mathbb{Z}_2^2$-souvislost dokážeme rozborem případů.

V prvních dvou kapitolách zavedeme pojmy tok, nikde-nulový tok a grupová souvislost. Podíváme se také na věty týkající se toků a grupové souvislosti a shrneme otevřené problémy, které s danou problematikou souvisí.

Další kapitola je věnována algoritmu pro testování grupové souvislosti, který jsem naprogramovala v jazyce C++. Také se budeme věnovat druhé nezávislé implementaci téhož problému, tentokrát v jazyce Prolog s využitím technik programování s omezujícími podmínkami (tzv. CSP).

Abychom mohli efektivně testovat grafy pomocí programů, musíme stanovit podmínky týkající se struktury takových grafů. Pomocí vět a tvrzení v Kapitole 5 vyřadíme grafy, u kterých je zřejmé, že nemohou být souvislé v žádné ze zkoumaných grup.

Poté uvedeme, které grafy jsme zkoumali a shrneme výsledky týkající se jejich souvislosti. Dokážeme, že jeden z nalezených grafů není grupově $\mathbb{Z}_2^2$-souvislý. Pomocí druhého programu pro kontrolu ověříme, že tento graf je $\mathbb{Z}_4$-souvislý.

Nakonec porovnáme doby běhu dvou nezávislých implementací testu na grupovou souvislost. Tyto implementace porovnáme v Kapitole 6.

2. Definice pojmů a značení

Graf $G = (V, E)$ je dán uspořádanou dvojicí množiny vrcholů V a množiny hran E. Neupřesníme-li, bereme grafy neorientované, bez smyček a multihran. *Most* je hrana, po jejímž odebrání se zvýší počet komponent grafu.

2.1 Abelovské grupy

Grupa je uspořádaná dvojice $(G, *)$, kde G je množina prvků a $*$ binární zobrazení splňující tyto podmínky:

- $\forall a, b, c \in G : (a * b) * c = a * (b * c)$ (asociativita),
- $\exists 0 \in G, \forall a \in G : a * 0 = 0 * a = a$ (neutrální prvek vzhledem k operaci $*$),
- $\forall a \in G, \exists b \in G : a * b = b * a = 0$ (jedná se o inverzní prvek a značíme ho $-a$).

V *abelovské grupě* navíc platí komutativita: $\forall a, b \in G : a * b = b * a$. *Řádem* grupy se nazývá mohutnost $|G|$ její nosné množiny. *Cyklická* grupa je taková grupa, která může být generována jedním jediným prvkem.

Věta 1 ([6]). *Každá konečná abelovská grupa je izomorfní součinu cyklických grup, jejichž řád je mocnina prvočísla.*

Z Věty 1 plyne, že existují právě dvě grupy se čtyřmi prvky ($\mathbb{Z}_2^2$ a $\mathbb{Z}_4$).

2.2 Toky

Nechť G je orientovaný graf a f je funkce přiřazující hranám prvky abelovské grupy A ($f : E(G) \to A$). Pro vrchol $v \in V(G)$ označíme $E^-(v)$ množinu *vstupních hran*, podobně $E^+(v)$ je množina *výstupních hran* a definujeme:

$$f^+(v) = \sum_{e \in E^+(v)} f(e),$$

$$f^-(v) = \sum_{e \in E^-(v)} f(e).$$

Tok na orientovaném grafu G je ohodnocení všech hran prvky grupy a pro všechny vrcholy $v \in V(G)$ platí $f^+(v) = f^-(v)$. Této vlastnosti se říká Kirchhoffův zákon, podle analogie s prvním Kirchhoffovým zákonem týkajícím se zachování elektrického náboje.

Přiřazuje-li funkce f hranám celočíselné hodnoty, pak se jedná o celočíselný tok. *Celočíselný k-tok* grafu G je ohodnocení hran grafu funkcí $f : E(G) \to \mathbb{Z}$ takové, že $|f(e)| < k$ pro každou hranu e grafu G.

Definice 2. *Nechť G je orientovaný graf a A je abelovská grupa (s 0 jako neutrálním prvkem). A-tok na G je zobrazení, kde $f : E(G) \to A$, které je tok.*

Definice 3. *Nechť $G = (V, E)$ a $A, B \subseteq V$ a f je tok na G. Pak definujeme:*

- $f(A, B) = \sum_{\substack{e=(a,b),\\ a\in A, b\in B}} f(e)$,
- $f^+(A) = f(A, \overline{A})$,
- $f^-(A) = f(\overline{A}, A)$,

kde $\overline{A} \in V(G) \setminus A$.

Pozorování 4. *Nechť f je tok na nějaké orientaci grafu G a $A \subseteq V$. Pro každou množinu A platí:*

$$f^+(A) = f^-(A).$$

Důkaz. Součet hodnot vtékajících do vrcholů množiny A:

$$\sum_{v\in A} f^+(v) = \sum_{\substack{v\in A,\\ u\ \in V}} f(v,u) = \sum_{\substack{v\in A,\\ u\ \in A}} f(v,u) + \sum_{\substack{v\in A,\\ u\ \notin A}} f(v,u) = f(A,A) + f(A,\overline{A}).$$

Součet hodnot vytékajících z vrcholů množiny A:

$$\sum_{v\in A} f^-(v) = \sum_{\substack{v\in A,\\ u\ \in V}} f(u,v) = \sum_{\substack{v\in A,\\ u\ \in A}} f(u,v) + \sum_{\substack{v\in A,\\ u\ \notin A}} f(u,v) = f(A,A) + f(\overline{A},A).$$

Z Kirchhoffova zákonu ($\forall v\ \in V : f^+(v) = f^-(v)$) plyne:

$$f(A, A) + f(A, \overline{A}) = f(A, A) + f(\overline{A}, A).$$

□

2.3 Nikde-nulové toky

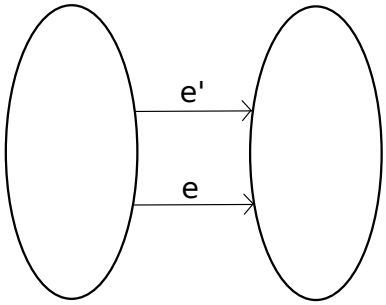
Nikde-nulový tok je takový tok, který nabývá na všech hranách hodnoty různé od nuly. Mezi hlavní důvody, proč zkoumat nikde-nulové toky patří, že pro rovinné grafy platí, že mají nikde-nulový k-tok právě tehdy, když jejich duální graf má obarvení pomocí k barev.

Učiníme několik jednoduchých pozorování. Začneme důsledkem Pozorování 4:

Důsledek 5. *Nechť G je graf a f je nějaký jeho tok:*

(1) Je-li e most, pak $f(e) = 0$. Z toho plyne, že graf G nemá nikde-nulový tok.

(2) Tvoří-li e a e' orientovaný 2-řez (na Obrázku 2.1), pak $f(e) + f(e') = 0$.



Obrázek 2.1: Orientovaný 2-řez.

Pozorování 6. *Nechť $G = (V, E)$ je orientovaný graf, f je tok na příslušné orientaci G a E_0 je podmnožina $E(G)$. Vyrobíme novou orientaci grafu G tak, že každé hraně z E_0 změníme její orientaci. Nechť f' je ohodnocení $E(G)$ dané předpisem:*

$$f'(e) = \begin{cases} f(e) & \text{pokud } e \notin E_0, \\ -f(e) & \text{pokud } e \in E_0. \end{cases}$$

Pak f' je tok na grafu G.

Z Pozorování 6 plyne následující lemma:

Lemma 7. *Nechť G je graf s nějakou orientací hran $E(G)$. Pak graf G má nikde-nulový k-tok na nějaké orientaci právě tehdy, když G má nikde-nulový k-tok pro každou orientaci hran $E(G)$.*

2.3.1 Vztahy mezi k-toky, $\mathbb{Z}_k$-toky a A-toky

Vztahy mezi nikde-nulovými k-toky, $\mathbb{Z}_k$-toky a A-toky na daném grafu G charakterizují následující věty a pozorování:

Věta 8 (Tutte [15], 1950)**.** *Graf G má nikde-nulový k-tok $\Leftrightarrow$ G má nikde-nulový $\mathbb{Z}_k$-tok.*

Důkaz. Implikace zleva doprava plyne z definice k-toku a $\mathbb{Z}_k$-toku. Je-li g k-tok, pak $g \bmod k$ je $\mathbb{Z}_k$-tok.

Nechť f je nikde-nulový $\mathbb{Z}_k$-tok. Zadefinujeme pro každý vrchol v:

$$b(v) = \sum_{e \in E^+(v)} f(e) - \sum_{e \in E^-(v)} f(e).$$

Povšimněme si, že $b(v)$ jsou násobky k.

Bez újmy na obecnosti zvolíme f (nikde-nulový $\mathbb{Z}$-tok) takový, že $\sum_{v \in V} |b(v)|$ je nejmenší možné mezi všemi toky f a orientacemi G. Dostáváme dva případy:

- $\sum_{v \in V} |b(v)| = 0$, z čehož plyne, že f je nikde-nulový k-tok.

- Nechť $\sum_{v \in V} |b(v)| \neq 0$. Vrcholy rozdělíme do dvou množin podle toho, zda $b(v)$ je kladné či záporné:

  $$S = \{v : b(v) > 0\}, T = \{v : b(v) < 0\}.$$

  Množiny S a T jsou neprázdné, protože $\sum_{v \in V} |b(v)| > 0$ a $\sum_{v \in V} b(v) = 0$ (plyne z definice).

  Nechť U jsou všechny vrcholy, které jsou spojeny cestou s vrcholy z množiny S.

  1. Pokud $U \cap T = \emptyset$, pak:

     $$0 < \sum_{v \in U} b(v) = \sum_{e \in E^+(U)} f(e) - \sum_{e \in E^-(U)} f(e).$$

     Ale $E^+(U) = \emptyset$, což implikuje:

     $$\sum_{v \in U} b(v) = \sum_{e \in E^+(U)} f(e) - \sum_{e \in E^-(U)} f(e) \leq 0.$$

     Dostáváme spor.

  2. $U \cap T \neq \emptyset$. Z toho plyne, že existuje cesta P spojující nějaké $s \in S$ a $t \in T$. Nyní obrátíme orientaci hran na cestě P a vytvoříme tak $\mathbb{Z}_k$-tok předpisem:

     $$f'(e) = \begin{cases} k - f(e) & \forall e \in P, \\ f(e) & \forall e \notin P. \end{cases}$$

     Zobrazení f' je $\mathbb{Z}_k$-tok (plyne z Pozorování 6). Zadefinujeme pro každé $v \in V$:

     $$b'(v) = \sum_{e \in E^+(v)} f'(e) - \sum_{e \in E^-(v)} f'(e).$$

     Všimněme si, že pro každé $v \in V \setminus \{s, t\}$ platí $b'(v) = b(v)$, $b'(s) = b(s) - k$ a $b'(t) = b(t) + k$. Z toho plyne, že $\sum_{v \in V} |b'(v)| < \sum_{v \in V} |b(v)|$, a tudíž dostáváme spor s minimalitou f.

  Dostáváme, že může nastat pouze možnost $\sum_{v \in V} |b(v)| = 0$, pro kterou existuje nikde-nulový $\mathbb{Z}_k$-tok.

□

Z předchozí věty plyne toto pozorování:

Pozorování 9 (Tutte)**.** *Pokud graf G má nikde-nulový k-tok, pak G má nikde-nulový h-tok pro každé $h \geq k$.*

Z následující věty plyne, že počet nikde-nulových toků je závislý na mohutnosti grupy A a je dán tzv. Tutteho tokovým polynomem.

Věta 10 (Tutte, 1954)**.** *Pro každý graf G existuje tokový polynom $P_G(x)$ takový, že pro každou grupu A je počet nikde-nulových A-toků na G roven $P_G(|A|)$.*

Jednoduchým důsledkem předešlé věty je existence nikde-nulových toků pro grupy se stejnou mohutností:

Důsledek 11 (Tutte, 1954)**.** *Nechť A a A' jsou dvě konečné abelovské grupy, pro které platí $|A| = |A'|$. Pak graf G má nikde-nulový A-tok právě tehdy, když má nikde-nulový A'-tok.*

Z Věty 8 a Důsledku 11 plyne obdoba Pozorování 9 pro grupové toky:

Důsledek 12 (Grupová monotonie)**.** *Jestliže G má nikde-nulový A-tok, pak G má nikde-nulový A'-tok pro každé $|A'| \geq |A|$.*

Z grupové monotonie plyne následující věta:

Věta 13. *(Tutte) Nechť A je abelovská grupa řádu k. Graf G má nikde-nulový k-tok právě tehdy, když G má nikde-nulový A-tok.*

2.3.2 Domněnky týkající se nikde-nulových toků

Nikde-nulových toků se týkají následující domněnky, které byly zformulovány Williamem Tuttem v polovině 20. století. U každé domněnky uvádíme i její anglický název, pod kterým je známá v literatuře.

Domněnka 14 (5-flow conjecture, 1954)**.** *Každý graf bez mostů má nikde-nulový 5-tok.*

Tato domněnka je u rovinných grafů ekvivalentní tomu, že všechny rovinné grafy jsou 5-obarvitelné, což je snadné dokázat (viz [2]). Bylo dokázáno, že pokud existuje protipříklad k této domněnce, pak je to cyklicky hranově 5-souvislý snark (pozn. *snark* je souvislý, kubický graf bez mostů s chromatickým číslem čtyři) s nejkratším cyklem délky alespoň 7.

Steinberg [11] dokázal tuto domněnku pro grafy vnořené do projektivní roviny.

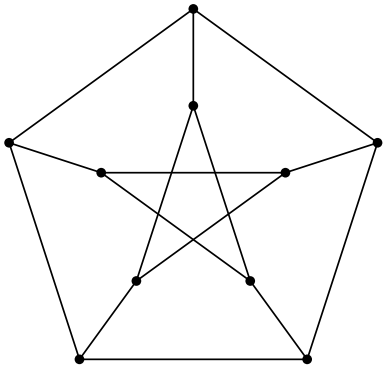
Jaeger v [3] dokázal, že každý graf bez mostů má nikde-nulový 8-tok. S touto oblastí také souvisí výsledek Paula Seymoura [12], který dokázal, že každý graf bez mostů má nikde-nulový 6-tok.

Domněnka 15 (4-flow conjecture, 1966)**.** *Každý graf bez mostů neobsahující podrozdělení Petersenova grafu má nikde-nulový 4-tok.*

Očekává se, že důkaz této domněnky bude obtížný, protože je to silnější tvrzení než Věta o čtyřech barvách [1]. Pokud Domněnku 15 omezíme jen na kubické grafy, dostáváme následující slabší verzi:

Domněnka 16 (Edge-3-coloring conjecture, 1966)**.** *Každý hranově 2-souvislý kubický graf, který neobsahuje podrozdělení Petersenova grafu, je hranově 3-obarvitelný.*

Robertson, Seymour a Thomas [10] v roce 1997 dokázali, že tuto domněnku stačí dokázat pro dva speciální druhy kubických grafů, které jsou téměř rovinné. Také oznámili, že tuto domněnku (Domněnka 16) vyřešili, ale dosud důkaz nepublikovali.



Obrázek 2.2: Petersenův graf.

Domněnka 17 (3-flow conjecture)**.** *Každý hranově* 4*-souvislý graf má nikde-nulový* 3*-tok.*

S touto domněnkou souvisí věta od Françoise Jaegera:

Věta 18. *Každý hranově* 4*-souvislý graf má nikde-nulový* 4*-tok.*

Jaeger také formuloval následující domněnku:

Domněnka 19 (The weak 3-flow conjecture)**.** *Existuje pevné celé číslo* k *takové, že každý hranově* k*-souvislý graf má nikde-nulový* 3*-tok.*

Tato domněnka byla Thomassenem [13] dokázána pro $k = 8$ a v roce 2012 vylepšena v [8] dokonce pro $k = 6$.

3. Grupová souvislost grafu

Pro graf $G = (V, E)$ s nějakou jeho orientací a abelovskou grupu A, značí $F(G, A)$ množinu všech funkcí $f : E(G) \to A$. Pro každou funkci f definujeme *hranici* $\partial f : V(G) \to A$ jako

$$\partial f(v) = \sum_{e \in E^+(v)} f(e) - \sum_{e \in E^-(v)} f(e),$$

což intuitivně znamená, že ∂f je *velikost přebytku* v každém vrcholu v. Nechť A^* je množina nenulových prvků grupy A ($A^* = A \setminus \{0\}$), pak $F^*(G, A) \subseteq F(G, A)$ obsahuje všechny funkce $f : E(G) \to A^*$.

Nikde-nulový A-tok je funkce $f \in F^*(G, A)$, pro kterou platí, že $\partial f = 0$.

Zobrazení $b : V(G) \to A$ je *funkce s nulovým součtem* na grafu G, platí-li $\sum_{x \in V} b(x) = 0$.

Definice 20. *Nechť $G = (V, E)$ je neorientovaný graf a A je abelovská grupa. Graf G je grupově A-souvislý, pokud platí: Pro každou orientaci G' grafu G každá funkce s nulovým součtem $b : V(G) \to A$ je hranicí ∂f nějaké funkce $f \in F^*(G', A)$.*

Z definice grupové souvislosti a nikde-nulového toku přímo plyne následující pozorování:

Pozorování 21. *Nechť G je grupově A-souvislý graf. Pak G má nikde-nulový A-tok.*

Z Pozorování 22 plyne, že na orientaci grafu nezáleží.

Pozorování 22. *Nechť G je orientovaný grupově A-souvislý graf a e je libovolná hrana. Změníme-li orientaci e, graf G bude i nadále grupově A-souvislý.*

Následující pozorování, která lze najít v [4], ukazují, že grupová souvislost je silnější než "normální" souvislost grafu.

Pozorování 23. *Graf $G = (V, E)$ je souvislý právě tehdy, když každá funkce s nulovým součtem $b : V(G) \to A$ je hranicí ∂f nějaké funkce $f \in (F, A)$.*

Důkaz. Implikace zprava doleva plyne z definice.

$\Rightarrow$: Tento případ stačí dokázat pro graf G, který je strom. Budeme postupovat matematickou indukcí podle počtu vrcholů stromu:

- Pokud je G jeden vrchol, pak je pozorování triviální.
- Nechť $G' = G - x$ je graf, který z G vznikne odtrhnutím vrcholu x, který je v původním grafu listem. Nechť e je hrana spojující vrcholy y a x. Na grafu G je definována funkce s nulovým součtem $b : V(G) \to A$. Nechť $b' : V(G') \to A$ je funkce, která je rovna b na množině vrcholů $V \setminus \{x, y\}$ a nechť $b'(y) = b(x) + b(y)$. Z toho plyne, že b' je funkce s nulovým součtem na grafu G'. Indukcí podle velikosti stromu dostáváme, že existuje $f' \in (G', A)$ s $b' = \partial f'$.

  Za předpokladu, že e vede z y do x, položíme $f(e) = b(x)$ a $f(a) = f'(a)$ pro $a \neq e$.

□

Pozorování 24. *Nechť $G = (V, E)$ je graf a A je abelovská grupa. Pak následující tvrzení jsou ekvivalentní:*

(1) G je A-souvislý.

(2) G je souvislý a je dána nějaká funkce $\overline{f} \in F(G, A)$ taková, že existuje nějaký A-tok daný funkcí f takový, že $f(e) \neq \overline{f}(e)$ pro každé $e \in E$.

(3) Je dána funkce s nulovým součtem $b : V(G) \to A$ a $\overline{f} \in F(G, A)$, pak existuje nějaká funkce $f \in F(G, A)$, která splňuje rovnost $\partial f = g$ a $f(e) \neq \overline{f}(e)$ pro každé $e \in E$.

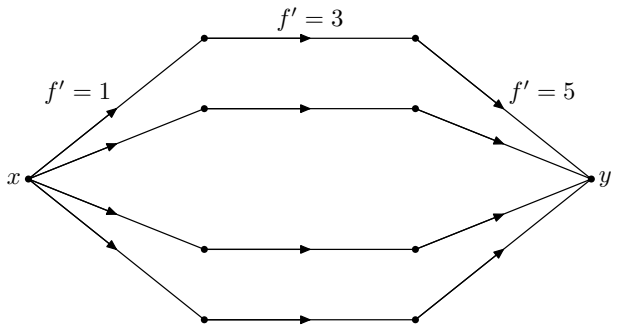
Důkaz. (1) $\Rightarrow$ (2): Z předpokladů je graf G A-souvislý. Nechť je dána nějaká funkce $\overline{f} \in F(G, A)$. Pak existuje $g \in F^*(G, A)$ taková, že $\partial g = -\partial \overline{f}$. Zadefinujeme funkci $f = \overline{f} + g$. Z toho plyne, že f je A-tok a je různý od $\overline{f}$ na všech hranách.

(2) $\Rightarrow$ (3): Z toho, že G je souvislý podle Pozorování 23 plyne, že pro každou funkci s nulovým součtem b existuje $g \in (F, A)$ taková, že $b = \partial g$. Z předpokladů (2) máme funkci $\overline{f} \in (G, A)$, která zaručuje existenci toku $f' \in F^*(G, A)$ a pro každou hranu platí, že $f'(e) \neq \overline{f} - g(e)$. Funkce $f = g + f'$ splňuje podmínky z bodu (3).

(3) $\Rightarrow$ (1): Vezmeme $\overline{f} = 0$, protože podle definice A-souvislosti potřebujeme funkci s nulovým součtem. □

V Pozorování 8 ukazujeme, že pokud existuje k-tok, pak existuje i h-tok ($\forall h \geq k$). V článku [4] ukazují, že podobná vlastnost u grupové souvislosti neplatí. Neboli, že graf, který je A-souvislý, nemusí být A'-souvislý, kde $|A'| > |A|$:

Pozorování 25. *Nechť $G = (V, E)$ je neorientovaný graf, který obsahuje čtyři paralelní disjunktní cesty délky 3 (na Obrázku 3.1) vedoucí z vrcholu x do vrcholu y. Pak G je $\mathbb{Z}_5$-souvislý a není $\mathbb{Z}_6$-souvislý.*



Obrázek 3.1: Graf se čtyřmi paralelními cestami.

Důkaz. Zorientujeme paralelní hrany ve směru z x do y, u ostatních hran grafu G vybereme libovolnou orientaci. Vrcholy na paralelních disjunktních cestách budeme nazývat *vnitřními* vrcholy.

Začneme tím, že dokážeme, že takový graf není $\mathbb{Z}_6$-souvislý. Definujeme funkci $b : V(G) \to \mathbb{Z}_6$ s nulovým součtem:

- $b(x) = 1$,
- $b(y) = -1$,
- $b(v) = 0$ pro všech 8 vnitřních vrcholů v.

Nechť $f' : E(G) \to \mathbb{Z}_6$ přiřadí liché hodnoty (1, 3 a 5) hranám každé cesty mezi vrcholy x a y. Každá cesta dostane prvky 1, 3 a 5 v nějakém pořadí. Nyní musíme ověřit, zda platí podmínka (3) z Porozování 24:

Funkce $f \in F(G, \mathbb{Z}_6)$ splňuje podmínku na nulovou hranici na vnitřních vrcholech a hodnoty jsou různé od hodnot určených f' pro každou hranu. Z toho plyne, že na cestách mezi vrcholy x a y tečou sudé hodnoty. Hranice $b(x) = 1$ znamená, že součet hodnot čtyř hran, které vytékají z x musí být 1, ale po těchto hranách můžou téct pouze sudé hodnoty. Součet sudých hodnot v $\mathbb{Z}_6$ nemůže být liché číslo, tudíž graf G není grupově $\mathbb{Z}_6$-souvislý.

K ověření, že G je $\mathbb{Z}_5$-souvislý využijeme druhou část Pozorování 24. Funkce $f' : E(G) \to \mathbb{Z}_5$ přiřadí hranám na každé cestě nejvýše 3 zakázané hodnoty, neboli vždy alespoň 2 hodnoty zůstanou volné. Hodnoty toku na hranách musíme vybrat tak, aby platilo $\partial(f)(x) = \partial(f)(y) = 0$. Součet čtyř hodnot musí být 0.

Nyní je potřeba rozmyslet, že vezmeme-li libovolné čtyři dvouprvkové množiny $A, B, C, D \subset \mathbb{Z}_5$, pak vždy existují čtyři prvky $a \in A$, $b \in B$, $c \in C$ a $d \in D$ takové, že $a+b+c+d = 0$. Neboli, že každé ze čtyř hran vycházejících z x můžeme přiřadit nějakou hodnotu tak, že jejich součet dá dohromady nulu. Pomůžeme si pozorováním z [4] týkajícím se cyklických grup a součtu jejich podmnožin:

Pozorování 26. *Nechť P je cyklická grupa, jejíž řád je prvočíslo. S je vlastní podmnožinou P a T je podmnožina P obsahující alespoň dva prvky. Pak $|S+T| > |S|$.*

Z Pozorování 26 plyne, že velikosti množin A, $A+B$, $A+B+C$, $A+B+C+D$ tvoří ryze rostoucí posloupnost (dokud nedosáhnou mohutnosti 5). Z toho plyne, že $A + B + C + D$ obsahuje všech 5 prvků grupy $\mathbb{Z}_5$ a tudíž i hodnotu 0.

□

3.1 Domněnky týkající se grupové souvislosti

Stejně jako v případě nikde-nulových toků, i u grupové souvislosti existují podobné domněnky. Jaegerova slabá domněnka o 3-toku má svůj ekvivalent v grupové souvislosti. V [4] je uveden důkaz, že jsou tyto dvě domněnky ekvivalentní.

Domněnka 27. *Existuje pevné celé číslo l takové, že každý hranově l-souvislý graf je grupově $\mathbb{Z}_3$-souvislý.*

Tutteho Domněnka 17 se týká hranově 4-souvislých grafů. V případě grupové souvislosti potřebujeme 5-souvislost:

Domněnka 28. *Každý hranově 5-souvislý graf je grupově $\mathbb{Z}_3$-souvislý.*

4. Algoritmus na testování grupové souvislosti grafu

V této kapitole se budeme věnovat analýze algoritmu, který testuje grupovou souvislost grafu s použitím výpočetní techniky. Zvolili jsme programovací jazyk C++ vzhledem k jeho rychlosti, variabilitě a multiplatformní podpoře.

4.1 Generování toků a zakázaných hodnot

Nyní objasníme, jakým způsobem budeme testovat grafy, zda jsou grupově souvislé. Využijeme druhou část Pozorování 24:

Graf G je A-souvislý $\Leftrightarrow$ G je souvislý a platí: Pro všechny $\overline{f} \in F(G, A)$ existuje nějaký A-tok daný funkcí f takový, že $f(e) \neq \overline{f}(e)$ pro každé $e \in E$.

Zakázanými hodnotami nazveme hodnoty přiřazené hranám funkcí $\overline{f}$. Vygenerujeme-li všechny m-tice ($m = |V(G)|$) zakázaných hodnot nad danou grupou a ke každé m-tici najdeme tok, který je na všech hranách různý od zakázaných hodnot, dokážeme tím, že graf je grupově A-souvislý.

4.1.1 Příprava grafu

Načteme graf a uložíme každou hranu do pole hran v obou jejích orientacích. Pole hran setřídíme podle počátečního vrcholu. Do pole vrcholů uložíme indexy počátků bloků hran pro jednotlivé vrcholy.

Pro jednoduchost budeme uvažovat pouze souvislé grafy. U grupové souvislosti nezáleží na orientaci grafu, určíme ji pořadím vrcholů na vstupu.

Pomocí DFS nalezneme libovolnou kostru grafu.

Výroba elementárních toků

Definice 29. *Elementární kružnice v grafu (V, E) vzhledem ke kostře (V, E') je kružnice v grafu $(V, E' \cup \{e\})$, kde $e \in E \setminus E'$.*

Pro každou nekosterní hranu nalezneme elementární kružnici. Je zřejmé, že pro každou nekosterní hranu taková kružnice vždy existuje.

Poté vytvoříme elementární tok.

Definice 30. *Elementární tok v grafu (V, E) je tok, který je ± 1 na elementární kružnici a 0 jinde.*

Elementární tok vyrobíme tak, že postupně (počínaje nekosterní hranou, která dostane hodnotu 1) projdeme hrany elementární kružnice a podle podmínky pro tok každou z nich odhonotíme 1 nebo -1, ostatní hrany budou mít hodnotu 0.

Až do tohoto momentu je příprava grafu nezávislá na volbě grupy.

4.1.2 Vygenerování toků

S pomocí elementárních toků vyrobíme všechny toky, které mohou na grafu existovat. Nechť $k = |E| - |V| + 1$ je počet elementárních toků, který je roven počtu nekosterních hran grafu. Vezmeme nějakou posloupnost p délky k, jejíž prvky jsou prvky grupy A. Hodnotu toku na hraně e spočítáme jako:

$$f = \sum_{j=0}^{k} h_j(e) \cdot p_j, \tag{4.1}$$

kde h je pole elementárních toků. Násobení není obvykle v grupě definované, ale zde násobíme jen čísly 0, ± 1, což lze vždy dobře definovat: $0 \cdot a = 0$, $1 \cdot a = a$ a $-1 \cdot a = -a$.

V Kapitolách z diskrétní matematiky [9] je dokázáno (Věta 13.4.3: O prostoru kružnic), že charakteristické vektory elementárních kružnic tvoří bázi vektorového prostoru. Analogii pro elementární toky jsme v žádné literatuře nenašli, tudíž ji dokážeme:

Věta 31. *Každý A-tok na grafu G lze jednoznačně vyjádřit ve tvaru 4.1.*

Důkaz. Nechť $e_1, \ldots, e_k$ jsou nekosterní hrany pro něž platí $h_j(e_i) = 1 \Leftrightarrow i = j$ (jinak $h_j(e_i) = 0$).

Pro f ve tvaru 4.1 je $p_j = f(e_j)$, odsud plyne jednoznačnost vyjádření.

Je-li g libovolný A-tok, volme g jako $p_j = f(e_j)$ a f buď dáno předpisem 4.1. Pak $f - g$ je A-tok, který je roven 0 na $e_1, \ldots, e_k$, tj. všude mimo kostru. Tudíž $f = g$. □

Z počtu elementárních toků triviálně plyne celkový počet toků na grafu:

Pozorování 32. *Nechť $G = (V, E)$ je neorientovaný graf a A je abelovská grupa, pak existuje celkem $|A|^{|E|-|V|+1}$ toků na grafu G.*

4.1.3 Porovnání zakázaných hodnot s hodnotami toku

Postupem popsaným v Kapitole 4.1.2 jsme vyrobili všechny toky na grafu G a ty uložíme do pole. Z části (2) Pozorování 24 plyne, že graf je grupově A-souvislý právě tehdy, když pro každou m-tici (kde $m = |E|$) zakázaných hodnot existuje nějaký tok. Vygenerujeme tudíž všechny m-tice $\overline{f}$ zakázaných hodnot a hledáme první tok f, který je na všech hranách různý od hodnot $\overline{f}$ ($f(e) \neq \overline{f}(e)$). Nenajde-li se takový tok, který by byl na všech hranách různý, pak jsme našli $\overline{f}$, které porušuje podmínku grupové souvislosti, a tudíž graf G nemůže být grupově A-souvislý.

Vygenerujeme-li a zkontrolujeme všechny možné m-tice zakázaných hodnot a pro každou z nich existuje tok, který je na všech příslušných hranách různý, pak je graf grupově A-souvislý.

Dvě různé implementace

Během psaní programu na testování grupové souvislosti vznikly dvě verze. První a druhá verze programu se liší pouze v tom, co udržujeme v paměti. V první verzi programu vygenerujeme všechny m-tice zakázaných hodnot a pro každou

z nich si pamatujeme, zda už byl nalezen příslušný tok. Pak postupně generujeme toky. Každý vygenerovaný tok porovnáme se všemi m-ticemi zakázaných hodnot. Tento přístup se ukázal jako nepoužitelný, neboť zabírá velké množství paměti. Jeho paměťovou složitost rozebereme v Kapitole 4.3.

V druhé verzi, která se ukázala jako značně použitelnější, máme v paměti uloženy všechny vygenerované toky. Postupně generujeme m-tice zakázaných hodnot a pro každou takovou m-tici procházíme toky do té doby, než najdeme takový, který je na všech příslušných hranách různý.

Součástí programu je také spočítání počtu m-tic zakázaných hodnot, pro které neexistuje tok. Pokud by nás tato hodnota nezajímala, mohl by se program zastavit při nalezení první m-tice, pro kterou neexistuje tok. U grupově nesouvislých grafů by došlo ke zkrácení doby běhu programu, u souvislých by bylo potřeba projít všechny m-tice zakázaných hodnot.

4.1.4 Shrnutí algoritmu

1. Načteme graf do vhodné reprezentace.
2. Nalezneme libovolnou kostru grafu a elementární kružnice pro nekosterní hrany.
3. Vyrobíme elementární toky.
4. Vyrobíme všechny toky. Pro každou posloupnost prvků grupy A délky k s pomocí elementárních toků vyrobíme tok.
5. Postupně generujeme všechny možné m-tice zakázaných hodnot a hledáme tok, který je na všech hranách různých.
6. Určíme, zda je graf grupově souvislý či nikoliv.

4.2 Časová složitost

Nyní rozepíšeme časovou složitost podle jednotlivých bodů z předchozí kapitoly:

1. Načtení grafu a setřídění hran: $O(m \cdot \log m)$.
2. Nalezení kostry a elementárních kružnic pro nekosterní hrany: $O(m+m^2) = O(m^2)$.
3. Vygenerování elementárních toků: $O(m^2)$.
4. K vyrobení všech toků na grafu je potřeba vygenerovat všechny posloupnosti délky k, kterých je $O(4^k) = O(4^{m-n})$. Výroba jednoho toku zabere $O(m \cdot k) = O(m^2)$, výroba všech toků trvá $O(m^2 \cdot 4^{m-n}) = O(4^m)$.
5. Generování všech m-tic zakázaných hodnot zabere $O(4^m)$. Kontrola, zda existuje tok pro danou zakázanou hodnotu, zabere nejvýše: počet toků · jejich délka $\Rightarrow O(4^m \cdot 4^{m-n}) = O(4^{2m-n})$.
6. V průběhu výpočtu se udržuje kolik existuje zakázaných m-tic, pro které neexistuje tok. Určení, zda je graf A-souvislý, je tedy $O(1)$.

Z předchozího rozboru plyne, že časová složitost je $O(4^{2m})$.

Očekáváme, že problém je NP-úplný, ale tuto otázku jsme nezkoumali. Holyer [5] dokázal, že rozpoznávání existence hranového 3-obarvení (a tedy nikde-nulového 4-toku) pro kubické grafy je NP-úplné. Proto očekáváme, že testování grupové souvislosti je také NP-úplné.

4.3 Paměťová složitost

V první verzi programu jsme udržovali v paměti všechny zakázané hodnoty, kterých je $4^m = 2^{2m}$. Toto řešení sice odpovídá části (2) Pozorování 24, ale není vhodné, neboť je potřeba $4^m/4^{m-n+1} = O(4^n)$-krát více paměti než v případě, kdy v paměti držíme všechny toky.

Pro ilustraci: Celkem máme 4^m zakázaných m-tic. Pokud bychom si pamatovali o každé hraně jen $1\,B$ informace a kdybychom na každý výpočet měli paměť $16\,GB$ ($16\,GB = 2^{34}\,B$), pak bychom mohli držet v paměti informace nejvýše o 15 hranách ($2^{34} = m \cdot 2^{2m} \Rightarrow m \doteq 15$).

Budeme tedy držet v paměti všechny vygenerované toky, kterých je $O(4^{m-n+1})$. Ostatní části (např. reprezentace grafu) jsou polynomiální.

Pro porování spočítáme, kolik hran mohou mít kubické grafy, budeme-li si pamatovat všechny toky. Tento typ grafů jsme si pro počítání paměťové složitosti nevybrali náhodou – v Kapitole 5.2 budeme zkoumat kubické grafy a grafy, které vzniknou podrozdělováním jejich hran.

Kubický graf, který má celkem n vrcholů, má $\frac{3}{2}n$ hran $\Rightarrow m \cdot 4^{m-2/3m+1} = 2^{32}$ $\Rightarrow m \doteq 37$.

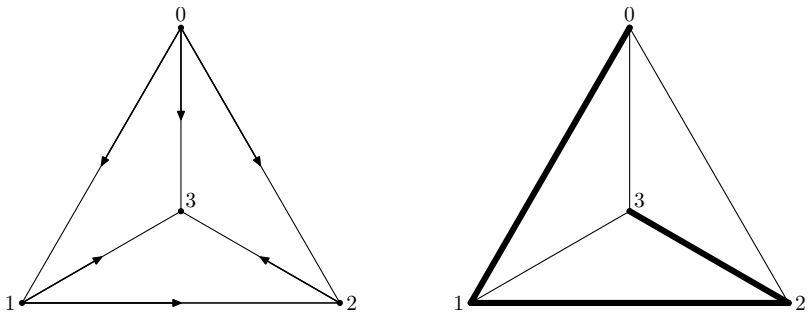
Pro kubické grafy se počet hran grafu, který můžeme v první a druhé verzi zpracovat, liší o 22.

Příklad

Na úplném grafu na 4 vrcholech tzv. K_4 a grupě $\mathbb{Z}_4$, budeme ilustrovat, co se děje v jednotlivých částech algoritmu.

Graf je dán tímto popisem:

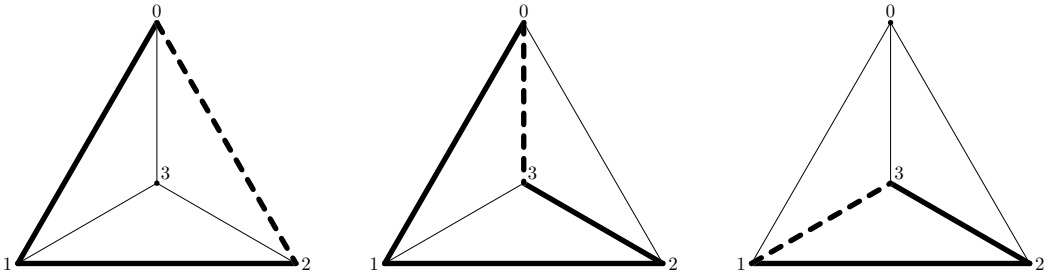
```
počet vrcholů: 4
počet hran: 6
hrany: (0,1), (1,2), (0,2), (0,3), (1,3), (2,3)
```



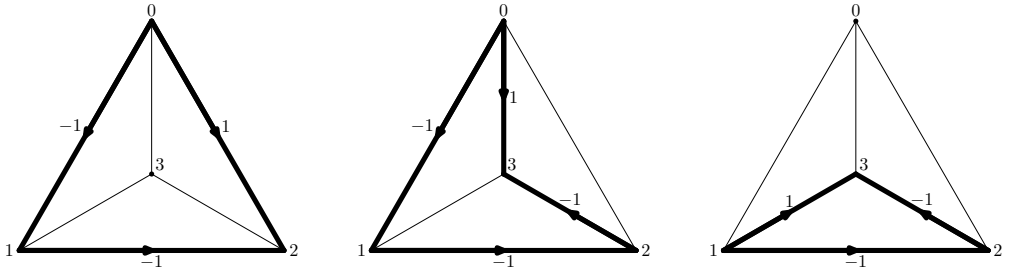
Obrázek 4.1: Orientace grafu (vlevo) a kostra grafu G.

Pomocí DFS nelezneme jeho libovolnou kostru, která je na Obrázku 4.1 zvýrazněna tučně.

Pro nekosterní hrany $(0,2)$, $(0,3)$ a $(1,3)$ najdeme elementární kružnice (Obrázek 4.2) pomocí kterých určíme elementární toky na hranách (Obrázek 4.3).



Obrázek 4.2: Elementární kružnice pro hrany $(0,2)$, $(0,3)$ a $(1,3)$.



Obrázek 4.3: Elementární toky.

Do tabulky zapíšeme hodnoty elementárních toků: sloupce budou určovat hrany, řádky elementární kružnice. Pro ilustraci vybereme posloupnost $(1,2,3)$ (jedná se o hodnoty p_j z 4.1) a spočítáme hodnotu toku na každé hraně. Tyto hodnoty jsou zapsány v posledním řádku tabulky.

	$(0,1)$	$(0,2)$	$(0,3)$	$(1,2)$	$(1,3)$	$(2,3)$
f_1	-1	1	0	-1	0	0
f_2	-1	0	1	-1	0	-1
f_3	0	0	0	-1	1	-1
f	1	1	2	2	3	3

Tabulka 4.1: Hodnoty elementárních toků.

S pomocí Kirchhoffova zákonu můžeme ověřit, že hodnoty z posledního řádku tabulky tvoří tok na příslušných hranách.

Nyní vygenerujeme a zapíšeme všechny možné $\mathbb{Z}_4$-toky na grafu G a poté vygenerujeme 4^6 šestic zakázaných hodnot a zkontrolujeme, zda pro každou šestici lze najít nějaký tok, který je na každé hraně různý od zakázané hodnoty na této hraně.

4.4 Alternativní přístup pomocí CSP

Pro kontrolu výsledků jsme využili alternativní přístup pomocí CSP.

Constraint Programming (zkráceně CP) neboli programování s omezujícími podmínkami je nástroj pro efektivní popis vztahů mezi proměnnými. Vztahy jsou zapisovány pomocí podmínek. Jedná se o deklarativní přístup, u kterého nedochází k přesnému popisu algoritmu, ale k popsání toho, jak má vypadat řešení. *Constraint Satisfaction* je proces, při kterém se hledá řešení tak, aby byly splněny všechny podmínky. Řešení dostaneme tehdy, pokud jsou všechny podmínky splněny.

Problém splňování podmínek (zkráceně CSP) se skládá z konečné množiny proměnných, jejich domén (konečná množina hodnot proměnné) a konečné množiny podmínek. Podmínka je libovolná relace nad množinou proměnných (např. $A \neq B$) a může být definována extenzionálně (jako množina kompatibilních n-tic) nebo intenzionálně (pomocí formulí). *Přípustné řešení* CSP je úplné konzistentní ohodnocení proměnných, což znamená, že každé proměnné je přiřazena hodnota z její domény a všechny podmínky jsou splněny. *Optimální řešení* je přípustné řešení, které minimalizuje/maximalizuje hodnotu objektivní funkce (funkce z množiny přípustných řešení do množiny reálných čísel).

Absolvování předmětu Programování s omezujícími podmínkami mě přivedlo k myšlence napsat program na testování grupové souvislosti grafu s využitím CSP. Pomocí tohoto programu nezávisle ověříme, zda zkoumané grafy jsou $\mathbb{Z}_4$- (resp. $\mathbb{Z}_2^2$-) souvislé. V Kapitole 6 porovnáme doby běhu programu v C++ a v Prologu (s využitím CSP).

V této kapitole popíšeme, jak bude vypadat problém a model pro jeho řešení. Jako programovací jazyk použijeme jazyk Prolog s knihovnou *clpfd*, která je součástí interpretru Sicstus Prolog. Více informací o projektu Sicstus Prolog lze najít na `http://sicstus.sics.se/`.

Využijeme toho, že úloha testování grupové souvislosti grafu, je totéž, co hledání uspořádané m-tice zakázaných hodnot na hranách grafu, pro kterou neexistuje tok. Graf je grupově A-souvislý, pokud neexistuje žádná taková m-tice.

4.4.1 Model

Modelem rozumíme způsob, jak pomocí proměnných a vztahů mezi nimi zapsat reálný problém.

Popíšeme model pro testování grupové $\mathbb{Z}_4$-souvislosti. Program lze jednoduše modifikovat pro testování $\mathbb{Z}_2^2$-souvislosti tím, že prvky této grupy zakódujeme pomocí čísel od 0 do 3 (např. bijekcí $(0,0) \to 0$, $(0,1) \to 1$, $(1,0) \to 2$ a $(1,1) \to 3$) a na těch zavedeme vlastní aritmetiku (př. $(0,1)+(0,1) = (0,0)$ resp. $1+1 = 0$).

Budeme mít dvě sady proměnných týkajících se hran grafu G: proměnné pro zakázané hodnoty a proměnné pro tok. Na každé hraně musí být hodnota toku různá od zakázané hodnoty. Jedná se o program testující grupovou $\mathbb{Z}_4$-souvislost, proto obě sady hodnot mají doménu obsahující prvky $0, 1, 2, 3$.

Dále stanovíme tokovou podmínku: pro každý vrchol grafu musí platit, že součet vstupních hran je roven součtu hran výstupních.

Zpočátku jsme rozdělili úlohu na dvě části: v první z nich jsme generovali všechny m-tice zakázaných hodnot a v druhé poté testovali, zda pro každou z nich existuje tok (pomocí CSP). Tato metoda, které se říká *Generate and test*, byla funkční nicméně velmi pomalá. Po konzultaci s profesorem Bartákem jsem vytvořila proceduru *search*, ve které dochází k vlastnímu řešení využívajícímu,

jak proměnné a podmínky mezi nimi, tak strukturu problému.

Tato procedura je založena na postupném generování zakázaných hodnot a ověřování, zda existuje tok. Na rozdíl od metody Generate and test dochází k testování na existenci toku již během generování m-tice zakázaných hodnot a ne až po vygenerování celé m-tice.

V proceduře *search* dojde k zafixování hodnoty proměnné pro zakázanou hodnotu na nějaké hraně a poté se zkoumá, zda ostatní proměnné pro zakázané hodnoty mají kompletní doménu. Stane-li se, že nějaké proměnné zmizela hodnota z domény, pak jsme našli konfiguraci, pro kterou nemůže existovat tok. Pokud se nikdy nestane, že by zmizela nějaká hodnota z domény, potom po zafixování všech hodnot otestujeme, zda existuje tok.

Pseudokód

V následujícím pseudokódu jsou vyjmenovány jednotlivé části, které rozebereme. Nejedná se o klasický pseudokód, na který jsme zvyklí, ale o popsání prologovských procedur.

```
vykonej
{
        pridej_promennouZH,
        prirad_zakazanouH,
        pridej_promennouTH,
        seznam_vrcholu,
        prirad_domenu,
        projdi_vrcholy,
        search,
}
```

Procedura *pridej_promennouZH* vyrobí pro každou hranu zakázanou hodnotu. V proceduře *prirad_zakazanouH* dostane každá zakázaná hodnota doménu. Prvky domény jsou čísla 0, 1, 2 nebo 3.

Po nachystání proměnných pro zakázané hodnoty dojde k přípravě proměnných pro tok. V rámci procedury *pridej_promennouTH* (*TH* znamená toková hodnota) dojde k vytvoření proměnných pro hrany, které určují hodnoty toku těchto na hranách. Dále potřebujeme vytvořit vrcholy a rozdělit hrany na tzv. *vstupní* a *výstupní*. Hranu ve tvaru *edge(V1,V2)* započteme pro vrchol V_1 jako výstupní, pro vrchol V_2 jako vstupní. Každý vrchol má dva seznamy: jeden se vstupními hranami, druhý s výstupními. Obě popsané procedury (*pridej_promennouPH* a *seznam_vrcholu*) jsou součástí procedury *udelej_graf*.

Procedura *prirad_domenu* určí každé proměnné pro tok její doménu (0, 1, 2 nebo 3) a to, že je různá od zakázané hodnoty pro příslušnou hranu.

V rámci *projdi_vrcholy* dojde k vytvoření vztahů mezi proměnnými pro hrany a vrcholy. Pro každý vrchol se sečtou proměnné jeho vstupních hran a proměnné jeho výstupních hran. Tyto dva součty se musí rovnat, aby byla zachována toková podmínka. Zatím jsme pouze určili vztahy mezi těmito proměnnými, nikoliv jejich hodnoty. K jejich přiřazení dojde v proceduře *search*. V této proceduře postupně fixujeme hodnoty na hranách a díváme se, zda se nám nezmenšila doména některé z proměnných. Dojde-li k tomu, že některá hodnota zmizí, pak pro tuto

konfiguraci nemůže existovat tok. Funkce *search* vrací jako seznam jednu m-tici zakázaných hodnot.

Abychom zjistili, zda je graf grupově A-souvislý, je potřeba všechny výsledky sloučit. Ke sloučení využijeme vestavěnou funkci *findall*. Dostaneme-li po sloučení 0 m-tic, pro které neexistuje tok, pak je graf A-souvislý. Pokud dostaneme kladné celé číslo, pak graf není grupově A-souvislý. U tohoto programu nedostaneme přesný počet pro kolik m-tic neexistuje tok, protože heuristiky použité v CSP postupně ořezávají prostor řešení.

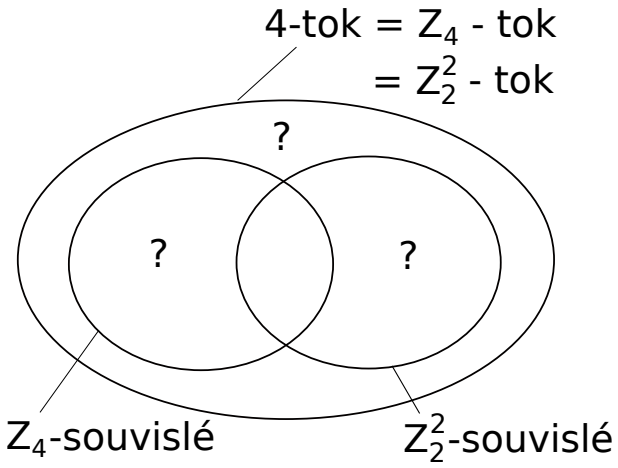
Délku běhu programu změříme pomocí vestavěné funkce *statistics*.

5. Grupová $\mathbb{Z}_4$- a $\mathbb{Z}_2^2$-souvislost

Cílem naší práce je popsat vztahy mezi grupovovou $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislostí z části za pomocí programů popsaných v Kapitole 4, z části analyticky.

V této kapitole popíšeme jakým způsobem jsme postupovali při hledání grafů, které budou v jedné grupě souvislé a v druhé nikoliv.

Podle Tutteho věty (Věta 11) jsou z pohledu nikde-nulových toků tyto grupy ekvivalentní. O tom, jak to vypadá s jejich grupovou souvislostí, se dosud mnoho nevědělo. Linial se v [4] zeptal, zda $\mathbb{Z}_4$-souvislost a $\mathbb{Z}_2^2$-souvislost jsou ekvivalentní.



Obrázek 5.1: Diagram vztahů $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-nikde-nulových toků a grupové souvislosti.

Diagram na Obrázku 5.1 znázorňuje všechny nikde-nulové $\mathbb{Z}_2^2$- (resp. $\mathbb{Z}_4$-) toky. Vnitřek diagramu obsahuje množinu $\mathbb{Z}_2^2$-souvislých grafů a množinu $\mathbb{Z}_4$-souvislých grafů. Tyto dvě množiny mají neprázdný průnik. V rámci této práce budeme zkoumat, zda existují souvislé grafy, pro které platí:

(1) jsou jak $\mathbb{Z}_2^2$-, tak $\mathbb{Z}_4$-souvislé,

(2) jsou pouze $\mathbb{Z}_4$-souvislé,

(3) jsou pouze $\mathbb{Z}_2^2$-souvislé,

(4) nejsou ani $\mathbb{Z}_2^2$-, ani $\mathbb{Z}_4$-souvislé, ale mají nikde-nulový $\mathbb{Z}_4$-tok.

Z Pozorování 21 plyne, že $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislé grafy jsou podmnožinou všech grafů, které mají $\mathbb{Z}_4$-tok.

Z výše uvedených bodů jsme nevyřešili otázku číslo (3). Primárně se věnujeme otázce (2), neboť zbývající dvě jsou snadné.

5.1 Podmínky pro grupovou souvislost

Na počátku jsme stanovili několik podmínek, které musí platit, aby graf mohl být souvislý alespoň v jedné ze dvou daných grup. Tyto podmínky nám měly vyloučit grafy, které nemohou být grupově souvislé ani v jedné z grup. Vše nyní shrneme za pomoci několika tvrzení. Některá z těchto tvrzení platí nejen pro grupy $\mathbb{Z}_2^2$ a $\mathbb{Z}_4$, ale pro libovolnou grupu.

Tvrzení 33. *Graf není grupově A-souvislý (pro libovolné A), pokud obsahuje most.*

Důkaz. Představíme si grupovou souvislost v řeči přiřazování zakázaných hodnot na hranách a následném hledání toků. Kvůli zachování tokových podmínek, musí na mostu téct hodnota 0. Zakážeme-li na mostě tuto hodnotu (na volbě ostatních zakázaných hodnot nezáleží), pak nemůže existovat tok. Našli jsme tudíž takovou konfiguraci zakázaných hodnot, pro kterou neexistuje tok a graf tedy není grupově A-souvislý pro libovolnou grupu. □

Následující věta nám říká, že graf, který je hranově 4-souvislý, bude souvislý v obou grupách a mezi těmito grafy zřejmě nenajdeme takový, který by souvislý nebyl.

Jaeger [3] dokázal, že každý hranově 4-souvislý graf má nikde-nulový 4-tok. V článku [4] tento důkaz modifikovali pro grupovou souvislost grafu:

Věta 34. *Každý hranově 4-souvislý graf je grupově A-souvislý pro každou abelovskou grupu A řádu $|A| \leq 4$.*

V důkazu Věty 34 se použije Nash-Williamsova [2] věta. Využije se toho, že každý hranově 4-souvislý graf má dvě hranově disjunktní kostry.

Důkaz Pozorování 25 nás inspiroval k tomu, abychom stanovili podmínku na délky cest v grafu:

Pozorování 35. *Obsahuje-li graf indukovanou cestu délky alespoň $|A|$, pak tento graf není A-souvislý.*

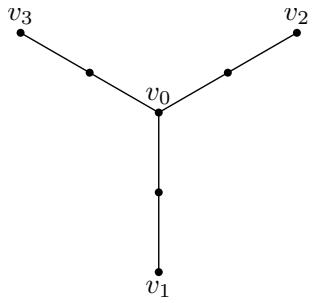
Důkaz. Nechť $G = (V, E)$ je graf obsahující cestu P délky alespoň $|A|$. Zvolíme všem hranám stejnou orientaci. Vybereme $|A|$ hran cesty P a přiřadíme jim různé prvky grupy A. Zbývajícím hranám přiřadíme libovolné prvky grupy A. Zakázali jsme $|A|$ prvků na cestě délky alespoň $|A|$. Nezáleží na tom, jaké zakázané hodnoty zvolíme na ostatních hranách grafu. Na hranách cesty P poteče stejná hodnota. Z volby zakázaných hodnot plyne, že nemáme volnou hodnotu z grupy A, která by mohla po této cestě téct, tudíž graf G není grupově A-souvislý. □

Z Pozorování 35 plyne, že existuje nekonečně mnoho grafů, které mají nikde-nulový $\mathbb{Z}_4$-tok (tedy i nikde-nulový $\mathbb{Z}_2^2$-tok), ale nejsou souvislé ani v jedné ze zkoumaných grup. Tyto grafy sestrojíme tak, že vezmeme libovolný graf se $\mathbb{Z}_4$-tokem (např. bipartitní kubické grafy) a vybereme hranu (případně hrany), kterou podrozdělíme alespoň třemi vrcholy.

Následující tvrzení jsme objevili až během zkoumání grafů, ale pro přehlednost jej uvádíme zde. Jeho zobecnění bude uvedeno později.

Tvrzení 36. *Existuje-li v grafu vrchol stupně 3, u kterého platí, že hrany (na Obrázku 5.2) s ním incidentní jsou každá součástí cesty délky alespoň 2, pak tento graf není souvislý ani v grupě $\mathbb{Z}_2^2$, ani v grupě $\mathbb{Z}_4$.*

Jinými slovy: Graf není grupově A-souvislý, pokud obsahuje 3 podrozdělené hrany potkávající se ve vrcholu stupně 3. Každá z těchto tří hran je podrozdělená alespoň jedním vrcholem.



Obrázek 5.2: Zakázaná konfigurace.

Důkaz. Opět použijeme zakazování hodnot na hranách a hledání toku. Podíváme se na to, jak to bude vypadat pro grupu $\mathbb{Z}_4$. Na cestách v_0v_1 a v_0v_2 zakážeme liché hodnoty (1 a 3), na cestě v_0v_3 zakážeme sudé hodnoty (0 a 2). Po cestách v_1v_0 a v_2v_0 tudíž přiteče do vrcholu v_0 dvakrát sudá hodnota, kdežto v případě cesty v_3v_0 to bude lichá hodnota. Při jakékoliv orientaci hran dojde k tomu, že u vrcholu v_0 bude parita toho, co do něho vtéká, různá od parity toho, co z něj vytéká. Tudíž s danou konfigurací zakázaných hodnot nelze vytvořit tok, a proto graf obsahující 3 podrozdělené hrany incidentní se stejným vrcholem stupně 3 nemůže být $\mathbb{Z}_4$-souvislý.

Pro grupu $\mathbb{Z}_2^2$ je to obdobné. Na cestách v_0v_1 a v_0v_2 zakážeme hodnoty $(0,1)$ a $(1,1)$, na cestě v_0v_3 zakážeme $(0,0)$ a $(1,0)$. Do vrcholu v_0 po cestách v_0v_1, v_0v_2 mohou přitéct v součtu pouze tyto hodnoty: $(0,0)$ a $(1,0)$. Očekává se proto, že příslušná hodnota přiteče do v_0 i z v_3, ale obě tyto hodnoty jsou zakázané. Z toho plyne, že takový graf nemůže být ani $\mathbb{Z}_2^2$-souvislý.

□

5.2 Zkoumání grafů

Na počátku jsme stanovili, že chceme najít graf, který bude souvislý v jedné z grup a nesouvislý v té zbývající. Z Tvrzení 33 a Věty 34 plyne, že má smysl zkoumat grafy, které jsou hranově 2- nebo 3-souvislé. Dále takový graf nesmí mít indukovanou cestu délky 4 a delší (plyne z Pozorování 35).

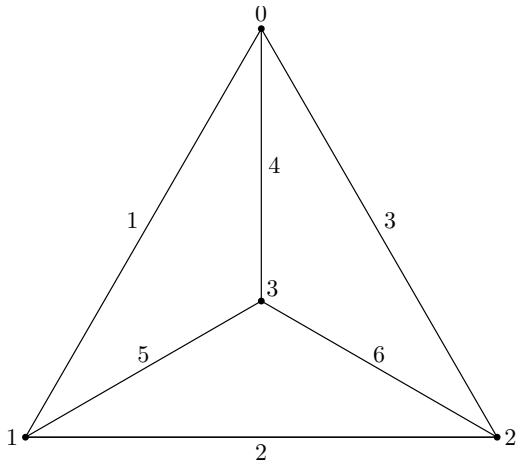
Začali jsme zkoumáním grafů pomocí programu napsaném v jazyce C++. Program v Prologu využívající CSP byl napsán až posléze pro kontrolu výsledků týkajících se grafů vzniklých podrozděleným grafu Cube.

Vrcholy, které podrozdělují hranu, budeme nazývat *podrozdělítky*.

5.2.1 Podrozdělený graf K_4

Jako první graf jsme použili graf K_4. Vygenerujeme všechny možné K_4 s žádným, jedním nebo dvěma podrozdělítky na hranách. Grafy budeme kódovat posloupností šesti čísel, kde každé číslo znamená, kolik podrozdělení (žádné, jedno nebo dvě) obsahuje příslušná hrana. Posloupnost podrozdělení na hranách nazveme P. Na Obrázku 5.3 jsou hrany K_4 a jim odpovídající indexy v posloupnosti P. Například K_4 zakódujeme jako 000000.

Celkem existuje $3^6 = 729$ různých posloupností. Abychom zbytečně neopakovali výpočty na izomorfních grafech, použijeme následující úvahu pro zredukování



Obrázek 5.3: Graf K_4 s indexy posloupnosti P.

množství grafů, které budeme zkoumat:

Na počátku jsou všechny vrcholy a hrany K_4 ekvivalentní. Bez újmy na obecnosti bude mít hrana $(0,1)$ nejmenší hodnotu (označme ji $h(1)$), její hodnoty tudíž budou od 0 do 2. Všichni její sousedé (hrany $(1,2)$, $(0,2)$, $(0,3)$ a $(1,3)$) jsou ekvivalentní, tudíž vybereme jednoho z nich, který bude minimální. Nechť je tímto sousedem hrana $(1,2)$, její hodnota (označme ji $h(2)$) bude nabývat hodnot od $h(1)$ do 2. Kdybychom vybrali jiného souseda, můžeme vrcholy a hrany grafu zpermutovat tak, aby odpovídaly grafu s minimální hranou $(1,2)$. Nedostali bychom tudíž žádné nové grafy, pouze grafy izomorfní s těmi, které jsme již vygenerovali.

Ostatní sousedé hrany $(0,1)$ budou nabývat hodnot od $h(2)$ do 2. Hrana $(2,3)$ bude mít hodnoty od $h(1)$ do 2.

Stanovili jsme podmínky pro hodnoty posloupnosti P: $h(1) \leq h(2)$, $h(2) \leq h(3)$, $h(2) \leq h(4)$, $h(1) \leq h(6)$, $h(2) \leq h(5)$.

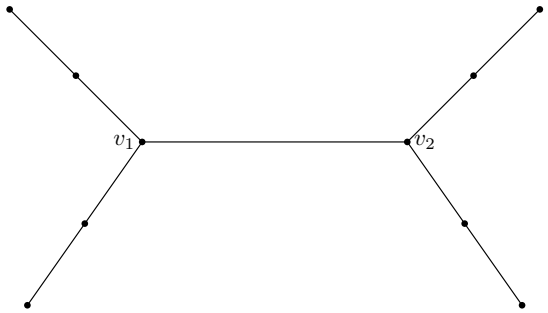
Pomocí programu napsaného v jazyce Perl jsme vygenerovali 127 posloupností (resp. grafů), pro které jsme nechali spočítat, zda jsou grupově $\mathbb{Z}_4$- a $\mathbb{Z}_2^2$-souvislé. Všechny tyto grafy jsou buď souvislé v obou grupách nebo v obou nesouvislé.

Během zkoumání K_4 a grafů vzniklých výše popsaným způsobem jsme stanovili podmínku, že nesmí existovat vrchol stupně 3, do kterého vedou tři hrany a každá z nich je podrozdělena alespoň jedním vrcholem. Tuto podmínku jsme formulovali jako Tvrzení 36, které je uvedeno na konci Kapitoly 5.1.

Také jsme formulovali následující tvrzení:

Tvrzení 37. *Graf, jehož vrcholy v_1, v_2 jsou stupně 3 a hrana v_1v_2 je nepodrozdělená, není grupově $\mathbb{Z}_4$- a $\mathbb{Z}_2^2$-souvislý, pokud obsahuje konfiguraci podrozdělítek jako na Obrázku 5.4.*

Důkaz. Důkaz je podobný důkazu Tvrzení 36. Podrozděleným hranám jdoucím do vrcholu v_1 zakážeme vždy sudé hodnoty, musí po nich tedy téct liché hodnoty. Na hraně v_1v_2 tudíž poteče sudá hodnota. Jedné z podrozdělených hran jdoucích z vrcholu v_2 zakážeme liché hodnoty, druhé podrozdělené hraně zakážeme sudé hodnoty. Nyní máme ve vrcholu v_2 dvakrát sudou a jednou lichou hodnotu. Z toho je zřejmé, že takový tok nemůže existovat, neboť je různá parita (nezávisle na volbě orientace grafu) toho, co vtéká a co vytéká z vrcholu v_2. □

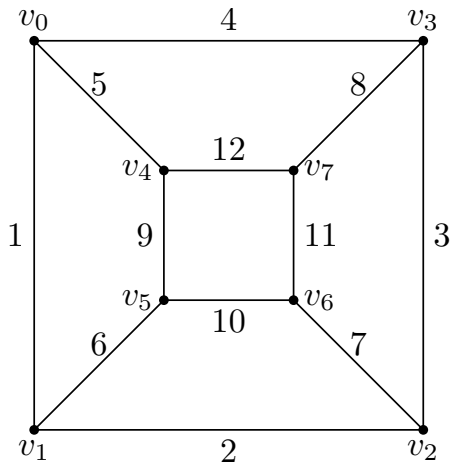


Obrázek 5.4: Konfigurace kvůli které nemůže být graf $\mathbb{Z}_4$- ani $\mathbb{Z}_2^2$-souvislý.

Toto tvrzení jsme použili u zkoumání grupové souvislosti grafu ve tvaru krychle. Jako jeho zobecnění vzniklo posléze Tvrzení 38.

5.2.2 Podrozdělený graf Cube

Dalším grafem, který jsme zkoumali, je krychle. Tento graf budeme nazývat *Cube*. Opět jsme využili již popsané kódování hran. Posloupnost P má délku 12, protože Cube má 12 hran. Na Obrázku 5.5 je u každé hrany napsán její index v posloupnosti P.

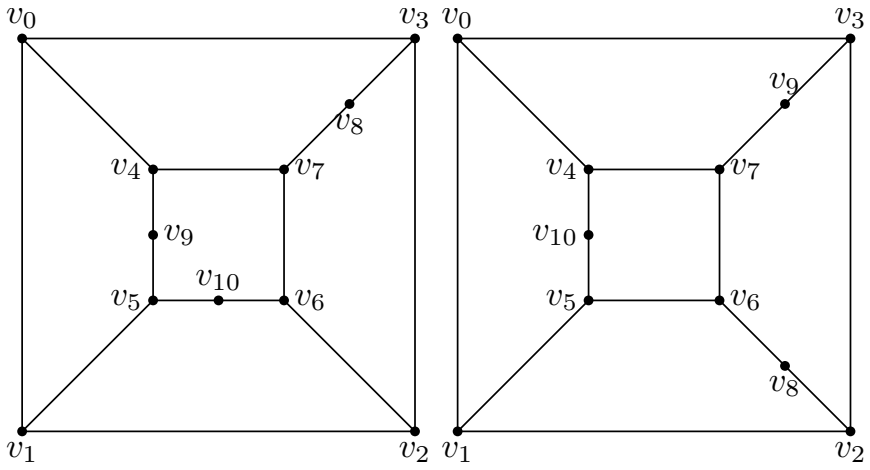


Obrázek 5.5: Cube s indexy hran v posloupnosti P.

Tentokrát jsme se z důvodu velikosti grafů omezili pouze na žádné nebo jedno podrozdělítko na hraně. Všech možných posloupností délky 12 obsahujících čísla 0 a 1 je celkem $2^{12} = 4\,096$. Pomocí programu napsaném v jazyce Python s využitím matematického rozšíření Sage jsme dostali z každé skupiny izormorfních grafů pouze jediného zástupce. Zůstalo nám celkem 144 grafů, ze kterých jsme pomocí programu v Perlu odstranili ty, které podle Tvrzení 36 a 37 nemohou být souvislé ani v jedné ze zkoumaných grup.

Po provedení této selekce nám zbylo pouhých 49 grafů, které jsme nechali otestovat programem v C++.

Mezi těmito grafy jsme našli dva, které jsou $\mathbb{Z}_4$-souvislé a nejsou $\mathbb{Z}_2^2$-souvislé. Jedná se o grafy na Obrázku 5.6 s posloupností podrozdělítek 000000011100 a 000000111000.



Obrázek 5.6: $\mathbb{Z}_4$-souvislé a $\mathbb{Z}_2^2$-nesouvislé grafy.

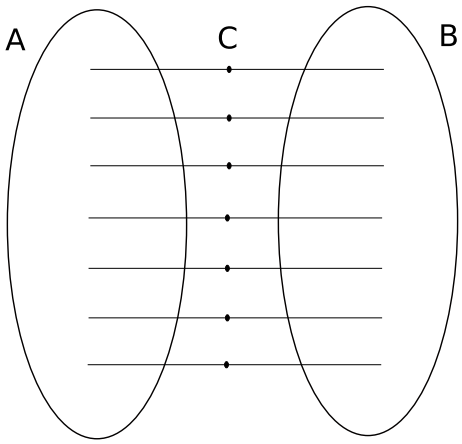
V Kapitole 7 dokážeme, že Cube s posloupností 000000011100 není grupově $\mathbb{Z}_2^2$-souvislý.

Kvůli ověření grupové $\mathbb{Z}_4$-souvislosti jsem napsala program v Prologu využívající CSP. Tímto programem jsme ověřili, že graf Cube s posloupností 000000011100 (resp. 000000111000) je grupově $\mathbb{Z}_4$- a není $\mathbb{Z}_2^2$-souvislý. V Kapitole 6 se podíváme na rozdíly v rychlosti obou implementací.

5.2.3 Další podmínky

Vzhledem k výsledkům testování podrozdělených grafů K_4 a Cube jsme objevili další podmínky týkající se vlastností grafů, které zobecňují Tvrzení 36.

Tvrzení 38. *Vrcholy grafu G jsou rozděleny na disjunktní množiny A, B a C (Obrázek 5.7). Každý vrchol množiny C má stupeň 2 a jednoho souseda v A a druhého v B. Pak tento graf není grupově $\mathbb{Z}_2^2$- ani $\mathbb{Z}_4$-souvislý.*



Obrázek 5.7: Řez rozdělující graf na dvě části.

Důkaz. Pro spor předpokládejme, že G je grupově souvislý. Z toho plyne, že existuje tok pro libovolnou m-tici zakázaných hodnot na hranách grafu G.

Řez je tvořen k podrozdělenými cestami. Můžeme předpokládat (Pozorování 22), že všechny hrany jsou orientovány z A do C a z C do B. Prvním $k-1$ cestám zakážeme vždy obě liché hodnoty. Vynutíme tudíž, že na těchto cestách poteče sudá hodnota. Na k-té cestě zakážeme sudé hodnoty, musí tudy téct lichá hodnota. Zde dostáváme spor s Pozorováním 4, neboť $f^-(A) = 0$ a $f^+(A)$ je liché. □

Věta 39 (Tutte [15]). *Kubický graf G má nikde-nulový 4-tok (resp. $\mathbb{Z}_4$-tok, resp. $\mathbb{Z}_2^2$-tok) právě tehdy, když G je hranově 3 obarvitelný.*

Z Tutteho věty plyne:

Důsledek 40. *Nechť G je kubický graf, který nemá nikde-nulový 4-tok, pak G není hranově 3-obarvitelný.*

Z následujícího tvrzení plyne, že oproti mnoha jiným problémům v této oblasti teorie grafů nemá smysl zkoumat snarky, ani jejich podrozdělení:

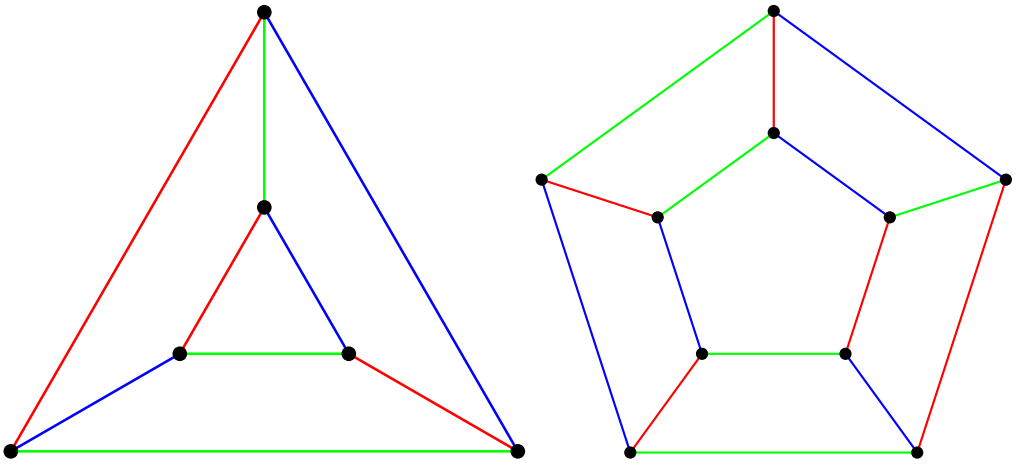
Tvrzení 41. *Snarky nejsou grupově $\mathbb{Z}_2^2$- ani $\mathbb{Z}_4$-souvislé.*

Důkaz. Nechť všechny zakázané hodnoty jsou rovny 0 (resp. $(0,0)$). Z neexistence nikde-nulového $\mathbb{Z}_4$-toku (resp. $\mathbb{Z}_2^2$-toku) plyne, že pro takovou konfiguraci zakázaných hodnot neexistuje tok. A tudíž tento graf nemůže být $\mathbb{Z}_2^2$- ani $\mathbb{Z}_4$-souvislý. □

5.2.4 Další zkoumané grafy

Zaměřili jsme se na kubické hranově 3-obarvitelné grafy. Graf Cube jsou vlastně dva čtverce, jehož odpovídající vrcholy jsou spojené hranou. Vzali jsme tudíž dva trojúhelníky a příslušné vrcholy spojili hranou (tento graf budeme nazývat *Dvojtrojúhelník*). Totéž jsme udělali u pětiúhelníku (budeme jej nazývat *Dvojpětiúhelník*). Oba tyto grafy jsou na Obrázku 5.8 a jejich hrany jsou barevně, aby bylo zřejmé, že jsou hranově 3-obarvitelné.

Dalším zkoumaným grafem bude $K_{3,3}$.

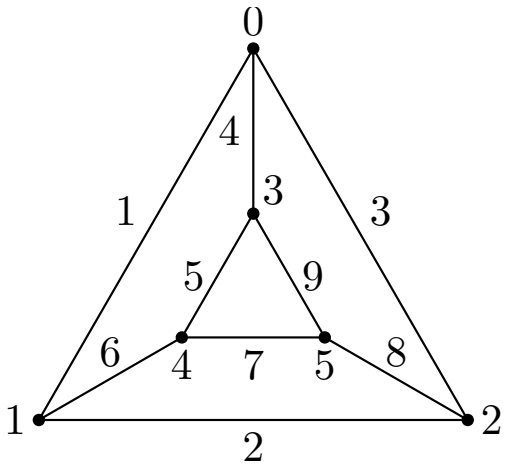


Obrázek 5.8: Dvojtrojuhelník a Dvojpětiúhelník.

Všechny tři uvedené grafy (Dvojtrojúhelník, Dvojpětiúhelník a $K_{3,3}$) jsou grupově $\mathbb{Z}_4$- i $\mathbb{Z}_2^2$-souvislé.

Podrozdělení dvojtrojúhelníku

Každou z hran Dvojtrojúhelníku jsme opět označili indexem v posloupnosti (viz Obrázek 5.9), která má nyní délku 9. Zpočátku jsme zkusili nejvýše jedno podrozdělítko na každé hraně.



Obrázek 5.9: Dvojtrojúhelník s indexy posloupnosti P.

Pomocí programu napsaného v C++ jsme vygenerovali všechny takové posloupnosti, kterých je celkem $2^9 = 512$. Poté jsme každý graf vzniklý aplikováním podrozdělítek na původní graf otestovali, zda po odebrání podrozdělených hran zůstane souvislý. Díky Tvrzení 38 víme, že nesouvislé grafy nemohou být grupově $\mathbb{Z}_4$- ani $\mathbb{Z}_2^2$-souvislé.

Po otestování souvislosti nám z 512 grafů zbylo 198, které jsme pomocí programu v Prologu s využitím CSP otestovali na grupovou $\mathbb{Z}_4$-souvislost. Celkem 98 grafů nebylo $\mathbb{Z}_4$-souvislých. Tyto grafy jsme opět pomocí CSP otestovali na $\mathbb{Z}_2^2$-souvislost. Zjistili jsme, že žádný z těchto grafů není $\mathbb{Z}_2^2$-souvislý, tudíž mezi grafy, které vznikly maximálně jedním podrozdělením každé hrany Dvojtrojúhelníku, není žádný takový, který by nebyl $\mathbb{Z}_4$-souvislý a byl $\mathbb{Z}_2^2$-souvislý.

Mezi grafy s jedním podrozdělítkem jsme nenašli graf, který by v jedné grupě byl souvislý a v druhé nikoliv. Proto jsme zkusili dogenerovat všechny grafy, které obsahují alespoň jedenkrát dvě podrozdělítka. Grafů, které obsahují 0, 1 nebo 2 podrozdělení, je $3^9 = 19\,683$, ale do těchto grafů jsou zahrnuty i grafy s maximálně jedním podrozdělením, kterých je 512. Proto grafů, které obsahují na alespoň jedné hraně 2 podrozdělení, je celkem $3^9 - 2^9 = 19\,171$. Tyto grafy jsme opět otestovali programem v C++, který nám vyřadil grafy, které podle Tvrzení 38 nebudou souvislé ani v jedné grupě. Dostali jsme celkem 1 781 grafů, které jsme nechali otestovat programem na testování grupové souvislosti napsaným v jazyce C++. Program jsme modifikovali tak, aby počítal grupovou $\mathbb{Z}_4$-souvislost a ve chvíli, kdy najde $\mathbb{Z}_4$-nesouvislý graf, otestuje ho na $\mathbb{Z}_2^2$-souvislost.

Po otestování všech 1 781 grafů jsme zjistili, že mezi nimi není žádný, který by nebyl $\mathbb{Z}_4$- a byl $\mathbb{Z}_2^2$-souvislý.

6. Srovnání CSP a C++

Na několika typech grafů porovnáme dobu běhu programů v C++ a v Prologu s využitím CSP. Měření byla prováděna na počítači s názvem Tikam (Intel(R) Core(TM) $i7-3770$ CPU @ $3.40\,GHz$, 8 jader, RAM: $16\,GB$).

Pro kompilaci programu v jazyce C++ používáme GNU g++ s flagem $-O3$ (`http://gcc.gnu.org/`) a pro interpretaci programu v jazyce Prolog využíváme Sicstus Prolog 4 (`http://sicstus.sics.se/`).

6.1 Podrozdělené K_4

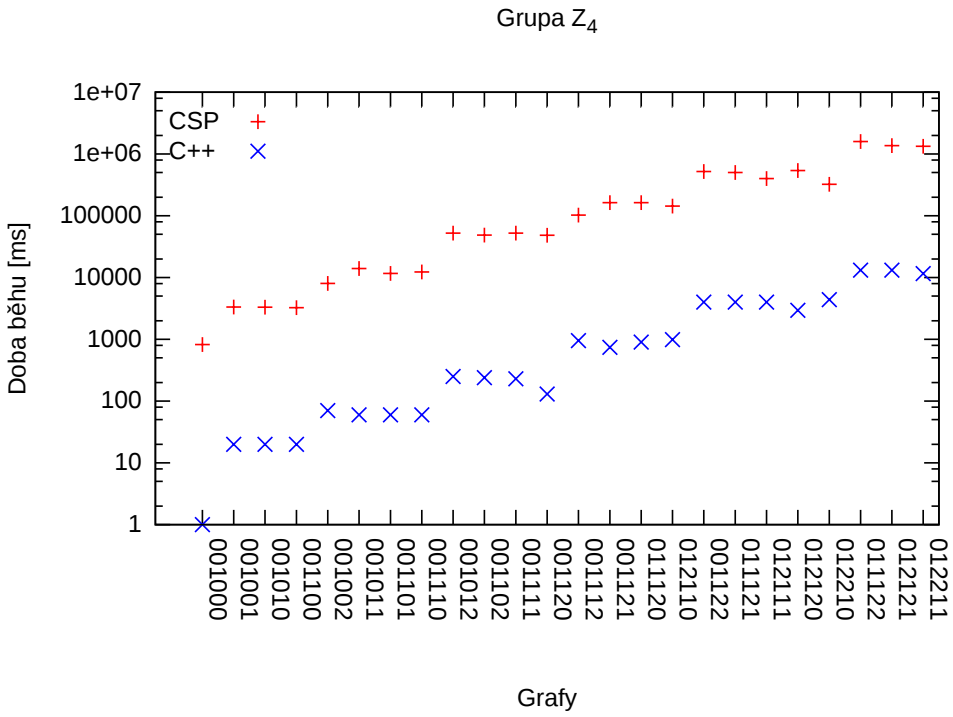
Z grafů (popsaných v Kapitole 5.2.1) vzniklých podrozdělením K_4, jsme vybrali náhodně 24. Pro tyto grafy jsme nechali otestovat, zda jsou grupově souvislé v obou grupách. Záleží nám pouze na době běhu, nikoliv na tom, zda jsou souvislé či nikoliv.

Z grafů na Obrázku 6.1 a 6.2 je viditelné, že doba běhu záleží na počtu podrozdělení grafu K_4.

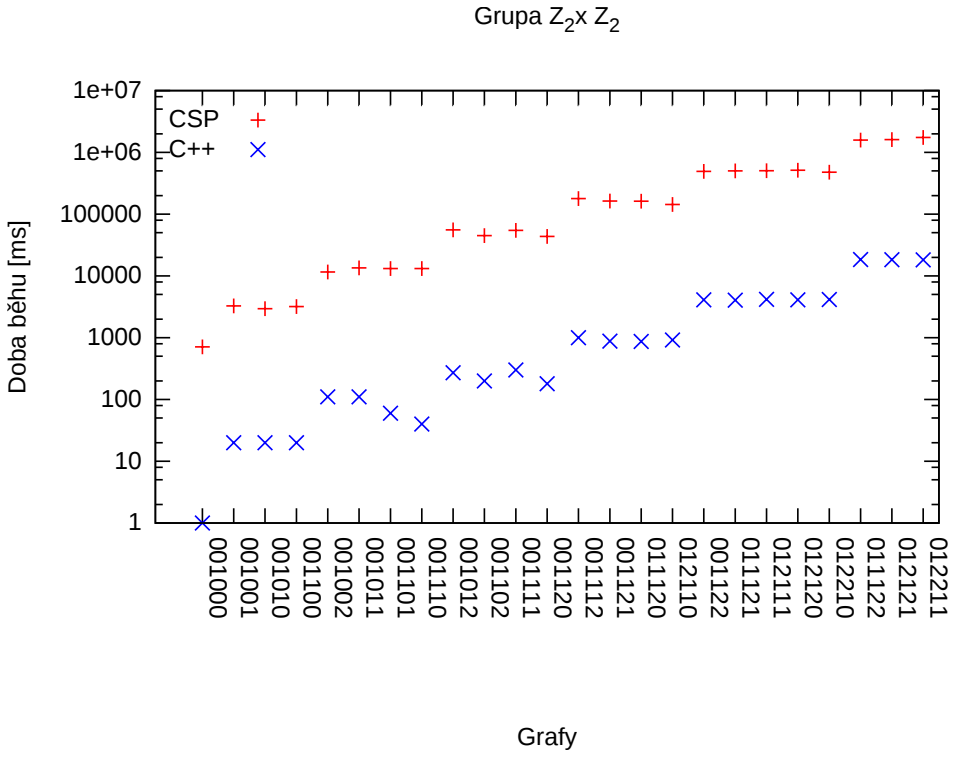
Na Obrázku 6.3 a 6.4 je graf poměru rychlostí obou implementací, ze kterého je vidět, že program napsaný v jazyce C++ je téměř vždy alespoň stokrát rychlejší než program napsaný v Prologu s využitím CSP.

6.2 Podrozdělený graf Cube

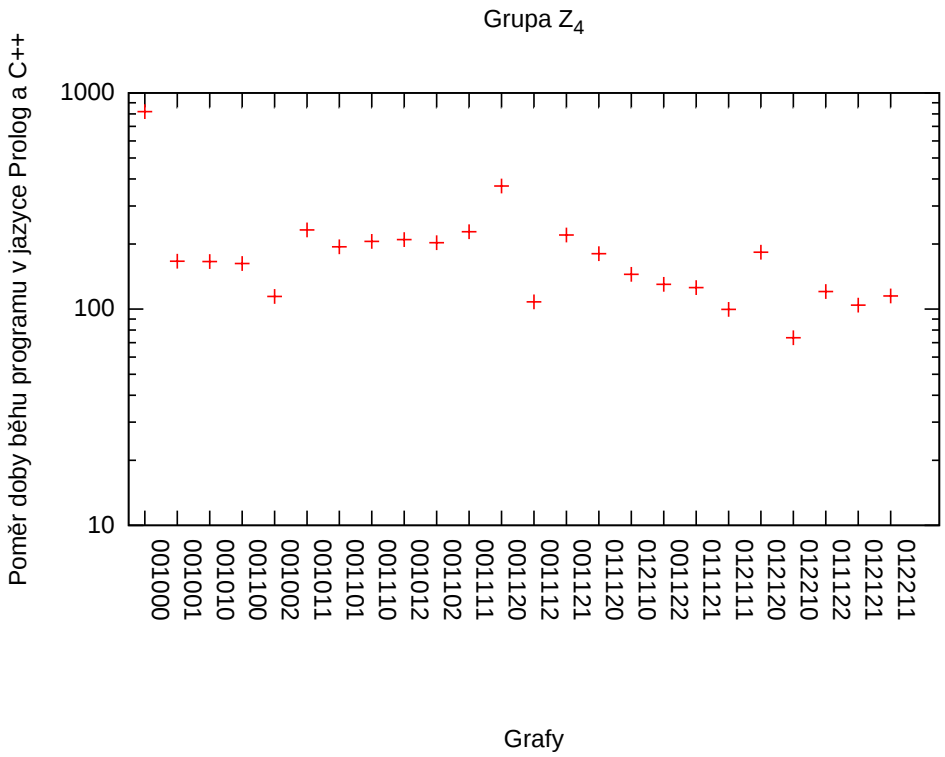
Z grafů vzniklých podrozdělením grafu Cube v Kapitole 5.2.2 jsme náhodně vybrali 7. Tyto grafy jsme nechali otestovat oběma programy. Z grafů na Obrázku 6.5 a 6.6 je zřejmé, že opět závisí na množství podrozdělítek. Tentokrát jsme dostali, že program v C++ je alespoň třicetkrát rychlejší než program napsaný v jazyce Prolog (viz Obrázek 6.7 a 6.8).



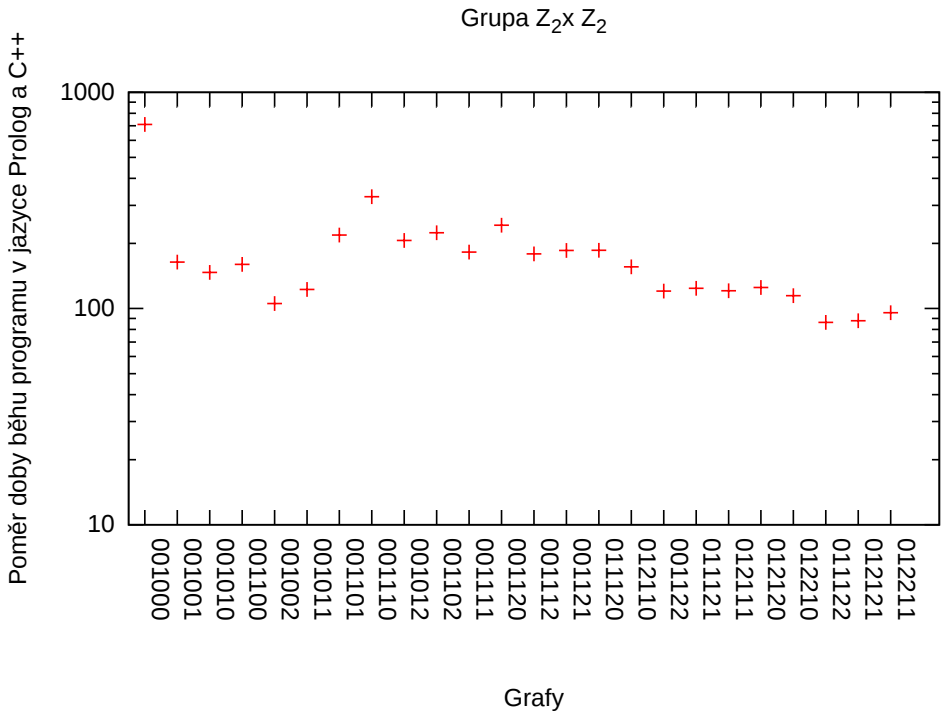
Obrázek 6.1: Graf závislosti doby běhu programů na volbě grafu v grupě $\mathbb{Z}_4$.



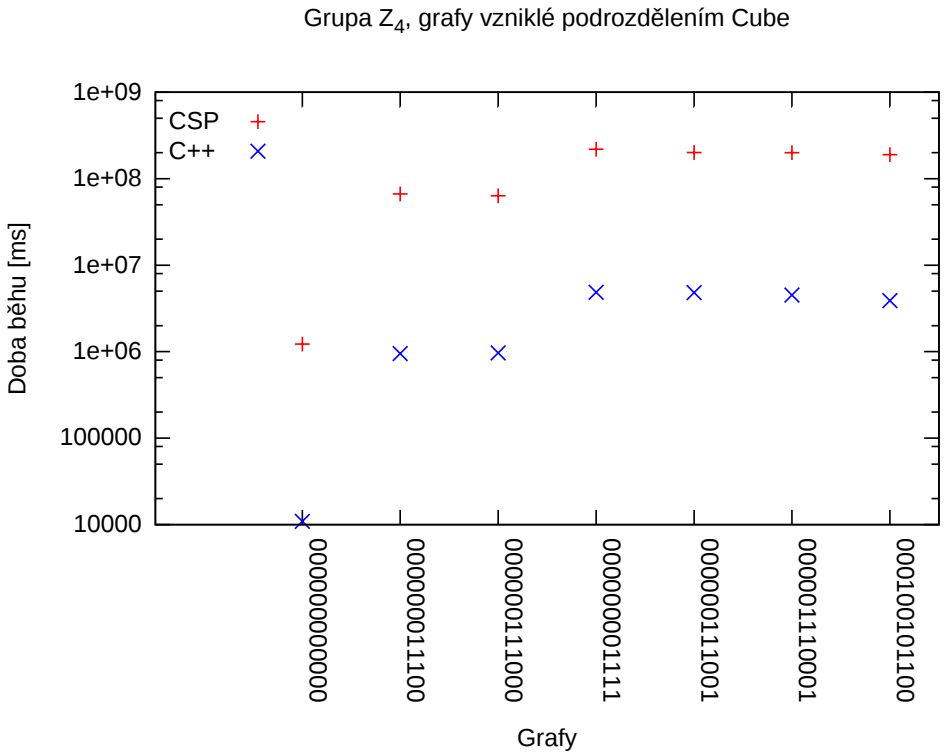
Obrázek 6.2: Graf závislosti doby běhu programů na volbě grafu v grupě $\mathbb{Z}_2^2$.



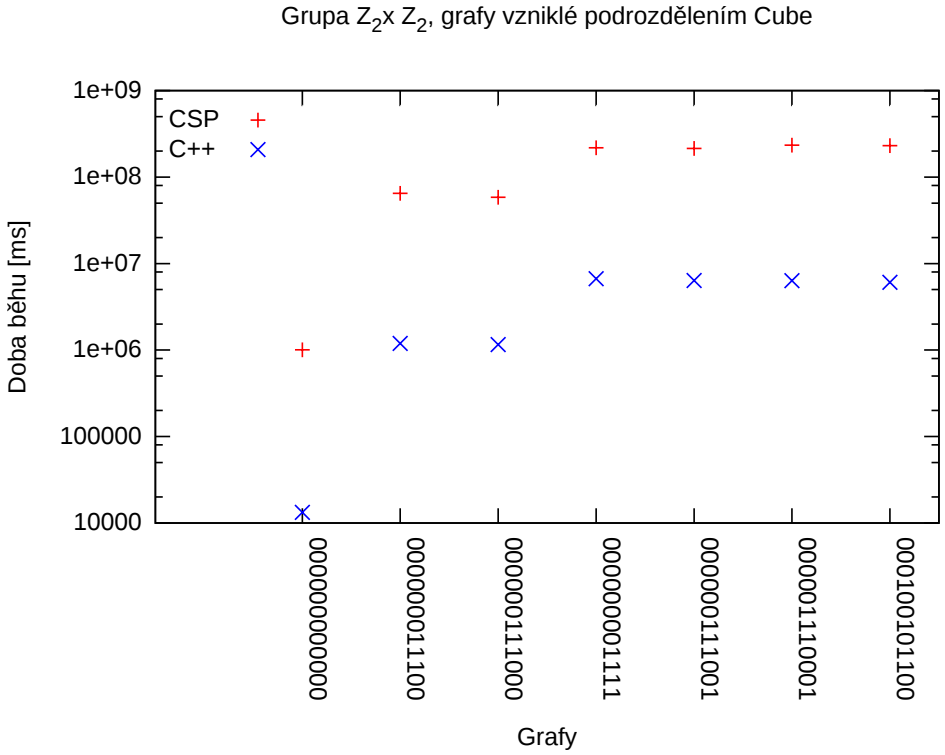
Obrázek 6.3: Graf závislosti poměru doby běhu programů v Prologu a C++ na volbě grafu v grupě $\mathbb{Z}_4$.



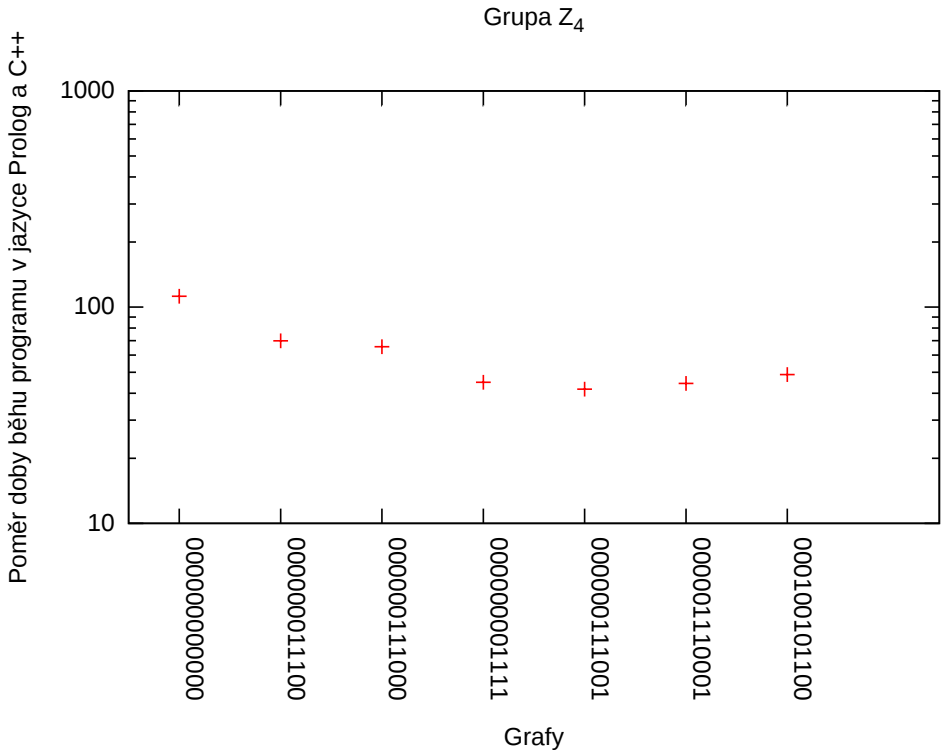
Obrázek 6.4: Graf závislosti poměru doby běhu programů v Prologu a C++ na volbě grafu v grupě $\mathbb{Z}_2^2$.



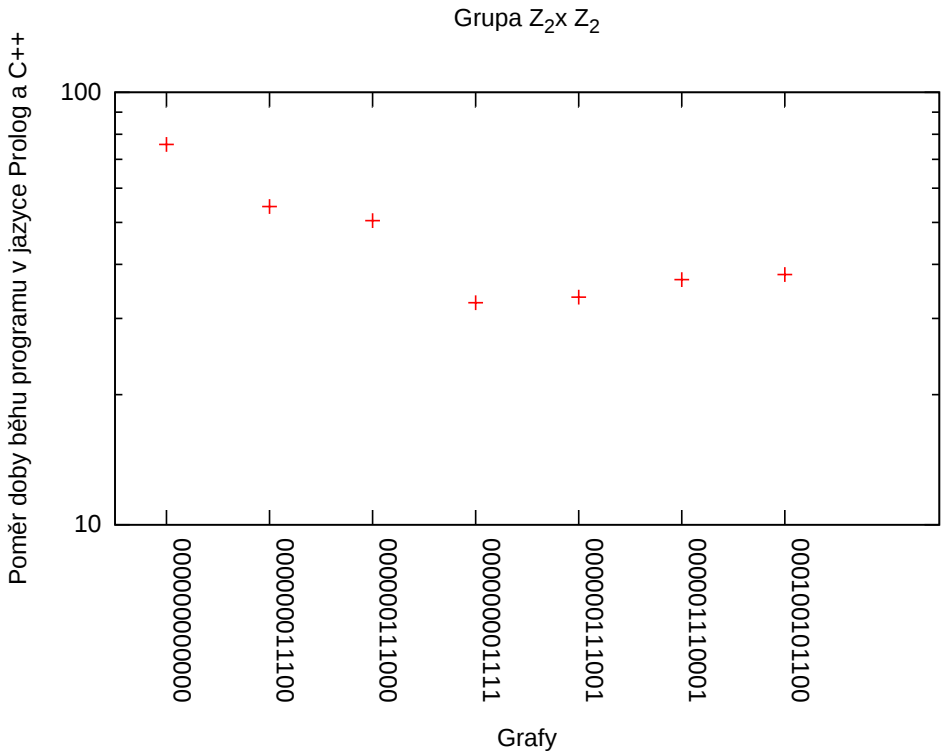
Obrázek 6.5: Graf závislosti doby běhu programů na volbě grafu v grupě $\mathbb{Z}_4$.



Obrázek 6.6: Graf závislosti doby běhu programů na volbě grafu v grupě $\mathbb{Z}_2^2$.



Obrázek 6.7: Graf závislosti poměru doby běhu programů v Prologu a C++ na volbě grafu v grupě $\mathbb{Z}_4$.

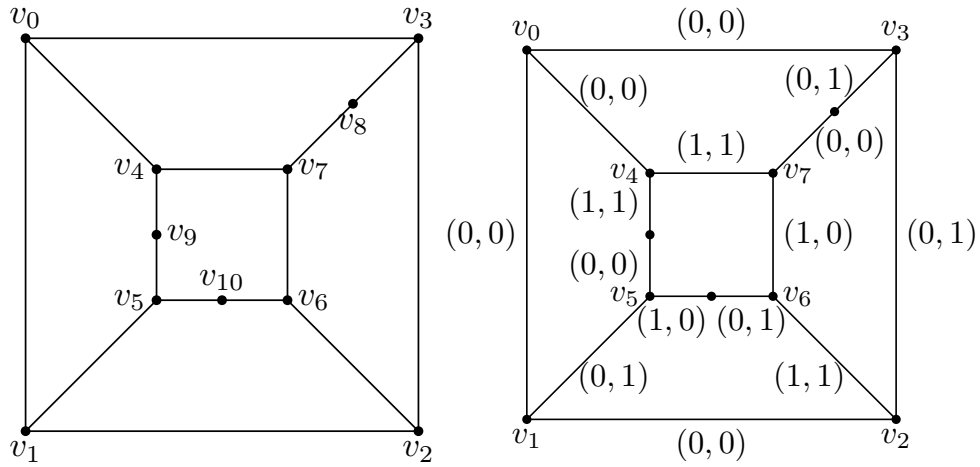


Obrázek 6.8: Graf závislosti poměru doby běhu programů v Prologu a C++ na volbě grafu v grupě $\mathbb{Z}_2^2$.

7. Cube3 není grupově $\mathbb{Z}_2^2$-souvislý

Pomocí programu v C++ jsme našli dva grafy (000000011100 a 000000111000), které jsou $\mathbb{Z}_4$-souvislé a nejsou $\mathbb{Z}_2^2$-souvislé. Vybereme jeden (000000011100) z nich a rozborem případů dokážeme, že není $\mathbb{Z}_2^2$-souvislý. O druhém grafu by se to dokazovalo obdobně.

Jedná se o graf (na Obrázku 7.1), ve tvaru krychle se třemi podrozdělenými hranami. Vrcholy označíme v_0 až v_{10}. Tento graf budeme nazývat *Cube3*.



Obrázek 7.1: Graf krychle s podrozdělenými hranami.

Lemma 42. *Graf Cube3 není grupově $\mathbb{Z}_2^2$-souvislý.*

Důkaz. Graf není grupově souvislý, pokud existuje alespoň jedna konfigurace zakázaných hodnot na hranách, pro kterou nemůže existovat tok. Nechali jsme programem napsaným v C++ vypsat m-tice, pro které neexistovaly tok a z nich jsme si jednu vybrali. Na Obrázku 7.1 vpravo je znázorněna jedna taková m-tice zakázaných hodnot.

Tvrzení 43. *Pro graf Cube3 s danými zakázanými hodnotami (na Obrázku 7.1) neexistuje tok.*

Důkaz. Uvědomíme si, že když hranu podrozdělíme vrcholem, dostaneme cestu se dvěma zakázanými hodnotami. Tudíž se zmenší počet možností, jaká hodnota toku může na této cestě téct.

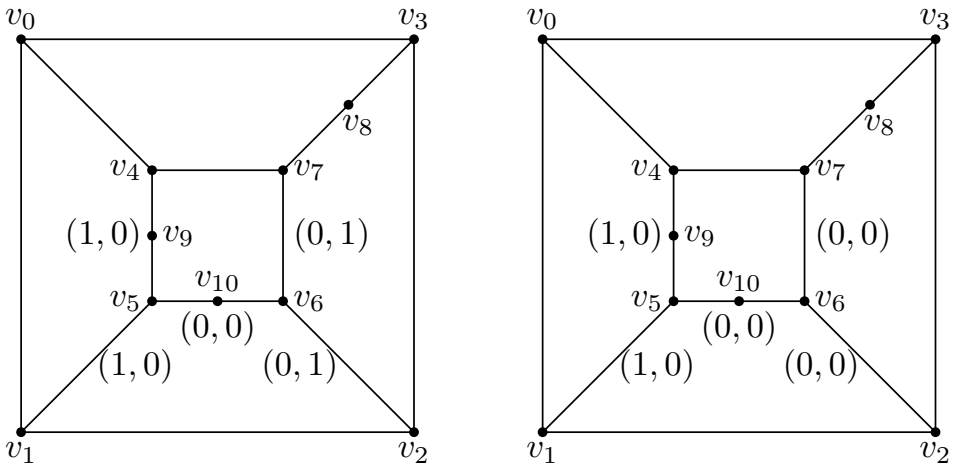
Na počátku si uvědomíme, že na cestách v_4v_5 a v_5v_6 jsou zakázány vždy dvě různé hodnoty. To znamená, že existují pouze čtyři možnosti, jak může na těchto cestách vypadat tok. Víme-li, jak budou vypadat hodnoty toku na těchto dvou cestách, pak rovnou můžeme určit, jak bude vypadat hodnota toku na hraně v_1v_5. Tuto hodnotu určíme tak, aby ve vrcholu v_5 byl součet hodnot roven $(0,0)$. Tabulka 7.1 ukazuje, jak mohou vypadat hodnoty na všech třech hranách. Je-li u hrany v_1v_5 napsáno nelze, znamená to, že hodnota, která tudy měla téct, byla rovna zakázané hodnotě.

Z Tabulky 7.1 je vidět, že existují pouze dvě možnosti. Každou možnost rozebereme zvlášť:

v_4v_5	$(1,0)$	$(1,0)$	$(0,1)$	$(0,1)$
v_5v_6	$(1,1)$	$(0,0)$	$(1,1)$	$(0,0)$
v_1v_5	nelze	$(1,0)$	$(1,0)$	nelze

Tabulka 7.1: Možné hodnoty pro hrany v_4v_5, v_5v_6 a v_1v_5.

1. Cesty v_4v_5 a v_5v_6 mají hodnoty $(1,0)$ a $(0,0)$, po hraně v_1v_5 teče hodnota $(1,0)$. Nyní určíme hodnotu hrany v_7v_6. Pro tuto hranu je zakázaná hodnota $(1,0)$. Tudíž tudy můžou téct hodnoty $(0,1)$, $(0,0)$ a $(1,1)$. Víme, že na cestě v_5v_6 poteče hodnota $(0,0)$, tudíž na hraně v_6v_2 poteče stejná hodnota jako na hraně v_7v_6. Proto musíme vyloučit hodnotu $(1,1)$, neboť tato hodnota je na hraně v_6v_2 zakázaná. Tudíž existují právě dvě možnosti, které jsou zachyceny na Obrázku 7.2.



Obrázek 7.2: Grafy s 5 zafixovanými hodnotami.

Nyní rozebereme první případ, kdy na hranách v_6v_7 a v_6v_2 poteče hodnota $(0,1)$:

(a) Na cestě v_3v_7 jsou zakázané hodnoty $(0,0)$ a $(0,1)$, proto tam mohou téci pouze hodnoty $(1,0)$ a $(1,1)$. To, že na cestě v_3v_7 poteče $(1,0)$ znamená, že by na hraně v_4v_7 musela téct hodnota $(1,1)$, což nelze, neboť tato hodnota je tam zakázaná. Z toho plyne, že tam tato hodnota nemůže téct.

Poteče-li na cestě v_3v_7 hodnota $(1,1)$, pak po hraně v_4v_7 teče $(1,0)$. Sečteme-li hodnoty na hraně v_4v_7 a cestě v_4v_5, pak do vrcholu v_4 musí po hraně v_0v_4 přitéct hodnota $(0,0)$, což není možné, neboť je tam tato hodnota zakázaná.

Z tohoto rozboru plyne, že na hranách v_7v_6 a v_6v_2 nemůže téct hodnota $(0,1)$.

(b) V tomto bodě rozebereme, jak to bude vypadat, když na hraně v_7v_6 (resp. v_6v_2) poteče hodnota $(0,0)$: na hraně v_4v_7 a cestě v_7v_3 dohromady teče $(0,0)$. Uvážíme-li zakázané hodnoty na cestě v_7v_3, pak dospějeme k tomu, že tam může téct hodnota $(1,1)$ nebo $(1,0)$. V prvním

případě dostaneme, že na hraně v_4v_7 by musela téct hodnota $(1,1)$, což nelze protože tato hodnota je zde zakázaná.

Bude-li hodnota toku na ceste v_7v_3 rovna $(1,0)$, pak dopočítáme hodnotu na hraně v_4v_7 (bude rovna $(1,0)$). Z toho, že hraně v_0v_4 a cestě v_4v_5 jsou přiřazeny stejné hodnoty, plyne, že na hraně v_0v_4 by musela téct hodnota $(0,0)$, což není možné, protože tato hodnota je zde zakázaná.

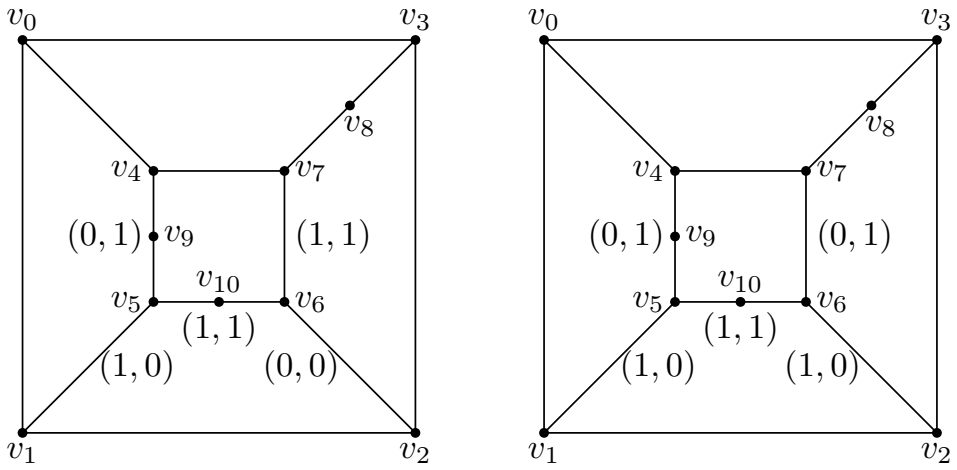
Z tohoto rozboru je vidět, že na hranách v_6v_7 a v_6v_2 nemůže téct hodnota $(0,1)$ ani $(0,0)$.

2. Rozebereme, jak to bude vypadat, když na cestě v_4v_5 poteče $(0,1)$ a na cestě v_5v_6 hodnota $(1,1)$. V Tabulce 7.2 jsou uvedeny obě možnosti, jak mohou vypadat hodnoty na hranách v_7v_6 a v_6v_2. Na hraně v_6v_2 by mohla téct ještě hodnota $(0,1)$, ale to by vynutilo na hraně v_6v_7 hodnotu $(1,0)$, která je tam zakázaná a nepřipadá tedy v úvahu.

v_6v_7	$(1,1)$	$(0,1)$
v_6v_2	$(0,0)$	$(1,0)$

Tabulka 7.2: Možné hodnoty pro hrany v_6v_7 a v_6v_2.

Na Obrázku 7.3 jsou dvě možnosti, jak mohou vypadat hodnoty toku na dosud ohodnocených pěti hranách.



Obrázek 7.3: Grafy s 5 zafixovanými hodnotami.

(a) Podíváme se na první případ: hranou v_6v_7 (resp. v_6v_2) teče hodnota $(1,1)$ (resp. $(0,0)$). Zaměříme se na cestu v_7v_3. Na této cestě mohou téct pouze dvě různé hodnoty: $(1,0)$ nebo $(1,1)$. První z nich ($(1,0)$) určí hodnotu hrany v_4v_7 jako $(0,1)$ a tudíž i hranu v_0v_4, která by musela být $(0,0)$, což nelze, neboť tato hodnota je tam zakázaná.

Nyní předpokládejme, že na cestě v_7v_3 poteče hodnota $(1,1)$. Díky této informaci víme, že na hraně v_4v_7 poteče $(0,0)$ a hranou v_0v_4 hodnota $(0,1)$.

Vybereme hranu v_0v_3 a postupně jí přiřadíme tyto hodnoty: $(0,1)$, $(1,0)$ a $(1,1)$. V prvním případě dostaneme, že na hraně v_0v_1 může téct hodnota $(0,0)$, což není možné, neboť je tam tato hodnota zakázaná. Přiřadíme-li hraně v_0v_3 hodnotu $(1,0)$, pak vynutíme na hraně v_2v_3 hodnotu $(0,1)$, která je tam zakázaná. Poslední možností je hodnota $(1,1)$. Nyní sice dopočítáme hodnotu toku na hraně v_2v_3 na hodnotu $(0,0)$, která je povolená. Ale pro hranu v_1v_2 dostáváme hodnotu $(0,0)$, která je na této hraně zakázaná.

Z toho plyne, že pro hodnotu toku $(1,1)$ (resp. $(0,0)$) na hraně v_6v_7 (resp. v_6v_2) neexistuje tok na zkoumaném grafu.

(b) Na hraně v_7v_6 (resp. v_6v_2) může téct také hodnota $(0,1)$ (resp. $(1,0)$). Na cestě v_3v_7 jsou zakázané hodnoty $(0,0)$ a $(0,1)$, a proto tam může téct hodnota $(1,0)$ nebo $(1,1)$. Sečteme-li hodnotu $(1,0)$ (pro cestu v_3v_7) s hodnotou $(0,1)$ na hraně v_6v_7, pak dostaneme, že na hraně v_4v_7 by musela téct hodnota $(1,1)$, ale to nelze, neboť je pro tuto hranu zakázaná.

Pokud by na cestě v_3v_7 tekla hodnota $(1,1)$, pak jsme schopni dopočítat hodnotu na hraně v_4v_7 (resp. v_0v_4): $(1,0)$ (resp. $(1,1)$). Nyní nám chybí určit pouze zbývající čtyři hodnoty na hranách v_0v_1, v_1v_2, v_2v_3 a v_3v_0.

Vezmeme hranu v_0v_3 a přiřadíme jí postupně všechny tři možné hodnoty toku: $(1,0)$, $(0,1)$ a $(1,1)$. Je zřejmé, že na této hraně nemůže téct první ani třetí hodnota, neboť u první hodnoty bude muset na hraně v_2v_3 téct hodnota $(0,1)$ (ale to nelze) a v druhém případě poteče na hraně v_0v_1 hodnota $(0,0)$, která je na této hraně zakázaná. Poteče-li na hraně v_0v_3 hodnota $(0,1)$, pak dopočítáme, že na hraně v_0v_1 poteče hodnota $(1,0)$. Z toho určíme, že na hraně v_1v_2 by musela téct hodnota $(0,0)$, která je zde zakázaná.

Rozebrali jsme všechny možnosti, jak může vypadat tok, když na hraně v_6v_7 (resp. v_6v_2) poteče hodnota $(0,1)$ (resp. $(1,0)$).

Rozebrali jsme všechny možnosti, jak mohou vypadat hodnoty toku na hranách a zjistili jsme, že na grafu Cube3 se zakázanými hodnotami nemůže existovat tok v grupě $\mathbb{Z}_2^2$. □

Z Tvrzení 43 plyne, že existuje zakázané ohodnocení zadaného grafu, pro který neexistuje tok. Graf tudíž není grupově $\mathbb{Z}_2^2$-souvislý.

□

8. Závěr

Napsala jsem program na testování grupové souvislosti grafů v jazyce C++. Pomocí tvrzení a vět jsme stanovili jednoduché nutné (neexistence mostu, omezení na délku cesty) a postačující (hranová 4-souvislost) podmínky pro grupovou souvislost. Vybrali jsme několik grafů (K_4, Cube, Dvojtrojúhelník a jiné) a vytvořili jejich podrozdělení. Všechny tyto grafy jsme nechali otestovat na grupovou $\mathbb{Z}_2^2$- a $\mathbb{Z}_4$-souvislost. Mezi grafy, které vznikly podrozdělením Cube, jsme našli dva (000000011100 a 000000111000), které jsou $\mathbb{Z}_4$- a nejsou $\mathbb{Z}_2^2$-souvislé. $\mathbb{Z}_4$-souvislost jsme potvrdili druhou nezávislou implementací, tentokrát v jazyce Prolog s využitím technik programování s omezujícími podmínkami. Při dokazování, že tento graf není $\mathbb{Z}_2^2$-souvislý, jsme využili toho, že stačí najít jednu m-tici zakázaných hodnot a dokázat, že pro ni neexistuje tok. Vybrali jsme jednu z m-tic zakázaných hodnot, pro kterou dle programu v C++ neexistuje tok a pomocí rozboru případů dokázali, že pro ni žádný tok opravdu neexistuje.

Rovněž jsme porovnali obě nezávislé implementace a dospěli k závěru, že program v jazyce C++ je vždy několikrát (u K_4 stokrát, u Cube třicetkrát) rychlejší než program v Prologu.

Na tuto práci by se dalo navázat zkoumáním, zda existují grafy, které jsou $\mathbb{Z}_2^2$- a nejsou $\mathbb{Z}_4$-souvislé, případně dokázat, že žádný takový graf neexistuje, což by znamenalo, že $\mathbb{Z}_2^2$-souvislé grafy jsou podmnožinou $\mathbb{Z}_4$-souvislých. Také by se dalo bez použití výpočetní techniky ověřit, že dva nalezené grafy jsou $\mathbb{Z}_4$-souvislé.

9. Příloha

Na přiloženém CD se ve složce *Programy* nalézají programy na testování grupové souvislosti, které vznikly během zkoumání tématu. Programy jsou rozděleny do tří složek – *C++*, *Prolog* a *Pomocné*.

9.1 C++

Program na testování grupové souvislosti je napsán v jazyce C++ a je rozdělen do dvou souborů – *toky.cpp* a hlavičkového souboru *grupy.h*.

V hlavičkovém souboru se nachází definice grup (GrupaZ a GrupaN) a funkce popisující aritmetické operace nad nimi. V rámci programu můžeme používat cyklické grupy $\mathbb{Z}_k$ (k tomu slouží třída GrupaZ) a jejich mocniny, tj. grupy $\mathbb{Z}_k^l$ (třída GrupaN).

9.1.1 Vstupy

Program očekává na vstupu jako parametr grupu, pro kterou má provést výpočet. V případě grupy $\mathbb{Z}_4$ je to číslo 4, u grupy $\mathbb{Z}_2^2$ použijeme pouze číslo 2. V případě potřeby se program dá upravit tak, aby testoval grupovou souvislost pro jakoukoliv grupu, která se dá vyjádřit jako GrupaZ či GrupaN.

Na standardním vstupu program očekává graf. Na prvním řádku jsou čísla n (počet vrcholů) a m (počet hran). Na dalších m řádcích jsou dvojice vrcholů, které značí, že mezi nimi vede hrana. Vrcholy jsou pojmenované čísly od 0 do $n-1$.

Příklad vstupu

```
7 9
0 1
1 2
0 4
4 2
0 5
5 3
1 3
2 6
6 3
```

9.1.2 Výstupy

Na standardním výstupu se vypíšou dva řádky – první z nich obsahuje počet neuspokojených m-tic a druhý dobu běhu v milisekundách. Je-li počet neuspokojených m-tic roven 0, pak to znamená, že graf je grupově souvislý.

Příklad

```
lysiii@boruvka:~/programování/diplomka$ ./gr_souv.out 4 < 001101.in
```

```
pocet neuspokojenych m-tic: 768
doba behu: 4090
```

9.2 Prolog s CSP

Zdrojový kód programu je v souboru s názvem *gr_souv.pl*. Je interpretovatelný Sicstus Prologem (`http://sicstus.sics.se/`). Využívá knihovnu *clpfd*, která není součástí jiných interpretrů Prologu.

Program je primárně určený pro grupu $\mathbb{Z}_4$. V případě, že chce uživatel program použít pro jinou grupu, musí nadefinovat pravidla pro sčítání na této grupě výčtem všech trojic (sčítanec, sčítanec, součet). Tato pravidla se umístí do predikátu *table* v proceduře *secti2hodnoty*. Ve zdrojovém kódu jsem uvedla tato pravidla pro grupu $\mathbb{Z}_2^2$.

9.2.1 Vstupy

Na vstupu předpokládáme graf tohoto tvaru: [edge(V1,V2), edge(V2,V3),...], kde V_i a V_j jsou koncové vrcholy hrany.

Příklad vstupu

Vytvoříme unární predikát pro graf – graph_name(Graph). Příklad unárního predikátu pro graf K_4:

```
k_4([edge(0,1),edge(1,2),edge(2,3),edge(3,1),edge(0,2),edge(3,0)]).
```

Puštění programu

Existují dvě možnosti, jak pustit program lišící se podle toho, co po programu chceme.

1. graph_name(Graph), run_all_CSP(Graph, Time, Bag, L) – v proměnné Bag na konci budeme mít všechny m-tice zakázaných hodnot, pro které neexistuje tok; L je počet zakázaných m-tic; Time značí dobu běhu programu v milisekundách.

2. graph_name(Graph), run_first_CSP(Graph, Time, Sol) – výsledkem volání tohoto predikátu je první nalezená m-tice zakázaných hodnot; Time značí dobu běhu do nalezení určitého řešení.

Výstup

Kromě klasického výpisu v interpretru, používáme vestavěný predikát *write* k formátování výpisu na standardním výstupu. Výstup je stejný jako v případě programu v C++. Na prvním řádku je počet neuspokojených m-tic, na druhém je doba běhu programu v milisekundách.

9.3 Pomocné programy

Kromě již popsaných programů pro testování grupové souvislosti jsme naprogramovali další drobné programy, které měli za cíl připravit data. Ke každému z nich uvedeme v několika slovech k čemu se používají, případně s jakými parametry se pouští.

Programy s příponou *pl* jsou napsány v jazyce Perl.

GenK4.pl

Vytvoří všechny posloupnosti délky 6 (popsáno v Kapitole 5.2.1), které využije ke generování všech podrozdělení grafu K_4. Poté pro každou z těchto posloupností vyrobí soubor s názvem obsahujícím posloupnost a *.in* (např. 000000.in). V tomto souboru bude graf v reprezentaci, kterou jsme popsali v Kapitole 9.1.

Graph_symetries.sage

Program napsaný v jazyce Python s matematickým rozšířením Sage. Pro graf Cube určí všechny jeho symetrie a vypíše seznam dvanáctic nul a jedniček oddělených čárkami. Každá posloupnost začíná a končí kulatými závorkami (př. (0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0),), na konci každého řádku je čárka. Mezi grafy, které tímto způsobem dostaneme, nejsou žádné dva vzájemně izomorfní.

GenQ3.pl

Na standardním vstupu očekává posloupnosti délky 12 (posloupnost podrozdělítek pro graf Cube, popsáno v Kapitole 5.2.2) z nichž vybere ty, které neobsahují podgrafy popsané v Kapitole 5.2.2 a vytvoří z nich grafy v reprezentaci popsané v Kapitole 9.1. Tyto grafy uloží po jednom do souboru s názvem posloupnosti a *.in*.

Posloupnosti na vstupu jsou očekávány jako posloupnost 0 a 1 oddělených čárkami blíže popsaná u programu *Graph_symetries.sage*. Na konci každého řádku je čárka.

ConvertC_prolog.pl

Tento program slouží ke změně reprezentace grafu z reprezentace pro program v C++ na seznam hran pro program v jazyce Prolog. Jako parametry program dostane jméno souboru, ve kterém je uložený graf v reprezentaci pro C++. Dále dostane jako parametr název soubor, ve kterém je zdrojový kód napsaný v Prologu. Výsledkem je funkční kód v jazyce Prolog, který se dá pustit v interpretru Sicstus Prologu. Tento kód zajistí načtení grafu a procedur pro testování grupové souvislosti do paměti interpretru a spuštění testu na grupovou souvislost.

Gen.cpp

Program v jazyce C++, který načte graf v podobě popsané v Kapitole 9.1 a vytvoří všechna možná podrozdělítka (0, 1 nebo 2) na hranách. Otestuje souvislost grafů po odebrání podrozdělených hran. Všechny souvislé grafy uloží do souborů

v reprezentaci popsané v Kapitole 9.1. Grafy nazve podle posloupnosti, ze které byly vytvořeny (stejně jako v Kapitole 9.3).

Seznam použité literatury

[1] Appel, Kenneth, Haken, Wolfgang, Every planar map is four colorable. Part I, *Discharging, Illinois J. Math.*, 1977, pp. 429-490.

[2] Diestel, Reinhard, *Graph Theory.* Springer-Verlag, New York, 2000. ISBN 0-387-98976-5.

[3] Jaeger, François. Flows and generalized coloring theorems in graphs. *Journal of combinatorial theory*, Series B, 1979, pp. 205-216.

[4] Jeager, François, Linial, Nathan, Payan, Charles, Tarsi, Michael. Group Connectivity of Graphs - A Nonhomogeneous Analogue of Nowhere-Zero Flow Properties. *Journal of Combinatorial Theory*, Series B, 1992, pp. 165-182.

[5] Holyer, Ian, The NP-completeness of edge-coloring. *SIAM J. Comput. 10*, 1981, pp. 718–720.

[6] Hungerford, Thomas W. *Algebra.* Springer-Verlag, New York, 1974. ISBN 3-540-90518-9.

[7] Lai, Hong-Jian, Li, Xiangwen, Shao, Yehong, Zhan, Mingquan, Group Connectivity and Group Colorings of Graphs – A Survey, *Acta Mathematica Sinica, English Series*, 2011, pp. 405-434.

[8] Lovász, László Miklós, Thomassen, Carsten, Wu, Yezhou, Zhang, Cun-Quan. Nowhere-zero 3-flows and modulo k-orientations, *Journal of Combinatorial Theory, Series B, Vol. 103*, 2013, pp. 587-598.

[9] Matoušek, Jiří, Nešetřil, Jaroslav. *Kapitoly z diskrétní matematiky*, Nakladatelství Karolinum, Praha, 2007.

[10] Robertson, Neil, Seymour, Paul, Thomas, Robin. Tutte's Edge-Colouring Conjecture. *Journal of combinatorial theory*, Series B, Vol. 70, 1997, pp. 166-183.

[11] Steinberg, Richard. Tutte's 5-flow conjecture for the projective plane. *Journal of Graph Theory*, 1984, pp. 277-285.

[12] Seymour, Paul D.. Nowhere-Zero 6-Flows. *Journal of combinatorial theory*, Series B 30, 1981, pp. 130-135.

[13] Thomassen, Carsten. The weak 3-flow conjecture and the weak circular flow conjecture. *Journal of combinatorial theory*, Series B, 2012, pp. 521–529.

[14] Zhang, Cun-Quan. *Integer Flows and Cycle Covers of Graphs.* Marcel Dekker, New York, 1997. ISBN 0-8247-9790-6.

[15] Tutte, William, A contribution on the theory of chromatic polynomial, *Canad. J. Math.*, 6, 1954, pp. 80-91.