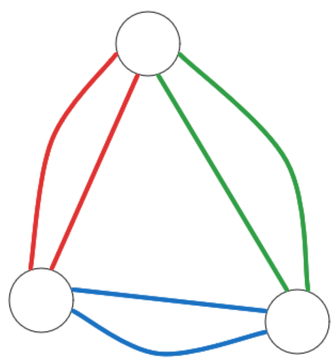
Report ročníkový projekt zimný semester 2024/2025

Alex Diko
`diko3@uniba.sk`

Pojmy: *Graf* je dvojica $G = (V, E)$ disjunktných množín V a E spolu so zobrazením $f : E \to V \cup \binom{V}{2}$. Prvky V sa nazývajú *vrcholy* grafu G a prvky E jeho *hrany*. f priraďuje každej hrane jej *konce* (teda povolujeme multigrafy). Ak je hrane priradený iba jeden vrchol, nazveme ju *slučka*. Hranu s koncovými vrcholmi x, y označíme xy. Množinu hrán grafu G označujeme ako $E(G)$. Vrchol v je *incidentný* s hranou e, ak $v \in f(e)$. *Stupeň vrcholu* je počet hrán s ním incidentných, pričom slučka pridáva k stupni vrchola 2. Dve hrany sú *susedné*, ak existuje vrchol, ktorý je s obidvomi incidentný. Ak všetky vrcholy grafu majú stupeň k, graf sa nazýva k*-regulárny*. *Cesta* je neprázdny graf $P = (V, E)$, kde $V = \{x_0, x_1, \ldots, x_k\}$ a $E = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\}$, pričom všetky x_i sú rôzne. Túto cestu označíme $x_0x_1 \ldots x_k$. Ak $P = x_0 \ldots x_{k-1}$ je cesta, potom graf $C := P + x_{k-1}x_0$ sa nazýva *kružnica*. *Dĺžka* kružnice je počet jej hrán. Graf $G'(V', E')$ je *podgrafom* grafu G ak $V' \subseteq V$ a $E' \subseteq E$. *Rozklad* množiny $E(G)$ je množina $R = \{R_1, \ldots, R_k\}$, kde R_i sú navzájom disjunktné a zjednotenie $\cup R$ všetkých množín $R_i \in R$ sa rovná $E(G)$. [1]

Kružnicovou dekompozíciou grafu nazveme rozklad $E(G)$ na kružnice, ktoré nezdieľajú žiadnu hranu. Ak každá kružnica tohto rozkladu má párnu dĺžku, nazveme tento rozklad *párnym* (ďalej *ECD* z anglického *Even cycle decomposition*). Pre ECD grafu zafarbíme každú jeho kružnicu tak, aby kružnice so spoločným vrcholom nemali rovnakú farbu. Potom zjednotenie kružníc v každej farebnej triede bude 2-regulárny podgraf pôvodného grafu. Ak najmenší počet farieb potrebných na takéto zafarbenie je m potom ECD rozklad má *veľkosť* m. [2]



Obr. 1: ECD rozklad veľkosti 3 na červenú, modrú a zelenú kružnicu

Práca za semester: Do grafovej knižnice `ba-graph` som v jazyku `c++` naimplementoval backtracking (prehľadávanie s návratom) algoritmus, ktorý nájde veľkosť ECD grafu (súbor `ecd.hpp`). Pseudokód algoritmu je uvedený nižšie.

```
function NájdiECD(graf)
    inicializuj počet farebný tried na 0
    SkúsZačaťNovúKružnicu
    if žiadne ECD sa nenašlo then
        return neexistuje ECD
    else
        return najmenšie ECD
    end if
end function
```

```
function SkúsZačaťNovúKružnicu
    if všetky hrany sú priradené farebnej triede then
        porovnaj veľkosť tohto ECD s doteraz najmenším nájdeným, ak je menšie zapamätaj si prira-
        denie hrán do farebných tried
        return
    else
        h ← zober nejakú nepriradenú hranu
        for all f ∈ aktuálne farebné triedy do
            skús priradiť h do f a zapamätaj si, že h bude v kružnici na párnej pozícii
            SkúsPokračovaťVAktuálnejKružnici(h)
        end for
        skús priradiť h do novej farebnej triedy a zapamätaj si, že h bude v kružnici na párnej pozícii
        if počet farebných tried je väčší alebo rovný ako aktuálne najmenšie nájdené ECD then
            odmietni aktuálne riešenie
            return
        end if
        SkúsPokračovaťVAktuálnejKružnici(h)
    end if
end function

function SkúsPokračovaťVAktuálnejKružnici(aktuálnaHrana)
    ▷ Over či je aktuálnaHrana správne zafarbená. V korektnej ECD musia byť práve 2 susedné hrany
      v rovnakej farebnej triede a na pozíciach inej parity ako aktuálnaHrana ◁
    p ← 0        ▷ Počet susedných hrán v rovnakej farebnej triede ale na pozíciach rôznej parity
    for all h ∈ hrany susedné s aktuálnaHrana do
        if h je v rovnakej farebnej triede ako aktuálnaHrana then
            if h je na pozícii rovnakej parity ako aktuálnaHrana then
                odmietni aktuálnu kružnicu
                return
            else
                p ← p + 1
            end if
        end if
    end for
    if p > 2 then
        odmietni aktuálnu kružnicu
        return
    else if p = 2 then
        SkúsZačaťNovúKružnicu        ▷ Bez sporu sa nám podarilo vytvoriť kružnicu párnej dĺžky
        return
    else
        for all h ∈ hrany susedné s aktuálnaHrana a aktuálne nepriradené do farebnej triedy do
            skús priradiť h do rovnakej farebnej triedy ako aktuálnaHrana ale na pozíciu inej parity
            SkúsPokračovaťVAktuálnejKružnici(h)
        end for
    end if
end function
```

K algoritmu sú urobené aj testy (`test_ecd.cpp`). Skúšal som viacero variant algoritmu (napríklad, že iterujeme cez všetky možné veľkosti ECD od najmenšej po najväčšiu a pre každú veľkosť skúšame nájsť jedno zafarbenie. Ak ho nájdeme, skončíme.), ale tento bol najrýchlejší. Pri výbere hrany na riadku 16 máme viacej možností. Skúšal som program, kde je poradie výberu hrán náhodne a spustenia programu na rovnakých vstupoch trvali rôzne dlho. Teda to, v akom poradí sú tam vyberané hrany, ovplyvňuje rýchlosť. Skúšal som heuristiku, kde sa vždy vyberie hrana, ktorá má najviac susedov priradených do nejakej farebnej triedy. Rýchlejší bol však variant, kde sa vrcholy vyberajú na základe poradia, v ktorom sú prehľadávané do šírky BFS algoritmom od prvej hrany, ktorá sa zafarbí. Varianty backtrackingu sa dajú pozrieť ako rôzne `git` vetvy.

Práca na letný semester: ECD by sa malo dať nájsť pomocou SAT solvera.

Referencie

[1] Reinhard Diestel. *Graph theory*. Springer (print edition); Reinhard Diestel (eBooks), 2024.

[2] Analen A. Malnegro a Kenta Ozeki. „H-colorings for 4-regular graphs“. In: *Discrete Mathematics* 347.3 (2024), s. 113844. ISSN: 0012-365X. DOI: `https://doi.org/10.1016/j.disc.2023.113844`. URL: `https://www.sciencedirect.com/science/article/pii/S0012365X23005307`.