# Genetický algoritmus - barvení grafu

Sebastián Krajňák - xkrajn05

Březen 2022

## Popis problému

Problémom farbenia grafu (Graph Colouring Problem) je problém ako nafarbiť množinu vrcholov spojených hranami čo najmenším počtom farieb tak, aby žiadne dva susedné vrcholy nemali rovnakú farbu. Tento najmenší počet farieb potrebný na ofarbenie všetkých vrcholov bez kolízií sa nazýva chromatické číslo. Tento problém je NP-úplný (neexistuje spôsob ako efektívne nájsť jeho optimálne riešenie) a teda je vhodný pre optimalizáciu pomocou evolučného algoritmu.

## Implementačné prostredie

Implementáciu riešenia problému budem riešiť v jazyku Python 3 s pomocou knižníc `numpy` napr. pri výpočtoch alebo generovaní náhodných čísel pomocou routiny `random` a `matplotlib` na vykresľovanie grafu priebehu fitness funkcie. Pri implementácii genetického algoritmu, výberu rodičov a kríženia budem používať už implementované algoritmy z 2. cvičenia predmetu EVO, ktorých autorom je Ing. Martin Hurta[1].

## Spôsob riešenia

Pre riešenie problému budem používať podklady z článku Genetic Algorithm Applied to the Graph Coloring Problem (Musa M Hindi, Roman Yampolskiy)[2]. Pri riešení problému budem používať viacero grafov so známym chromatickým číslom a pomocou evolučného algoritmu sa pokúšať nájsť kombináciu farieb jednotlivých vrcholov pre dané chromatické číslo. Tieto grafy sú voľne dostupné od DIMACS (Centrum for Discrete Mathematics and Theoretical Computer Science). Pre riešenie problému pomocou evolučného algoritmu je potrebné:

- navrhnúť reprezentáciu chromozómu,
- vytvoriť fitness funkciu,
- vymyslieť genetické operátory ako sú kríženie a mutácia,
- zvoliť vhodný spôsob selekcie, vytvorenia ďalšej generácie a ukončenia EA[3].

Graf samotný bude reprezentovaný pomocou matice susednosti vrcholov, pričom 1 v matici znamená, že vrcholy sú susedné. Keďže sa jedná o neorientovaný graf, bude matica symetrická. Chromozóm bude potom reprezentovať jednotlivé vrcholy, podľa indexu v chromozóme, a hodnota na danom indexe bude reprezentovať farbu. Cieľom optimalizácie je minimalizovať počet kolízií susedných vrcholov s rovnakou farbou teda fitness funkcia bude penalizovať každú kolíziu hodnotou 1 a výsledná fitness hodnota je suma všetkých kolízií. Pre selekciu rodičov použijem turnajovú selekciu. Kríženie bude následne jednobodové a mutácie budú

[1] `https://drive.google.com/file/d/1IppOWW2MNuabvtei5a0yQOpWIZu0TPzy/edit?pli=1`
[2] `https://www.researchgate.net/publication/256169514`
[3] evolučný algoritmus

inšpirované z vyššie zmienenej práce. Nová generácia bude tvorená len z potomkov, teda žiaden z rodičov z predošlej generácie sa nedostane do novej. Keďže chceme nájsť kombináciu farieb pre všetky vrcholy bez kolízií, tak ukončovacou podmienkou bude nulová fitness hodnota chromozómu.

## Čiastkové výsledky

V čase písania pojednania bol implementovaný celý algoritmus, metódy mutácie podľa už zmienenej práce, turnajová selekcia a jednobodové kríženie. Vykonal som 30 nezávislých behov pre graf `queen5_5.col`, ktorý má 23 vrcholov a 160 hrán, s chromatickým číslom 5 s nasledujúcimi parametrami EA:

- počet generácií: 1000,
- veľkosť populácie: 50,
- pravdepodobnosť kríženia: 0.2,
- pravdepodobnosť mutácie: 0.8.

Výsledná celková úspešnosť algoritmu bola 0.3 s priemernou hodnotou fitness 4.841460785581857. Výsledný priebeh fitness funkcií jednotlivých behov je zobrazený na Obr.1.

Obr. 1: Priebeh hodnôt fitness jednotlivých behov