Jednoduchý genetický algoritmus

Úlohou bolo pomocou evolučného algoritmu vytvoriť jedinca, ktorý by v sebe obsahoval čo najviac po sebe striedajúcich čísiel 1 a 0. Riešenie je v zdrojovom súbore *ExampleFitnessFunction.java*. Na vygenerovanie takéhoto jedinca som použil cyklus, ktorý iteruje po 2 prvkoch a v každej iterácii kontroluje, či je prvý prvok rovný jednotke a druhý rovný nule. Riešením sa vo veľa prípadoch dostaneme k optimálnemu jedincovi, no v niektorých sa tiež dostávame do lokálnych maxím, z ktorých sa dostaneme len vhodnou mutáciou.

Alternatívy :

- (Horšie) Pri písaní funkcie mi v prvom rade napadlo iterovať po jednom prvku. V takomto prípade síce berieme do úvahy aj to, že medzi opakujúcimi 10 sekvenciami môže byť ľubovoľný odpad, no zároveň vygenerovanie optimálneho jedinca trvá dlhšie než v mojom riešení. Dôvod je hlavne kvôli možnej veľkej zmene (pozície núl a jedničiek) medzi horším a lepším jedincom, ktorý vedie k veľkému oscilovaniu jedincov.
- Podobne by sa dala nastaviť počiatočná fitness na dĺžku *genes.length*, iterovať po 2 a pri každej rozdielnej sekvencii odpočítať 1. V takomto prípade by malo byť riešenie ekvivalentné môjmu danému.

Zmeny parametrov

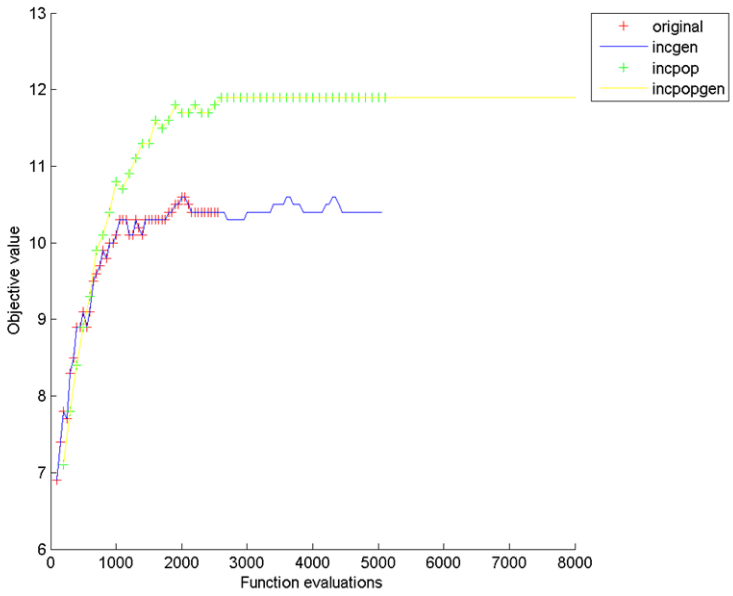
Na nižšie uvedenom obrázku je vidieť vývoj priemeru jedincov (max. 12) pri zmene určitých parametrov. Legenda popisuje jednotlivé zmeny takto :

- Original – Originálne nastavené parametre.
- IncGen – Zväčšenie počtu generácii
- IncPop – Zväčšenie veľkosti populácie
- IncPopGen – Zväčšenie počtu generácii a veľkosti populácie.

Z grafu je vidieť, že iba pri zmene počtu generácii je graf rovnaký ako originál. Jedincov je málo a algoritmus zostane v lokálnom extréme. Algoritmus pomaly hľadá optimálne riešenie (potrebuje viacero generácii na dosiahnutie takého riešenia => pomaly rastúca krivka) a dosiahnutie lepších výsledkov je možné iba vhodnou mutáciou. Pri veľkom počte generácii je toho možné dosiahnuť. Pri zmene mutation rate by algoritmus rýchlejšie vyhľadal optimum.

Pri zmene počtu populácii je vidieť viditeľné zlepšenie. Algoritmus rýchlejšie vytvorí lepšieho jedinca (rýchlejšie rastúca krivka). To je hlavne kvôli väčšej variabilite jedincov, na rozdiel od predchádzajúceho prípadu

V poslednom prípade je zvýšená veľkosť populácie aj počet generácii. Výsledok je rovnaký, ako v predchádzajúcom prípade.



Na 2. obrázku sa nachádzajú porovnania priemerov vývinu jedinca (max 25), keď zvýšime veľkosť jedinca na 50.

Legenda popisuje jednotlivé zmeny takto :

- OriginalIncSize – Originálne nastavené parametre
- IncPopGenSize – Zvýšený počet generácii aj veľkosť populácie.

Obrázok len potvrdzuje predchádzajúce spomenuté vlastnosti parametrov. Zvýšenie veľkosti populácie urýchlil vývin lepšieho jedinca.