Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií
Ilkovičova 3, 842 16 Bratislava 4

Umelá inteligencia - Hľadanie pokladu

Róbert Dudáš

Študijný program: Informatika
Ročník: 2, Krúžok: streda 8:00
Predmet: Umelá inteligencia
Cvičiaci: Ing. Ivan Kapustík
Ak. rok: 2013/14

Riešený problém – Hľadanie pokladu

Úlohou je nájsť najlepšieho jedinca, ktorý nájde všetky poklady na mapke. Každý jedinec ma 64 génov ktoré spracúva virtuálny stroj. Program vypíše prvé nájdené riešenie. V prípade, že nenájdete riešenie, vypíše zatiaľ najlepšieho nájdeného jedinca .

Stručný opis riešenia a podstatných častí

V implementácií riešenia som použil algoritmus, ktorý sa skladá z nasledujúcich krokov:

1. Vytvorenie počiatočnej populácie. Množstvo jedincov v počiatočnej populácií si môžeme ľubovoľne zvoliť, pričom počet jedincov v populácií je počas všetkých generácií rovnaký. Každý jedinec (program) je vlastne chromozóm, ktorý obsahuje gény (inštrukcie). V počiatočnej populácií som si vytvoril daný počet jedincov a prvých 32 génov každého jedinca (jedinec má 64 génov) som náhodne inicializoval v rozsahu od -128 po 127 ostatných 32 génov je doplnený 0.
2. Ďalším krokom je vykonanie každého jedinca v populácií na virtuálnom stroji a na danej mape. Výstupom sú fitnes jedinca. V mojom programe mam dva fitnes-i. Jeden mám na počet nájdených pokladov a druhý je počet inštrukcií. Ak nejaký jedinec našiel všetky poklady na mape, program vypíše výslednú cestu hľadača na mape a jeho chromozóm. V prípade, že bol dosiahnutý maximálny počet generácií, tak sa program zastaví a čaká sa na rozhodnutie používateľa, či chce program ukončiť, alebo pokračovať znova.
3. Výber elity do novej populácie, tzv. elitárstvo. Elita sa nemení z predošlej generácie, čím dosiahneme, že nová generácia bude minimálne taká dobrá, ako tá stará. Tento výber prebieha tak, že si usporiadam jedincov podľa ich fitnes zostupne a vyberiem z nich daný počet.
4. Výber jedincov (ruletou) do novej populácie, ktorý sa budú mutovať. Metoda mutovanie si môže náhodne vybrať z troch mutovaní:
   A. Prepíše gén na 0.
   B. Prepíše gén na náhodnú hodnotu od -128 po 127 .
   C. Náhodný bajt zmení z 1 na 0 a naopak.
5. Výber jedincov (ruletou) do novej populácie, ktorý sa budú medzi sebou krížiť. Užívateľ si môže zvoliť ktorý typ kríženia použije:
   A. nastriedačku - nepárny gén je od otca a párny od matky
   B. napol- prvých 32 génov od matky a ďalších 32 génov je od otca
   C. nahoda- náhodne sa vyberie či jedince bude mať gén od matky alebo otca

6. Dogenerovanie zvyšných jedincov. Počet nových jedincov je dogenerovaný tak, aby bol počet jedincov v novej generácií vždy rovnaký.
7. Pokračuje sa krokom 2.

Stavový priestor

V mojej úlohe je stavový priestor reprezentovaný ako mapka rozloženia pokladov a začiatočnej pozícii. Na pohyb jedinca v mapke používam štyri operátory, ktoré zodpovedajú ťahom t.j. posun vľavo, vpravo, hore a dole. Všetky operátory majú rovnakú váhu. Cieľový stav je keď hľadač nájde všetky poklady.

Jedinec

Jedinec je reprezentovaný triedou *Jedinec* ktorá má pole bytov, v ktorom sa nachádza program jedinca, *fitnespokladov* je počet nájdených pokladov týmto jedincom, *fitnesinstrukcie* je počet inštrukcii vykonaných virtuálnym strojom, *posun* je cesta po mape prejdená týmto jedincom napr. U-up, D-down, L-left, R-right. Gén je teda jeden prvok pola v *chromozóme*, ktorý je typu *byte*. Celý *program* je chromozóm

Virtuálny stroj

Virtuálny stroj je trieda, ktorá spracúva jedincov chromozóm po génoch. Každý gén je inštrukcia, ktorú musí stroj vyhodnotiť a spracovať podľa zadania.

Nová generácia

Nová generácia je teda vždy tvorená elitou z predošlej generácie, zmutovanými jedincami z pôvodnej generácie, jedincami, ktorí sa medzi sebou skrížili a dogenerovanými novými náhodnými jedincami. V programe máme možnosť nastaviť **koľko percent** z novej populácie bude tvoriť **elita**, **zmutovaný jedinci** a **skrížení jedinci**. Zvyšok tvoria dogenerovaný náhodnými jedinci.

Kríženie

Keď mám jedincov, ktorý sú určený na kríženie medzi sebou, tak potom vytvorím nového jedinca z rodičov. Nový jedinec je potom zmutovaný s **danou pravdepodobnosťou**. Táto pravdepodobnosť je vstupný argument evolučného algoritmu. V programe mám na implementované tri druhy kríženia:

A. nastriedačku - nepárny gén je od otca a párny od matky
B. napol- prvých 32 génov od matky a ďalších 32 génov je od otca
C. nahoda- náhodne sa vyberie či jedince bude mať gén od matky alebo otca

Mutácia

V programe mám implementované dva druhy mutácie. Jedno mutovanie prebieha v cykle ide po génoch a s istou pravdepodobnosťou sa v génoch náhodný bit zmení z 1 na 0 alebo naopak toto mutovanie používam iba pri vytvorení jedinca krížením. Druhé mutovanie používam pri vytvorení nového jedinca čisto len mutovaním, ktoré sa dá zvoliť. Náhodne sa vyberie, ktoré mutovanie nastane:

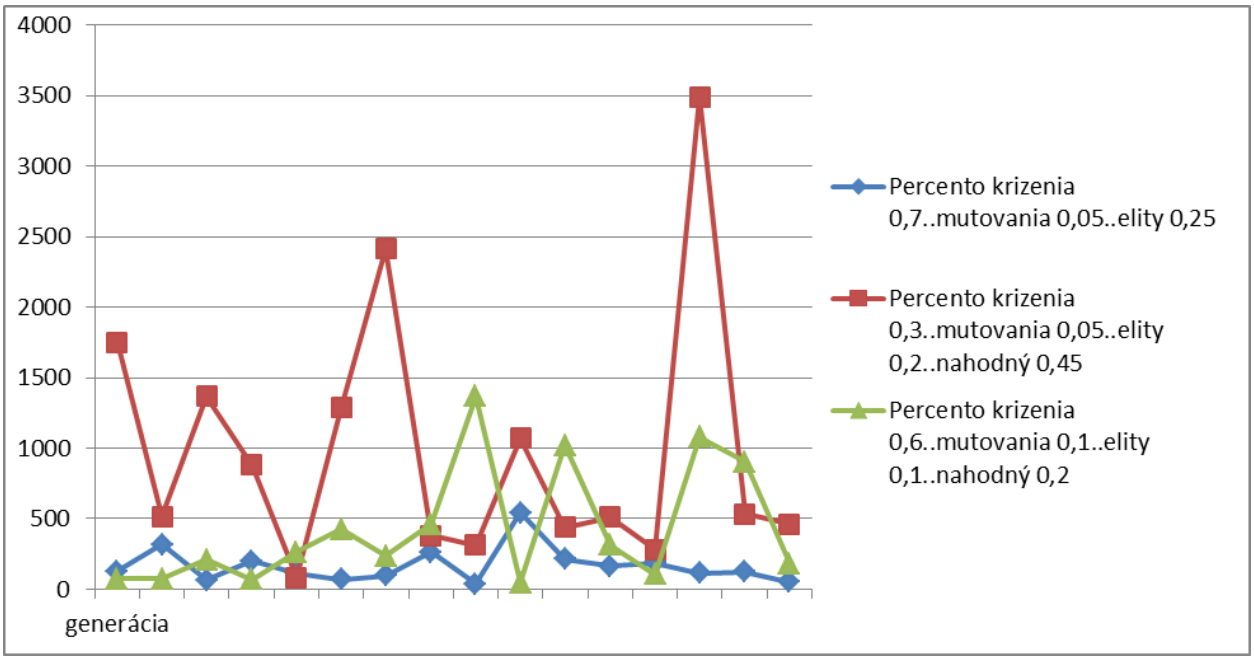
A. Prepíše gén na 0.
B. Prepíše gén na nahodnú hodnotu od -128 po 127 .
C. Náhodný bajt zmení z 1 na 0 a naopak.

Typ selekcie

V programe mám implementovaný jeden druh selekcie- **ruleta- výber podľa poradia** na základe fitnes pokladov a fitnes inštrukcií.
Celkový fitnes sa vypočítava ako:
1+(hp.maxinstrukcii*(1+(populacia.get(i).getfitnespoklady()))-
populacia.get(i).getfitnesinstrukcia())
maxinstrukcii- maximálny počet inštrukcii pre jedného jedinca
týmto výpočtom získame jedinca ktorý nazbieral najviac pokladov za čo najmenší počet inštrukcií.
Algoritmus je ten istý ako v prednáške, každý jedinec ma svoj segment na intervale od 0 po 1 a náhodne hádžem čísla z intervalu od 0 po 1, ktoré priradím segmentu a následné jedincovi ktorému daný segment patrí.

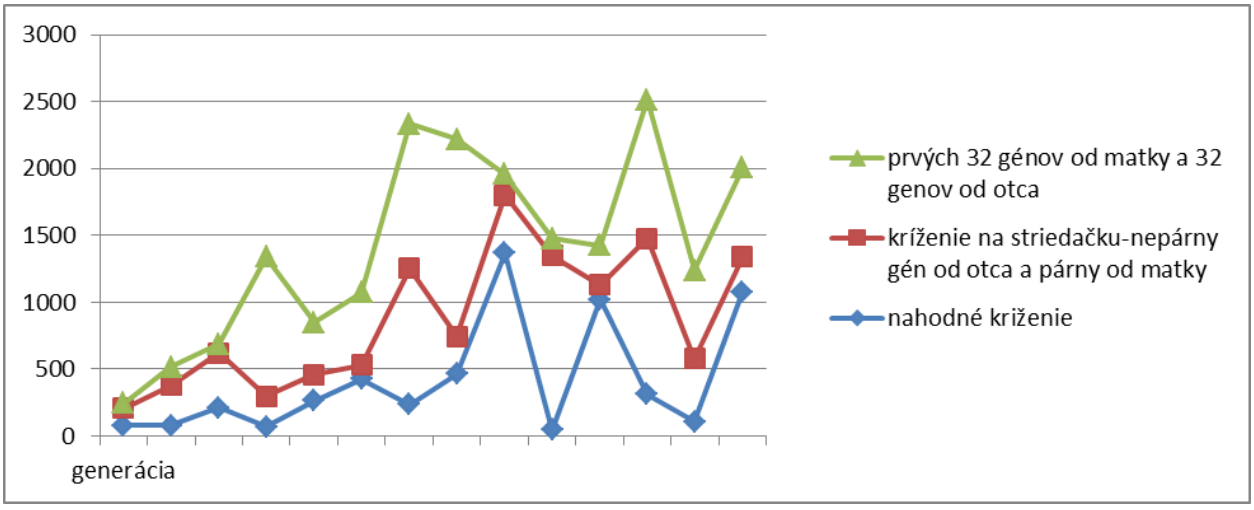
Porovnanie výsledkov

Na nasledujúcom grafe je porovnanie percenta kríženia, mutovania, elity a doplnenie náhodnými jedincami. Na osi y sú generácie jedincov, v ktorých sa našlo riešenie. Počet populácií bol 50 a mapa je tá istá ako zo zadania.



Z grafu vyplýva, že najrýchlejšie nájde riešenie populácia, ktorá je tvorená 70% krížením, 5% mutovaním, 25% elita.

V ďalšom grafe je porovnanie typ krížení. Na osi y sú generácie jedincov, v ktorých sa našlo riešenie. Percento kríženia bolo 60%..mutovania 10%..elity 10%..nahodný 20%. Počet populácií bol 50 a mapa je tá istá ako zo zadania.



Z grafu vyplýva, že najrýchlejšie nájde riešenie populácia, ktorá bola krížená náhodným krížením.

Možné spôsoby vylepšenia

Na vylepšenie algoritmu by mohli byť na implementované ďalšie druhy selekcie jedincov do novej generácie, nové druhy mutácie jedincov, nové druhy kríženia jedincov, experimentovať s percentom elity v novej generácií, percentom mutujúcich jedincov, percentom krížených jedincov, veľkosťou generácie. Alebo s rôznymi pravdepodobnosťami mutácie génu v jedincovi, pravdepodobnosťami kríženia jedincov.

Testovanie

Program som otestoval tak ako je to napísané v zadaní. Tu sa nachádza ukážkový výstup programu :

```
Hladac nasiel vsetky poklady jeho putovanie po mapke bolo nasledovne:
R , U , U , R , D , U , U , D , U , R , D , U , U , D , D , U , U , D , D , U , U , D
, U , U , D , U , D , U , D , U , D , U , D , U , D , U , D , D , U , D , L , U , L ,
D , U , U , L , D , L , D , U , D , L , D ,
Jedinec generacii: 9
11001010,11101100,11100010,11010110,1000101,1111011,11010100,10111,1111110,1001100,11
101011,10010010,11000011,11000000,11011000,11110000,11000011,10110,11110011,10100001,
1010101,10110,1111110,10001,1,1000011,11110010,10110100,1110010,10011,1101110,1000000
,1,0,10,0,1,10000,1100000,0,0,1000000,101000,100000,10,0,10,100,1,10000000,1000000,10
010000,11,0,0,0,11000000,0,0,0,1000,0,0,1001000,

Prajete si zacat znova? Ano=1/Nie=0
```

Zhodnotenie

Keďže bol program implementovaný v jazyku *Java*, tak by sa dal veľmi jednoducho prerobiť do iného objektovo orientovaného jazyka. Použil som dátové štruktúry (*Arraylist*), ktoré dostupné aj v iných programovacích jazykoch.

Použitá literatúra

Pavol Návrat a kol. – Umelá inteligencia
http://javafactsonline.blogspot.sk/2013/01/how-to-sort-java-arraylist-in.html
http://stackoverflow.com/questions/4844342/change-bits-value-in-byte