Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológii

Ilkovičova 2, 842 16 Bratislava 4

Zadanie č. 3, Hľadanie pokladu

Dokumentácia

Predmet: Umelá Inteligencia

Autor: Ivan Gulis

Študijný program: Informatika

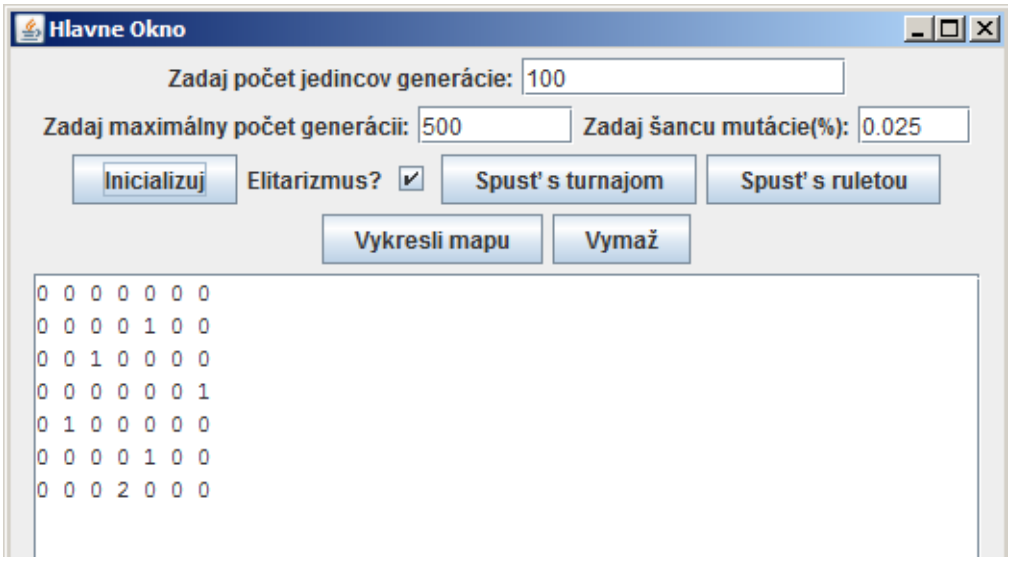
Semester: 4.

Ak. rok: 2014/2015

Zadanie úlohy:

Úlohou je vytvoriť hľadača pokladov, ktorí sa pohybuje po dvojrozmernej mapke a zbiera poklady. Pohybuje sa v 4 smeroch: H-hore, D-dole, P-doprava, L-doľava. Úlohou je nájsť čo najviac pokladov, najlepšie všetky. Za nájdenie pokladu sa považuje pozícia, kde stojí na políčku, kde sa nachádza poklad.

Používateľské rozhranie:



Gui je pomerne jednoduché. Na začiatku je potrebné inicializovať prvú generáciu hľadačov. Po vyplnení políčka pre počet jedincov v generácii je možné inicializovať generáciu stlačením tlačidla Inicializuj. Po stlačení tohto tlačidla sa vykreslí mapa a zviditeľnia zvyšné tlačidlá pre spúšťanie algoritmu. Je možné vybrať, či sa použije elitarizmus alebo nie. Zachováva sa vždy iba 1 jedinec do ďalšej generácie. Tlačidlo Vymaž vymaže textové pole a tlačidlo Vykresli mapu vykreslí zadanú mapu, po ktorej behajú jedinci na cvičisku.
Mapa sa načítava z textového súboru „mapa.txt“. Súbor vyzerá takto:
7
7
0000000
0000100
0010000
0000001
0100000
0000100
0002000

Poklady sú označené ako 1, počiatočná pozícia je 2, zvyšné políčka sú 0. Následne stlačením tlačidiel Spusť s turnajom alebo Spusť s ruletou sa spúšťa hlavný algoritmus s vybranou selekciou.

Popis algoritmu:

Najskôr sa inicializuje prvá generácia. Vytvorí sa zadaný počet jedincov, ktorým sa postupne inicializujú atribúty.

```
private String[] bunky;
private Mapa mapa;
private double fitnes;
private int riadok;
private int stlpec;
private List<Character> postupnost = new ArrayList<Character>();
```

Každý jedinec má v sebe pole svojich 64 pamäťových buniek, vlastnú mapu (na ktorej si vyškrtáva nájdené poklady), fitnes hodnotu (počiatočná fitnes je 1), aktuálny riadok a stĺpec(kde na mape sa práve nachádza), a svoju postupnosť krokov(H,D,L,P), ktoré prešiel na cvičisku.

Mapa: Mapa má nasledovné atribúty:

```
private int[][] mapa;
private int rozmerR;
private int rozmerS;
private int startriadok;
private int startstlpec;
private List<Poklad> poklady = new ArrayList<Poklad>();
private int pocetPokladov = 0;
```

Samotnú mriežku, rozmery (riadky a stĺpce), počiatočnú pozíciu pre jedincov, list pokladov a počet pokladov.

Poklad má len 2 atribúty, riadok a stĺpec.

```
private int riadok;
private int stlpec;
```

Pamäť: Najskôr sa naplní všetkých 64 pamäťových buniek jedinca náhodnými hodnotami (64*8 bitov, 1 alebo 0), ktoré predstavujú jednotlivé inštrukcie.
Prvé 2 bity určujú typ inštrukcie:
00xxxxxx = inkrementácia bunky na adrese xxxxxx
01xxxxxx = dekrementácia bunky na adrese xxxxxx
10xxxxxx = skok na bunku xxxxxx
11xxxxxx = pohyb po mape
Pohyb po mape má 4 smery (H,D,P,L), ktoré sa určujú podľa posledných 2 bitov inštrukcie na adrese xxxxxx.

H = 00, D = 01, P = 10, L = 11

Cvičisko: Jedince sa po inicializácii vysielajú na cvičisko. Tam sa vo while cykle do 500 inštrukcii testuje ich program. Od 1. Inštrukcie do 64 a znova, až kým nevybehnú z mapy alebo nenájdu všetky poklady alebo neubehne 500 inštrukcii. Inštrukcie inkrement a decrement sú cyklické, takže pri hodnote 255+1=0, a pri 0-1=255. Tu využívam funkcie Integer.toBinaryString.
Keď jedinec nájde poklad, zvýši sa mu fitnes +1 a odškrtne si z mapy nájdený poklad a zistí, koľko pokladov mu ešte ostáva nenájdených. Každý krok mu z fitnes odráta -0,001, takže čím menej krokov, tým lepšie pre jedinca.
Jedinec z cvičiska vychádza s novou fitnes hodnotou a vrátia sa mu pôvodné bunky. Je pripravený na ďalšiu fázu, selekciu. Pred každým pridaním jedinca do prvej generácie sa ešte skontroluje, či niektorý nenašiel všetky poklady, vtedy program skončí a vypíšu sa výsledné údaje.

Obmieňanie generácii:

Hlavný while cyklus sa spúšťa po stanovený maximálny počet generácii.
Podľa výberu selekcie sa ďalej rozvetvuje do dvoch častí: turnaj(0) a ruleta(1).
Ak je zaškrtnutý elitarizmus, najskôr sa vyberie najlepší jedinec podľa fitnes, a pridá sa priamo do novej generácie.
Cesta s turnajom: Najskôr sa vyberie náhodne 5 jedincov zo starej generácie, z nich sa vyberú 2 najlepší (podľa ich fitnes) a skrížia sa. Nasleduje prípadná mutácia, pobehanie po cvičisku pre získanie fitnes a zaradenie do novej generácie.
Cesta s turnajom: Najskôr sa spočítajú fitnes hodnoty celej generácie. Následne sa vygenerujú 2 čísla v danom rozmedzí po vypočítaný súčet. Rodičia sa vyberú tak, že od nultého jedinca sa spočítavajú hodnoty až po vygenerované náhodné čísla, a vezme sa jedinec, na ktorom sa cyklus zastaví.
Kríženie: Kríženie prebieha vo for cykle 0-64. Náhodne sa generujú čísla 0 a 1.
0 = nový jedinec dostane pamäťovú bunku od 1. rodiča
1 = nový jedinec dostane pamäťovú bunku od 2. rodiča
Mutácia: Každému jedincovi sa so zadanou šancou pregeneruje pamäťová bunka. Šanca mutácie je maximálne 100% a minimálne 0%.

Ak sa nenájde jedinec, ktorý by našiel všetky poklady, vypíše sa najlepší jedinec, a je možné pokračovať opätovným stlačením toho istého tlačidla. Zvýši sa maximálny počet generácii a while cyklus pokračuje.

Príklad výpisu k neúspešnému hľadačovi:
Už bolo vygenerovaných 50 generácii a nikto zatiaľ všetky poklady nenašiel.
Najlepší jedinec našiel pokladov: 4/5
Jeho cesta je [P, H, L, H, H, P, P, L, H, L, P, P, L, H, L, D, D, L, H, L, D, D, L, H, L]
Jeho počet krokov: 25
Pre pokračovanie spustite znova vybraný štýl.

Ak sa jedinec, ktorý našiel všetky poklady nájde, program končí a vypíšu sa výsledné údaje o jedincovi.
Pôvodná prvá generácia sa nastaví ako stará, resetne sa aktuálny počet generácii a je možné spustiť algoritmus znova na pôvodnú generáciu napr. s iným módom (ruleta/turnaj).
Tiež je možné re-inicializovať prvú generáciu stlačením tlačidla Inicializuj.

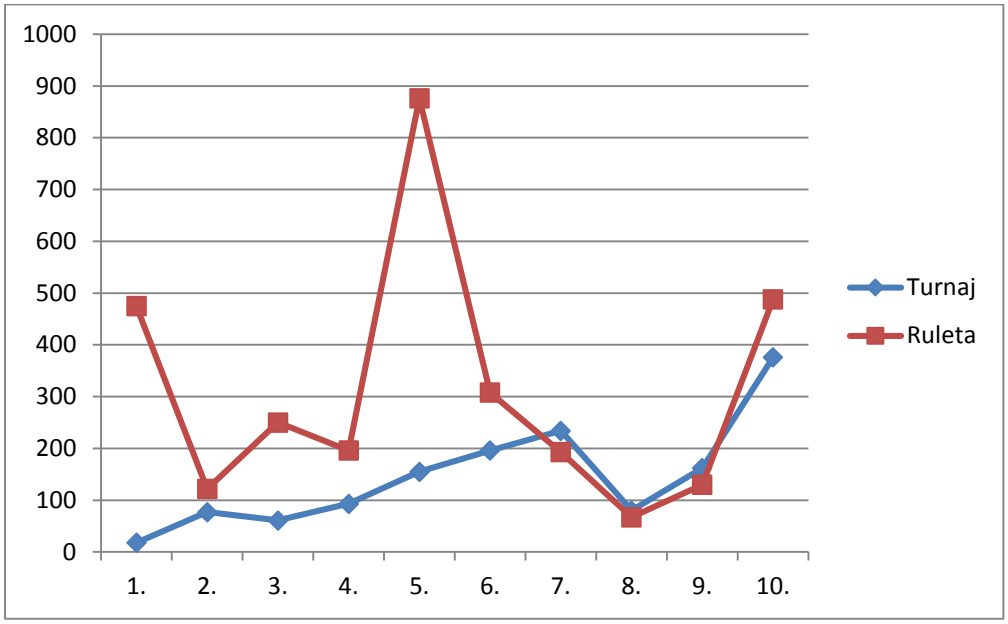
Príklad výpisu k úspešnému hľadačovi:
Jedinec našiel všetky poklady, jeho postupnosť krokov:
[P, L, H, P, D, L, H, P, H, L, P, P, L, L, D, P, D, L, H, P, P, L, H, P, H, L, P, P, L, L, D, P, L, L, H, P, H, L, L, P, L, L, D, P, H, L, P, P, L, L, D, P, D, L, L, P, H, L, P, P, P, L, L, P, D, L, H, P, P, L, P, P, H, L, H, P]
Jeho počet krokov: 76
Je z generácie číslo 164

Vlastnosti selekcii (porovnanie a testovanie):

Testoval som na generáciách so 100 jedincami, 0,025% šanca mutácie a základ max 1000 generácii na jednej inicializovanej generácii. Výsledky 10 pokusov:

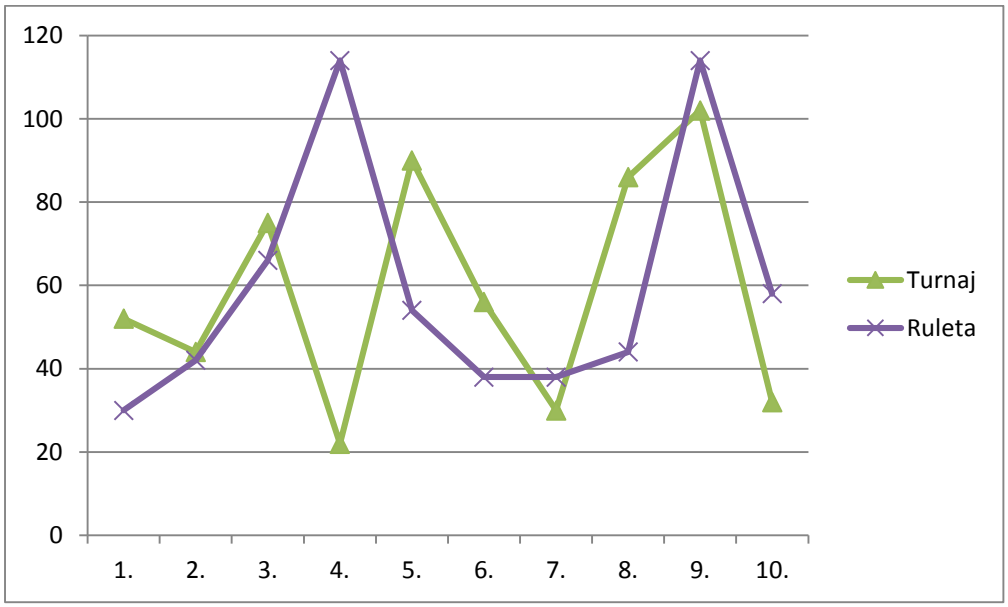
	Počet generácii		Počet krokov	
Spustenia programu	Turnaj	Ruleta	Turnaj	Ruleta
1.	18	475	52	30
2.	77	122	44	42
3.	61	250	75	66
4.	93	196	22	114
5.	155	876	90	54
6.	196	308	56	38
7.	234	193	30	38
8.	80	67	86	44
9.	162	130	102	114
10.	376	488	32	58

Graf, v ktorej generácii sa našiel jedinec, ktorý našiel všetky poklady:



Z grafu je vidieť, že selekcia turnajom je úspešnejšia (pretože výber najlepšieho z 5 jedincov nie je náhodný).

Graf pre počet krokov jedinca, ktorý našiel všetky poklady:



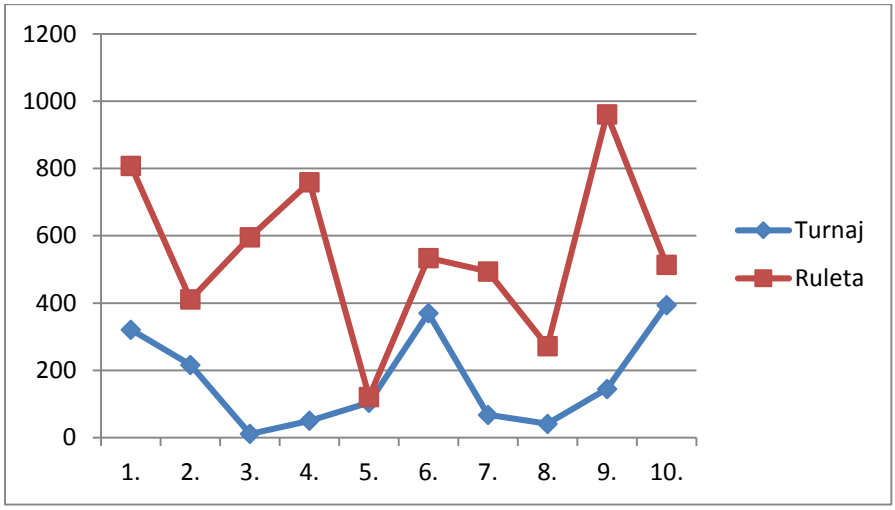
Výsledky sú veľmi podobné, turnaj je miestami lepší. Dĺžka cesty totiž priamo nezávisí od použitej selekcie, ale ak selekcia vyberie lepších jedincov, a lepší jedinec má lepšiu fitnes, prešiel aj kratšiu cestu.

Bez elitarizmu:

Pre porovnanie ešte tabuľka a grafy zo spustení programu, kde nebol použitý elitarizmus:

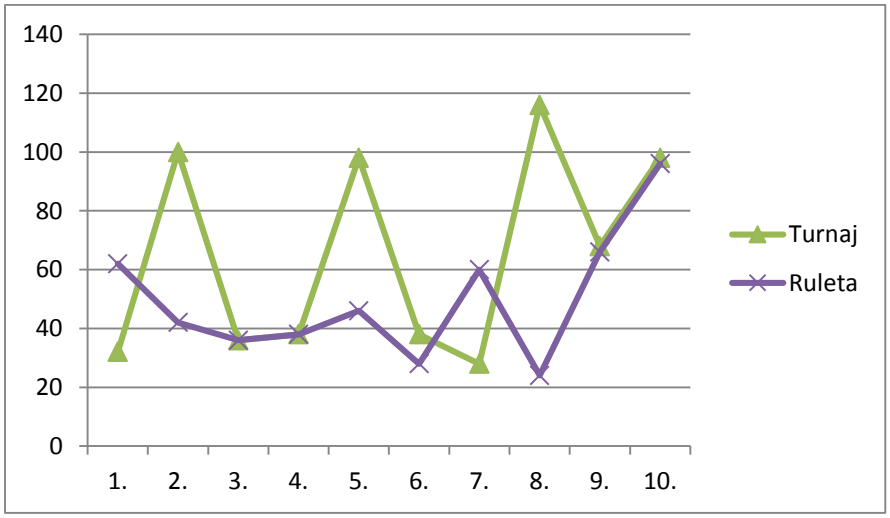
	Počet generácii		Počet krokov	
Spustenia programu	Turnaj	Ruleta	Turnaj	Ruleta
1.	321	808	32	62
2.	216	411	100	42
3.	11	596	36	36
4.	50	759	38	38
5.	104	121	98	46
6.	370	534	38	28
7.	68	494	28	60
8.	41	272	116	24
9.	145	961	68	66
10.	394	514	98	96

Graf, v ktorej generácii sa našiel jedinec, ktorý našiel všetky poklady:



Rozdiel medzi turnajom a ruletou sa ešte viac prehĺbil.

Graf pre počet krokov jedinca, ktorý našiel všetky poklady:



Ruleta má mierne lepšie výsledky, ale turnaj má veľa amplitúd. Tieto výsledky závisia stále z veľkej časti na náhode, viac ako na type selekcie.

Príklady výstupov programu:

Jedinec našiel všetky poklady, jeho postupnosť krokov:
[P, H, P, P, H, H, H, D, H, D, H, H, L, D, P, D, H, H, L, D, L, D, H, H, L, D, H, D, H, H, L, D, D, D, H, H, L, D, P, D, H, H, L, D, D, D]
Jeho počet krokov: 46
Je z generácie číslo 30

Jedinec našiel všetky poklady, jeho postupnosť krokov:
[H, P, H, L, L, L, H, H, D, D, H, H, P, D, H, P, D, H, H, P, D, H, H, P, D, D, H, H, P, D, D, D]
Jeho počet krokov: 32
Je z generácie číslo 321

Jedinec našiel všetky poklady, jeho postupnosť krokov:
[P, H, D, L, P, P, H, D, L, H, P, L, L, P, H, D, D, H, P, L, H, P, P, D, H, H, D, L, H, P, L, D, H, H, D, L, P, P, H, D, L, H, H, L, L, P, L, D, L, H, L, L, P, P, L, D, D, H, D, L, D, P]
Jeho počet krokov: 62
Je z generácie číslo 808

Zhodnotenie:

Program funguje pre 2 typy selekcie, 1 spôsob kríženia a 1 typ mutácie. Požívateľské prostredie dovoľuje nastaviť skoro všetky potrebné parametre a program pracuje s externým textovým súborom pre načítanie mapy a pokladov. Hlavné rozšírenie teda môže smerovať k pridaniu nových typov selekcie, nových typov kríženia (napr. polovica buniek od jedného rodiča, polovica od druhého), nových typov mutácii (napr. inverzia bitov bunky, zmena jedného bitu bunky). Tiež je možné zmeniť počet inicializovaných buniek jedinca (napr. naplniť len prvých 20 buniek, a zvyšných 44 len vynulovať).
Vylepšenie môže byť aj v zmene Gui a pohľadu na vygenerovanú mapu. Pridanie mriežky prípadne iných grafických prvkov (mapa je zatiaľ len skupina čísel), alebo zmeny typu integer na char a tak použiť písmená na označenie objektov na mape.