Slovenská technická univerzita v Bratislave

Fakulta informatiky a informačných technológií

Ilkovičova 2, 842 16 Bratislava 4

Zenová záhrada

Robert Yamkovyi

Študijný program: Informatika

Ročník: 2

Krúžok: Ut 10:00, 1.30a

Predmet: Umela Inteligencia

Vedúci projektu: Mgr. Irina Malkin Ondik, PhD.

Ak. rok: 2017/2018

Оглавление

[Definovanie problému 3](#_Toc511077748)

[Použitý algoritmus a jeho vlastnosti: 4](#_Toc511077749)

[Opis riešenia 4](#_Toc511077750)

[Zhodnotenie 5](#_Toc511077751)

# Definovanie problému

Zenová záhradka je plocha vysypaná hrubším pieskom (drobnými kamienkami). Obsahuje však aj nepohyblivé väčšie objekty, ako napríklad kamene, sochy, konštrukcie, samorasty. Mních má upraviť piesok v záhradke pomocou hrablí tak, že vzniknú pásy ako na nasledujúcom obrázku.



Pásy môžu ísť len vodorovne alebo zvislo, nikdy nie šikmo. Začína vždy na okraji záhradky a ťahá rovný pás až po druhý okraj alebo po prekážku. Na okraji - mimo záhradky môže chodiť ako chce. Ak však príde k prekážke - kameňu alebo už pohrabanému piesku - musí sa otočiť, ak má kam. Ak má voľné smery vľavo aj vpravo, je jeho vec, kam sa otočí. Ak má voľný len jeden smer, otočí sa tam. Ak sa nemá kam otočiť, je koniec hry. Úspešná hra je taká, v ktorej mních dokáže za daných pravidiel pohrabať celú záhradu, prípade maximálny možný počet políčok. Výstupom je pokrytie danej záhrady prechodmi mnícha. Pokrytie zodpovedajúce presne prvému obrázku (priebežný stav) je napríklad takéto:

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 8 | 9 |
| 0 | 0 | 1 | 0 | 0 | K | 0 | 0 | 10 | 10 | 8 | 9 |
| 0 | K | 1 | 0 | 0 | 0 | 0 | 0 | 10 | 10 | 8 | 9 |
| 0 | 0 | 1 | 1 | K | 0 | 0 | 0 | 10 | 10 | 8 | 9 |
| 0 | 0 | K | 1 | 0 | 0 | 0 | 0 | 10 | 10 | 8 | 9 |
| 2 | 2 | 2 | 1 | 0 | 0 | 0 | 0 | 10 | 10 | 8 | 9 |
| 3 | 3 | 2 | 1 | 0 | 0 | 0 | 0 | K | K | 8 | 8 |
| 4 | 3 | 2 | 1 | 0 | 0 | 0 | 0 | 5 | 5 | 5 | 5 |
| 4 | 3 | 2 | 1 | 0 | 0 | 0 | 11 | 5 | 6 | 6 | 6 |
| 4 | 3 | 2 | 1 | 0 | 0 | 0 | 11 | 5 | 6 | 7 | 7 |

Uvedenú úlohu riešte pomocou evolučného algoritmu. Maximálny počet génov nesmie presiahnuť polovicu obvodu záhrady plus počet kameňov, v našom príklade podľa prvého obrázku 12+10+6=28. Fitnes je určená počtom pohrabaných políčok. Výstupom je matica, znázorňujúca cesty mnícha. Je potrebné, aby program zvládal aspoň záhradku podľa prvého obrázku, ale vstupom môže byť v princípe ľubovoľná mapa

# Použitý algoritmus a jeho vlastnosti:

Jedná sa o klasický genetický algoritmus, takže na začiatku sa vytvorí prvá generácia jedincov s náhodne nastavenými génmi. Potom sa všetky jednice ohodnotia, teda pre každého jedinca sa vytvorí matica s prechodmi mnícha a zistí sa, koľko políčok sa podarilo pokryť. Na základe ohodnotenia sa vyberú jedince na tvorbu novej generácie - kríženie a s určitou pravdepodobnosťou aj na mutáciu. Vytvorí sa nová generácia a tak to ide dokola, až kým sa nepodarí pokryť všetky políčka alebo sa dosiahne stanovený počet nových generácií.   
Pre gény je najvhodnejšie, keď reprezentujú priamo miesto na obvode záhrady, kde mních vstúpi (ak môže) a začne hrabať. Zvyšné gény môžu reprezentovať rozhodnutia mnícha, či sa pri najbližšej možnosti voľby dá vpravo alebo vľavo.   
Vstupom sú rozmery záhrady a súradnice kameňov, výstupom je mapa pohrabanej záhrady, podobne ako na druhom obrázku.

# Opis riešenia

Mam 2 triedy “Person” a “Genetics”

Trieda “Person” obsahuje hodnotu fitness funkcii, chromozomu a mapu.

Trieda “Genetics” obsahuje vsetky vypocty

Na začiatku vygenerujem prvu generaciu. Prva generacia ma 100 jedincov a nahodne chromozomy. Pre každeho jedinca vypočitam fitness (počet projdenych poličok), a usporiadam jedincov podľa fitnessu. Potom mam na vyber dve možnosti selekcii:

1. Selekcia Turnament – zvoli 80 jedincov zo 100 tak že vyberie dvoch náhodných ľudí a pridá do zoznamu osobu s vyššou hodnotou fitness.
2. Selekcia Rullet – zvoli 80 jedincov zo 100 tak že vyberie jedinca z vačšou hodnotou fitness z vačšou pravdepodobnosťou.

Potom robim križenie tak že budem križiť 1. a 2. jedinca, 3. a 4. ... 79. a 80. jedinca. Z jedneho križenie dostanem dve nove chromozomy(dva jedinca):

1. Jedinec bude obsahovať *n*  genov prvej chromozomy a *28 – n* genov z druhej chromozomy.
2. Jedinec bude obsahovať *28 - n*  genov prvej chromozomy a *n* genov z druhej chromozomy.

n = random od *0* po *28*

A pridam do novej generacii 20 jedincov z najlepšou hodntotou fitness zo starej generacii.

Každa nova chromozoma može dostať mutaciu z pravdepodobnosťou 30%

Mutacia prehadza tak že zvoli nahodne(0-28) počet genov, nahodny gen sa zvačsi o 3.

Pre novu generaciu musim vypočitať fitness každeho jedinca. Metoda „fitness“ zavola metodu „makeNewMap“. Cieľom tejto metody je najsť suradnicu(X a Y) vstupu na mapu.Tato metoda funguje tak že zoberie 1. Gen chromozomy a ak on je menši alebo rovna sa 13(polovica chromozomy) a je parny, tak X = 0, ak neparny X = 11(max dlžka po osi X). Y rovna sa hodnote sledujuceho genu ale %= 10(max vyška po osi Y).

Inak (vačši ako 13) ak je parny Y=0, ak je neparny Y=9. X rovna sa hodnote sledujuceho genu.

Ak X = 0 smer je zľava doprava

Ak X = 11 smer je sprava doľava

Ak Y = 0 smer je zhora nadol

Ak Y = 9 smer je zdola nahor

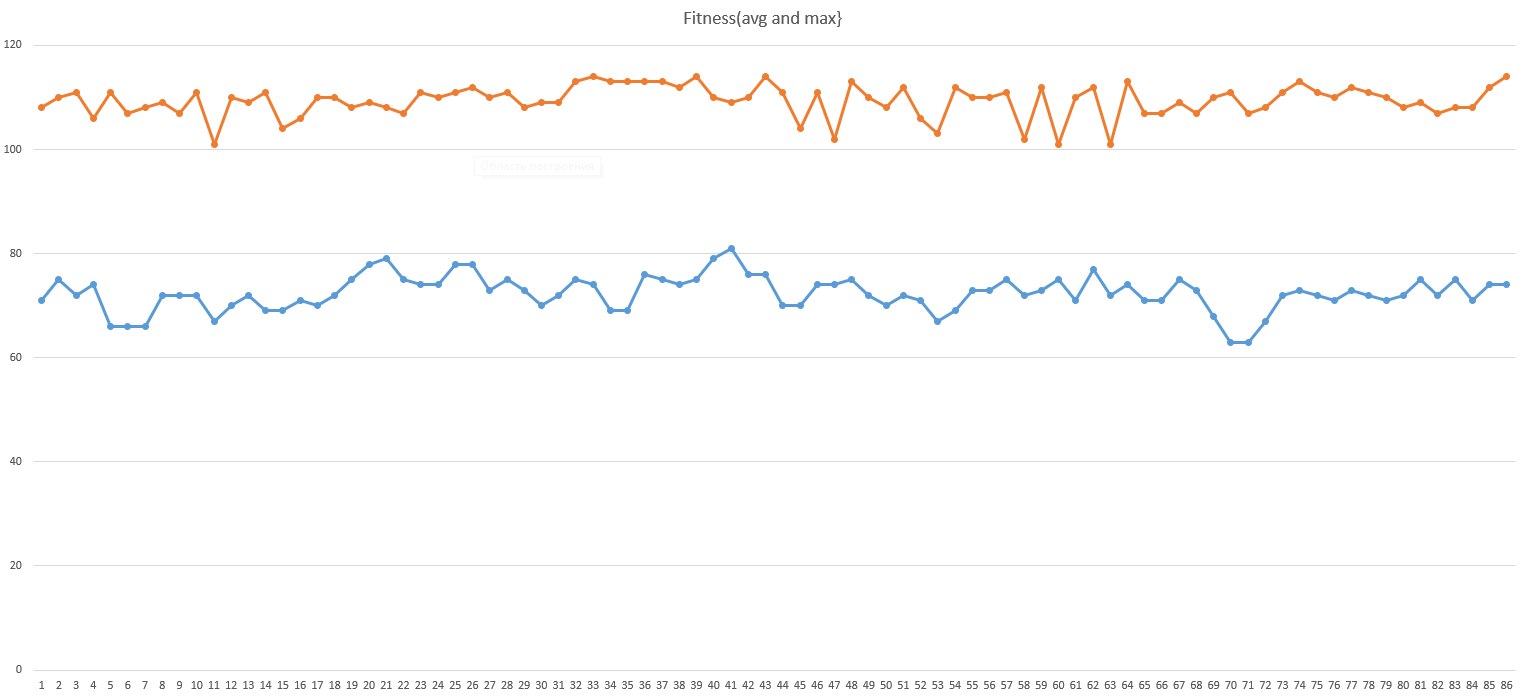
Ak sa cesta stretne s prekážkou, najprv sa pozerá doprava, ak nemôžete ísť doprava, pozrite sa doľava, ak nemôžete ísť doľava, poď von, ak nemôžete ísť von, potom sme na mŕtvom bode a už nehľadáme cestu.

Pre každeho jedinca vypočitavame fitness, ak fitness rovna sa 114(ploha mapy - počet kameňov) tak je to riešenie. Ak najdene riešenie menej ako za 500 generacij program vypiše chromozomu jedinca a jeho mapu.

# Zhodnotenie

V tomto projekte som sa naučil pracovať s genetickym algoritmom v praxi. Podľa testov možem povedať, že z vačšou pravdepodobnosťou selekcia turnajom je lepšia ako selekcia rullet a je to videť aj na grafoh. Podľa testov zistil že hlavnu rolu hra implementacia časti kde musime najsť vhod v mapu. Program funguje iba z mapou čo je v zadanii. Na zlepšenie efektivnosti programu bolo by vhodne použiť TreeSet.

Selekcia Rullet



Selekcia Turnaj

