Zadanie 3

Zenova záhrada

Sofia Mlynarčíková
ID: 103058
FIIT STU

Zadanie

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.

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 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

Implementačné prostredie

Program bol implementovaný použitím IDE Pycharm 2020.2.3 a je naprogramovaný v jazyku Python verzie 3.8. Použitá bola taktiež funkcia *deepcopy* z prídavného modulu *copy* na kopírovanie objektov, modul *time* na meranie času a *random* na generovanie náhodných čísel.

Reprezentácia údajov

Záhrada a umiestnenia kameňov sú načítané z textového súboru, ktorý na prvom riadku obsahuje rozmer mapy (počet riadkov a stĺpcov) a na každom ďalšom riadku súradnice jednotlivých kameňov. Mapa je reprezentovaná 0 na miestach, kde nie je pohrabaná a -1 na miestach kde sú kamene.

Každý mních obsahuje zoznam génov a fitness. Gény sú reprezentované vstupným políčkom na mape od 0 po obvod mapy - 5 (rohy nie sú započítané dvakrát a indexuje sa od 0). Napríklad pre mapu rozmeru 3 x 3 by možné vstupy vyzerali následovne:

0	1	2
7		3
6	5	4

Pre vstupy od 0 - 2 by mních začal hrabať smerom dole, pre vstup 3 smerom doľava, pre vstup 4 - 6 hore a 7 doprava.

Nastavenia algoritmu sú uložené v zozname *setting*, ktorý obsahuje následovné údaje:

[počet jedincov v populácii, maximálny počet generácii, pravdepodobnosť mutácie, podiel elitizmu, podiel novej krvi, počet políčok na pohrabanie]

Algoritmus hrabania

Program si načíta údaje o záhrade, vo funkcii *init_garden* ju vytvorí na podobu opísanú v reprezentácii údajov. Vypočítajú sa dodatočné informácie potrebné počas programu a záhradka je pripravená na pohrabanie.

Vo funkcii *rake* sa skopíruje pôvodná záhrada do premennej *test_garden*, v ktorej sa testujú pohyby mnícha, ktorý hrabe. Pri jeho prvom géne je poradie hrabania nastavené na 1 - mních za sebou necháva políčka označené 1. Pokiaľ je hrabanie pre tento gén neúspešné, jeho cesta sa z mapy zmaže a poradie sa zníži o 1, pretože na konci jedného cyklu pre gény sa vždy zvyšuje o 1 bez ohľadu na to, či sa mních zasekol alebo nie.

Smer hrabania je určený podľa toho kde na mape sa gén nachádza. Ak mních začína hrabať hore, pokračuje smerom dole a podobne. Pohyb génu je zaznamenávaný vo funkcii *draw_garden*, ktorá vracia hodnotu fitness. Ak sa gén mnícha zasekne, vrátená hodnota je 0 a funkcia sa s rovnakým génom zavolá znova aby premazala na *test_garden* pohyby génu.

Argumenty *draw_garden* sú x a y súradnice vstupného políčka, poradie hrabania, záhrada na ktorej sa bude hrabať, políčko, na ktoré má mních vstúpiť (*path* - pri prvom hrabaní je to 0, pokiaľ sa volá táto funkcia znova aby sa vymazala cesta, tak je táto hodnota vymenená s poradím) a počiatočný smer, akým má mních hrabať. Fitness je na začiatku *draw_garden* inicializovaná na 1, pretože pohrabanie posledného políčka sa nezapočítava. Ak je vstupné políčko už pohrabané alebo mních sa zasekne, funkcia vráti 0.

Vnútri sa nachádza cyklus na prechádzanie políčok. Podľa smeru mnícha sa najprv mních pomocou jednej z pomocných funkcií *check* pozrie, či môže pohrabať políčko v jeho smere. Pokiaľ nie, tak pomocou *check_borders* zistí, či nie je na konci svojho ťahu a vráti *fitness*. Ak nie je na okraji mapy, buď sa môže pohnúť do iného smeru, alebo je zaseknutý a funkcia vráti 0. Pri rozhodovaní, akým smerom pôjde ďalej sa pri vertikálnom hrabaní vždy najprv pozerá doľava a potom doprava a pri horizontálnom hrabaní naopak. Testované bolo aj pridanie ďalšieho poľa génov mnícha s inštrukciami, akým smerom má hrabať, avšak riešenie nebolo nájdené rýchlejšie, preto toto riešenie bolo vymazané.

Evolučný algoritmus

Inicializovanie prvej generácie prebieha vo funkcii *init_population*. Pre nastavený rozsah jedincov sa vygenerujú jedince s náhodnými génmi. Počet génov je určený vzorcom:

$$\frac{obvod}{2} - x \bmod 3$$

x = počet kameňov na obvode mapy, mod 3 aby sa zabránilo nízkemu počtu génov

Náhodné gény vracia funkcia *get_random*, ktorá pre vygenerovaný gén zistí, či nie je duplikát, poprípade či je možné na vygenerované políčko vstúpiť. Každý mních sa taktiež pošle do funkcie *rake*, aby pohrabal záhradu a bol zistený jeho fitness. Potom sa s ďalšími populáciami pokračuje už iba do funkcie *evolve*. Tá zoradí populáciu od jedinca s najvyšším fitness po najslabšieho. Pokiaľ je najsilnejší jedinec aj výsledný, vráti sa naň odkaz, čo indukuje ukončenie evolúcie a riešenie sa vypíše. Ak nie, tak sa pokračuje elitizmom a výmenou najslabších jedincov za novú krv – jedincov s náhodnými génmi. Pokým nie je nová generácia zaplnená na požadovaný počet jedincov, vyberajú sa rodičia do kríženia a vytvárajú sa nové jedince. Výber štandardne prebieha turnajom dvoch jedincov a zároveň ruletou. Efektivita rôznych výberov je porovnaná v kapitole *Testovanie*. Prví dvaja rodičia sa vyberú turnajom, kde vstupujú dva náhodné jedince a vyberie sa ten silnejší. Títo rodičia sa skrížia a vymenia sa novými vybranými ruletou. Do rulety vstupuje celková hodnota fitness, ktorá je hornou hranicou pre náhodne vybranú hodnotu. Od tejto hodnoty sa odpočítavajú fitness jedincov v generácii od najslabšieho. Pokiaľ pri odčítaní fitness niektorého z jedincov dosiahne táto hodnota menej ako 1, daný jedinec bol úspešne vybraný.

Pred tým, než sa jedince skrížia určí sa, ktorý z nich má silnejšie gény a ktorý slabšie. Do funkcie *crossover* vstupuje ako prvý rodič s dominantnými - silnejším génmi, a druhý s recesívnymi génmi. Pri krížení dostane dieťa prvú časť od dominantného rodiča a druhú od recesívneho. Bod, kde sa majú gény meniť sa určuje podľa toho, aký je rozdiel vo fitness rodičov. Premenná *ratio* určuje tento bod podľa vzorca:

$$\frac{x - y + max}{2 * max}$$

x = fitness dominantného rodiča, **y** = fitness recesívneho rodiča, ***max*** = maximálny možný fitness rodiča

Premenná *ratio* tak naberie percentuálnu hodnotu, ktorá keď sa navyše vynásobí s počtom génov v jedincovi, určí sa presné miesto výmeny rodiča. Napríklad ak je premenná $y = 0$ a $x = 57$ (polovica maximálnej hodnoty 114 pre ukážkovú zenovu záhradku), tak *ratio* bude rovné 75% a teda dieťa bude mať prvých 15 génov od dominantného rodiča a zvyšných 5 od recesívneho.

Mutovanie sa uskutočňuje taktiež vo funkcii *crossover*. Pri každom géne novo vytvoreného dieťaťa sa inicializuje náhodné číslo z intervalu <0;1> a porovná sa s pravdepodobnosťou mutácie. Pokiaľ je toto číslo menšie, znova sa zavolá funkcia *get_random*, ktorá vráti neduplicitný a prístupný bod na mape. Dieťa tak môže mať zmutovaný náhodný počet génov.

Po vytvorení novej populácie sa jej jedinci vyšlú pohrabať záhradku a prekopíruju sa do globálnej populácie. Poradie generácie sa navýši a proces sa opakuje pokým nedosiahneme stanovený maximálny počet generácií. Ak ani dovtedy nie je nájdené riešenie, vypíše sa mních s najvyšším fitness v poslednej generácii.

Testovanie a zlepšovanie algoritmu

Testovanie bolo uskutočnené dvoch rôznych mapách - pôvodnej s rozmerom 10x12 a druhou s rozmerom 6x6 a dvoma kameňmi na okrajoch. Výsledné hrabania sú zobrazené nižšie spolu so zodpovedajúcim grafom evolúcie. Efektivita metód výberu bola skúmaná pri vzorke 100 pokusov.

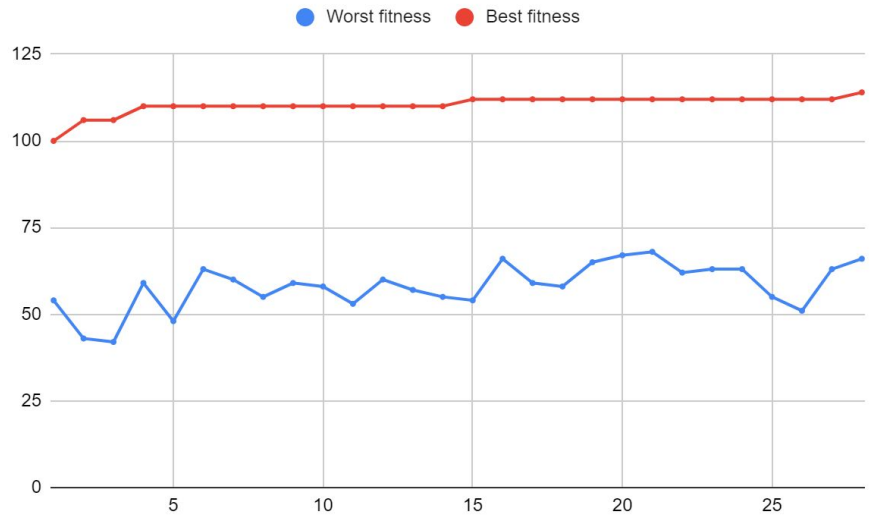
Najlepšie výsledky boli dosiahnuté pri pravdepodobnosti mutácie od 15 do 25%, elitizmom menším ako 5% a novou krvou od 40 do 60%. Samozrejme, čím viac mníchov v generácii bolo,

tým rýchlejšie bol nájdený výsledok. Štandardný počet jedincov je nastavený na 100 a počet generácii na 1000, avšak nestáva sa aby riešenie bolo dovtedy nenájdené vďaka vysokému percentu novej krvi a mutácie. Zaseknutie v lokálnom maxime teda nemôže nastať a riešenie sa po čase nájde.

Pokusy zlepšiť evolúciu pozostávali z implementovania iných funkcií pre kríženie a pre rozšírenie génov mnícha. Testovaný typ kríženia bol založený na overovaní, či dieťa daný gén v poradí dostane od dominantného alebo recesívneho rodiča. Znova sa vypočítalo *ratio*, ktoré vyjadrovalo pravdepodobnosť prevzatia génu z dominantného rodiča. Pokiaľ bolo náhodne vygenerované číslo z intervalu <0;1> nižšie ako *ratio*, gén bol prevzatý od dominantného rodiča, inak od recesívneho. Zaručilo to vyšší počet dominantných génov na náhodných miestach dieťaťa, avšak riešenie to neurýchlilo.

Pri rozšírení génov mnícha obsahoval mních ďalší zoznam s inštrukciami pozostávajúci z 0 a 1, ktoré mali rozhodnúť pri otočení mnícha ak narazil na prekážku. 1 znamenala doľava a 0 doprava. Mnísi sa pri hrabaní riadili podľa týchto génov, ktoré boli neskôr taktiež krížené a mutované rovnako ako gény so vstupmi na záhradku. Jediný rozdiel, ktoré toto riešenie poskytlo bolo viacej riadkov. Riešenia boli nájdené trochu pomalšie, pretože takéto hľadanie cesty bolo rovnako ako štandardné založené na náhode.

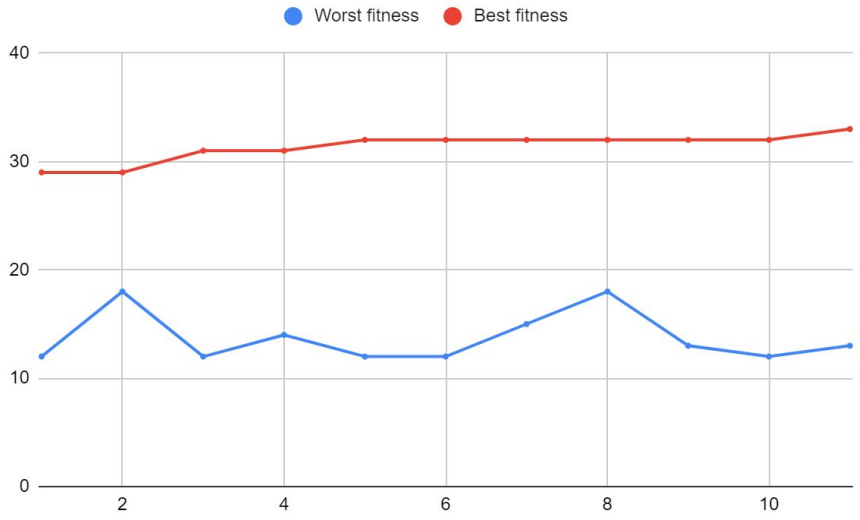
```
Best solution found in generation 28
Genes: [7, 34, 5, 3, 24, 31, 21, 8, 6, 22, 35, 38, 29, 15, 0, 27, 16, 19, 4, 1]
 12 12  9  4  4  3  3  1  7  7  6 11
 12 12  9  4  4 -1  3  1  7  7  6 11
  9 -1  9  4  4  4  3  1  7  7  6 11
  9  9  9  4 -1  4  3  1  7  7  6 11
  9  9 -1  4  4  4  3  1  7  7  6 11
  3  3  3  3  3  3  3  1  7  7  6 14
  2  2  2  2  2  2  2  1 -1 -1  6 14
  5  5  5  5  5  5  2  1  8  8  6 14
  5 10 10 13 13  5  2  1  8  8  6 14
  5 10 10 13 13  5  2  1  8  8  6 14
```



Riešenie nájdené priemerne v generácii:

- 47 pri použití rulety aj turnaja zároveň
- 43 pri použití turnaja
- 68 pri použití rulety

```
Best solution found in generation 11
Genes: [6, 4, 2, 10, 16, 19, 12, 3]
  3  3  3  1  2  2
  4  4 -1  1  1  1
 -1  4  4  4  4  4
  5  5  5  5  5  4
  5  5  5  5  5  4
  6  6  6  6 -1  4
```



Riešenie nájdené priemerne v generácii:

- 136 pri použití rulety aj turnaja zároveň, v 2 prípadoch nebolo riešenie nájdené
- 180 pri použití turnaja, v 5 prípadoch nebolo riešenie nájdené
- 82 pri použití rulety

Záver

Evolučný algoritmus úspešne kríži jedincov a dokáže pri správnych nastaveniach dosiahnuť rýchlo výsledok. V tomto zadaní avšak o tom, ako rýchlo a či vôbec sa nájde riešenie nerozhoduje hlavne evolučný algoritmus, ale algoritmus hrabania. Pre rozdielne mapy je rozdielny ideálny smer odbočenia mnícha v prípade prekážky, preto by možno pri riešení viacerých rozdielnych máp stálo za to nechať si v génoch mnícha aj inštrukcie ku otáčaniam.