Zadanie 3 - Zenová záhrada

Michal Franczel, FIIT, STU 19/04/2020

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.

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.

Implementácia

Zadanie bolo implementované v jazyku Python, s použitím knižnice numpy a matplotlib na vytváranie grafov a knižnice PyQt5 na GUI na tvorbu počiatočnej mapy. Program sa spúšťa spustením súboru Zen.py. Na nájdenie riešenia bol použitý konvenčný genetický algoritmus.

Genetický algoritmus

Pri spustení sa načíta požadovaná konfigurácia, pričom táto konfigurácia môže obsahovať viac setov parametrov pre genetický algoritmus. Zároveň (ak nie je stanovené inak) sa vytvorí mapa s náhodne rozmiestenými prekážkami, ktorých počet je stanovený užívateľom. V prípade, že chce užívateľ mať na mape aj listy, tie sa taktiež vygenerujú v zvolenom množstve. Samotný algoritmus prebieha nasledovne:

1. Prebehne **inicializácia**, v ktorej sa vytvorí počiatočná populácia, ktorá obsahuje *n* jedincov, ktorý majú náhodne zvolené gény.
2. Vypočíta sa *fitness* pre každého jednotlivca z populácie, pričom sa populácia zoradí na základe vypočítanej *fitness*

3. Ak je zapnutý elitarizmus, 20% jedincov z populácie sa presunie do ďalšej generácie a množstvo jedincov, ktorý sa budú krížiť sa zníži na 80%. Ak nie je zapnutý, ich počet bude tvoriť 100%.

4. Pokiaľ nie je naplnená nová generácia, generujú sa noví jedinci krížením.

   (a) Prebehne **selekcia**. Ak je zvolená selekcia ruletou, vyberú sa dvaja jedinci z predošlej generácie *ruletou*, inak sa vyberú *turnajom*.

   (b) Prebehne **kríženie**. Ak je zvolená možnosť náhodného kríženia, náhodne sa zvolí množstvo génov kopírovaných z jedného a druhého rodiča. Inak sa kopírujú gény v pomere 50:50.

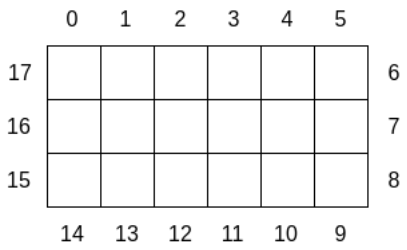
   (c) Výsledkom kríženia je jeden jedinec, u ktorého s pravdepodobnosťou *p* nastane **mutácia**. Ak nastane, vyberie sa náhodný gén a na jeho miesto sa vytvorý nový náhodný gén.

5. Vypočíta sa *fitness* populácie. Ak nie je súčasná generácia posledná, prejde sa na 3. bod. Inak pokračujeme.

6. Vyberieme prvého jedinca s najvyšším *fitness* z populácie a vypíšeme ho.

Gény

Gény sú reprezentované poľom čísel veľkosti polovice obvodu vstupnej mapy, pričom jedno číslo symbolizuje jeden vstup do mapy. Vstúpiť do mapy môže mních zo všetkých strán a teda číslo je maximálne veľkosť obvodu.



Pohyb mnícha

Vzhľadom na to, že gény znázorňujú vstupy na mapu, mních začína vstupom v prvom géne, pričom vždy keď vyjde použije ďalší gén v poradí ako vstup. V prípade, že sa zasekne, skončí. Ak ide smerom zhora alebo zdola a narazí na prekážku, tak skúsi ísť najprv vpravo, pričom ak aj tam je prekážka, ide vľavo. Ak ide smerom z ľava alebo z prava a narazí na prekážku, najprv skúša ísť dole, pričom ak sa tam nachádza prekážka, ide hore.

Listy

Miernou zmenou funkcie na zisťovanie *fitness* som implementoval možnosť zohľadnenia listov na mape. Existujú tri druhy listov - žlté, oranžové a červené. Na to aby sa mohli zbierať oranžové,

musia byť pozbierané žlté a na to, aby sa dali zbierať červené musia byť pozbierané aj oranžové. Počas prechádzania mapou sa postupne zbierajú listy, pričom sa zaznamenáva ich počet. Za každý pozbieraný žltý list sa pripočíta ku celkovej *fitness* 20, za oranžový 40 a červený 60, čím sa incentivizuje ich zbieranie.

Tvorba generácie

Tvorba novej generácie závisí na:

1. Elitarizme - či a koľko najlepších jedincov z predchádzajúcej generácie bude prenesených do novej
2. Selekcii - ako vybereme jedincov na párenie a teda tvorbu zvyšnej časti novej populácie
3. Krížení - aké časti využijeme z jedného a z druhého rodiča
4. Mutácii - či, ako a koľko génov bude náhodne nanovo vytvorených

V mojej implementácii je možné nastaviť či sa použije alebo nepoužije **elitarizmus**. V prípade, že je použitý, je 20% jedincov s najlepším *fitness* prekopírovaných do novej generácie, pričom títo nie sú nijakým spôsobom zmenení.

Následne zvyšnú časť (80% v prípade, že sme použili elitarizmus, inak 100%) tvoríme krížením jedincov z predošlej generácie. Pre každého jedinca, ktorého chceme vytvoriť, musíme **selekciou** vybrať rodičov, od ktorých zdedí gény. Tá môže prebiehať v závislosti od konfigurácie dvoma spôsobmi:

1. Turnajom - náhodne sa vyberie *n* (v našom prípade 3) jedincov z popúlácie, z ktorých sa porovnávaním vyberie najlepší. Táto metóda je časovo menej náročná ako nasledujúca.
2. Lotériou - spočíta sa celková *fitness* populácie a vyberie sa náhodné číslo menšie ako je celková *fitness* (zatočí sa kolesom). Postupne sa prechádza zoradenými hodnotami fitness a vypočítava sa suma týchto hodnôt. Ak je prekročené náhodne vybrané číslo, jedinec v aktuálnej iterácii je vybraný.

Dvaja jedinci vybraní selekciou sa *skrížia*, pričom sú implementované dva spôsoby kríženia:

1. Náhodné kríženie - vyberie sa náhodná časť génu jedného rodiča, pričom zvyšok sa doplní z druhého.
2. Kríženie, počas ktorého sa vyberie polovica genómu jedného rodiča a spojí sa s druhou polovicou genómu druhého rodiča

Takto vytvorený gén sa následne môže zmutovať, pričom sú implementované dva spôsoby mutácie:

1. S pravdepodobnosťou *p* prebehne mutácia, počas ktorej sa vyberie náhodný gén. ten sa nahradí náhodným číslom
2. Prechádza sa genómom jedinca a s pravdepodobnosťou *p* sa gén nahradí náhodným číslom.

Optimalizácia parametrov

Na zistenie vhodnej kombinácie parametrov som skúšal ich rôzne kombinácie, pričom som z nich vytváral grafy kvôli lepšej reprezentácii výsledkov. Testoval som mapu s rozmermi 20x15, na ktorej bolo 15 náhodne umiestnených prekážok a neboli používané listy. Každú konfiguráciu som spustil 50 krát, pričom počas každej iterácie boli prekážky generované na iných miestach. Výsledky som zaznamenal, zpriemeroval a na základe nich som vytvoril grafy.

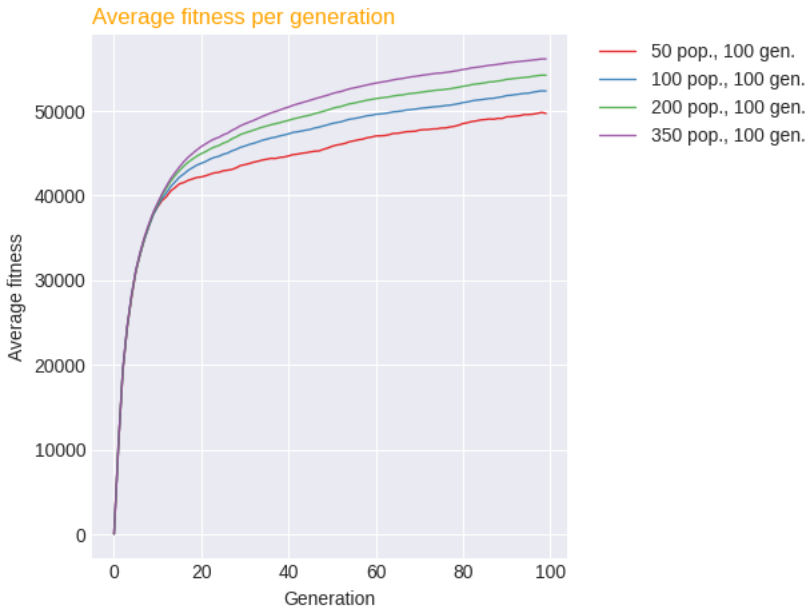


Figure 1: Graf zobrazujúci rozdieľ vo vypočítanej fitness vzhľadom na veľkosť populácie

V prvom grafe som znázornil rozdiely v priemernej fitness v štyroch situáciach rozlíšených veľkosťou populácie. Čím bola použitá populácia, tým vyššia bola výsledná priemerná fitness. Pri zvyšujúcej sa populácii ale zároveň aj rástla časová náročnosť a preto som použil pri ďalších testoch populáciu veľkosti 200, nie 350.

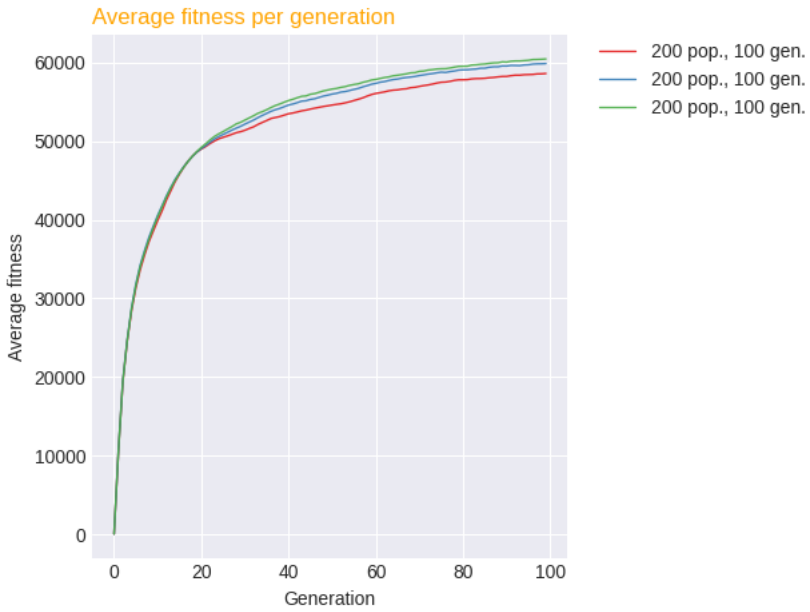


Figure 2: Graf zobrazujúci rozdieľ vo vypočítanej fitness vzhľadom na pravdepodobnosť mutácie

Pri veľkej populácii nie je až tak znateľný rozdiel spôsobený pravdepodobnosťou mutácie, no vyššia miera mutácie spôsobila lepšie výsledky. Červená čiara signalizuje 2% pravdepodobnosť mutácie, modrá 5% a zelená 10%. Tu treba však upozorniť na to, že bol použitý prvý spôsob mutácie. Pri vysokej miere mutácií v malých populáciach sa môže stať, že populácia bude príliž náhodná.

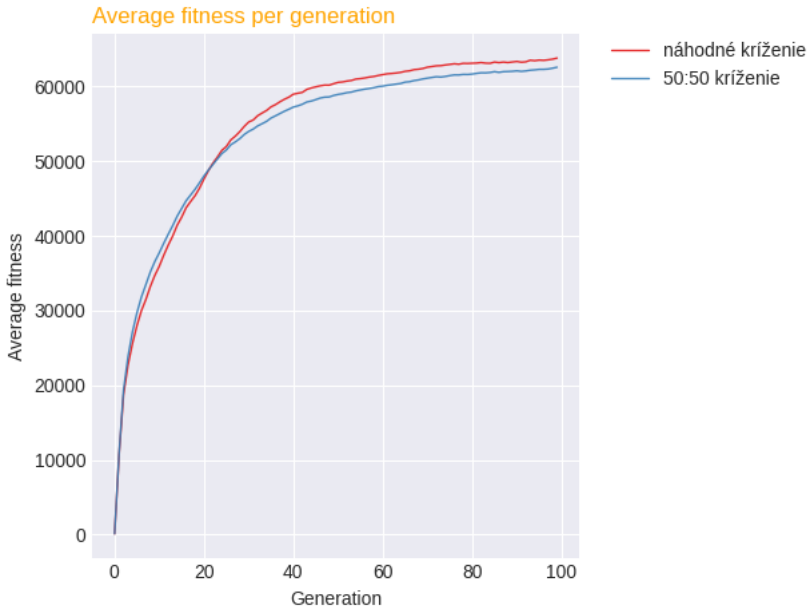


Figure 3: Graf zobrazujúci rozdieľ vo vypočítanej fitness vzhľadom na spôsob kríženia

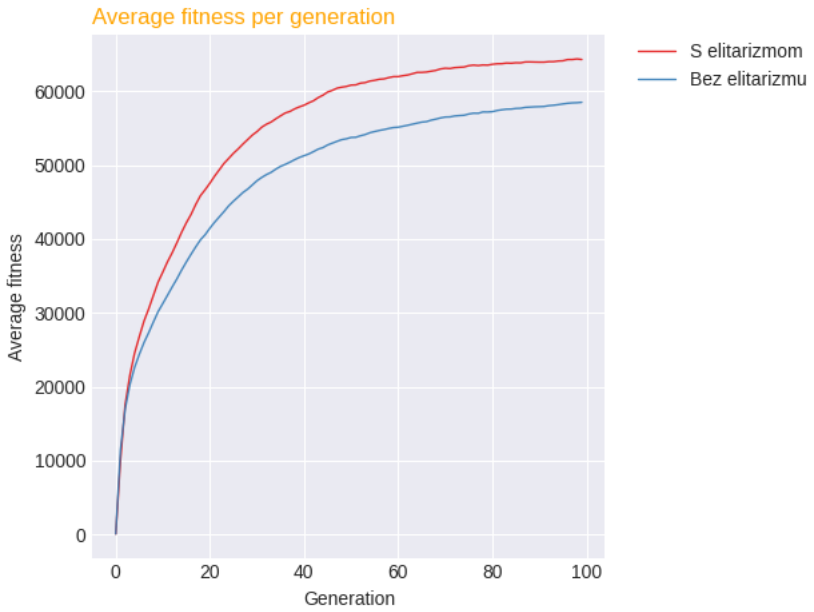


Figure 4: Graf zobrazujúci rozdieľ vo vypočítanej fitness vzhľadom na zapnutie/vypnutie elitarizmu

Pri spôsobe kríženia vyhralo náhodné kríženie. V prípade elitarizmu je omnoho lepšie mať zapnutý elitarizmus, pretože vďaka nemu sa uchovávajú najlepší jedinci a tým sa zvyšuje priemerná fitness.

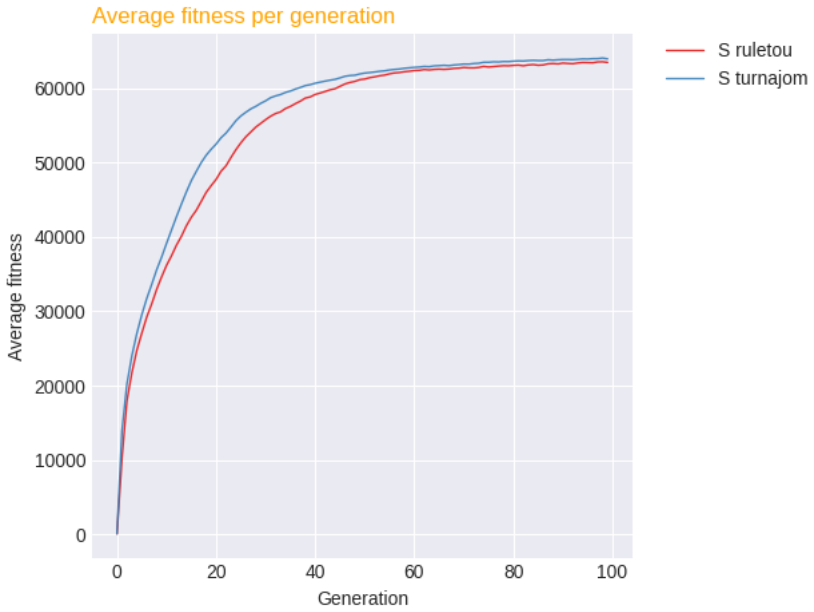


Figure 5: Graf zobrazujúci rozdieľ vo vypočítanej fitness vzhľadom spôsob selekcie

Selekcia jedincov nehrá až takú veľkú rolu vzhľadom na fitness, takže je lepšie v konečnom dôsledku použiť turnaj, ktorý má podobný výsledok s rýchlejším nástupom, pričom je zároveň aj rýchlejší a jednoduchší.

Zároveň som zoptimalizoval funkciu na výpočet fitness exponenciálnym počítaním fitness.

To spôsobilo, že čím je jeho hodnota vyššia, tým je väčšia incentíva na to, aby sa zlepšoval. Pri vysokých hodnotách nie je za normálnych okolností až taký priepastný rozdiel ak je hodnota o 1 vyššia, no mal by byť. Pre prípad, že by hodnota fitness viacej generácií stagnovala, som implementoval aj možnosť zastavenia, čím som znížil celkový čas vykonávania programu v prípade, že je tento parameter zle určený.