Jozef Waldhauser

Úloha 4

Generácia terénu pomocou knižnice DEAP a evolučného algoritmu

Obsah

[Úloha 4 – zadanie 2](#_Toc197116812)

[Úloha 4 – riešenie 2](#_Toc197116813)

[Design algoritmu 3](#_Toc197116814)

[Redesign sofistikovanej fitness funkcie 3](#_Toc197116815)

[Implementácia algoritmu 4](#_Toc197116816)

[Implementácia DEAP: 5](#_Toc197116817)

[Analýza 6](#_Toc197116818)

[Výstupy úlohy 4 7](#_Toc197116819)

[Podrobnejšia analýza fitness funkcie 8](#_Toc197116820)

[Experimenty s parametrami 10](#_Toc197116821)

[Mutácia 10](#_Toc197116822)

[Kríženie 10](#_Toc197116823)

[Súboj / Tournament 10](#_Toc197116824)

[Elitizmus 11](#_Toc197116825)

[Teoretické prepojenie 11](#_Toc197116826)

[Mutácia 11](#_Toc197116827)

[Kríženie 12](#_Toc197116828)

[Súboj / Tournament 12](#_Toc197116829)

[Elitizmus 12](#_Toc197116830)

[Fitness funkcia 12](#_Toc197116831)

[Záver 12](#_Toc197116832)

# Úloha 4 – zadanie

1. nainstalovat a rozchodit knihovnu DEAP
2. pomocí DEAP evolučně vygenerovat funkci terénu v závislosti na sofistikované fitness funkci

**Úlohou bolo nainštalovať a rozbehnúť knižnicu DEAP v pythone, implementovať evolučný algoritmus pre generovanie funkcie terénu vrátane sofistikovanej fitness funkcie.**

# Úloha 4 – riešenie

Skúsil som cez Claude AI vygenerovať kód, ktorý vygeneroval toto:

A graph showing a graph showing a graph

AI-generated content may be incorrect.

Dalo mi to pár nápadov, čiže dobrý odrazový bod.

Nasledovne som si prečítal učebnicu UI z predmetu a našiel na internete pár videí ako napr. [Genetic Algorithms in Python - Evolution For Optimization](https://www.youtube.com/watch?v=CRtZ-APJEKI)

Je tu pár konceptov, s ktorými budem pracovať:

* Populácia – populácia, ktorá bude (počiatočne) vygenerovaná náhodne
* Genóm – genetický materiál člena (počiatočne) generovaný náhodne  
  určený na zlepšovanie sa
* Fitness – kritériá, ktoré určujú „aký dobrý“ je genóm individuála z populácie
* Generácia – jedno
* Crossover – výmena genómu medzi dvoma indivíduami – rodičmi
* Mutácia – náhodná zmena v genóme
* Elitizmus – výber najlepších individualít a ich ponechanie do ďalšej generácie

# Design algoritmu

Umelá inteligencia mi dala dobrý nápad, je možné rozšíriť tento 2D terén o viac „2D terénov“ a z tohto urobiť 3D terén, skúsim aj to keď budú populanti veľmi podobní:  
či sa z toho nebude dať urobiť 3D terén resp. 3D mesh.

1. Vygenerovať počiatočnú populáciu:
   1. Vygenerovať náhodný genóm pre danú populáciu
2. Vykonať na populácií pomocou DEAP knižnice:
   1. Crossover
   2. Mutáciu
   3. Selekciu
   4. Elitismus
   5. Tournament
3. Evaluovať populáciu pomocou „sofistikovanej“ fitness funkcie:
   1. Zatiaľ budem pozerať na to, aby v genóme neboli medzi troma bodmi veľké rozdiely: 1-3-1 nie 1-9-1 (takýmto spôsobom)   
      číže niečo na báze  
      x-y-z fitness evaluácia
4. Najlepšieho individuála naplotuj do 2D reprezentácie
5. Hall of fame naplotuj do 3D reprezentácie s farbami

## Redesign sofistikovanej fitness funkcie

Keďže môj počiatočný plán nefungoval tak ako som si to predstavoval (veľký rozdiel medzi peakmi) tak som zmenil evaluáciu či je sledovaný element pod dvojnásobkom alebo rovnaký ako sused na X-násobok – teda urobil som to ako variable.

Toto tiež nepomáhalo vyrovnať náhodné peaky,  
tak som prerobil celú funkciu na to aby evaluovala či je sused in-range:  
čiže vynásobí jedinca EVALUATION\_DIFFERENCE a nasledovne skontroluje či je sused in-range (viď [Podrobnejšia analýza fitness funkcie](#_Podrobnejšia_analýza_fitness)).

# Implementácia algoritmu

Zaimplementoval som následovnú fitness funkciu:

|  |
| --- |
| def sophisticatedFitnessFunction(individual):      score = 0      variable = EVALUATION\_DIFFERENCE  # Variable used to define the range      genome = individual  # The genome is the individual itself        # Beginning: Check if the next neighbor is in range of the first element      if (genome[1] >= genome[0] / variable and          genome[1] <= genome[0] \* variable):          score += 1        # Middle: Check if both neighbors are in range of the current element      for i in range(1, len(genome) - 1):          if (genome[i - 1] >= genome[i] / variable and              genome[i - 1] <= genome[i] \* variable and              genome[i + 1] >= genome[i] / variable and              genome[i + 1] <= genome[i] \* variable):              score += 1        # End: Check if the previous neighbor is in range of the last element      if (genome[-2] >= genome[-1] / variable and          genome[-2] <= genome[-1] \* variable):          score += 1        return score,  # Return a tuple because DEAP expects fitness values to be tuples |

# Implementácia DEAP:

|  |
| --- |
| # initialize deap toolbox  toolbox = base.Toolbox()  # register functions for generating genes, individuals, and population  toolbox.register("**gene\_maker**", random.randint, MIN\_HEIGHT, MAX\_HEIGHT)  toolbox.register("**individual\_creator**", tools.initRepeat, creator.Individual,                   toolbox.gene\_maker, n=GENOME\_LENGTH)  toolbox.register("**population**", tools.initRepeat, list, toolbox.individual\_creator)  # register genetic algorithm operators  toolbox.register("**evaluate**", sophisticated\_fitness\_function)  toolbox.register("**mate**", tools.cxTwoPoint)  toolbox.register("**mutate**", tools.mutUniformInt, low=MIN\_HEIGHT, up=MAX\_HEIGHT,                   indpb=CROSSOVER\_RATE)  toolbox.register("**select**", tools.selTournament, tournsize=TOURNAMENT\_SIZE)  toolbox.register("**elitism**", tools.selBest, k=ELITISM\_SIZE) |

* **Gene\_maker** - generuje jeden gén (náhodné celé číslo) v rozmedzí od ‚MIN\_HEIGHT‘ po ‚MAX\_HEIGHT’ Používa sa na vytváranie hodnôt v genóme jedinca.
* **Individual\_creator** - vytvára jedinca (individuála) ako zoznam génov. Používa ‚tools.initRepeat‘ na opakovanie volania.
* **Population** - generuje populáciu ako zoznam jedincov.
* **Evaluate** – registruje funkciu sophisticated\_fitness\_function na evaluáciu jedinca - kontroluje, či susedné body v genóme sú v rozsahu určenom ‚EVALUATION\_DIFFERENCE’.
* **Mate** - Vymieňa časť genómu medzi dvoma rodičmi na dvoch náhodne vybraných bodoch, čím vytvára dvoch nových potomkov.
* **Mutate** - náhodne mení hodnoty génov v genóme. Každý gén má pravdepodobnosť mutácie ‚indpb’ (rovnú ‚CROSSOVER\_RATE’, 0.75) a nová hodnota je náhodné celé číslo v rozsahu ‚MIN\_HEIGHT’ až ‚MAX\_HEIGHT’.
* **Select** - vykonáva turnajovú selekciu. Vyberá ‚TOURNAMENT\_SIZE’ jedincov náhodne a z nich vyberie najlepšieho podľa fitness hodnoty. Tento proces sa opakuje na výber rodičov pre novú generáciu.
* **Elitism** - vyberá ‚ELITISM\_SIZE‘ najlepších jedincov z populácie podľa ich fitness hodnoty. Títo jedinci sú zachovaní do ďalšej generácie, aby sa nestratili kvalitné riešenia. Zároveň slúži ako „šírka“ pre graf.

Aby som vytvoril čo najpriaznivejší terén, tak som začal experimentovať s vstupnými parametrami.

Nakoniec som zistil že s touto konfiguráciou som najspokojnejší:

|  |
| --- |
| GENOME\_LENGTH = 40          # length of the terrain genome MIN\_HEIGHT = 0             # minimum height value MAX\_HEIGHT = 20            # maximum height value  POPULATION\_SIZE = 500      # number of individuals in population MUTATION\_RATE = 0.3        # probability of mutation CROSSOVER\_RATE = 0.7      # probability of crossover TOURNAMENT\_SIZE = 3        # size of tournament selection ELITISM\_SIZE = 10          # number of best individuals to keep EVALUATION\_DIFFERENCE = 1.2 # range factor for fitness evaluation MAX\_GENERATIONS = 200      # number of generations to run |

# Analýza

**GENOME\_LENGTH** = 40 - určuje dĺžku genómu, čiže počet bodov (40) v teréne, ktoré reprezentujú výšky na jednotlivých pozíciách

**MIN\_HEIGHT** = 0 - minimálna výška terénu, najnižší možný bod v genóme

**MAX\_HEIGHT** = 20 - maximálna výška terénu, najvyšší možný bod v genóme

**POPULATION\_SIZE** = 500 - počet jedincov v populácii, väčší počet zvyšuje rôznorodosť a šancu na nájdenie dobrého riešenia

**MUTATION\_RATE** = 0.3 - pravdepodobnosť mutácie génu

**CROSSOVER\_RATE** = 0.7 - pravdepodobnosť kríženia medzi jedincami, podporuje kombináciu dobrých vlastností rodičov

**TOURNAMENT\_SIZE** = 3 - počet jedincov v turnajovej selekcii, určuje koľko jedincov súťaží o výber

**ELITISM\_SIZE** = 10 - počet najlepších jedincov, ktorí sú automaticky prenesení do ďalšej generácie

**EVALUATION\_DIFFERENCE** = 1.2 - faktor určujúci rozsah povolených rozdielov medzi susednými bodmi v teréne, ovplyvňuje hladkosť terénu

**MAX\_GENERATIONS** = 200 - počet generácií, koľko krát sa populácia vyvíja, dostatočný počet na konvergenciu k dobrému riešeniu

## A graph showing a graph AI-generated content may be incorrect.Výstupy úlohy 4

A graph with a line

AI-generated content may be incorrect.A graph of a graph

AI-generated content may be incorrect.

## Podrobnejšia analýza fitness funkcie

Fitness funkcia priraďuje skóre  na základe počtu susedných bodov  
ktoré spĺňajú podmienku , kde = EVALUATION\_DIFFERENCE.

|  |  |
| --- | --- |
| EVALUATION\_DIFFERENCE | VÝSTUP |
| 1 |  |
| 1.1 |  |
| 1.4 |  |
| 1.8 |  |
| 2 |  |
| 10 |  |

**Zvyšovaním EVALUATION\_DIFFERENCE** umožňujeme aby aj **menej optimálne riešenia dosahovali vyššie skóre.** To znamená že aj ak je v danom genóme medzi individuálnymi génmi veľmi veľký rozdiel, stále budú mať vysoké skóre **a tým pádom sa jedinec počíta ako viac kvalitný, pričom prakticky nie je.**

## Experimenty s parametrami

Všetky experimenty boli konané s počiatočným nastavením „spokojnej konfigurácie“ ktorú je možno nájsť v tabuľke nad nadpisom [Analýza](#_Analýza).

### Mutácia

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Hodnota | Priemerný fitness | Kvalita terénu | Pozorovaná zmena |
| MUTATION\_RATE | 0.01 | 39.912 | Dobrá | Homogénnosť |
| MUTATION\_RATE | 0.02 | 39.136 | Chválitebná | Včš.variabilita |
| MUTATION\_RATE | 0.3 | 27.486 | Výborná | Krajšie |
| MUTATION\_RATE | 0.5 | 16.136 | Zlá | Viac hôr |
| MUTATION\_RATE | 0.9 | 2.054 | Veľmi zlá | Chaos |

Za optimálna hodnotu mutácie považujem interval (0.3, 0.5) väčšia hodnota spôsobuje prílišné zmeny medzi génmi v riešení a taktiež chaotický tvar terénu.

**Veľmi vysoká hodnota nedovolí nikdy nájsť optimálne riešenie.**

### Kríženie

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Hodnota | Priemerný fitness | Kvalita terénu | Pozorovaná zmena |
| CROSSOVER\_RATE | 0.01 | 35.72 | Zlá | Viac hôr |
| CROSSOVER\_RATE | 0.1 | 37.272 | Dobrá | Homogénnosť |
| CROSSOVER\_RATE | 0.3 | 32.89 | Dobrá | Včš.variabilita |
| CROSSOVER\_RATE | 0.7 | 28.516 | Výborná | Njvčš.variabilita |
| CROSSOVER\_RATE | 0.9 | 27.888 | Dobrá | Homogénnosť |

Zvyšovaním parametru kríženia zvyšujeme variabilitu konečného riešenia, optimálna hodnota sa nachádza v intervale (0.1, 0.3) kde je fitness najväčší. Subjektívne to vyzerá lepšie ak je väčšia variabilita terénu.   
**Veľmi vysoká hodnota značne predĺži hľadanie optimálneho riešenia.**

### Súboj / Tournament

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Hodnota | Priemerný fitness | Kvalita terénu | Pozorovaná zmena |
| TOURNAMENT\_SIZE | 1 | 1.82 | Veľmi zlá | Chaos |
| TOURNAMENT\_SIZE | 2 | 24.196 | Dobrá | Znormalizovanie |
| TOURNAMENT\_SIZE | 3 | 29.192 | Výborná | Včš.variabilita |
| TOURNAMENT\_SIZE | 4 | 29.238 | Dobrá | Včš.variabilita |
| TOURNAMENT\_SIZE | 10 | 20.436 | Zlá | Njvčš.variabilita |

Zvyšovaním parametru tournament size zväčšujeme počet jedincov, ktorí „súperia“. Hodnota 1 je nedáva zmysel, keďže súperí sám so sebou tým pádom vždy vyhrá. Za optimálne hodnoty považujem rozsah od 3 do 5 kde je najväčší fitness a aj dobrá variabilita. **Príliš nízkou alebo príliš vysokou hodnotou TOURNAMENT\_SIZE nedosiahneme optimálne riešenie.**

### Elitizmus

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Parameter | Hodnota | Priemerný fitness | Kvalita terénu | Pozorovaná zmena |
| ELITISM\_SIZE | 1 | 29.044 | Dobrá | Grf.neexistuje |
| ELITISM\_SIZE | 2 | 28.232 | Dobrá | Grf.nakreslený |
| ELITISM\_SIZE | 10 | 29.038 | Dobrá | Očak. výstup |
| ELITISM\_SIZE | 15 | 27.058 | Výborná | Včš.variabilita |
| ELITISM\_SIZE | 40 | 28.074 | Zlá | Homogénnosť |

Keďže som dosť nešťastne zvolil ELITISM\_SIZE ako šírku grafu, tak pri hodnote 1 sa graf vôbec nezobrazil – na vytvorenie grafu sú potrebné aspoň dve spojnice. Pri zvýšení tejto hodnoty na 2 je graf už správne vykreslený.

Každý jedinec v HallOfFame (hof) reprezentuje jeden 2D terén. Tieto terény sú zoradené pozdĺž y-osi, čím vytvárajú 3D mesh:

* **x-osa** zodpovedá pozícii v genóme,
* **y-osa** predstavuje index jedinca,
* **z-osa** udáva výšku terénu.

**Pri hodnotách ELITISM\_SIZE medzi 10 a 15 je dosiahnutá dostatočná variabilita, takže výstupný graf pôsobí rôznorodo a vizuálne atraktívne.**

**Pri hodnote 40 je nedostatočná variabilita kvôli tomu, že elitný jedinci sú si veľmi podobní.**

# Teoretické prepojenie

Implementácia evolučného algoritmu v tejto úlohe je založená na princípoch genetických algoritmov, ako sú opísané v učebnici [Umelá inteligencia](https://uuapp.plus4u.net/uu-bookkit-maing01/20b99da3ef5c44e988225284047e6958/book/page?code=home).

Genetické algoritmy napodobňujú proces prírodnej evolúcie, kde populácia jedincov prechádza iteráciami (generáciami) s cieľom optimalizovať danú úlohu podľa fitness funkcie.

Kľúčovým konceptom je rovnováha medzi exploráciou (hľadaním nových riešení) a exploatáciou (vylepšovaním už existujúcich riešení).

## Mutácia

A graph of green and grey triangles

AI-generated content may be incorrect.**Vysoká hodnota MUTATION\_RATE=0.3** podporuje exploráciu tým, že zavádza náhodné zmeny do genómov, čo umožňuje objavovať nové konfigurácie terénu. P**ríliš vysoká mutácia môže narušiť konvergenciu**, čo bolo pozorované pri MUTATION\_RATE=0.9, kde terény boli chaotické (priemerná fitness ~2.054).

Naopak, **nízka mutácia** (napr. 0.01) viedla k homogénnym, menej variabilným terénom, čo **obmedzuje rôznorodosť** (priemerná fitness ~39.912, ale s minimálnou variabilitou).

## Kríženie

**CROSSOVER\_RATE=0.7** zaisťuje exploatáciu kombináciou génov od rodičov, čím sa prenášajú dobré vlastnosti (napr. hladké úseky terénu) do potomkov. Dvojbodové kríženie, (z DEAP - tools.cxTwoPoint), zachováva štruktúru genómu lepšie ako jednobodové, čo podporuje postupné vylepšovanie terénov. Experimentovanie ukázali, že optimálna hodnota CROSSOVER\_RATE v intervale (0.1, 0.3) maximalizuje fitness (~37.272 pri 0.1), zatiaľ čo vyššie hodnoty (0.9) predlžujú konvergenciu.

## Súboj / Tournament

**TOURNAMENT\_SIZE=3** vytvára mierny selekčný tlak, čím sa vyvažuje rôznorodosť a kvalita. Teoreticky, turnajová **selekcia s nízkym počtom účastníkov** (napr. 3) **umožňuje aj menej fit jedincom prežiť, čím sa predchádza predčasnej konvergencii.** Pri TOURNAMENT\_SIZE=10 (priemerná fitness ~20.436) bol tlak príliš vysoký, čo viedlo k strate rôznorodosti a horším terénom.

## Elitizmus

ELITISM\_SIZE=10, **je kľúčový pre zachovanie najlepších riešení**, ako spomenuté v učebnici. Tým sa zamedzuje strate kvalitných terénov / riešení (napr. jedincov s fitness blízkou 40). **Vysoká hodnota** ELITISM\_SIZE=**40 viedla k homogénnosti, pretože elitní jedinci boli príliš podobní,** čo obmedzilo variabilitu grafu.

## Fitness funkcia

**Fitness funkcia minimalizuje výškové skoky,** čím napodobňuje plynulé prírodné terény. **Použitím EVALUATION\_DIFFERENCE=1.2 sa dosiahla rovnováha medzi prísnosťou** (hladké prechody) **a variabilitou** (priemerná fitness ~29.192 pri TOURNAMENT\_SIZE=3). Fitness funkcia stimuluje lokálnu hladkosť, ale nezohľadňuje globálnu štruktúru terénu, čo je obmedzenie.

# Záver

Vo vypracovaní bola použitá knižnica DEAP na implementáciu evolučného algoritmu pre generovanie hladkého terénu. Fitness funkcia s minimalizovala výškové skoky. Mutácia, kríženie, elitizmus optimalizovali variabilitu a kvalitu terénu. 2D graf zobrazoval najlepší terén, 3D graf vizualizoval hall of fame. Algoritmus simuluje do určitej miery prírodné terény a demonštruje princípy genetických algoritmov.