Slovenská technická univerzita v Bratislave
Fakulta Informatiky a Informačných Technológií

Umelá inteligencia

Evolučný algoritmus – Zenova záhrada

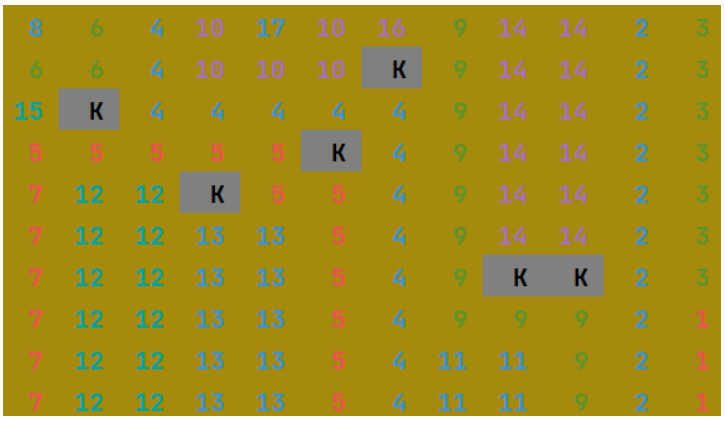
Emma Macháčová

Meno cvičiaceho : Ing. Ivan Kapustík
Čas cvičení : štvrtok, 12:00
Dátum vytvorenia : 17.Nov.2021, ZS 2021/22

Obsah

1. Zenova záhrada

Zenova 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.



Obrázok 1 Zenova záhradka (K - kameň)

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.

2. Zadanie

Uvedenú úlohu budeme riešiť 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.

Jedná sa o **klasický genetický algoritmus**, takže na začiatku, po načítaní rozmerov záhrady a pozícií kameňov, sa **vytvorí prvá generácia** jedincov s náhodne nastavenými génmi. Potom sa všetky jedince 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 takto vytvorení jedinci môžu s určitou pravdepodobnosťou aj **mutovať**. Vytvorí sa tak nová generácia a to sa vykonáva 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.

3. Opis programu

Program je písaný v jazyku Python (ver. 3.9). Pozostáva z nasledovných funkcií:

- ***najdiZaciatky****(x, y)* – slúži na zistenie súradníc okraju záhradky, odkiaľ vie na začiatku mních vojsť

- ***zistiUmiestnenie****(x, y)* – vie zistiť v ktorej časti záhradky sa nachádza mních (pravý okraj, ľavý...)
- ***porovnajSmer****(startX, startY, posX, posY)* – vie zistiť či je daný smer mnícha kolmý na smer odkiaľ vošiel do záhradky (tieto dve funkcie sa v aktuálnej verzii programu nepoužívajú)

- ***vytvorZahradu****(source)* – načíta vstup zo súboru a vytvorí maticu záhrady
- ***vypisZahradu****(policka)* – vypíše záhradu (v aktuálnom stave)
- ***pohrab****(mnich, flag)* – funkcia na prejdenie záhrady mníchom
- ***vytvorMnicha****()* – vytvorí nového mnícha
- ***sparuj****(rodic1, rodic2)* – funkcia na vytvorenie nového mnícha spárovaním predchodcov

- ***elitizmus****(populacia, generacia, maxMnich, maxMnichZaseknuty)* – výber do ďalšej generácie podľa najlepšieho percenta mníchov (podľa fitness)
- ***ruleta****(populacia, generacia, maxMnich, maxMnichZaseknuty)* – výber mníchov do ďalšej generácie ruletou

- ***vyries****(source)* – hlavná funkcia na riešenie zadania

Program využíva tieto **triedy a globálne premenné**:

```
class zahrada:
    zaciatky = []       # odkial sa da vstupit do zahradky - poradove cislo podla obvodu
    policka = []        # zahradka
    zahradaX = 0        # rozmer
    zahradaY = 0        # rozmer
    pocetKamenov = 0
    pocetPolicok = 0    # pocet policok zahradky
    obvod = 0           # pocet policok tvoriacich obvod
```

Trieda *zahrada* uchováva informácie o aktuálne používanej záhrade – jej rozmer, maticu, miesta odkiaľ sa do nej dá vojsť, počet kameňov ktoré v nej sú, dĺžku obvodu a počet políčok. Tieto informácie sa menia podľa aktuálnej záhrady.

```
class pohyb:
    POHYB_X = [1, 0, -1, 0]     # DOLE, DOPRAVA, HORE, DOLAVA
    POHYB_Y = [0, 1, 0, -1]     # 0       1       2      3
```

Trieda pohyb slúži na pohyb po matici – zmena súradnice X a Y.

```python
class glob:                                                    ⚠ 1  ⚠ 66
    START = 0       # urcuju kde u mnicha najdeme ktore informacie
    OPER = 1
    POC = 2
    KAMEN = 3

    VYBER = 0       # flag pre vyber sposobu vyberu mnichov
    ZASEKNUTIE = 0  # flag ci moze ukoncit hrabanie v zahradke

    LIMIT = 95000

    POCET_MNICHOV = 20          # pre (dalsie generacie) mnichov
    POCET_MNICHOV_PRVA_GEN = 20
    POCET_MAX_MNICHOV = 1
    POCET_MAX_MNICHOV_ZASEK = 1
    TOP_PERCENTO = 5
    POCET_NOVYCH_MUTACII = 5
    ROZSAH_PAROVANIA = 10
```

Trieda *glob* obsahuje najdôležitejšie (statické) premenné, ktorými sa program konfiguruje.

Premenné START až KAMEN určujú to, kde si mních v pamäti uchováva ktoré informácie:

- mnich[START, OPER, POC, KAMEN]
    - START – odkiaľ vchádza do záhrady (pole)
    - OPER – poradie rozhodovania sa ako zabočí pri stretnutí prekážky (pole)
    - POC – počet pohrabaných políčok (int)
    - KAMEN – koľkokrát narazil do kameňa – aktuálna verzia programu to nevyužíva (int)

Premenné VYBER a ZASEKNUTIE slúžia ako flagy – VYBER určuje, či sa do novej generácie vyberá pomocou rulety alebo elitisticky, ZASEKNUTIE určuje to, či mních môže na konci skončiť v záhradke (1 môže / 0 a iné nemôže).

LIMIT určuje po koľkých generáciách sa program ukončí ak nenájde riešenie.

Premenné pre konfiguráciu:

- POCET_MNICHOV určuje počet mníchov v jednej generácií
- POCET_MNICHOV_PRVA_GEN umožňuje iný počet mníchov v prvej generácií ako v nasledujúcich
- POCET_MAX_MNICHOV určuje, koľkokrát sa mních s najlepším výsledkom bude nachádzať v novej generácií
- POCET_MAX_MNICHOV_ZASEK určuje, koľkokrát sa mních s najlepším výsledkom (ktorý sa na konci pokusu zasekol v záhrade) bude nachádzať v novej generácií
- TOP_PERCENTO určuje aké veľké percento najlepších mníchov sa vyberie pri elit. metóde výberu
- POCET_NOVYCH_MUTACII udáva koľko mutácií má nastať
- ROZSAH_PAROVANIA určuje to, ktorí mnísi sa budú pri elit. výbere párovať

3.1. Použitý algoritmus

Program funguje nasledovne:

1. používateľ zadá v menu vstup,
2. program prečíta informácie o záhradke zo súboru a vytvorí maticu,
3. vytvorí sa prvá generácia mníchov (s nastavenými vlastnosťami),
4. mnísi sa pokúsi pohrabať záhradu a na základe ich výsledkov sa ohodnotia (fitness),
5. ak nenašli riešenie pokračuje sa výberom jedincov na tvorbu novej generácie,
6. vybraní mnísi sa spária,
7. pokračuje sa v hľadaní riešenia (bod 4)

3.1.1. Algoritmus tvorby mníchov

Každému mníchovi sa vytvoria dve polia: štartovacie pozície a poradie operátorov (zabočení).

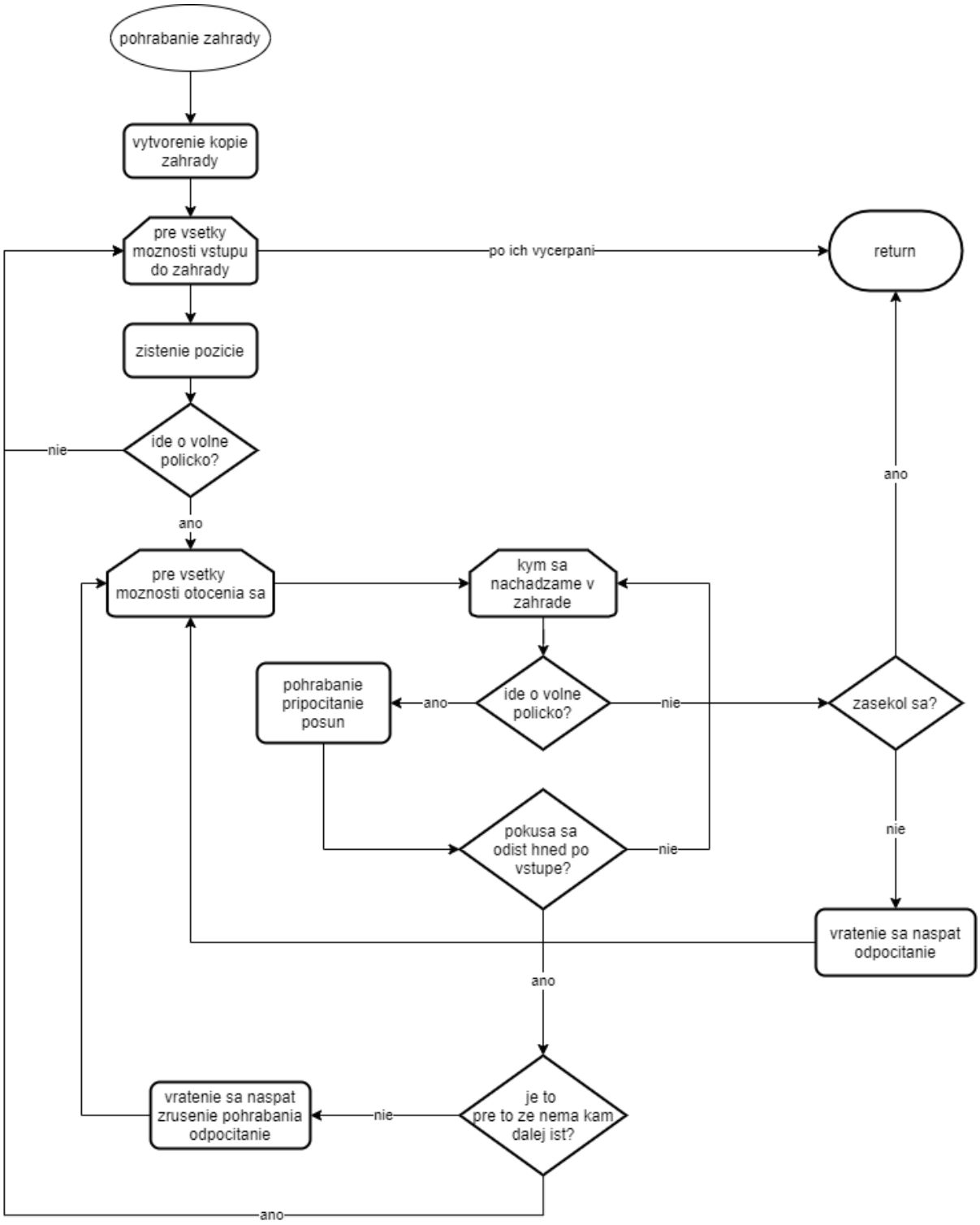
K štartovacím pozíciám - najskôr som sa pokúšala na začiatku každému mníchovi prideliť určitý počet náhodných vstupov – každému rovnaký počet ale rozdielne štartovacie umiestnenia v náhodnom poradí, tento spôsob sa ale ukázal veľmi neefektívny a riešenia nemali porovnateľný počet krokov. Tým pádom som zvolila iné riešenie – v **prvej generácií** môže mních vstúpiť do záhradky **hocikde**, no v závislosti od toho aké smery pohybu má určené, každý mních prvej generácie nakoniec **vstúpi z iných pozícií**, a pamätá si len tie pozície, odial **naozaj vstúpil** (teda jeho gén nemá dĺžku rovnú celému obvodu záhrady, ale iba jeho malú časť). **Nasledovné generácie** dedia už iba otestované vstupy.

Poradie zmeny smeru (operátorov) má každý mních iný (náhodný), o presne stanovenej dĺžke (pri ich vytváraní sa priebežne odstraňujú za sebou nasledujúce duplicitné operátory).

```
    # informacie o PORADI ZMENY SMERU
    while (len(operatory) < glob.POCET_OPERATOROV):
        sanca = randint(0, 80)

        if sanca < 20:
            operator = randint(0, 1)
        elif sanca < 40:
            operator = randint(1, 3)
        elif sanca < 60:
            operator = randint(1, 2)
        else:
            operator = randint(0, 3)

        operatory.append(operator)
        operatory = [x[0] for x in groupby(operatory)]  # odstranenie dupl. operatorov

    mnich = [starty, operatory, 0, 0]
    return mnich
```

3.1.2 Algoritmus hrabania záhrady

Hrabanie záhradky prebieha nasledovne:



Obrázok 2 Algoritmus hrabania záhrady

3.1.3. Algoritmus výberu – ruleta

Tento výber mníchov prebieha nasledovne:

- vypočíta sa fitness populácie
- každý mních dostane pridelenú pravdepodobnosť podľa fitness
- čím vyššia fitness tým viac „dielikov rulety“ je mníchovi pridelená
- na základe pravdepodobnosti sa náhodne vyberie mních (opakovane)
- ak je určené, pridá sa najúspešnejší mních viackrát
- ak je určené môže dôjsť k novým mutáciam
- potom prebieha párovanie medzi náhodne vybratými mníchmi z celej vybratej skupiny – počet mníchov novej generácie je určený v glob. premennej

3.1.4. Algoritmus výberu – elitný výber

Tento výber mníchov prebieha nasledovne:

- mnísi sa zoradia podľa fitness
- najlepších x % sa bez zmeny posunie do novej generácie
- ak je určené, pridá sa najúspešnejší mních viackrát
- ak je určené môže dôjsť k novým mutáciam
- určené percento vybraných mníchov sa spáruje a generácia sa doplní ich potomkami
- počet mníchov novej generácie je určený v glob. premennej

3.1.5. Algoritmus párenia mníchov

Párenie mníchov prebieha nasledovne:

1. vstup do záhradky

```
for i in range(0, lim_start):

   sanca = randint(0, 100)

   if poc1 > poc2 and sanca > 30:              # od rodica 1
      new_start.append(rodic1[glob.START][i])
   elif poc1 > poc2 and sanca <= 30:           # od rodica 2
      new_start.append(rodic2[glob.START][i])
   elif poc1 == poc2 and sanca < 20:           # mutacia
      gen = randint(0, zahrada.obvod - 1)
      new_start.append(gen)
   else:
      gen = int((rodic1[glob.START][i] + rodic2[glob.START][i]) / 2 )
      new_start.append(gen)
```

- pre všetky gény rodiča určujúce vstup do záhradky
    - vypočíta sa pravdepodobnosť
    - ak rodič 1 pohrabal viac políčok ako rodič 2, má väčšiu šancu posunúť ďalej svoje gény
    - v závislosti od pravdepodobnosti a pomeru pohrabaných políčok sa potom buď vyberie gén jedného z rodičov, alebo dôjde k mutácií

- keďže takto môže dôjsť k tomu, že by mal nový mních duplicitné gény a strácal by postupne informácie, dôjde k malej mutácií – vývinu – a dostane od rodičov pár génov naviac
- **počet génov však v konečnom dôsledku nenarastá, lebo nevyužité gény sa eliminujú**

```
for i in range(0, 5):                          # vyvin

   sanca = randint(0, 100)

      if poc1 > poc2 and sanca > 30:              # od rodica 1
         new_start.append(rodic1[glob.START][i])
      elif poc1 > poc2 and sanca <= 30:           # od rodica 2
         new_start.append(rodic2[glob.START][i])
      elif poc1 == poc2 and sanca < 20:           # mutacia
         gen = randint(0, zahrada.obvod - 1)
         new_start.append(gen)
      else:
         gen = int((rodic1[glob.START][i] + rodic2[glob.START][i]) / 2)
         new_start.append(gen)
```

2. operandy

Toto dedenie funguje obdobne, až na to, že ich počet je fixne daný a môžu sa vyskytovať duplikáty.

3.2. Opis vlastností génov

Mních si nesie tieto informácie:

- mnich[START, OPER, POC, KAMEN]
    - START – odkiaľ vchádza do záhrady (pole)
        - vstupy zistí a očísluje samostatná funkcia a uloží ich do globálnej premennej
        - je označený číslom (0 až počet políčok obvodu)
        - počet štartov nikdy nepresiahne polovicu obvodu, lebo to nie je možné – len prvá generácia nemá limitované odkiaľ vie vojsť do záhradky
        - ak mních pri hrabaní niektorý zo svojich vstupov nevyužije, nebude si ho ďalej pamätať
        - toto pole neobsahuje duplicitné čísla
    - OPER – poradie rozhodovania sa ako zabočí pri stretnutí prekážky (pole)
        - sú reprezentované číslami od 0 po 3
        - každé číslo znamená iný smer
        - operátory sa môžu v poli opakovať, nie však hneď za sebou
        - počet operátorov je daný globálnou premennou
    - POC – počet pohrabaných políčok (int)
    - KAMEN – koľkokrát narazil do kameňa – aktuálna verzia programu to nevyužíva (int)
        - tieto dve posledné informácie nepatria do genómu – nededia sa

3.3. Opis pohybu mnícha

Mních sa vie pohybovať v 4 smeroch (hore, dole, doprava, doľava). Tieto smery ale každý mních volí inak – pri strete s prekážkou sa vždy pozrie do svojej pamäte a skúsi nasledovný smer, po vyčerpaní všetkých smerov vie zistiť, že sa zasekol. Nie každý mních musí mať všetky možnosti zmeny smeru (niektorý napríklad vôbec nemusí vedieť ísť hore, atď.) Mních mení smer keď narazí na prekážku. Zo štartovacieho políčka vie vyštartovať hociktorým dostupným smerom – nezáleží na tom, na ktorej strane záhrady vstúpil, samotný vstup sa neráta ako smer pohybu (po odkomentovaní časti kódu je možná aj taká podmienka, kedy by sa vstup rátal ako začiatok smeru prechádzania).

3.4. Opis rozhodovania mnícha

Všetky rozhodnutia mnícha vyplývajú z jeho génov. Postupne volí začiatky a smery pohybov tak, ako je to už vysvetlené v predchádzajúcich bodoch.

- ak sa pokúsi vstúpiť sa políčko s prekážkou, nepostaví sa tam ale skúsi ďalší možný vstup – políčko s prekážkou odstráni z možných vstupov
- ak vstúpi na voľné políčko, vezme prvý smer z pamäte a pokračuje tým smerom, kým nenarazí na prekážku
- ak pri prechádzaní záhrady narazí na prekážku, pokračuje ďalším smerom, ktorý má v pamäti
- ak ďalší smer nepozná, alebo je obklopený prekážkami, jeho pokus končí
- ak vyjde von zo záhrady, opakuje postup z ďalšieho možného vstupu

3.5. Opis tvorby novej generácie

Viď bod 3.1.5.

3.6. Možnosti nastavenia parametrov

Hlavné parametre ovplyvňujúce chod programu sú nasledovné:

```
POCET_MNICHOV = 20
POCET_MNICHOV_PRVA_GEN = 20
POCET_MAX_MNICHOV = 1
POCET_MAX_MNICHOV_ZASEK = 1
TOP_PERCENTO = 5
POCET_NOVYCH_MUTACII = 5
ROZSAH_PAROVANIA = 10
```

Počet mníchov generácie sa ukázal najefektívnejší keď ide o nižšie čísla, aj keď tým pádom generácie narastajú rýchlejšie, program funguje tiež rýchlejšie (celkový počet použitých mníchov je tak však najmenší).

Počet najlepších mníchov, ktorí sa bez zmeny dostanú do ďalšej generácie je nastavený na 1 (resp. 2). Tento parameter nerobí až také veľké zmeny sám o sebe.

Top percento (ktoré sa posunie do ďalšej generácie) sa ukázalo efektívnejšie pri nižších hodnotách – nekrížia sa mnísi, ktorí nedosiahli tak dobré výsledky. Rozsah párovania sa správa obdobne.

Nové mutácie zabezpečujú to, že nepríde k „inbreeding-u“ a celý genóm populácie sa postupne neochudobňuje – nemôže ísť však o moc veľké číslo, lebo postupnosť nekonverguje dosť výrazne.

Pri všetkých možných parametroch je veľmi obtiažne nájsť presne najideálnejšiu kombináciu, hlavne pre náhodnosť génov prvej populácie.

3.7. Formát súboru záhrady

Vstup pre vytvorenie záhrady číta program z textového súboru v nasledujúcom tvare:

- rozmer záhrady Y – šírka (počet políčok)
- rozmer záhrady X – výška (počet políčok)
- súradnica X kameňa (0 až výška)
- súradnica Y kameňa (0 až šírka)
- ... ďalšie kamene

Napr. súbor pre záhradu 10x12 zo zadania vyzerá nasledovne:

4. Zhodnotenie vlastností systému

V aktuálnom stave program funguje **pomerne optimálne** – pre záhradku zo zadania by mal nájsť riešenie **vždy** (v závislosti od limitu), a to tiež pomerne rýchlo (pri rozumnom počte generácií – vzhľadom na nízky počet mníchov v jednej generácií).

Najlepšie funguje pri **nižších počtoch mníchov v jednej generácií** – čo sa týka času riešenia aj počtu mníchov potrebných pre nájdenie riešenia – ideálna kombinácia parametrov sa javí byť:

- výber ruletou,
- pri možnosti ukončenia trasy v záhrade,
- pri prvej generácií o veľkosti cca 30 mníchov,
- a pri veľkosti ostatných generácií medzi 10 a 20.

Pri vyššie uvedenom nastavení získame takýto výsledok:

```
Vysledok pre 10 testov:
pocet generacii na riesenie pri vybere ruletou:   5728.8 ; priemerne pohrabanych     114.0 / 114 policok,
priemerny cas na zahradku:      0.2680423561731974 min
```

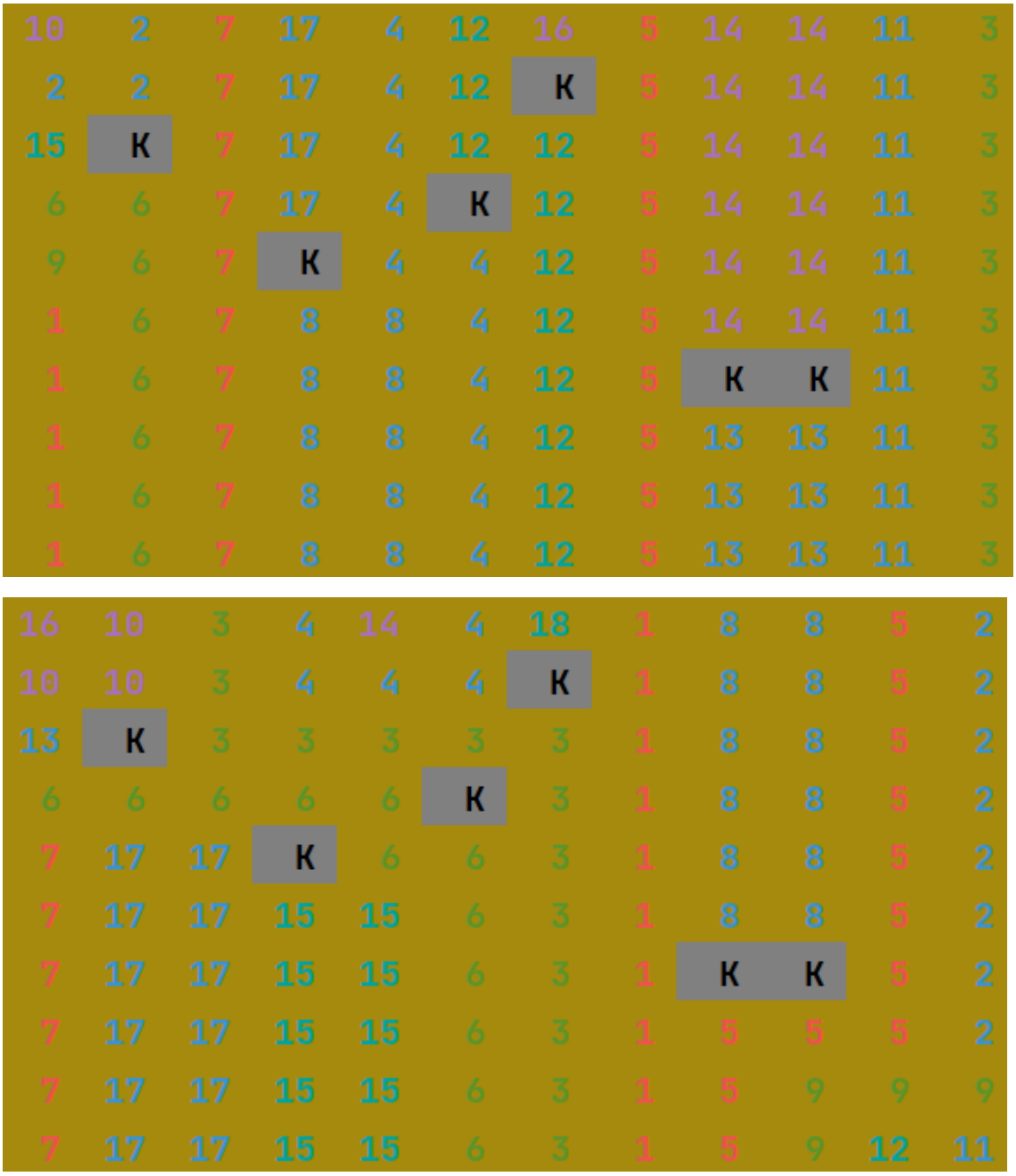
```
NASLI SME RIESENIE ! gen.c. 648
17  3  2  5 19  5 18  1 12 12 11  4
 3  3  2  5  5  5  K  1 12 12 11  4
10  K  2  2  2  2  2  1 12 12 11  4
 6  6  6  6  6  K  2  1 12 12 11  4
 7  8  8  K  6  6  2  1 12 12 11  4
 7  8  8 13 13  6  2  1 12 12 11  4
 9  8  8 13 13  6  2  1  K  K 11  4
 9  8  8 13 13  6  2  1 14 14 11  4
 9  8  8 13 13  6  2  1 14 14 11 15
16  8  8 13 13  6  2  1 14 14 11 15
```

V prípade, že máme takú požiadavku na mnícha, aby ukončil hrabanie mimo záhrady, počet generácií potrebných na nájdenie riešenia sa výrazne zvýši.

Ukážka výsledku testov pre 20 mníchov v generácií:

```
Vysledok pre 20 testov:
pocet generacii na riesenie pri vybere ruletou:   26017.5 ; priemerne pohrabanych    114.0 / 114 policok
pocet generacii na riesenie pri vybere elitizmom: 16953.4 ; priemerne pohrabanych    114.0 / 114 policok
Vysledok pre 100 testov:
pocet generacii na riesenie pri vybere ruletou:   29179.68 ; priemerne pohrabanych   113.96 / 114 policok
pocet generacii na riesenie pri vybere elitizmom: 22795.92 ; priemerne pohrabanych   113.98 / 114 policok
```

Ukážky pohrabania bez zaseknutia sa:



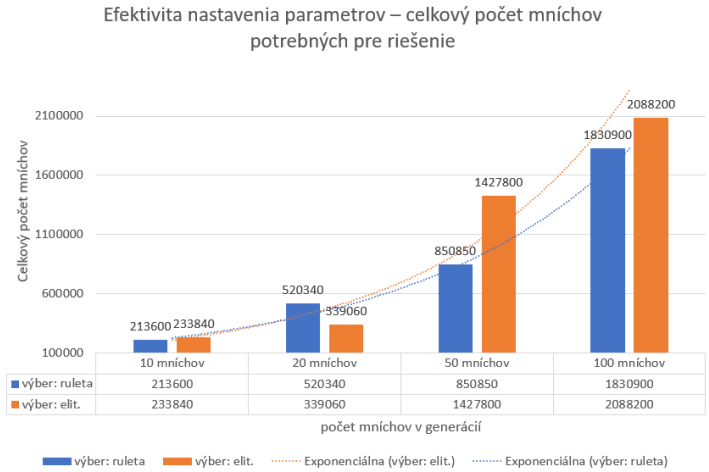
Obrázok 3 Ukážky pohrabania bez zaseknutia sa pre 10 mníchov v generácií

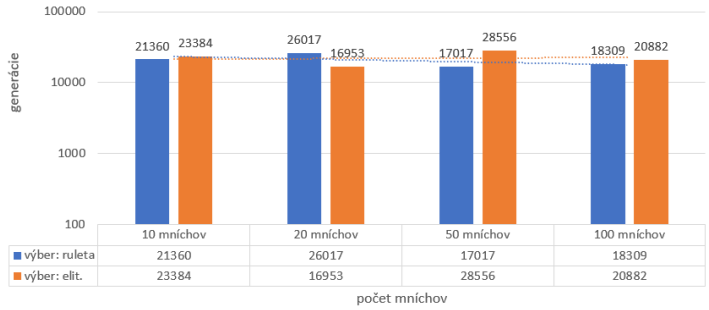
4.1. Porovnanie výsledkov pre rôzne nastavenia parametrov

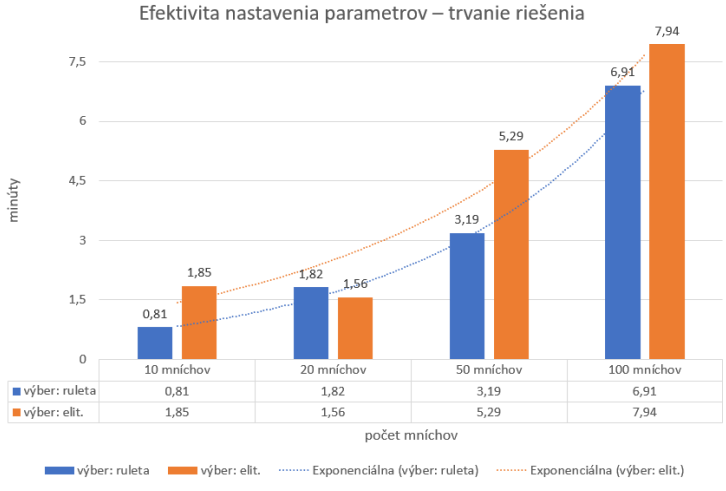
4.1.1. Pri možnosti ukončenia trasy vo vnútri záhrady

Nastavenie parametrov pre testovanie je nasledovné (mení sa počet mníchov) :

- POCET_MAX_MNICHOV; POCET_MAX_MNICHOV_ZASEK = 1
- TOP_PERCENTO; POCET_NOVYCH_MUTACII = 5
- ROZSAH_PAROVANIA = 10



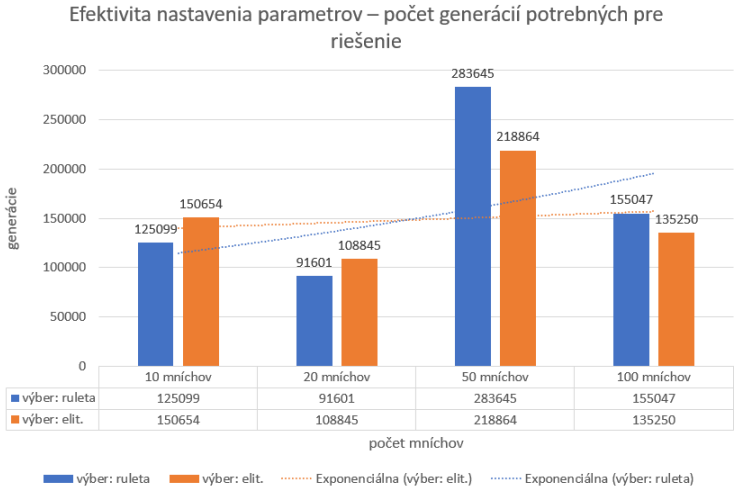


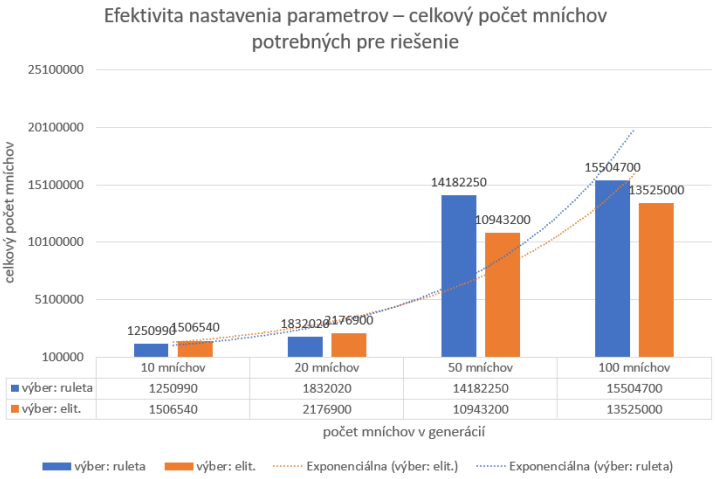


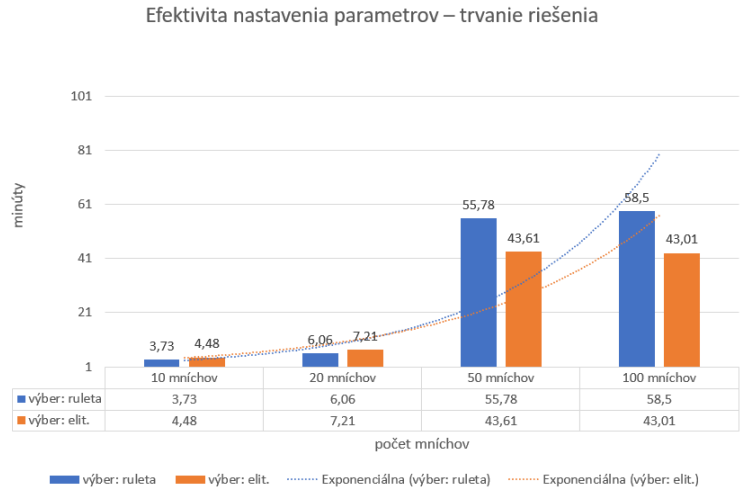
Ako najefektívnejší výber jedincov sa ukázala ruleta, a ako najvyrovnanejší variant počtu jedincov v každej generácií medzi 10 a 20 jedincov.

4.1.2. Bez možnosti ukončenia trasy vo vnútri záhrady

Nastavenie parametrov pre testovanie je rovnaké ako v predchádzajúcom bode, jediná zmena je možnosť ukončenia.







Výsledkom tohto merania je to, že efektívnejšou metódou výberu jedincov sa ukázal elitný výber. Program funguje optimálne pre počet jedincov v jednej generácií medzi 10 a 20.

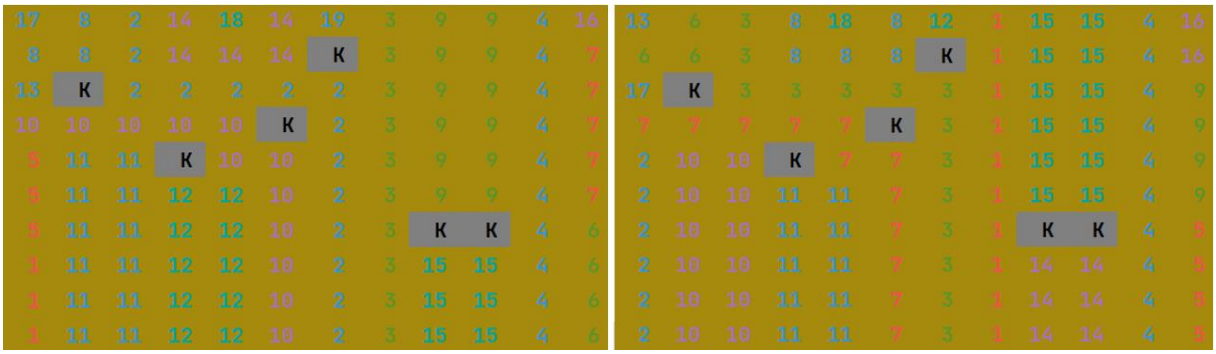
4.1.3. Porovnanie (náročnosti) riešenia v závislosti od miesta skončenia

```
najviac pohrabanych policok je:  109 / 114  generacia c.:    1926   (so zaseknutim)
najviac pohrabanych policok je:  110 / 114  generacia c.:    2486   (so zaseknutim)
najviac pohrabanych policok je:  100 / 114  generacia c.:    2591
najviac pohrabanych policok je:  104 / 114  generacia c.:    2732
najviac pohrabanych policok je:  114 / 114  generacia c.:    2828   (so zaseknutim)
najviac pohrabanych policok je:  111 / 114  generacia c.:    5388
najviac pohrabanych policok je:  113 / 114  generacia c.:    58371
najviac pohrabanych policok je:  114 / 114  generacia c.:    146701
```

Obrázok 4 Ukážka rozdielu náročnosti riešenia (pre 10 mníchov v generácií)

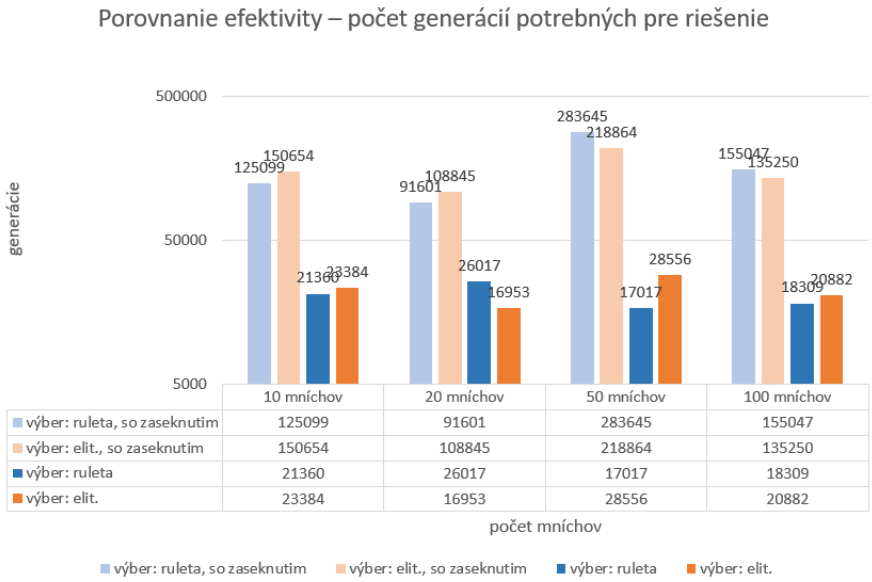
Vo výsledku, náročnosť riešenia za predpokladu, že mních na konci hrabania opustí záhradku, sa ukázala omnoho vyššia, ako v prípadoch keď na konci zostane zaseknutý v záhradke.

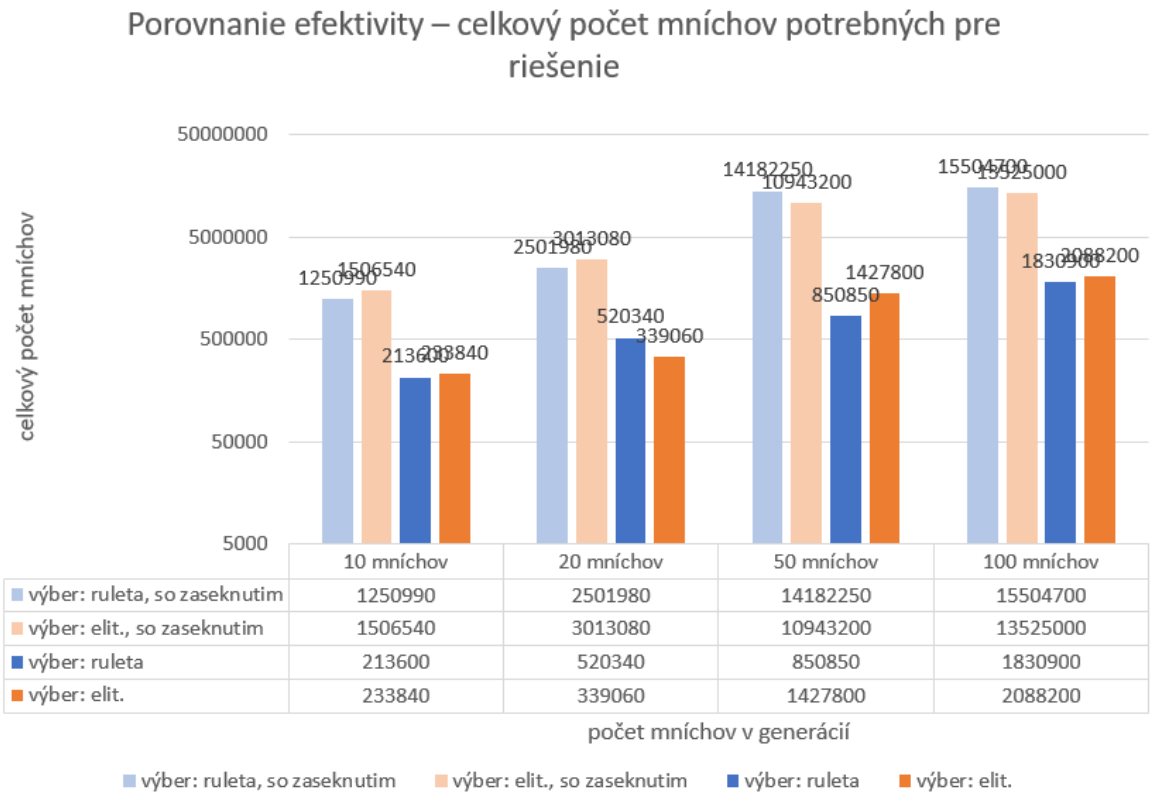
Riešenia v tom prípade, že mních záhradu opustí, boli takmer identické, len s rozdielnym poradím cestičiek a menšími zmenami trasy.



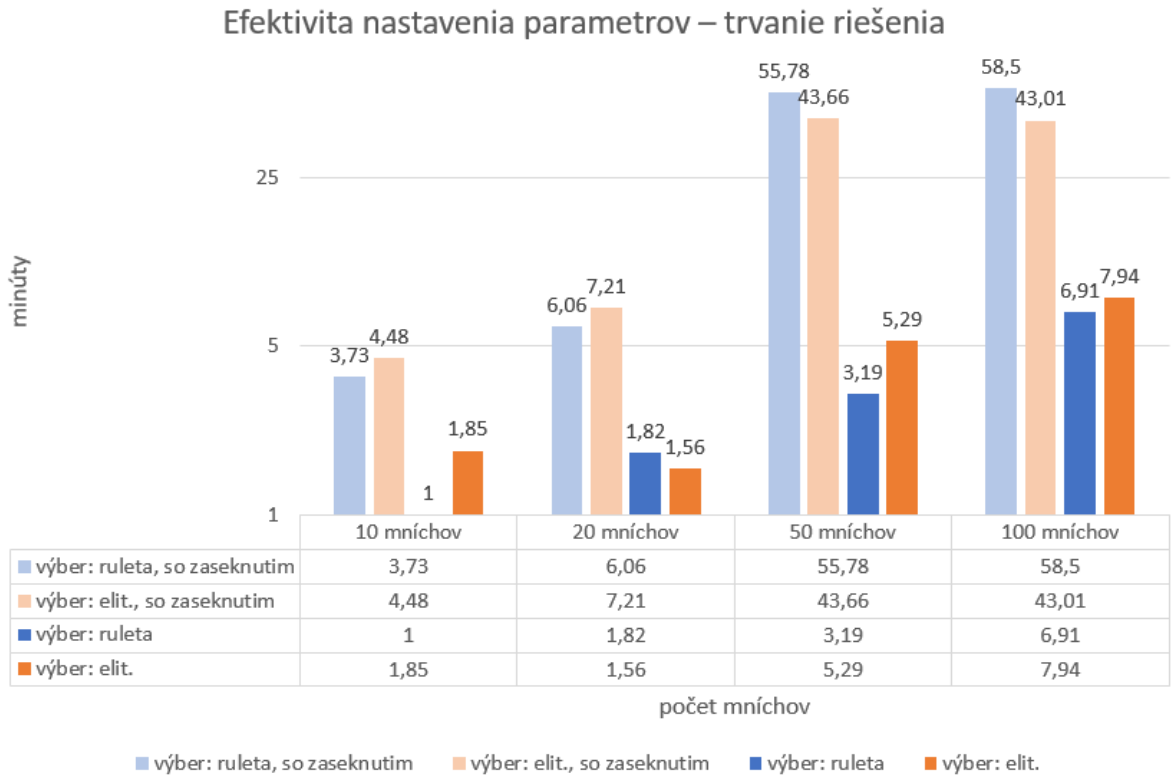
Obrázok 5 Ukážka totožnosti riešení

Pri porovnaní výsledkov predchádzajúcich testov pozorujeme to, že druhý spôsob ukončenia programu je výrazne náročnejší časovo aj pamäťovo.





	10 mníchov	20 mníchov	50 mníchov	100 mníchov
výber: ruleta, so zaseknutim	1250990	2501980	14182250	15504700
výber: elit., so zaseknutim	1506540	3013080	10943200	13525000
výber: ruleta	213600	520340	850850	1830900
výber: elit.	233840	339060	1427800	2088200



	10 mníchov	20 mníchov	50 mníchov	100 mníchov
výber: ruleta, so zaseknutim	3,73	6,06	55,78	58,5
výber: elit., so zaseknutim	4,48	7,21	43,66	43,01
výber: ruleta	1	1,82	3,19	6,91
výber: elit.	1,85	1,56	5,29	7,94

Kým v prvej sade testov bol efektívnejší výber ruletou, v druhej sade mierne vedie elitný výber.

4.2. Vývoj programu

4.2.1 Prvá fáza vývoja

```
Zahrada vytvorena, rozmer 10 x 12
pocet kamenov 6 pocet policok 114 obvod 40
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  K  0  0  0  0  0
 0  K  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  K  0  0  0  0  0  0
 0  0  0  K  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  K  K  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
Najviac pohrabanych policok je: 63 / 114 generacia c.: 1
Najviac pohrabanych policok je: 66 / 114 generacia c.: 2
Najviac pohrabanych policok je: 79 / 114 generacia c.: 10
Najviac pohrabanych policok je: 87 / 114 generacia c.: 16
Najviac pohrabanych policok je: 89 / 114 generacia c.: 79
Najviac pohrabanych policok je: 91 / 114 generacia c.: 40820
```

Maximálny počet pohrabaných políčok sa nedostal cez 91 – parametre boli nastavené nasledovne:

```
POCET_MNICHOV = 20
TOP_PERCENTO = 40
POCET_NOVYCH = 5
ROZSAH_PAROVANIA = 10
```

4.2.2. Druhá fáza vývoja

V tejto fáze bola prerobená funkcia na párovanie mníchov – program ale stále nepresiahol 91 políčok.

```
Zahrada vytvorena, rozmer 10 x 12
pocet kamenov 6 pocet policok 114 obvod 40
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  K  0  0  0  0  0
 0  K  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  K  0  0  0  0  0  0
 0  0  0  K  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  K  K  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0

najviac pohrabanych policok je: 57 / 114 generacia c.: 1
najviac pohrabanych policok je: 63 / 114 generacia c.: 2
najviac pohrabanych policok je: 70 / 114 generacia c.: 2
najviac pohrabanych policok je: 89 / 114 generacia c.: 6
najviac pohrabanych policok je: 91 / 114 generacia c.: 5626
```

```
MAX_POSUN = 3

POCET_MNICHOV = 20
TOP_PERCENTO = 40
POCET_NOVYCH = 5
ROZSAH_PAROVANIA = 10
```

4.2.3. Tretia fáza vývoja

```
Zahrada vytvorena, rozmer 10 x 12
pocet kamenov 6 pocet policok 114 obvod 40
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  K  0  0  0  0  0
 0  K  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  K  0  0  0  0  0  0
 0  0  0  K  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  K  K  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0

najviac pohrabanych policok je: 86 / 114 generacia c.: 1
najviac pohrabanych policok je: 87 / 114 generacia c.: 4
najviac pohrabanych policok je: 89 / 114 generacia c.: 11
najviac pohrabanych policok je: 91 / 114 generacia c.: 13
najviac pohrabanych policok je: 93 / 114 generacia c.: 14
najviac pohrabanych policok je: 97 / 114 generacia c.: 29
najviac pohrabanych policok je: 99 / 114 generacia c.: 54
najviac pohrabanych policok je: 104 / 114 generacia c.: 175
najviac pohrabanych policok je: 105 / 114 generacia c.: 260
najviac pohrabanych policok je: 107 / 114 generacia c.: 730
```

```
MAX_POSUN = 3

POCET_MNICHOV = 50
POCET_MAX_MNICHOV = 15
TOP_PERCENTO = 15
POCET_NOVYCH = 20
ROZSAH_PAROVANIA = 20
```

Po vylepšení funkcie na pohrabanie záhradky sme sa dostali cez 100 pohrabaných políčok, tam sa progres ale zastavil.

4.2.4. Štvrtá fáza vývoja

V tejto fáze sa program dlho nevedel dostať k poslednému políčku.

```
Zahrada vytvorena, rozmer 10 x 12
pocet kamenov 6 pocet policok 114 obvod 40
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  K  0  0  0  0  0
 0  K  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  K  0  0  0  0  0  0
 0  0  0  K  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  K  K  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0

najviac pohrabanych policok je: 95 / 114 generacia c.: 1
najviac pohrabanych policok je: 97 / 114 generacia c.: 33
najviac pohrabanych policok je: 100 / 114 generacia c.: 34
najviac pohrabanych policok je: 105 / 114 generacia c.: 42
najviac pohrabanych policok je: 107 / 114 generacia c.: 97
najviac pohrabanych policok je: 110 / 114 generacia c.: 228
najviac pohrabanych policok je: 111 / 114 generacia c.: 2207
najviac pohrabanych policok je: 113 / 114 generacia c.: 3552
```

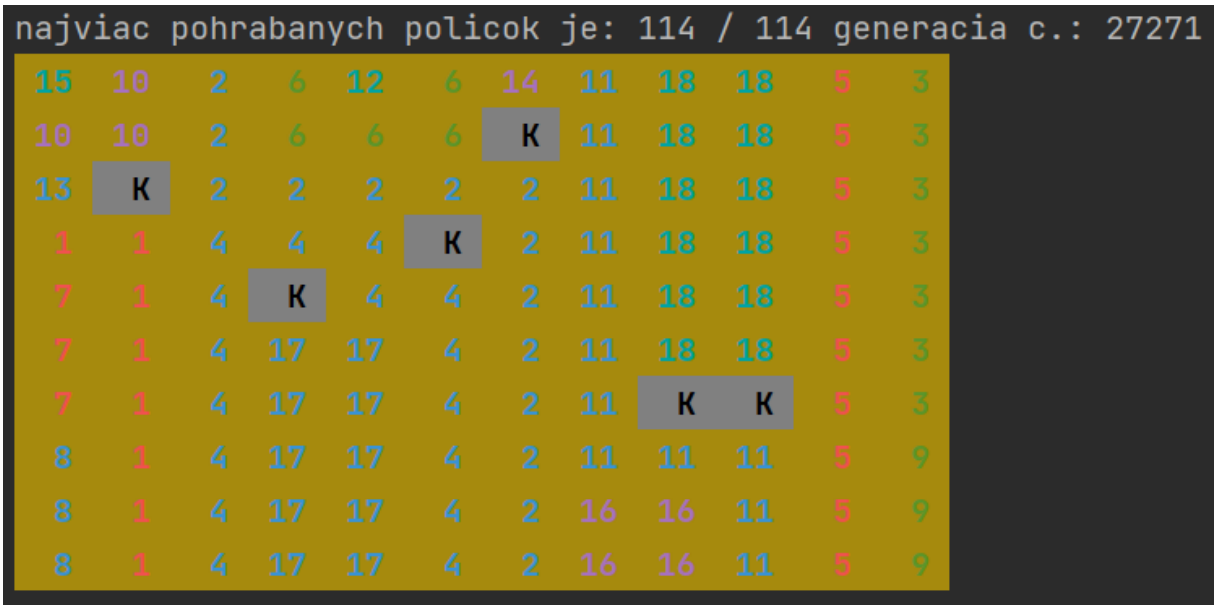
```
MAX_POSUN = 3

POCET_MNICHOV = 50
POCET_MAX_MNICHOV = 0
TOP_PERCENTO = 10
POCET_NOVYCH = 20
ROZSAH_PAROVANIA = 45
```

```
Zahrada vytvorena, rozmer 10 x 12
pocet kamenov 6 pocet policok 114 obvod 40
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  K  0  0  0  0  0
 0  K  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  K  0  0  0  0  0  0
 0  0  0  K  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  K  K  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0
 0  0  0  0  0  0  0  0  0  0  0  0

najviac pohrabanych policok je: 75 / 114 generacia c.: 1
najviac pohrabanych policok je: 79 / 114 generacia c.: 2
najviac pohrabanych policok je: 84 / 114 generacia c.: 2
najviac pohrabanych policok je: 91 / 114 generacia c.: 14
najviac pohrabanych policok je: 98 / 114 generacia c.: 17
najviac pohrabanych policok je: 103 / 114 generacia c.: 47
najviac pohrabanych policok je: 107 / 114 generacia c.: 145
najviac pohrabanych policok je: 113 / 114 generacia c.: 763
```

```
MAX_POSUN = 15

POCET_MNICHOV = 50
POCET_MAX_MNICHOV = 0
TOP_PERCENTO = 5
POCET_NOVYCH = 20
ROZSAH_PAROVANIA = 40
```

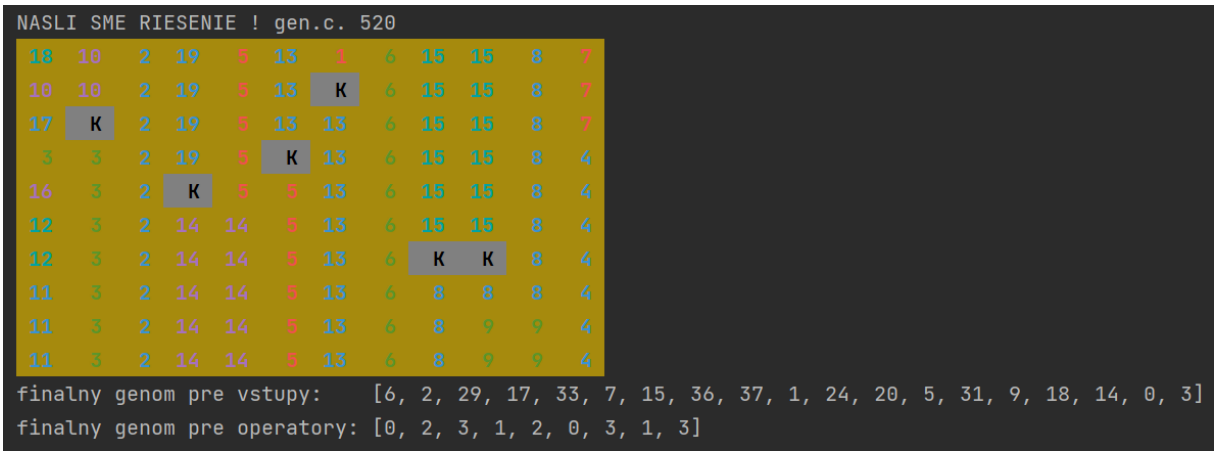
4.2.5. Piata fáza – prvá funkčná verzia

V tejto fáze bolo spravených viacero výrazných zmien – celá funkcia na pohrabanie záhradky bola prerobená, a mních si začal pamätať informácie iným spôsobom. Stále bol ale jeho genóm priveľmi rozsiahly a program bol neefektívny – príliš pomalý a nepredvídateľný.



4.2.6. Aktuálna fáza

Funkcia na kríženie mníchov bola prerobená, rovnako aj inicializácia mníchov a konfigurácia parametrov. Tiež bol pridaný nový spôsob výberu mníchov.



5. Ďalšie možné vylepšenia

Program síce prešiel veľkým množstvom optimalizovania a momentálne funguje pomerne dobre, stále by bolo možné nájsť spôsoby vylepšenia:

- nájdenie lepšej kombinácie parametrov,
- mnísi by si okrem pokynov, ktoré majú robiť mohli zo skúsenosti pamätať veci, ktoré nemôžu robiť, a pri odovzdávaní informácií by mohli tie informácie, ktoré už vie, že ho nikam neposunú, ignorovať,
- najväčší priestor na vylepšenie má v danej fáze podľa mňa funkcia na kríženie mníchov,
- mních by mohol vedieť pozerať sa okolo seba – smer cesty by nevolil podľa svojej pamäte, ale pozeral by sa, kam môže a kam nemôže ísť,
- mních by mohol vedieť, ktoré políčka ešte nepohrabal a mohol by sa zámerne snažiť ísť ich smerom