PA026

AI Platformer

Matěj Bukáček

Úvod

V rámci předmětu PA026 jsem se rozhodl vytvořit projekt skládající se ze dvou hlavních částí. První část je 2D platformer, kde uživatel vytvoří úroveň a následně zvolí algoritmus, který se ji následně pokusí vyřešit. Druhá potom je nekonečná úroveň, kde umělá inteligence optimalizuje své parametry s cílem dostat se co nejdále.

Projekt je implementovaný v programovacím jazyce Python, hlavní použité knihovny jsou **pygame** (vzhled), **numpy** a **matplotlib** (pouze interpretace výsledků), **pickle** (ukládání, načítání) a **random**.

1. část - Řešení uživatelem vytvořené mapy

Editor

Vytváření úrovní se odehrává v jednoduchém editoru, kde se kliknutím a tažením myší vytváří platformy, pravým tlačítkem myši se mažou, klávesou *S* se vytvoří start, klávesou *C* checkpoint a klávesou *F* cíl (finish). Pro uložení mapy lze využít klávesu *X* nebo zavřít editor křížkem v pravém horním rohu.

Implementované algoritmy

Pro první část projektu je implementovaný genetický algoritmus a dvě verze mravenčí kolonie. Každý algoritmus umožňuje při spuštění specifikovat vnitřní parametry, které jsou rozepsané u jednotlivých algoritmů.

Společné modifikovatelné parametry:

- **population** - počet agentů
- **steps** - počáteční počet kroků
- **step_inc** - zvýšení počtu kroků po každé iteraci

Genetický Algoritmus (EvolAI)

Každý agent v genetickém algoritmu dostane akce (plán) které má postupně provést. Po každé iteraci se vyhodnotí úspěšnost jednotlivých agentů. Několik nejlepších jedinců se zachová a prodlouží se jim plán o náhodné akce. Nová generace se vytvoří z přeživších a jedinců kteří vzniknou kombinací dvou členů staré generace, kteří jsou vybráni pomocí turnaje. Kombinace probíhá náhodným zvolením indexu plánu, část plánu nového jedince od začátku po index se naplní plánem prvního rodiče, část od indexu do konce se naplní plánem druhého rodiče, konec plánu se doplní náhodnými akcemi. Následně každý jedinec s určitou pravděpodobností mutuje, což změní náhodnou část jeho plánu.

Modifikovatelné parametry:

- **survivors** - počet přeživších po každé iteraci
- **mut_rate** - pravděpodobnost mutace
- **tournament_size** - velikost turnaje

Mravenčí kolonie 1 (AntColonyOrigin)

V první iteraci každý agent vykoná náhodné akce. Po každé iteraci se vyhodnotí úspěšnost jednotlivých agentů. Cesta nejúspěšnějšího agenta se posílí vytvořením “feromonové cesty”, která během několika následujících iterací slábne a následně zmizí. V každé iteraci agent v každém kroku volí akci s váženou pravděpodobností, podle toho, kterých feromonů se v danou chvíli dotýká.

Modifikovatelné parametry:

- **pheromone_strength** - počáteční síla feromonu
- **pheromone_evaporation** - kolik procent ze síly feromonu zůstane iteraci
- **pheromone_lifetime** - počet iterací, po kterých feromony mizí

Mravenčí kolonie 2 (AntColony)

Základní fungování algoritmu je stejné jako u *AntColonyOrigin,* změna je v optimalizování cesty. Narozdíl od základní verze, tato neoptimalizuje celou cestu jako celek, ale optimalizuje jednotlivé části mezi checkpointy. Finální cestu musí však stále projít jeden

agent, z důvodu potenciální nespojitosti částí cesty (průchod není jednoduchý markovův řetězec, záleží i na několika předchozích akcích a stavech).

Modifikovatelné parametry jsou stejné jako *AntColonyOrigin*

Spuštění

Pro spuštění simulace stačí spustit soubor *simulation.py.*

Po spuštění *simulation.py* uživatel zvolí jednu z map, kterou dříve vytvořil. Poté zvolí algoritmus ze seznamu. Ten následně uživatele vyzve ke specifikování jednotlivých parametrů (prázdné pole zachová defaultní hodnoty). Uživatel zvolí počet iterací a potom už jen sleduje pokusy algoritmu. Po skončení se vypíší základní informace o výsledku. Pro náročnější uživatele je připravena funkce tests, kterou lze upravit pro spuštění více úrovní na různých algoritmech v sekvenci a následně je porovnat.

2. část - Nekonečná mapa

Pro nekonečnou mapu je implementovaný pouze jeden algoritmus, a to verze genetického algoritmu, která funguje pomocí rozhodovacího stromu a optimalizuje thresholdy v jednotlivých uzlech.

Uživatelské rozhraní v této části není implementováno, parametry (počet snímků za vteřinu, šířka platforem atd.) lze modifikovat přímo v kódu na začátku souboru *infMapAlt.py*. počet iterací lze upravit na konci téhož souboru.

Implementovaný algoritmus

Pro tuto část projektu je implementovaný pouze jeden algoritmus, který funguje na principu rozhodovacího stromu a thresholdů, podle kterých se agent rozhoduje.

Na konci každé iterace se vyhodnotí agenti, několik nejlepších se zachová, vytvoří se několik úplně nových a několik se vytvoří zkombinováním agentů předchozí iterace.

(Následující odstavec nepopisuje rozhodovací algoritmus jedna ku jedné, pouze nastiňuje základní fungování. přesná specifikace rozhodování je v InfMapAlt.py ve třídě InfPlayer v metodě next_step)

Pro agenta existují 2 základní situace - stojí na platformě nebo je ve vzduchu. Pokud stojí na platformě, tak si volí novou platformu, na kterou se pokusí dostat. Jeden z parametrů

agentova genomu určuje, jaká je maximální vertikální vzdálenost mezi ním a platformou, kterou zvolí. Pokud je agent ve vzduchu, tak se většinu času buď snaží dostat k cílové platformě (je pod ní a pohybuje se vzhůru), nebo cílovou platformu minul (je pod ní a pohybuje se dolů) a v tom případě hledá platformu pod sebou, kde by se mohl zachránit.

Spuštění

Pro spuštění stačí spustit soubor *infMapAlt.py.*

Popis prostředí

Obsah souborů

ai_base.py

- obsahuje “template”, kterým se řídí implementace algoritmů

ant_colony.py

- obsahuje implementaci *AntColony*

ant_colony_original.py

- obsahuje implementaci *AntColonyOrigin*

constants.py

- obsahuje pomocné konstanty

editor.py

- soubor určený ke spuštění
- umožňuje vytvořit vlastní mapu

editor_constants.py

- obsahuje konstanty editoru

evolutionary_alg.py

- obsahuje implementaci *EvolAI*

infMapAlt.py

- soubor určený ke spuštění
- obsahuje vše pro část projektu s nekonečnou mapou

main.py

- soubor určený ke spuštění
- umožňuje zvolit mapu a ručně ji projít (ovládání pomocí šipek)

misc.py

- obsahuje různé pomocné funkce

simulation.py

- soubor určený ke spuštění
- umožňuje zvolit mapu a algoritmus, který se má pokusit ji vyřešit

sprites.py

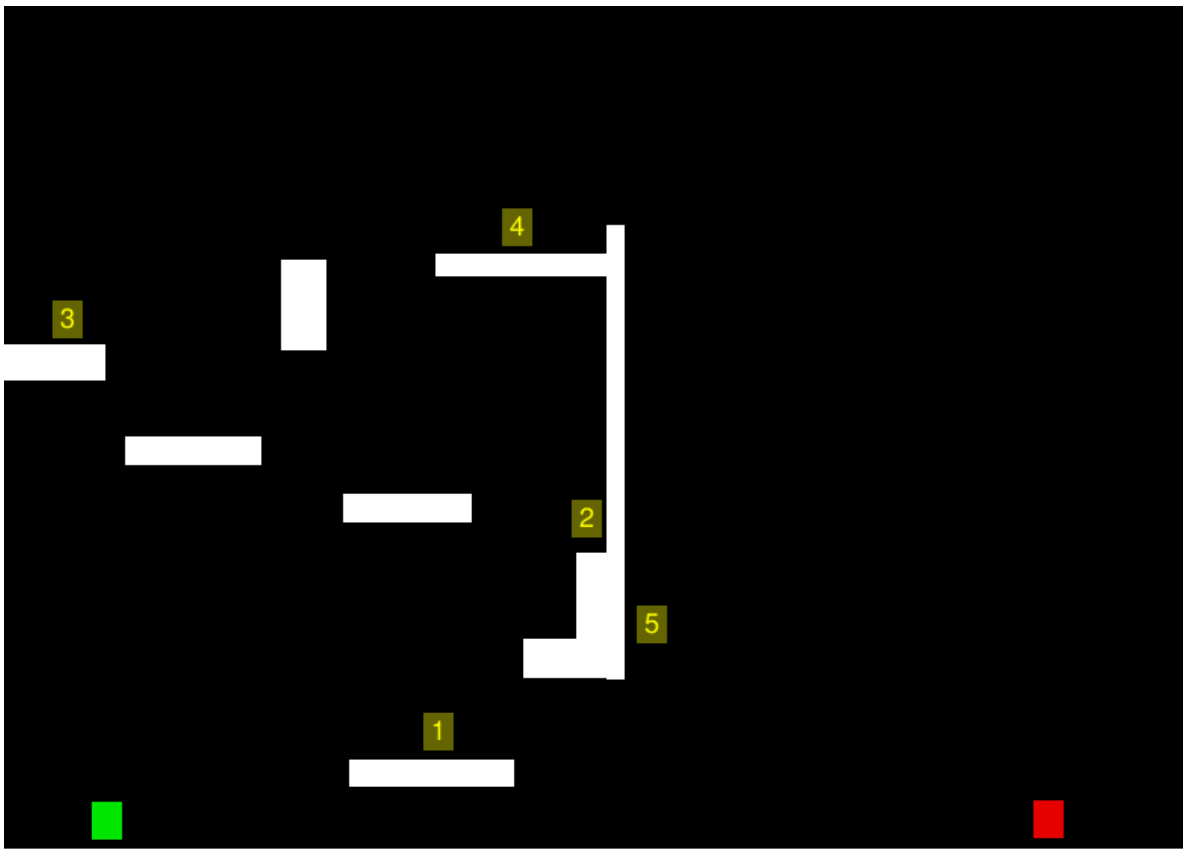
- obsahuje třídy implementující platformy, pohyb agentů a speciální body (start, checkpoint, cíl)

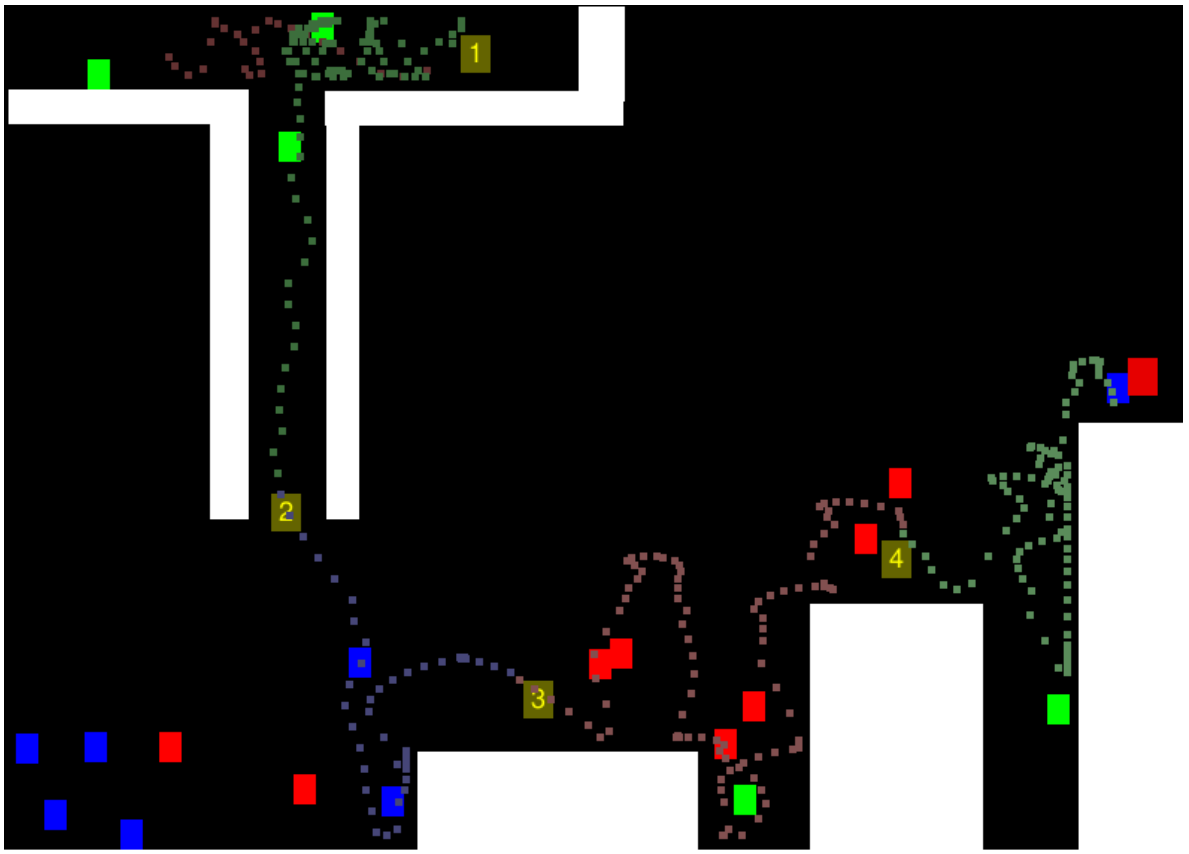
levels

- tato složka se vytvoří po uložení 1. mapy, slouží pro ukládání uživatelských map

GUI - obrázky

První obrázek ukazuje mapu v editoru. Druhý obrázek je průběh řešení jiné mapy algoritmem *AntColonyOrigin*.


4
3
2
5
1


1
2
4
3

Vyhodnocení

1. Část

Vytvořil jsem sadu map, které se všechny implementované algoritmy následně pokoušely vyřešit. Podle výsledků je nejúspěšnější *AntColonyOrigin* a nejhůře na tom je *EvolAI*. Jednalo se však o malou testovací sadu a při opakovaných spouštěních se výsledky lišily. Všechny testovací algoritmy využívají k hledání cesty náhodu, tudíž nelze jednoznačně říct, který algoritmus je nejlepší.

Tabulka výsledků

map	AI	max it	first path iteration	first path length	best path length
e1	EvolAI	80	45	196	166
e1	AntColony	80	41	170	129
e1	AntColonyOrigin	80	34	164	80
e2	EvolAI	80	36	134	134
e2	AntColony	80	33	172	70
e2	AntColonyOrigin	80	32	107	98
e3	EvolAI	80	47	229	214
e3	AntColony	80	45	234	183
e3	AntColonyOrigin	80	36	119	119
m1	EvolAI	100	50	257	237
m1	AntColony	100	33	163	163
m1	AntColonyOrigin	100	43	152	123
m2	EvolAI	100	-	-	-
m2	AntColony	100	77	373	228
m2	AntColonyOrigin	100	90	428	428
h1	EvolAI	120	104	522	522
h1	AntColony	120	66	289	274

h1	AntColonyOrigin	120	105	411	411
h2	EvolAI	120	-	-	-
h2	AntColony	120	-	-	-
h2	AntColonyOrigin	120	73	194	194
h3	EvolAI	120	-	-	-
h3	AntColony	120	91	425	347
h3	AntColonyOrigin	120	97	365	337
vertical2	EvolAI	140	-	-	-
vertical2	AntColony	140	85	406	307
vertical2	AntColonyOrigin	140	100	478	312
vertical3	EvolAI	140	81	410	395
vertical3	AntColony	140	86	412	280
vertical3	AntColonyOrigin	140	71	322	203
hole	EvolAI	160	-	-	-
hole	AntColony	160	87	414	248
hole	AntColonyOrigin	160	67	290	214

2. část

V této části jsem měřil, jestli se Agenti se zvyšujícím se počtem iterací zlepšují a v jakém poměru vyhrávají jednotlivé skupiny agentů - *survivors, offspring, new blood.*

pro účely testování byly parametry nastaveny tak, aby byly všechny skupiny stejně početné (5 jedinců v každé skupině).

Očekávané výsledky:

- se zvyšujícím se počtem iterací se bude snižovat poměr výher *new blood*
- nejvyšší poměr výher budou mít *survivors*, nejnižší *new blood*
- prvně se bude zvyšovat poměr výher *offspring* a od určité chvíle se začne snižovat ve prospěch *survivors*
- se zvyšujícím se počtem iterací se bude zvyšovat dosažená výška

Reálná pozorování:

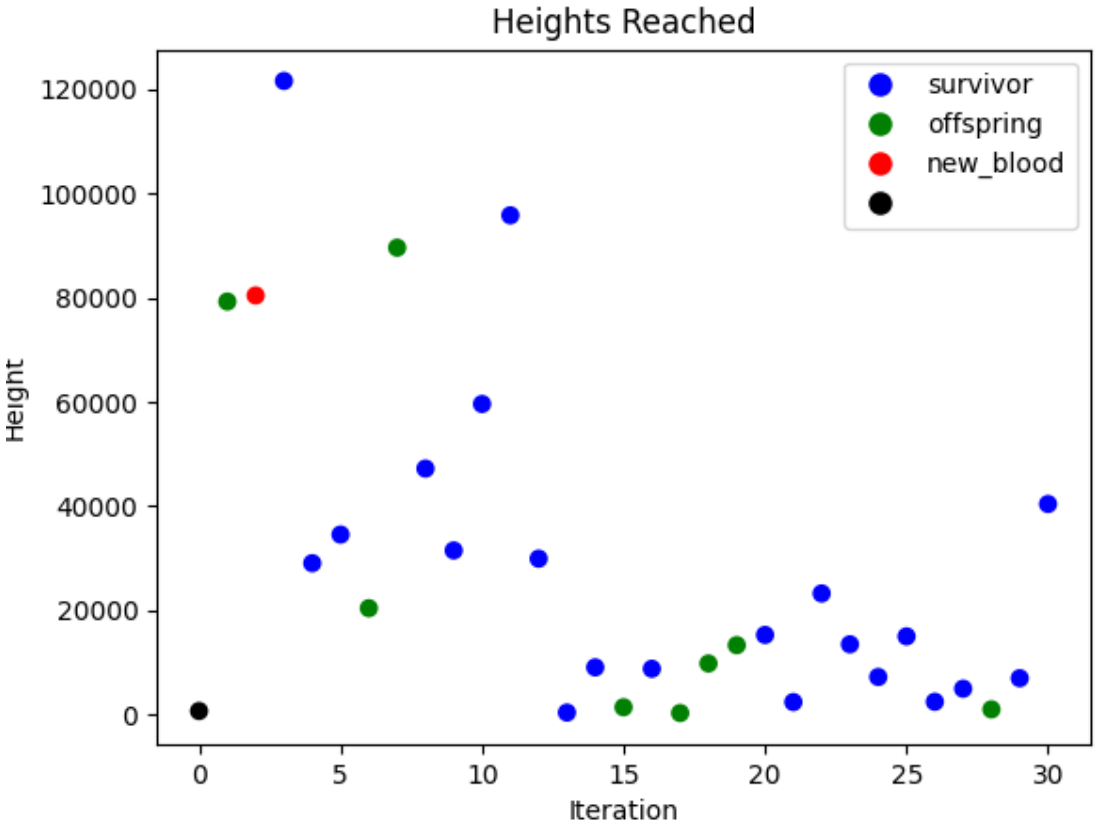
- poměry mezi skupinami se nijak zásadně nemění
- základní poměr mezi skupinami je podle očekávání
- průměrná dosažená výška se s počtem iterací nezvyšuje

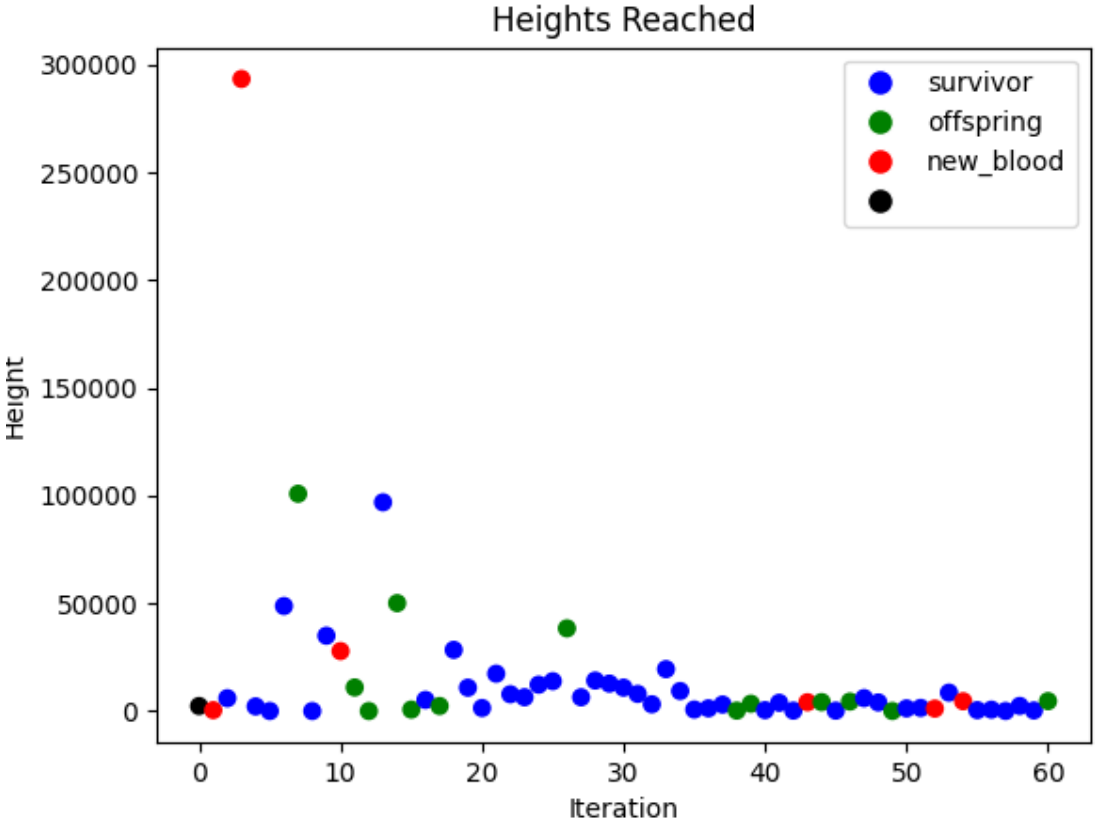
Možné důvody:

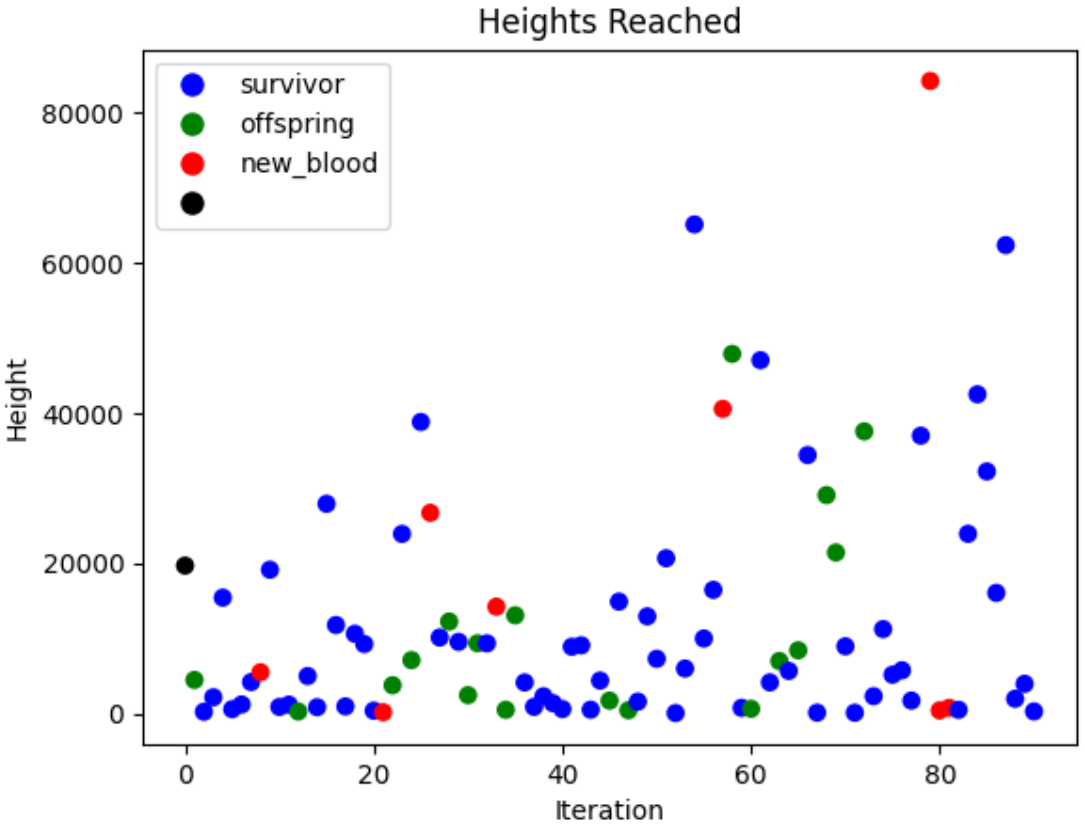
Agenti se mohou nepřímo ovlivňovat posunem obrazovky (jakmile se agent dostane mimo obrazovku, umírá). Chyba v programu (podle grafů to vypadá, že některé iterace jsou velmi krátké).

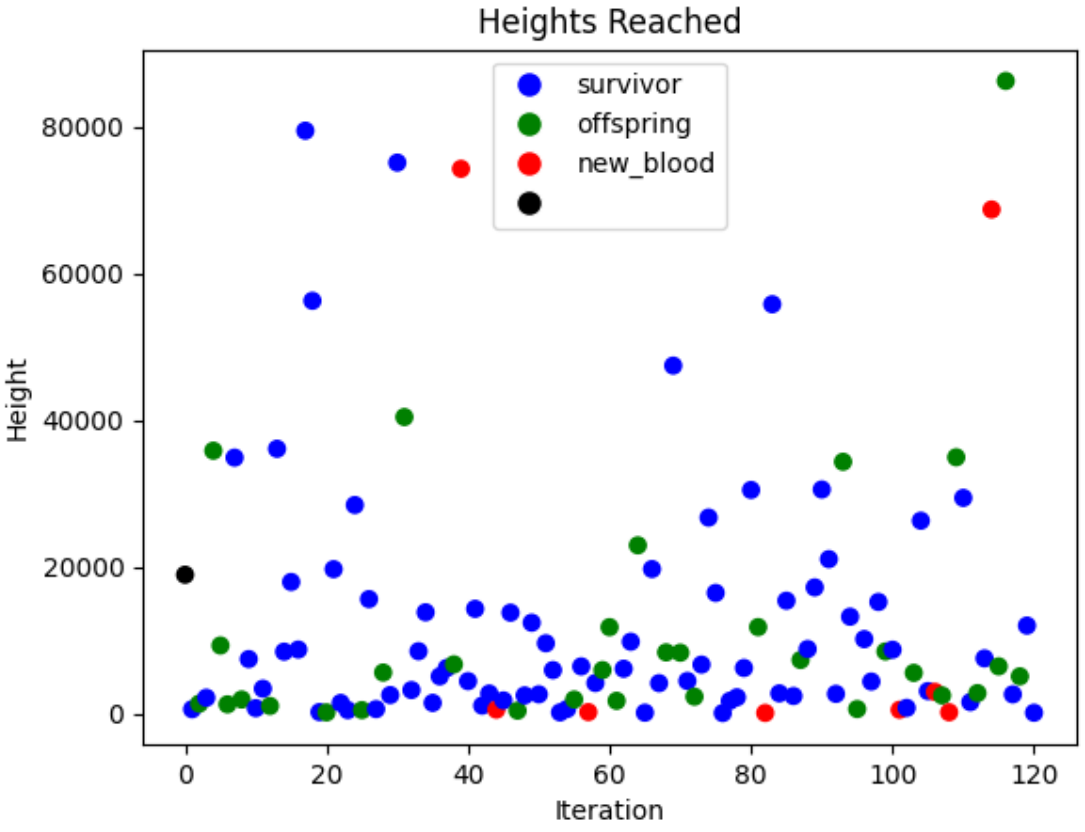
Tabulka a grafy

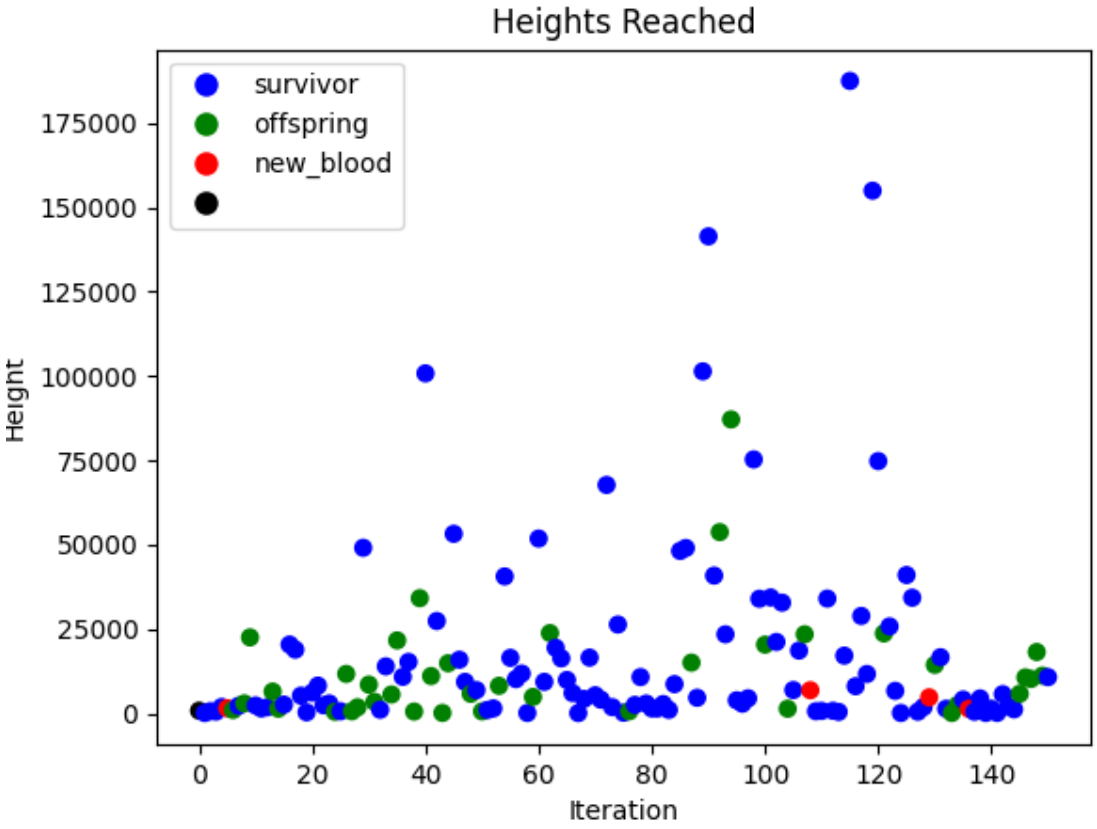
počet iterací	jakou část iterací jednotlivé skupiny vyhrály		
	offspring	survivors	new blood
30	0.27	0.70	0.03
60	0.22	0.68	0.10
90	0.20	0.71	0.09
120	0.27	0.67	0.07
150	0.26	0.71	0.03
180	0.19	0.73	0.08
210	0.27	0.68	0.06

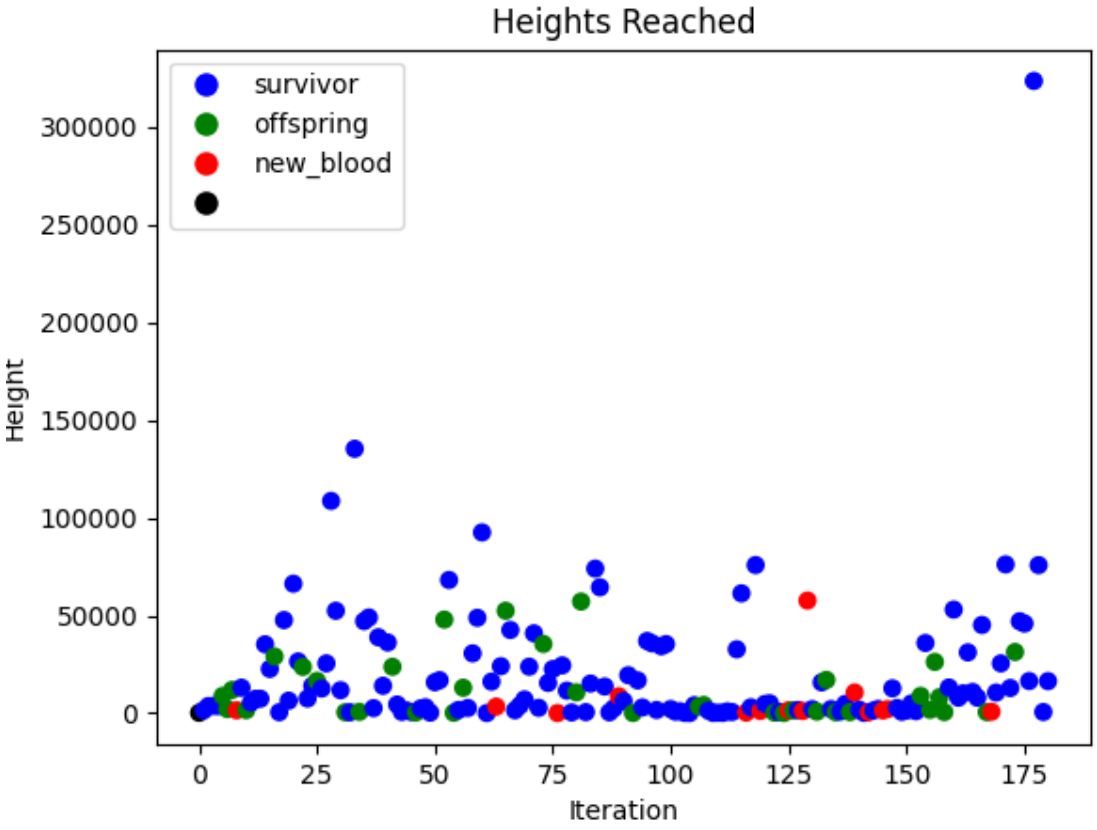

Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
20000
40000
60000
80000
100000
120000
0
5
10
15
20
25
30


Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
50000
100000
150000
200000
250000
300000
0
10
20
30
40
50
60


Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
20000
40000
60000
80000
0
20
40
60
80


Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
20000
40000
60000
80000
0
20
40
60
80
100
120


Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
25000
50000
75000
100000
125000
150000
175000
0
20
40
60
80
100
120
140


Heights Reached
survivor
offspring
new_blood
Height
Iteration
0
50000
100000
150000
200000
250000
300000
0
25
50
75
100
125
150
175