**Umelá inteligencia zadanie č.2b**

**Hľadanie pokladu**

Marek Smutný

Programoval som v jazyku Python. Podľa zadania som použil evolučný algoritmus a na reprezentáciu jedincov virtuálny stroj ako v zadaní.

**Opis kódu:**

Každý jedinec je objektom triedy **Jedinec**. Jeho parameter **hodnoty** predstavuje adresy s inštrukciami (64 adries). Parameter **fitnes** zaznamenáva hodnotu, koľko pokladov jedinec našiel.

Obrázok, na ktorom je text

Automaticky generovaný popis

Trieda **Pole** reprezentuje mriežku s pokladmi, po ktorej každý jedinec chodí. Pri každom novom spustení programu je rozloženie pokladov iné. Štartovacia pozícia je vždy [3, 6] a veľkosť mriežky 7x7.

Obrázok, na ktorom je text

Automaticky generovaný popis

Po spustení programu je od používateľa vyžiadaný počet jedincov v generácii a typ kríženia. Pre fungovanie program je potrebné aby bol počet jedincov také číslo, že jeho odmocnina je celé čislo (napr. 25, 36, 100, 121,…). Typy kríženia sa líšia tým, že v type 1 sa za gén považuje celá adresa jedinca a v type 2 sa za gén považuje 1 alebo 0 v každej adrese jedinca. Dieťa sa pri krížení vytvorí tak, že sa náhodne vyberajú gén z oboch rodičov.



Po získaní potrebných inputov od používateľa sa vytvorí **pole**, vypíše sa a vytvorí sa prvá generácia jedincov. Toto zabezpečuje funkcia **vytvor\_prvu\_gen,** ktorá vytvorí taký počet prázdnych jedincov ako zvolil používateľ a následne im na prvých 16 adries pridá náhodné 8 miestne stringy jednotiek a núl.

Obrázok, na ktorom je text

Automaticky generovaný popis

Následne sa v nekonečnom loope striedajú funkcie na vyhodnotenie jedincov (prejdenie jedincov po ploche), funkcia krizenia (vytvorenie novej generacie) a funkcia mutacie. Vo funkcii **vyhodnot\_jedinca** je maximálny počet inštrukcií 500, v prípade, že funkcia sa dostane na poslednú adresu jedinca a ešte stale sa nachádza na hracej mriežke, nevykonal 500 inštrukcií a nenašiel všetky poklady, funkcia pokračuje od začiatku, od prvej adresy. Pri vyhodnocovaní fitnes sa neberie do úvahy dĺžka cesty.

Obrázok, na ktorom je text

Automaticky generovaný popis

Inštrukcie inkrementácia, dekrementácia, výpis a posun sú reprezentované samostatnými funkciami a sú volané vo funkcii **vyhodnot\_jedinca.**

Obrázok, na ktorom je text

Automaticky generovaný popis

Nakoniec funkcia mutácie (**mutacia**) zmení v náhodných jedincoch v novej generácií na náhodnej adrese jeden bit (jednotku na nulu a nulu na jednotku).

Obrázok, na ktorom je text

Automaticky generovaný popis

**Porovnanie dvoch typov kríženia:**

Moje dva typy kríženia jedincov (vysvetlené vyššie) som porovnával pri identických príkladoch (mriežkach). Už po prvom spustení bolo zrejmé, že prvý typ kríženia je časovo menej náročný, pretože typ 2 potrebuje o jeden loop naviac (pracujeme s génmi na nižšej úrovni). Avšak po 10 testoch som zistil, že typ 1 vytvára v priemere o 1100 generácií viac. Pre definitívne rozhodnutie, ktorý typ kríženia je lepší, by bolo potrebné urobiť omnoho viac testov.

