Umelá inteligencia pre hru Dicewars

Umelá inteligencia a strojové učenie

Fakulta Informačních technologií VUT v Brně
2019/2020
Tomáš Lapšanský (xlapsa00), Anton Firc (xfirca00), Jakub Filo (xfiloj02)

1 Úvod

Hra dicewars je ťahová nedeterministická hra pre 2 a viac hráčov, založená na pravdepodobnosti hodu kockou pri boji jednotlivých hráčov. Každý hráč môže urobiť nula, jeden alebo viac ťahov v jednom kole na základe svojho uváženia alebo predom definovaných kritérií (AI). Hráč následne ukončí svoj ťah a v hre pokračuje ďalší hráč podľa predom definovaného poradia.

Cieľom hry je obsadiť všetky územia na mape a tak poraziť svojich jednotlivých oponentov.

2 Umelá inteligencia

Táto dokumentácia popisuje implementáciu umelej inteligencie (ďalej len AI) do hry dicewars. Popíšeme ideu implementácie AI a riešenia vyhodnotenia jednotlivých ťahov pre čo najlepší priebeh hry.

2.1 Idea

Implementácia AI využíva ideu algoritmu ExpectiMiniMax. Algoritmus ExpectiMiniMax využíva pravdepodobnosti vykonania jednotlivých ťahov hráčov a predpokladá, že každý protivník urobí ťah tak, aby spôsobil čo najväčšiu škodu nám ako ich protivníkovi. Rieši samozrejme aj pravdepodobnosť výhry ťahu jednotlivých protivníkov a počíta s nimi pri ohodnoteniach svojho ťahu. Keďže v hre dicewars potrebujeme jednotlivo predikovať každý krok v našom ťahu a nie ťah samotný, zredukovali sme implementáciu algoritmu na jeden alebo žiadny ťah pre jednotlivých hráčov.

Algoritmus využíva základné preddefinované prevdepodobnosti podľa počtu jednotlivých kociek hráčov. Pre každú kombináciu kociek je definovaná pravdepodobnosť výhry hráča ktorý útočí, čo prakticky znamená že na jeho kockách padne vyšší súčet ako na kockách oponenta. AI každý svoj potencionálny ťah ohodnotí podľa tejto funkcie a odsimuluje do hĺbky ťahy, pri ktorých máme pravdepodobnosť výhry väčšiu ako 50% (s výnimkou útoku 8 kockami na 8 kociek, kde je pravdepodobnosť výhry niečo málo pod 50 %, no ak by sme takéto ťahy nevykonali, neboli by sme schopný ukončiť hru).

Pre začatím simulácie ťahu ohodnotí hraciu plochu podľa ohodnocovacej funkcie (bude popísaná nižšie). Pri zanorení do hĺbky ohodnotí ťahy každého nášho oponenta podobným spôsobom ako ten náš, teda hodnotí hraciu plochu v závislosti na našom hráčovi, a za najlepší ťah vyhodnotí ten, ktorý ma pravdepodobnosť vykonania nad 50% a zároveň najviac zníži ohodnotenie hernej plochy. Pri protivníkovi počítame s tým, že sa jeho útok vždy podarí (pravdepodobnosť 1). Tento ťah odsimulujeme na našej simulovanej hracej ploche.

Po odsimulovaní ťahov všetkých oponentov AI ohodnotí hraciu plochu pre daný ťah. Rozdiel pôvodného ohodnotenia a aktuálneho ohodnotenia plochy je vynásobený pravdepodobnosťou úspechu daného ťahu. Ak táto hodnota vyjde kladná, prakticky to znamená, že simulovaný ťah je výhodný a zväčší ohodnotenie našej hracej plochy. Tento ťah je teda zaradený do poľa možných ťahov.

Po dokončení simulácie všetkých ťahov je pole zoradené, a najvyššia preferencia je vybraná ako najlepší ťah, ktorý bude vykonaný.

AI nevykoná žiadny ďalší ťah v prípade, že pole vhodných ťahov je prázdne.

2.2 Implementácia

AI je implementovaná v jazyku python.

2.2.1 Inicializácia AI

Pri inicializácii AI je vytvorený objekt ktorý ju reprezentuje a sú inicializované a nastavené atribúty ako identifikátor hráča, pole obsahujúce poradie hráčov zoradené tak aby začínalo našim AI a logger.

2.2.2 Ťah AI

Ťah začína volaním funkcie `ai_turn(board)`, čím je predané riadenie našej AI. Prvotne je uložená referencia na hernú plochu predaná parametrom `board`, následne je získaný zoznam možných ťahov pre aktuálny stav hernej plochy pomocou volania funkcie `possible_turns(board, player_name)`. Táto funkcia prejde všetky možné ťahy pre náše AI a každý jeden ťah ohodnotí. Pre ohodnotenie ťahu je použitá funkcia `calculate_possible_turns(source, target)` ktorá najprv podľa počtu kociek na oboch poliach (útočník, obranca) vyhodnotí, či má zmysel nad týmto ťahom vôbec uvažovať. Pokiaľ áno, ohodnotí počiatočný stav hernej plochy a uloží ho. Následne vytvorí kópiu hernej plochy, a vykoná úspešný ťah z poľa source na pole target. Po vykonaní ťahu postupne prechádza všetkých ostatných hráčov v poradí ako hrajú. Pre každého hráča hľadá jeho možný ťah ktorým by najviac znížil ohodnotenie hernej plochy pre naše AI. Túto funkcionalitu implementuje funkcia `do_ai_turn(board, player_name)`. Táto funkcia najprv ohodontí hernú plochu pre naše AI a toto ohodnotenie uloží ako počiatočné, následne skúma všetky možné ťahy ktoré môže daný hráč vykonať a majú pravdepodobnosť úspechu väčšiu ako 50%. Pre vhodné ťahy potom vytvorí ďalšiu kópiu hernej plochy, odsimuluje ťah a znovu ohodnotí hernú plochu. Po získaní ohodnotenia po vykonaní ťahu zistí rozdiel medzi počiatočným a koncovým ohodnotením, pokiaľ je kladné, a teda došlo k zníženiu ohodnotenia pre naše AI uloží tento ťah do množiny možných ťahov ktorá je potom vrátená. Následne je odsimulovaný najlepšie ohodnotený ťah pre daného hráča. Po odsimulovaní najlepších ťahov všetkých hráčov je znovu ohodnotená herná plocha. Následne je vypočítané zlepšenie ohodnotenia ako rozdiel koncového a počiatočného ohodnotenia. Tento rozdiel je následne vynásobený pravdepodobnosťou toho že tento ťah uspeje a táto hodnota je následne vrátená spať do funkcie `possible_turns`. Pokiaľ je vrátená nejaká číselná hodnota je trojica útočník, obranca, ohodnotenie ťahu pridaná do poľa možných ťahov ktoré budú vrátené do funkcie `ai_turn()`. Pokiaľ vrátená množina ťahov nie je prázdna je vykonaný najlepšie ohodnotený ťah, ak je prázdna tak sa AI vzdá riadenia a je celý ťah ukončený.

2.2.3 Ohodnocovacia funkcia

Ohodnocovacia funkcia `calc_board(bot_board, player_name)` vypočítava ohodnotenie hernej plochy pre naše AI. Ohodnotenie vypočítava podľa nasledujúcich faktorov:

- počet vlastnených polí
- počet polí v najväčšom regióne
- počet slabých polí, teda polí ktoré ležia na hranici a príľahlé nepriateľské polia majú o dve alebo viac kociek viac

- počet polí nachádzajúcich sa mimo najväčšieho regiónu
- počet ľahkých cieľov, teda príľahľých nepriateľských polí ktoré majú o dve alebo viac kociek menej

Ohodnocovacia funkcia následne podľa predom definovaných váh faktorov vypočíta ohodonotenie hernej plochy.

Výstup funkcie má nasledovný tvar:

$$value = \log_2(0.3 * owned\_fields + 0.35 * largest\_region - 0.15 * weak\_lands \\ -0.2 * (owned\_fields - largest\_region) + 0.3 * easy\_tgt)$$

kde premenné na pravej strane postupne reprezentujú uvedené faktory.

3 Záver

Táto sekcia popisuje spôsob testovania implementácie, dosiahnuté výsledky našej práce a návrhy na možné vylepšenie riešenia popísaného v tejto práci.

K finálnemu výsledku popísanému v tejto kapitole sme sa dopracovali iteratívnym upravovaním nami navrhnutých riešení. Každé navrhnuté riešenie bolo podrobené testovaniu na základe ktorého sme overili jeho efektivitu. Podľa výsledkov testovania bolo takéto riešenie buď použité pre ďalší vývoj alebo zahodené.

Medzi testované riešenia, ktoré boli prijaté, no následne ešte upravené patrí napríklad reálna simulácia priebehu hry. Takáto simulácia zahŕňala výpočet súboja o pole, a pokračovala podľa získaného výsledku. Tento postup sa ukázal ako nevhodný nakoľko mohol nastať stav, kedy percentuálne silný ťah bol zahodený, a naopak percentuálne slabý ťah bol vyhodnotený ako vhodný.

3.1 Testovanie

Testovanie prebiehalo počas celého procesu vývoja, čím bolo zaručené stále zlepšovanie výsledkov implementovaného riešenia.

Prvotné testy prebiehali v hrách proti jednému oponentovi, s využitím skriptu *dicewars-ai-only*. Po dosiahnutí nami požadovaných výsledkov[1] sme AI podrobili komplexnejším testom. Tieto testy zahŕňali hry proti viacerým oponentom a turnaje s rôznymi nastaveniami hry.

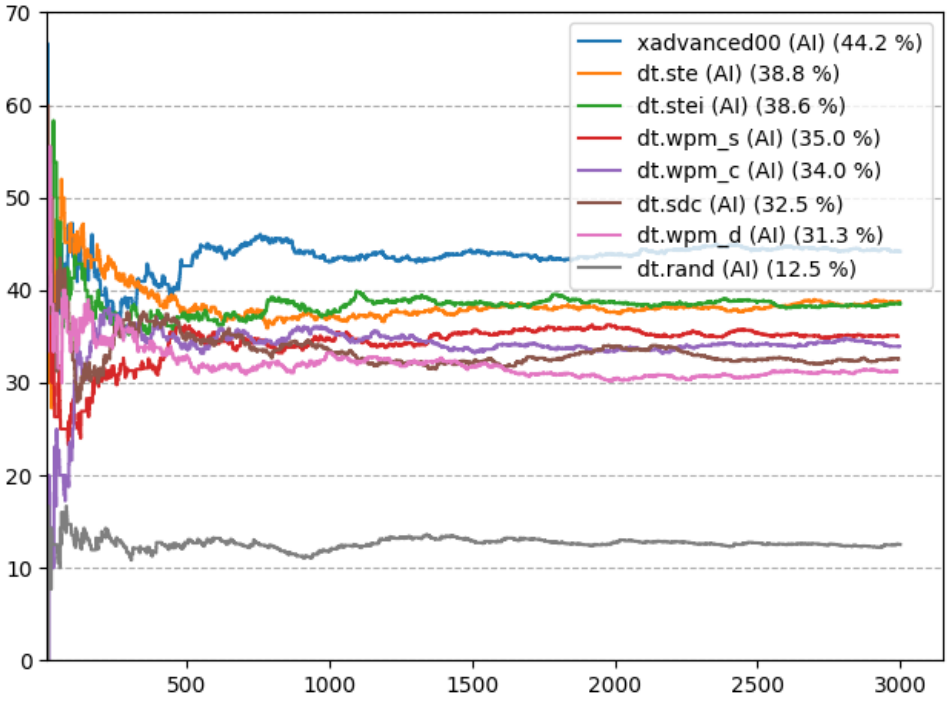
3.2 Vyhodnotenie

Nami implementované riešenie poráža všetky poskytnuté referenčné riešenia s úspešnosťou 75% a viac v súboji 1v1.

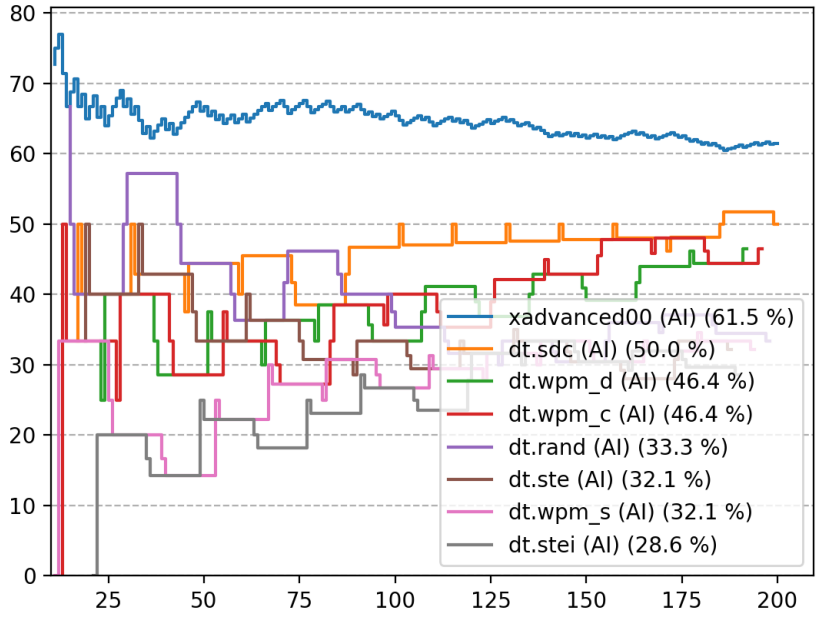
V turnajoch pre dvoch a viac hráčov sa naše riešenie umiestňuje vždy na prvých troch priečkach. Pokiaľ neobsadí prvé miesto, je rozdiel v percentuálnom pomere vyhratých hier oproti víťazovi zanedbateľný(víťaz vyhral o jednu hru viac,pri testovaní na 100 kôl). Predpokladáme, že toto správanie je spôsobené počiatočným rozložením mapy a poradím hráčov ktoré sú jednými z dôležitých faktorov tejto hry.

Úspešnosť nášho AI demonštrujeme nasledujúcimi grafmi (naše AI je reprezentované názvom *xadvanced00*).

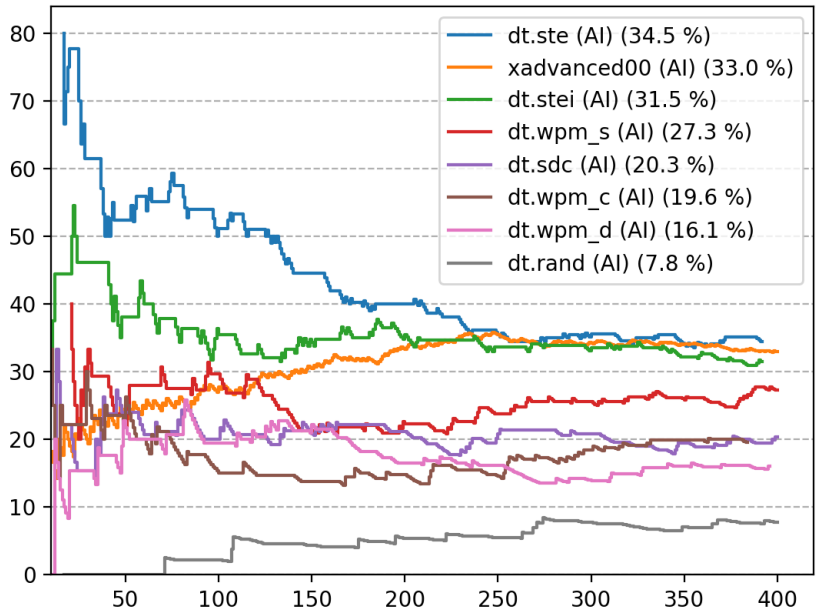
[1] stabilné percentá výhier nad ostatnými oponentami



Obr. 1: turnaj pre 3 hráčov s 1000 hrami



Obr. 2: turnaj pre 2 hráčov s testovaním nášho AI pri 100 hrách



Obr. 3: turnaj pre 4 hráčov s testovaním nášho AI pri 100 hrách

3.3 Možné vylepšenia

V ďalšom pokračovaní tejto práce sa javí ako vhodné optimalizovať proces simulácie ťahov kedy sa kopíruje celá herná plocha, nakoľko je tento proces výpočtovo náročný a pomere často opakovaný pri vyhodnocovaní jednotlivých ťahov.

Takisto sa javí ako viac než vhodné podrobnejšie preskúmať vplyv parametrov ohodnocovacej funkcie. Zároveň sa ukazuje možnosť použitia strojového učenia pre skúmanie a následnú úpravu váh jednotlivých faktorov (parametrov).

Posledne, môže byť v budúcnosti vylepšený algoritmus ExpectiMiniMax tak, aby pracoval do väčšej hĺbky než do jednej úrovne. V takomto prípade však bude potrebná aj úprava počtu krokov vykonaných (simulovaných) jednotlivými hráčmi.