Fakulta informačních technologií
Vysoké učení technické v Brně

SUI - Umělá inteligence a strojové učení

Umělá inteligence pro Válku kostek

2021

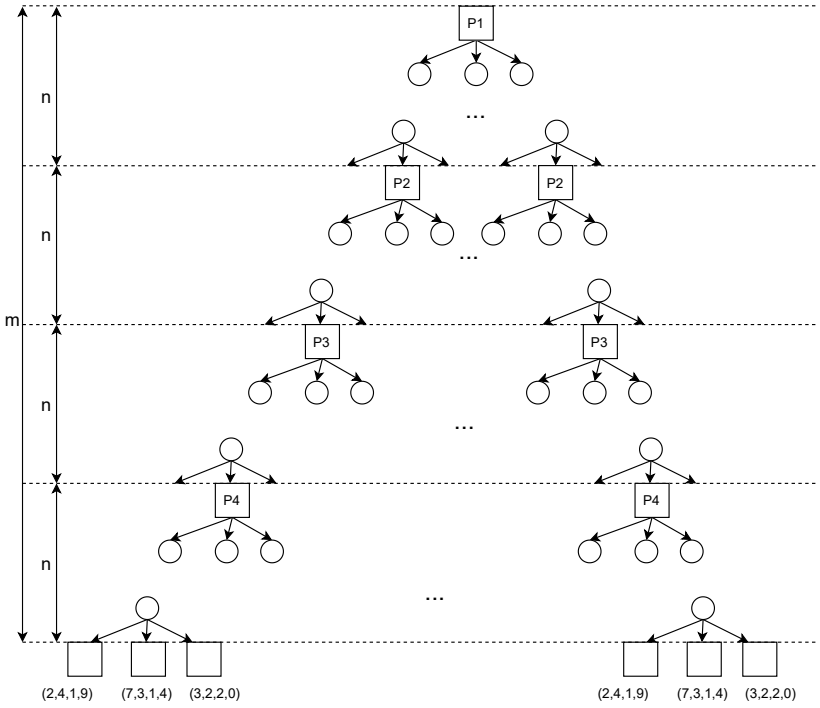
Doležel Marek (xdolez67)

Holub Juraj (xholub40)

Mateáš Branislav (xmatea00)

Rádsetoulal Vlastimil (xradse00)

Obsah



Obr. 1: Prehľadávanie stavového priestoru pomocou *Max-N*.

1 Úvod

2 Prehľadávanie stavového priestoru

Naša implementácia prehľadávania stavového priestoru je založená na algoritme *Max-N*;[1]. Algoritmus je vizualizovaný na obrázku 1. Hranaté uzly označujú prvý ťah každého hráča v aktuálnom kole (hráči P1, P2, P3 a P4). Aktuálny hráč na ťahu určí nasledujúce možné ťahy. Avšak v našej hre počas jedného kola môže hráč podľa pravidiel urobiť viacero ťahov. Preto sa každý hráčov ťah expanduje ďalej až do úrovne n, kde n je konfigurovateľné a obmedzuje rozsah prehľadávaných stavov. Kruhové uzly teda predstavujú jednotlivé ťahy v rámci jedného kola hráča. V úrovni n sa začína kolo pre nasledujúceho hráča.

V úrovni $4n$ (alebo tiež m) odohral každý hráč jedno kolo. V hĺbke m môžeme ohodnotiť listové uzly a ohodnotenie propagovať vyššie. Ak chceme prehľadávať do väčšej hĺbky tak vykonáme ďalším $4n$ expandovaní a prehľadávanie vyhodnocujeme vždy v hĺbke ktorá je násobkom m. Parameter m je teda ďalší konfigurovateľný atribút, ktorým obmedzujeme rozsah prehľadávania.

Uzly na listovej úrovni sú ohodnotené a výsledkom je štvorica (p1, p2, p3, p4), kde p1 až p4 je skóre hráča P1 až P4 v danom stave hry. Uzol o jednu úroveň vyššie zvolí za najlepšieho potomka ten uzol v ktorom má ohodnotenie aktuálne hrajúceho hráča najvyššiu hodnotu (algoritmus *Max-N*).

3 Implementácia prehľadávania

3.1 Expandovanie stavu

Expandovanie uzlu hráča vykoná funkcia `possible_moves(board, player)`, ktorá pre aktuálny stav hracej dosky (`board`) a hráča (`player`) určí nasledujúce možné ťahy. Možné ťahy sa počítajú podobne ako v botovi STEI. Množina všetkých možných ťahov je ohraničená len na také, ktoré spĺňajú aspoň jedno z nasledujúcich kritérií:

[1] `https://docplayer.net/131108557-Expectimax-n-a-modification-of-expectimax-algorithm-to-solve-multiplayer-stochast html`

- Pravdepodobnosť úspešného útoku na oblasť a súčasne udržanie tejto oblasti v nasledujúcom ťahu je väčšia alebo rovná $\frac{1}{2}$.
- Hráč má v útočiacom poli plný počet kociek (8).

3.2 Simulácia ťahu hráča

Každý expandovaný ťah je simulovaný funkciou `apply_move_to_board(board, action)`. Simulácia zoberie aktuálny stav hracej dosky (`board`), navrhovaný ťah (`action`) a vytvorí nový stav hracej dosky. Útočný ťah je v tejto hre stochastický avšak simulácia je deterministická a uvažuje, že útočník zvíťazí vždy ak má väčšie množstvo kociek ako obranca.

3.3 Ohodnotenie listového uzlu

Na ohodnotenie listového uzlu používame metriku, ktorá zohľadňuje najväčší región vlastnený každým hráčom. Región je súvislá časť polí na hracej doske, ktorá patrí jedinému hráčovi. V danom listovom uzle sa nájde najväčší región pre každého hráča a počet oblastí v tomto regióne je ohodnotením skóre pre daného hráča.

3.4 Konfigurácia parametrov prehľadávania

Parameter n (hĺbka prehľadávania ťahov v rámci jedného hráča) je nastavená na 1 pretože väčšia hĺbka znižovala pomer výhier nášho AI.

Parameter m (hĺbka prehľadávania v rámci kôl) je nastavená taktiež na 1. Prehľadávame teda stavový priestor v ktorom každý hráč odohrá jednu sériu svojich ťahov.

4 Stratégia

Naše AI funguje podľa stratégie, ktorá je podobná STEI_ADT. Najsilnejší protihráč je práve bot STEI_ADT. Preto je naša stratégia podobná jeho. Avšak naša AI ju vylepšuje o prehľadávanie stavového priestoru. Prehľadávanie by malo byť efektívne práve preto, že poznáme implementáciu súperovho bota. Naša simulácia súperovho útočného ťahu teda vychádza zo stratégie STEI, ktorú súper používa. Preto predpokladáme, že vieme pomerne dobre predvídať súperove ťahy. Na druhej strane, hra je pomerne silne stochastická čo simuláciu značne znepresnuje.

Voľba ťahu v každom kole, pričom v jednom kole má hráč viac ťahov, postupuje podľa nasledujúcej stratégie:

1. Prvé 4 ťahy AI presunie svoje kocky z vnútra poľa na hranice zo súpermi tak aby malo čo najväčšiu útočnú silu (ofenzíva podobná STEI_ADT).
2. AI vykoná prehľadávanie stavového priestoru pomocou *Max-N* a zvolí útočný ťah s najlepším ohodnotením. Tento krok opakuje dokedy mu *Max-N* poskytuje možné útoky. (STEI_ADT v tejto fáze neprehľadáva stavový priestor ale používa heuristickú funkciu)
3. Na záver vykoná AI defenzívny presun. Zvyšné 2 ťahy na presun vlastných kociek využije tak aby svoje kocky presunula od hraníc (defenzíva podobná STEI_ADT).

5 Umelá inteligencia

Neurónová sieť implementuje heuristickú funkciu, ktorá sa potom využíva na ohodnotenie listových uzlov v rámci procedúry MaxN. Vstupom neurónovej siete je vektor reprezentujúci aktuálny stav hry, výstupom je potom vektor štyroch reálnych čísel z intervalu $\langle 0, 1\rangle$, ktoré udáva pravdepodobnosť výhry daného hráča v aktuálnej situácii.

5.1 Serializácia hry

Pro účely serializace byla upravena serverová část hry, a to konkrétně soubor `dicewars/server/game.py`. Tato drobná úprava umožňuje uložit všechny stavy her(pole objektů `board`), které nastávají v rámci jednoho turnaje do souboru `.pickle`.

Samotná serializace a transformace do HDF[2] souboru

6 Záver

[2]HDF - Hierarchical Data Format