UNIVERZITA KOMENSKÉHO V BRATISLAVE
FAKULTA MATEMATIKY, FYZIKY A INFORMATIKY
KATEDRA APLIKOVANEJ INFORMATIKY

Úvod do Umelej Inteligencie
Cvičenie 1 - reflexný agent

September 21, 2022

1. REFLEXNÝ AGENT

Úlohou je naprogramovať reflexného agenta, ktorý bude hrať piškvorky.
Táto **úloha je za 1 bod.**

Reflexný agent je jednoduchý typ agenta (hráča), ktorý sa rozhoduje **iba na základe aktuálneho stavu sveta**. Nepamätá si predchádzajúci vývoj hry, ani neplánuje nadchádzajúce ťahy.

Program:

V súbore *games.py* je definovaná trieda *Game* a od nej odvodené *TicTacToe* a *Gomoku*, ktoré nemusíte vôbec meniť.
Pracovať budete so súborom *players.py*, v ktorom je definovaná abstraktná trieda *Player* s povinnou funkciou *choose_move*. Je tu aj trieda *MyPlayer*, ktorú máte doprogramovať. Na konci súboru si vyberiete, akú verziu hry chcete hrať.

Vašou úlohou je teda napísať funkciu **MyPlayer.choose_move(game, state)** tak, aby hráč vykonal "rozumný" ťah v hre piškvorky. Funkcia dostáva dva argumenty:

- **game** - inštancia triedy **Game**, v ktorej sú implementované všetky funkcie týkajúce sa samotnej hry (viď nižšie).
- **state** - aktuálny stav hry (t. j. stav hracej plochy, hráč ktorý je na ťahu,...), na ktorý je potrebné odpovedať čo najlepšou možnou akciou. Stav netreba parsovať, stačí ho použiť ako argument do ďalších funkcií (viď nižšie).

Výstupom z funkcie je akcia (t. j. ťah, ktorý sa má vykonať) - teda číslo zo zoznamu možných akcií **game.actions(state)** pre daný stav **state**.

Niektoré funkcie triedy **Game**, ktoré môžu byť nápomocné:

- **game.player_at_turn(state)** - zistí, kto je v stave **state** na ťahu ('X' alebo 'O')
- **game.other_player(player)** - ako argument dostane písmenko hráča, vráti písmenko druhého hráča
- **game.board_in_state(state)** - vráti stav hracej plochy *(list(list(char)))* v stave **state**
- **game.w** - šírka hracej plochy

- **game.h** - výška hracej plochy
- **game.k** - koľko znakov v rade musím mať, aby som vyhral
- **game.actions(state)** - vráti zoznam *(list(int))* možných akcií (platných ťahov) zo stavu **state**
- **game.state_after_move(state, a)** - vráti nový stav, ktorý vznikne vykonaním akcie **a** zo stavu **state**
- **game.is_terminal(state)** - otestuje, či je stav **state** terminálny, t.j. niekto vyhral alebo je remíza
- **game.utility(state, player)** - vráti 'utilitu' stavu **state** pre hráča **player** : 1 ak vyhral, -1 ak prehral, 0 inak. Majte na pamäti, že jednotlivé utility sú známe až na konci hry, t.j. nemá zmysel sa pýtať na utilitu stavu **state**, pokiaľ to nie je terminálny stav.

Jednoduché ukážky týchto funkcií sú zahrnuté aj v kóde.

Na konci kódu sa spúšťajú hry - odkomentujte čo budete potrebovať. Pri debuggingu odporúčame hrať aj počítač vs. človek. Pri hodnotení budeme používať posledné dva riadky (play_n_games), váš agent by mal presvedčivo vyhrávať.
Hodnotenie bude pozostávať zo štyroch spustení play_n_games - 2x proti náhodnému hráčovi (piškvorky a gomoku) a 2x proti nám navrhnutému (dosť slabému) agentovi.
Vyhrať proti náhodnému hráčovi v Gomoku naozaj nie je zložité, no pri 3x3 piškvorkách sa očakáva minimálne skóre 0.4 (po n hrách).

Čas programu nebudeme merať ale v prípade, že Váš agent bude extrémne pomalý, úlohu môžeme penalizovať. Očakáva sa, že doba, za akú váš agent urobí ťah pri piškvorkách bude zanedbateľná (a stihne sa cca 100 hier za sekundu), na druhej strane pri Gomoku doba trvania jednej hry sa pohybuje v sekundách.

Po odovzdaní všetkých úloh sa uskutoční turnaj funkčných agentov v TicTacToe a Gomoku, v ktorom si zahrá každá dvojica hráčov proti sebe a vytvorí sa tabuľka úspešnosti. Následne sa udelia bonusové body **(0.6b)** za prvé miesto, **(0.4b)** za druhé a **(0.4b)** za miesto v jednotlivých hrách.