WSTĘP DO SZTUCZNEJ INTELIGENCJI

Ćwiczenie 3 – Minimax z obcinaniem α − β

MICHAŁ MIZIA 331407

SPIS TREŚCI

Wstęp

Treść zadania:

1. Zaimplementować algorytm minimax z obcinaniem $\alpha - \beta$ grający w kółko i krzyżyk, używając przygotowanego kodu

2. Przeprowadzić następujące symulacje gier próbując znaleźć jak najlepsze parametry algorytmu minimax:

   1. Gracz minimax vs gracz losowy
   2. Gracz minimax vs gracz minimax

3. Rozegrać samemu kilka gier przeciwko:

   1. Graczowi losowemu
   2. Graczowi minimax ze znalezionymi parameterami

4. Zbadać eksperymentalnie wpływ głębokości odcinania drzewa gry.

Symulacja

GRACZ MINIMAX KONTRA GRACZ LOSOWY

Gracz minimax z pruning_depth = 5 jest w stanie wygrać prawie zawsze chyba że w sytuacjach gdy zawodnik zaczynający ‘przypadkowo’ wybierze najlepsze ruchy.

```
player_x = build_player(config["minimax"], game)
player_o = build_player(config["random"], game)
if player_o is None or player_x is None:
    exit()

wins = {
    "x_starting": {"o": 0, "x": 0, "t": 0},
    "o_starting": {"o": 0, "x": 0, "t": 0},
}

for i in range(100):
    simulate_game(game, player_x, player_o, i % 2 == 0)
    # print(game.get_winner())
    wins["x_starting" if i % 2 == 0 else "o_starting"][game.get_winner()] += 1
    game.play_again()

print(wins)
```

```
{'x_starting': {'o': 0, 'x': 50, 't': 0}, 'o_starting': {'o': 0, 'x': 44, 't': 6}}
```

GRACZ MINIMAX KONTRA GRACZ MINIMAX

Ponieważ kółko i krzyżyk jest grą rozwiązaną, przy pruning_depth = 5, wszystkie gry kończą się remisem ponieważ obydwaj gracze minimax grają w sposób optymalny

Co ciekawe, przy zestawieniu gracza z pruning_depth = 5 oraz drugiego z wartością 3, gracz z wartością 5 jest w stanie wygrać wszystkie swoje gry, a gry w których zaczyna przeciwnik dalej remisuje:

```
player_x = build_player(config["minimax"], game)
player_o = build_player(config["minimax_weak"], game)
if player_o is None or player_x is None:
    exit()

wins = {
    "x_starting": {"o": 0, "x": 0, "t": 0},
    "o_starting": {"o": 0, "x": 0, "t": 0},
}

for i in range(100):
    simulate_game(game, player_x, player_o, i % 2 == 0)
    # print(game.get_winner())
    wins["x_starting" if i % 2 == 0 else "o_starting"][game.get_winner()] += 1
    game.play_again()

print(wins)
```

```
{'x_starting': {'o': 0, 'x': 50, 't': 0}, 'o_starting': {'o': 0, 'x': 0, 't': 50}}
```

WNIOSKI

Gracz z głębokością 5 jest dużo lepszy od gracza random oraz znacząco lepszy od gracza z głębokością 3. 5 wydaje się być wystarczającym poziomem głębokości do optymalnej gry.