Politechnika Wrocławska

Projektowanie Algorytmów i Metody Sztucznej Inteligencji

Projekt nr 3

Zadanie na ocenę bdb (5.0)

Autor:	Aleksander Łyskawa 275462
Wydział i kierunek studiów:	W12N, Automatyka i Robotyka
Termin zajęć:	pon 15:15-16:55
Prowadzący:	dr inż. Witold Paluszyński
Data:	3.06.2024

Spis treści

1 Wprowadzenie

Celem tego projektu jest opracowanie sztucznej inteligencji zdolnej do gry w warcaby. AI wykorzystuje algorytm Minimax, wzbogacony o technikę przycinania alpha-beta, do oceny możliwych ruchów. Dokument ten zawiera szczegółowe wyjaśnienia dotyczące użytych algorytmów i heurystyk.

2 Struktura programu

Opracowany program powinien obsługiwać następujące pozycyjne parametry wywołania:

```
prog interface turn depth random_seed ip-address ip-port
```

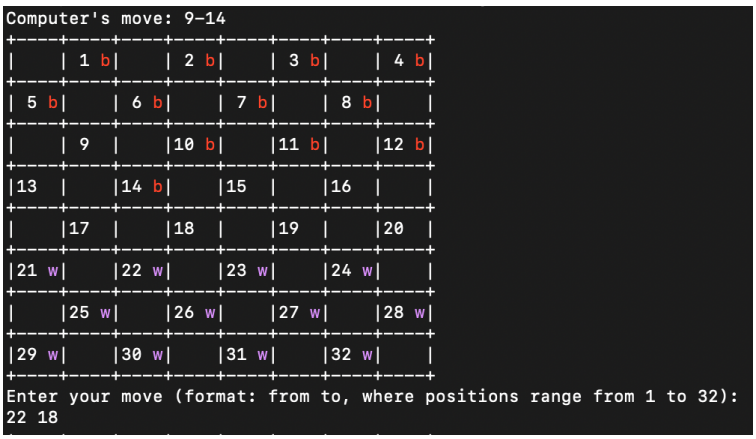
Program obsługuje parametry:

- **interface**: GUI lub NET (gra interakcyjna lub sieciowa).
- **turn**: WHITE lub BLACK (kolor gracza).
- **depth**: Głębokość analizy algorytmu MinMax.
- **random_seed**: Opcjonalne ziarno dla generatora liczb losowych.
- **ip-address, ip-port**: Opcjonalne nr portu i adresu IP, dla gry sieciowej.

Domyślne wartości: localhost i 12345 dla adresu IP i portu.

3 Interfejs gry

Zaimplementowano, wyposażony w numeracje pól,interfejs gry w terminalu, który obsługuje ruchy gracza (*from* - skąd) i zwraca ruch komputera.



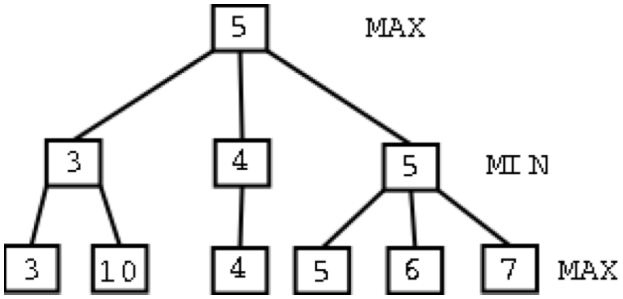
Rysunek 1: Interfejs gry

4 Algorytm Minimax

Algorytm Minimax jest rekurencyjną metodą stosowaną w podejmowaniu decyzji i teorii gier, mającą na celu minimalizowanie możliwej straty w najgorszym scenariuszu. W kontekście gry zakłada, że przeciwnik gra optymalnie, aby zmaksymalizować własny wynik.

4.1 Zastosowanie algorytmu

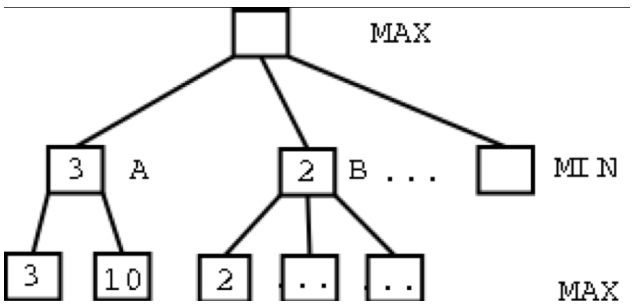
Zaimplementowany przeze mnie algorytm minimax wykorzystuje przeszukiwanie drzewa wszystkich możliwych stanów gry, opartych na ruchach legalnych dla każdego z graczy. Algorytm ten symuluje wykonanie ruchów na przemian przez graczy do określonej głębokości, minimalizując maksymalne straty komputera i maksymalizując jego szanse na zwycięstwo. Ocena stanu gry, czyli kombinacji ruchów graczy, jest dokonywana przy użyciu ustalonej heurystyki, która nie jest uczona przez algorytm, lecz jest zewnętrzną funkcją oceniającą stan planszy.



Rysunek 2: Algorytm MinMax

5 Odcięcia Alpha-Beta

Do optymalizacji algorytmu zastosowalem technikę alfa-beta cięć, które są usprawnieniem algorytmu minimax. Polegają one na pomijaniu niektórych gałęzi drzewa możliwych stanów gry, jeśli można przewidzieć, że te gałęzie nie mają wpływu na wynik algorytmu. To znaczy jeśli wartości węzłów w tych gałęziach są gorsze niż aktualnie najlepsza znaleziona wartość w górnej części drzewa, to te gałęzie mogą być pominięte. Dzięki temu można skrócić czas przeszukiwania drzewa bez zmiany wyniku działania algorytmu minimax.



Rysunek 3: Odcięcia Alpha-Beta

6 Heurystyka

Funkcja oceny używana w algorytmie Minimax przypisuje każdemu stanowi planszy wartość numeryczną, reprezentującą pożądanie tego stanu dla gracza. Funkcja określa ewaluacje pozycji na planszy.

6.1 Funkcja Oceny

Funkcja oceny bierze pod uwagę liczbę pionów i damek oraz ich pozycje na szachownicy, aby określić wartość stanu planszy, gdie :

- Wartość damki: 15 punktów
- Wartość piona: 5 punktów + numer wiersza, w którym się znajduje

Dodatkowo funkcja uwzględnia bonusowe punkty wg. zasad:

- Dodatkowy 1 punkty za każdy pionek w centrum
- Dodaktowy 1 punkt za bezpieczeństwo pionka znjadującego się na krawędziach planszy

Wartości są sumowane do zmiennych myRating oraz opponentRating, a następnie odejmowane od siebie zwracając ewauluacje na planszy.

7 Testy

7.1 Wykonane testy

Program został poddany testom za pomocą interfejsu gry, gdzie zwiększanie głębokości przeszukiwania sprawiło, że komputer stał się coraz trudniejszy do pokonania, co uważam za udane. Dodatkowo, przetestowałem działanie programu z wykorzystaniem brokera, gdzie komputer nie tylko unikał nielegalnych ruchów, ale także wykonywał je w zadowalającym czasie, przeprowadzając pełną rozgrywkę w zaledwie kilka sekund na głebokośći równej 8.

7.2 Sprzęt

Wszystkie testy działania programu zostały przeprowadzone na laptopie MacBook Air M1. Laptop ten jest wyposażony w procesor Apple M1 z 8 rdzeniami CPU i 8 GB pamięci RAM. Testy zostały wykonane na systemie macOS przy użyciu domyślnego środowiska uruchomieniowego. 14

8 Źródła

Implementacja zasad gry:

- ChatGPT
- Github Copilot
- https://en.wikipedia.org/wiki/English_draughts
- https://en.wikipedia.org/wiki/Portable_Draughts_NotationMovetext

Implemnetacja heurystyki:

- ChatGPT
- Github Copilot
- https://www.mini.pw.edu.pl/ mandziuk/PRACE/es_init.pdf
- https://kcir.pwr.edu.pl/ witold/ai/English-Draughts.pdf
- https://kcir.pwr.edu.pl/ witold/ai/Checkers_assignment.html

Powyższe źródła nie zostały skopiowane bezpośrednio do programu. Zamiast tego, zostały wykorzystane jedynie jako wsparcie merytoryczne do samodzielnego rozwiązania zadania.