Michał Więcek
248956
29.05.2020

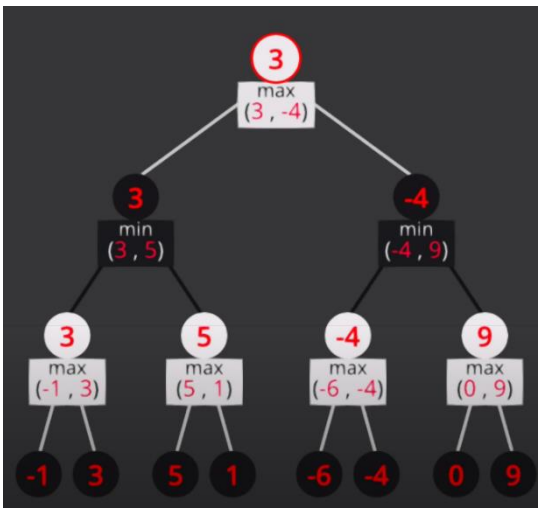
Projektowanie Algorytmów i Metody Sztucznej Inteligencji
Projekt 3

Mgr inż. Marcin Ochman
Grupa: Wtorek 15^{15}-16^{55}

1. Wstęp

Celem projektu było stworzenie algorytmu Minimax oraz zaimplementowanie go do gry dwuosobowej. W tym projekcie wybraną grą są **warcaby**.

2. Algorytm Minimax



Algorytm minimax rekurencyjnie tworzy drzewo możliwych ruchów naprzemiennie dla każdego z graczy po czym ocenia sytuację na planszy dla każdego pola w drzewie. Następnie zakłada, że kiedy ruch wykonuje algorytm to chce zmaksymalizować zyski z danego ruchu. Natomiast kiedy ruch wykonuje przeciwnik algorytm zakłada, że ten będzie próbował zminimalizować zyski dla algorytmu.
Obok przedstawione jest przykładowe drzewo gdzie białe pola to ruch algorytmu a pola czarne to ruch przeciwnika.

3. Implementacja algorytmu

Algorytm zaimplementowany w programie działa tak samo jak wyżej wymieniony. Jednak dla ułatwienia implementacji rekurencja nie rozpoczyna się w momencie kiedy gracz biały ma wykonać ruch. Algorytm tworzy wektor możliwych ruchów i symuluje te ruchy na planszach. Następnie dla każdej z tych planszy rekurencyjnie bada który liść w drzewie ma szansę na otrzymanie najwyżej oceny. Kiedy uda się określić który ruch może być najbardziej owocny, wykonuje go na oryginalnej planszy.

4. Głębokość rekurencji algorytmu

W programie głębokość rekurencji jest ustawiona na 6. Jednak biorąc pod uwagę fakt, że przed samym algorytmem sprawdzamy możliwe ruchy, należy określić głębokość rekurencji na 7 czyli 7 ruchów w przód.
Dla takiej głębokości rekurencji algorytm nie jest najlepszym graczem ale wykonuje się w czasie niezauważalnym dla użytkownika.
Poniżej przedstawione są czasy oczekiwania dla następujących głębokości rekurencji.

Głebokość rekurencji	Czas [s]
6	0,36
7	1,52
8	5,41
9	22,57
10	106,32

Czas wykonywania algorytmu około 1,5 sekundy lub 5,5 sekund można uznać za akceptowalny. Jednak uznałem, że nie będzie to komfortowe dla gracza czekać na ruch wirtualnego przeciwnika. Dlatego rekurencja w wersji tutaj prezentowanej została ustawiona na 7.

Michał Więcek
248956 29.05.2020

5. Prezentacja możliwości rozgrywki

Poniżej wypisane zostaną funkcje jakie zaimplementowałem do samej gry. Nie dołączałem zdjęć ponieważ nie prezentują się zbyt funkcjonalnie i estetycznie. Wszystkie zasady poruszania i bicia pionkami są również odpowiednio dostosowane dla damek.

a. Wybór pionka.

i. Jeśli bicie jest możliwe, gracz zostaje zmuszony do wyboru pionka z biciem.
ii. Jeśli bicie jest niemożliwe, gracz zostaje zmuszony wybrać pionek z możliwym ruchem.

b. Ruch wybranym pionkiem.

i. Jeśli bicie jest możliwe, gracz zostaje zmuszony do wykonania jednego z możliwych bić.
ii. Jeśli gracz zbije pionek i możliwe jest kolejne bicie tym samym pionkiem gracz zostaje zmuszony wykonać kolejne z możliwych bić.
iii. Jeśli bicie jest niemożliwe, gracz może przemieścić wybrany pionek na wybrane przez siebie pole.

c. Nieprawidłowy wybór pionka, niepoprawne dane wejściowe.

i. Jeśli gracz wybierze nieodpowiedniego pionka lub poda niepoprawne dane wejściowe program go o tym poinformuje. Doda również przyczynę nieprawidłowości.

d. Zamiana pionów na damki po przejściu na koniec planszy.

6. Bugi, niedociągnięcia

W sytuacji kiedy gracz będzie chciał wykonać kilka bić z rzędu. Musi wpisać odpowiednie koordynaty. Jeśli te wartości będą poza planszą program zgłosi wyjątek.
W przypadku kiedy po pierwszym zbiciu pojawią się dwie możliwości zbicia, algorytm nie stworzy dodatkowych dzieci do drzewa. Zamiast tego wybierze pierwsze bicie które jest w stronę lewego górnego rogu. Jeśli takiego bicia nie będzie sprawdzi kolejne bicia według ruchu wskazówek zegara.

7. Wnioski

- Implementacja algorytmu jest bardzo prosta. Można ulepszyć go np. implementując cięcia alfa-beta. Pozwoliło by to na głębsze przeszukiwanie drzewa w krótszym czasie.
- Zwiększenie efektywności można też uzyskać poprzez generowanie i zapis drzewa możliwych ruchów. Wtedy wystarczyłoby raz wygenerować drzewo o wysokości 7. Następnie do odpowiednich liści dodawać drzewa o wysokości 2.
- Implementacja interfejsu i sposób zarządzania planszą rozgrywki nie są zbyt odpowiedni. Uważam, że powinienem podzielić na warstwy w jaki sposób dane przechodzą od gracza do planszy gry i kontrolować je na odpowiednich etapach.
- Algorytm mimo osiągnięcia dość głębokiej rekurencji nie jest najbardziej wymagającym graczem. Z reguły przegrywa z początkującym graczem ale stara się nie „podkładać” graczowi.
- Najbardziej wymagającą częścią tworzenia projektu było „nauczenie” algorytmu zasad oraz egzekwowanie ich w przypadku gry człowieka. Największą trudność sprawiały damki które mogą wykonać bardzo wiele ruchów.