

Pamsi kolko i krzyzyk Denis Firat

Denis Firat 249031

May 2020

1 Wstep

Moim zadaniem było stworzenie prostego programu, gry komputerowej, kółko i krzyżyk. Wyzwaniem i celem projektu było zaimplementowanie algorytmu minimax, który imitowałby sztuczna inteligencja, bota, z którym gracz mógłby grać. Program umożliwia zmianę wielkości planszy, na której gra się z botem. Plansza wyświetlana jest w terminalu z pomocą znaków ASCII.

2 Opis algorytmu

Funkcja MiniMax(char **plansza, char PChar, int rozmiar, int depth, int a, int b) ocenia, jakość ruchu jaki wykonać ma sztuczna inteligencja. Funkcja działa rekurencyjnie, wybiera "najlepszy" ruch dla danej planszy, następnie jeżeli nie nastąpiła wygrana, to "symuluje" najlepszy ruch przeciwnika, a potem własny i tak dalej, aż do głębokości 6. Dodatkowo zostały dodane ciecica alfa, beta, których zadaniem jest zatrzymywać rekurencje funkcji, gdy okaże się, że ruch nie zmienia przewagi na planszy.

3 Opis gry

Na początek wita nas menu z trzema możliwościami:

- Start
- Zmien rozmiar planszy
- Wyjście

```
1. Start
2. Zmien rozmiar planszy
3. Wyjście
```

Po wybraniu Zmien rozmiar planszy, użytkownik ma okazję zmienić rozmiar planszy. Dostępne rozmiary są od 3x3, ale zaleca się nie przekraczać więcej niż 10x10. Pole wyboru rozmiaru prezentuje się tak:

Podaj rozmiar planszy:

Po rozpoczęciu gry wybierając Start, naszym oczom ukazuje się plansza o wybranym rozmiarze.

```
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
Tura gracza: _
```

Zadaniem gracza jest wpisać współrzędne pola, do którego gracz chciałby wpisać swój znak. Pierwsza cyfra oznacza wiersz, a druga kolumnę. Ważne jest, aby pamiętać o tym, że współrzędne pól numerowane są od 0. Gracz gra kółkiem i zaczyna rozgrywkę. Po wybraniu pola, naszym oczom ukazuje się uaktualniona plansza wraz z wyborem komputera, którego reprezentuje znak X. Proces wybierania pól trwa, aż ktoś wygra albo uzupełni się cała plansza.

```
+---+---+---+---+
| x | - | - | - |
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
| - | o | - | - |
+---+---+---+---+
| - | - | - | - |
+---+---+---+---+
Tura gracza: _
```

Program, po każdym ruchu sprawdza czy nastąpiła wygrana jednej z stron, gdy ma to miejsce ukazuje się napis o wygranej i możliwość rozpoczęcia gry jeszcze raz.

```

+---+---+---+
| o | o | x |
+---+---+---+
| o | x | - |
+---+---+---+
| x | - | - |
+---+---+---+
X wygrywa
1. Start
2. Zmien rozmiar planszy
3. Wyjscie

```

4 Wnioski

Projekt ten, był chyba jednym z ciekawszych, nad jakim pracowałem w ramach tego kursu. Najważniejszy, wniosek dla mnie z tego projektu to zrozumienie, że świat sztucznych inteligencji, programów decyzyjnych, nie jest aż tak niedostępny dla początkującego programisty. Zawsze przekonany, że coś takiego jak bot do gry jest już highendowym osiągnięciem zszokowałem się, że minimax wystarczający do obsługi kółka i krzyżyk jest parolinijkowym algorytmem. Dzięki temu zainteresowałem się praktycznym wdrażaniem AI do swoich projektów, a nie tylko oglądaniem jak efektów końcowych programistów na youtube. Jeżeli chodzi o wydajność działania algorytmu to w przypadku 3x3, 4x4 czas działania algorytmu jest prawie niezauważalny. Przy 5x5 czas na ruch przeciwnika trwał 14,88 sekundy, dla większych plansz czas jest już na tyle długi, że granie mija się z celem.

References

- [1] https://pl.wikipedia.org/wiki/Algorytm_min-max
- [2] <https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-1-introduction/>
- [3] <https://stackoverflow.com/>