Krystian Matusiak 249460
Czwartek 9:00-11:00
Dr inż. Łukasz Jeleń

Projektowanie Algorytmów i Metody Sztucznej Inteligencji

Projekt 3 – Gry & AI

1. Opis projektu

Celem projektu było stworzenie gry z wykorzystaniem algorytmu Min-Max. Z podanych gier wybrano grę „Kółko i krzyżyk”. Po uruchomieniu programu gracz ma możliwość zdefiniowania rozmiaru planszy (kwadratowej) oraz ilości znaków potrzebnych do wygrania.

2. Budowa programu i przebieg gry

Program, ze względu na niskie wymagania związane z mechaniką gry, podzielony został na dwa pliki :

- Tab.h
  Plik zawierający klasę Tablica, która jest odpowiedzialna za przechowywanie informacji dotyczących planszy, tj. rozmiar, ilość znaków potrzebnych do wygrania oraz funkcje odpowiedzialne m.in za sprawdzanie czy wygrał krzyżyk/kółko, czy plansza jest pusta itp.
- ttt.cpp
  Plik zawierający główną część kodu. W nim zdefiniowana jest funkcja realizująca algorytm Min-Max, komendy przyczyniające się do stworzenia interfejsu graficznego (wykorzystując bibliotekę SFML) oraz funkcja main, w której realizowana jest gra.

Włączając plik wynikowy tictactoe.out terminal pyta użytkownika o rozmiar planszy, ilość znaków potrzebnych do wygranej oraz kto ma zacząć grę. Następnie ukazuje się interfejs graficzny, w którym gracz podejmuje decyzje myszką. Po zakończonej grze wynik pokazuje się w terminalu. Dodatkowo plansza ukazuje się w terminalu w celu sprawdzania poprawności działania interfejsu oraz aby zobaczyc wynik końcowy.

3. Algorytm Min-Max oraz Alfa-Beta

Algorytm min-max oblicza drzewo wszystkich możliwych stanów w grze oraz każdemu stanowi końcowemu oblicza wartość stanu gry. Dla gry kółko i krzyżyk przy ostatnich stanach gry sprawdzane jest który gracz wygrał i zwracana jest odpowiednia wartość heurystyczna. Następnie przeglądane jest dane drzewo od ostatnich stanów i symulowane są optymalne wybory dla obu graczy. Gracz min wybera ruch który prowadzi do mniejszej wartości końcowej

a gracz max – przeciwnie. Po przeprowadzeniu tych symulacji gracz, który znajduje się w korzeniu drzewa ma pewność, że jego ruch jest optymalny w kontekście informacji o stanie gry z przeprowadzonej symulacji algorytmem minimax.

Ze względu na długi czas pracy algorytmu min-max, obligatoryjne stało się wykorzystanie cięć alfa-beta, które redukują liczbę węzłów, które muszą być rozwiązywane w drzewach przeszukujących przez algorytm min-max. Warunkiem stopu jest znalezienie przynajmniej jednego rozwiązania czyniącego obecnie badaną opcję ruchu gorszą od poprzednich opcji. Wybranie takiej opcji ruchu nie przyniosłoby korzyści graczowi ruszającemu się, dlatego też nie ma potrzeby przeszukiwać dalej gałęzi drzewa tej opcji. Ta technika pozwala zaoszczędzić czas poszukiwania bez zmiany wyniku działania algorytmu.

4. Wnioski

Na początku gra była oparta tylko na algorytmu min-max bez wykorzystania cięć alfa-beta. Niestety na planszy 3x3 tylko drugi i każdy następny ruch był niemal natychmiastowy. Dla planszy 4x4 pierwszy ruch komputera trwał zbyt długo aby można było poprowadzić grę, dlatego też rozszerzono ten algorytm o alfa-beta, co przyczyniło się bardzo krótkiego czasu pracy komputera dla planszy 3x3.

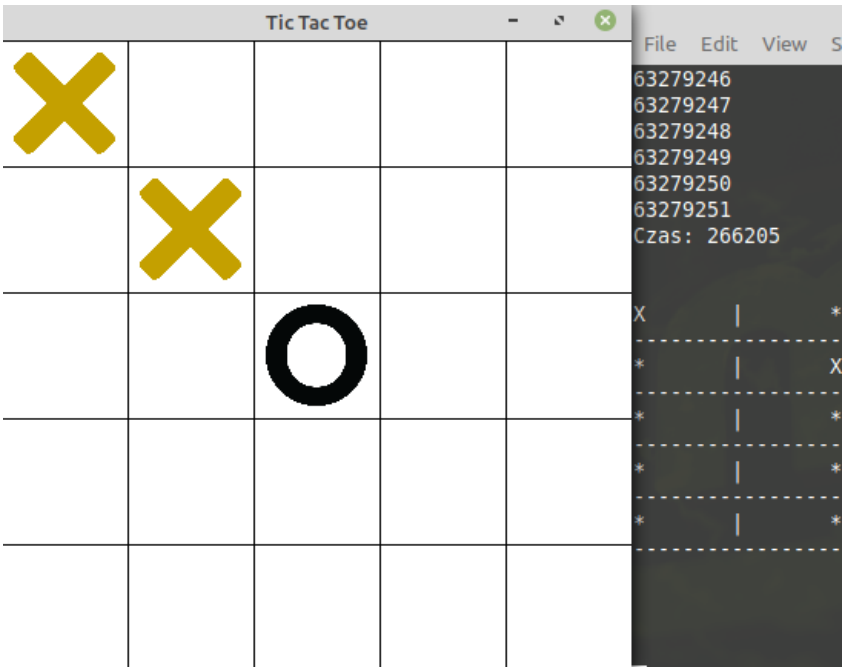
Niestety, dla planszy 4x4 oraz większych, ruch w dalszym ciągu trwał bardzo długo. Wprowadzono więc zmienną wartość głębokości rekurencji według podanego wzoru:

$$glebokosc = \frac{100}{\sqrt{10 * n}}$$

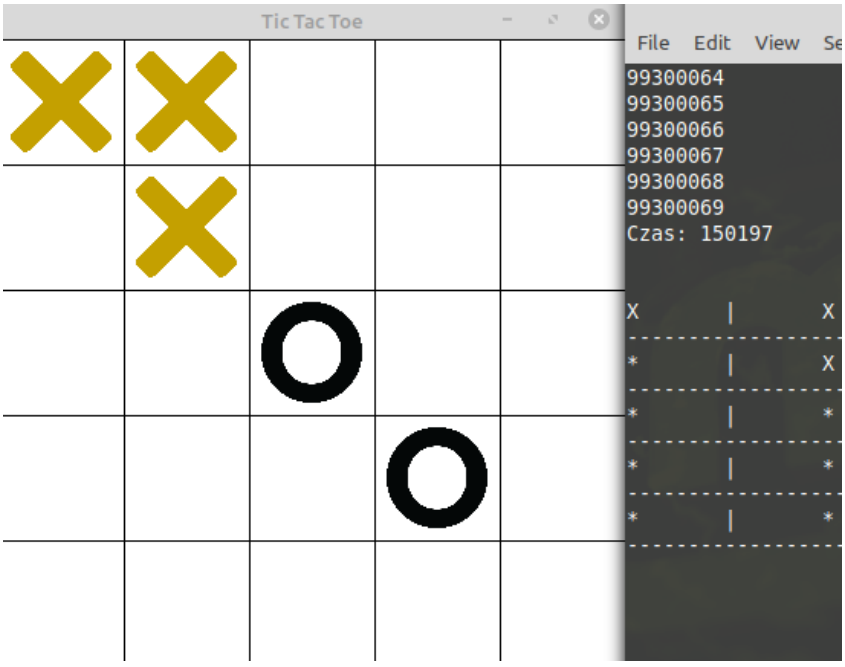
gdzie n-liczba wolnych miejsc na planszy (bez X oraz O).

Dodatkowym usprawnieniem okazało się także dodanie kilka ruchów predefiniowanych. Przykładem tego jest automatyczne zajęcie środkowego pola (o ile nie jest zajęte) przy pierwszym ruchu komputera, aby nie stracić czasu na rozważanie najlepszego ruchu, gdyż pierwszy ruch zajmuje najwięcej czasu ze względu na ilość możliwych ruchów do rozważenia.

Niestety, mimo wszystkich usprawnień, algorytm dalej potrzebuje dużo czasu na podjęcie decyzji, zwłaszcza dla większych rozmiarów plansz (6x6 i większych).



Rys.1 Czas działania algorytmu przy planszy 5x5 przy drugim ruchu AI



Rys.2 Czas działania algorytmu przy planszy 5x5 przy trzecim ruchu AI

Ponadto parametr głębokości rekurencji wiąże się z osłabieniem poziomu sztucznej inteligencji. Im mniejsza głębokość tym większa szansa na wygraną ze strony gracza. Komputer potrafi nie dostrzec wygranej pozycji przeciwnika, gdyż scenariusz opisujący tą pozycję znajduje się poniżej maksymalnej głębokości, której nie można zmienić ze względu na wydłużenie się czasu działania ruchu.

5. Bibliografia

Alfa-Beta 14.05.2020
https://www.youtube.com/watch?v=xBXHtz4Gbdo

Mini-Max 14.05.2020
https://www.youtube.com/watch?v=l-hh51ncgDI
https://pl.wikipedia.org/wiki/Algorytm_min-max

Pobieranie i instalacja SFML 24.05.2020
https://www.youtube.com/watch?v=9ZINo9QrJJg

SFML 31.05.2020
https://www.sfml-dev.org/tutorials/2.5/window-window.php