Sprawozdanie 3

„Gry & AI”

5 czerwca 2019

Temat projektu: Gry & AI

Autor : Paweł Gajda

Termin zajęć: Środa 7.30-9.00

Prowadzący: Dr inż. Łukasz Jeleń

1 Wprowadzenie

1.1 Opis projektu

Projekt opierał się na stworzeniu aplikacji gry wykorzystującej algorytm sztucznej inteligencji. Użytym algorytmem jest minima. Dodatkowo gra kółko i krzyżyk została wyposażona w graficzny interfejs użytkownika.

1.2 Opis algorytmu

Algorytm minimax jest drzewowym algorytmem rekurencyjnym polegającym na minimalizowaniu strat i maksymalizowaniu zysków. W postaci przyjętej dla „kółko-krzyżyk” rozważany jest aktualny stan planszy i symulacyjnie przewidywane są ruchy gracza oraz bota. Na podstawie założenia o warunku wygranej wyciąga się odpowiednie wnioski czy dany ruch jest opłacalny czy nie.

W projekcie wykorzystano usprawnienie tego algorytmu tzw. Prunning. Polega od na pomijaniu niektórych gałęzi drzewa, jeśli da się przewidzieć, że gałęzie te nie mają wpływu na wynik algorytmu (wartości węzłów stanowią gorszą alternatywę do obecnie najlepszej wyliczonej wartości w górnej części drzewa).

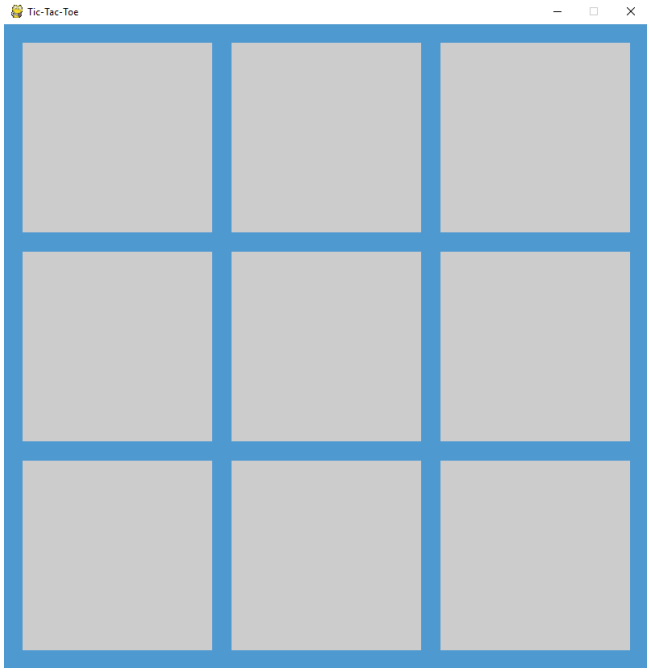
2 Opis gry

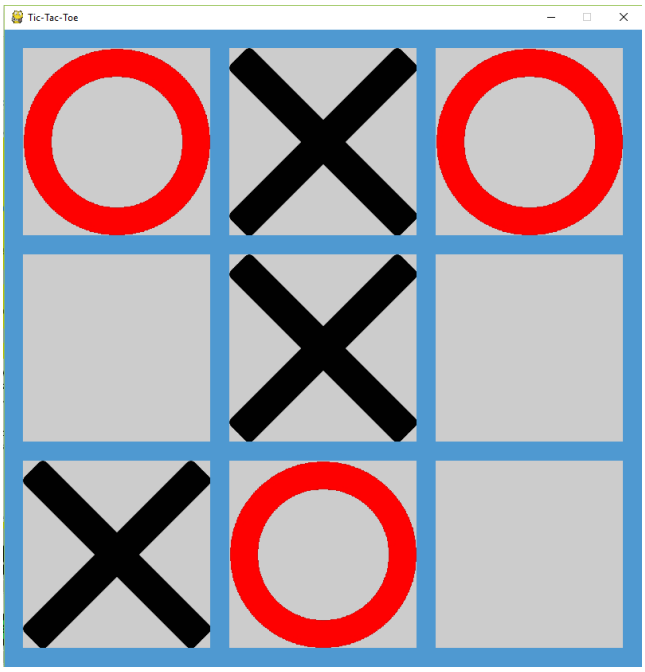
Gra została zaimplementowana przy użyciu języka Python wspomaganego biblioteką pygame, która umożliwia pracę z graficznym interfejsem użytkownika. Gra składa się z dwóch ekranów głównych (ekran startowy oraz ekran planszy) i jednego wspomagającego (ekran końcowy).

2.1 Ekran startowy

Na tym ekranie użytkownik jest w stanie wybrać wielkość planszy oraz warunek zwycięstwa czyli ilość tych samych figur w linii, które dają zwycięstwo. Warunek zwycięstwa nie może być większy niż rozmiar planszy. Dla przyśpieszenia działania algorytmu warunek zwycięstwa wzrasta wraz z rozmiarem planszy, ponieważ im większa różnica pomiędzy rozmiarem planszy a warunkiem zwycięstwa tym większy czas oczekiwania na poszczególnych ruch bota. Wielkość planszy oraz warunek zwycięstwa zostały ograniczone od 3 do 7. Po wybraniu odpowiednich parametrów rozgrywki użytkownik może uruchomić grę wciskając przycisk START.

2.2 Ekran planszy





Po wciśnięciu przycisku START pojawia się ekran z pustą planszą, która na przemian zapełniać będą gracz wraz z komputerem. Gracz ustawia figury ‘X’ natomiast komputer wstawia ‘O’. Domyślnie grę rozpoczyna Gracz.

2.3 Ekran końcowy

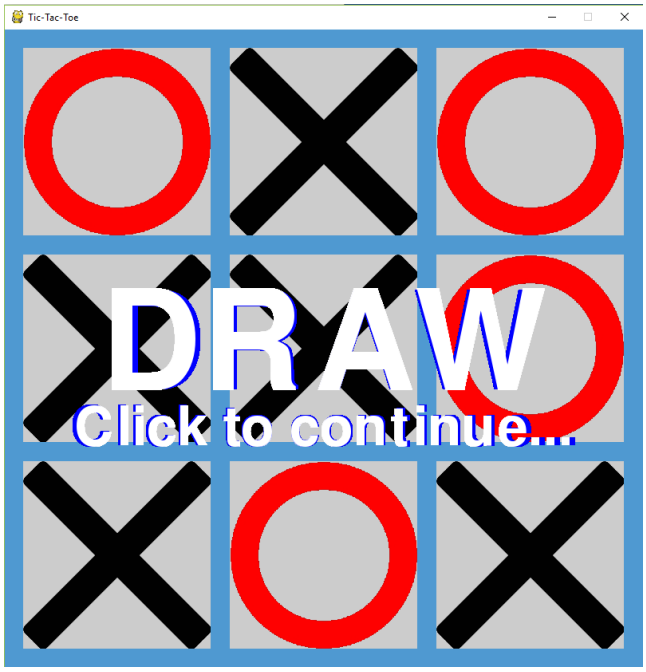
Istnieją 3 możliwe zakończenia rozgrywki, o których gra poinformuje nas komunikatem:

-wygrana, gracz umieścił odpowiednią ilość ‘X’ w linii

-przegrana, komputer umieścił odpowiednią ilość ‘O’ w linii

-remis, w przypadku gdy żadnej ze stron nie udało się wygrać a plansza jest już pełna

Po kliknięciu na ekran gra przekieruję nas na ekran startowy abyśmy mogli zacząć grę jeszcze raz.



3 Podsumowanie i wnioski

- Język Python, oraz jego biblioteka pygame, okazał się być znacznym usprawnieniem w tworzeniu gry.

- Dużym usprawnieniem działania programu okazało się dodanie tzw. Pruninng`u co sprawiło, znaczne przyśpieszenie pracy algorytmu.

- Ważnym elementem było również dobranie odpowiedniej głębokości rekurencji algorytmu w poszukiwaniu optymalnych ruchów.

- W przypadku wybrania płytkiej głębokości rekurencji program ogranicza się jedynie do wypełniania pól planszy oraz blokowania wygranej gracza. Algorytm minimax w programie działa odpowiednio gdy podczas wyszukiwania najlepszego ruchu dojdzie do zakończenia rozgrywki.

4 Bibliografia

- Algorytm minimax http://lukasz.jelen.staff.iiar.pwr.edu.pl/styled-2/page-2/index.php [22.05.2019]
- Podstawy tworzenia gier w Python przy pomocy biblioteki pygame https://python101.readthedocs.io/pl/latest/pygame/tictactoe_str/ [25.05.2019]
- Algorytm minimax https://www.geeksforgeeks.org/minimax-algorithm-in-game-theory-set-1-introduction/ [25.05.2019]
- Prunning https://pl.wikipedia.org/wiki/Algorytm_alfa-beta [29.05.2019]