Projektowanie i Analiza Algorytmów

Sprawozdanie

Projekt 3 – Algorytm MinMax z alfa-beta cięciami w „kółku i krzyżyk”

Jakub Piekarek

Indeks 264202

Prowadzący dr inż. Krzysztof Halawa

Kod grupy K00-37d

Poniedziałek 1115 – 1300

Obraz zawierający tekst, Czcionka, logo, biały

Opis wygenerowany automatycznie

1. Wstęp

Celem tego projektu jest zaimplementowanie gry w kółko i krzyżyk z wykorzystaniem algorytmu MinMax z alfa-beta cięciami. Gra w kółko i krzyżyk, jest to prosta gra logiczna, w której dwóch graczy kolejno stawia znaki "X" i "O" na planszy o rozmiarze NxN. Gracz, który jako pierwszy ułoży M znaków w jednym rzędzie (poziomo, pionowo lub na skos), wygrywa grę. W naszym projekcie gracz będzie miał możliwość definiowania rozmiaru planszy kwadratowej oraz ilości znaków w rzędzie potrzebnych do wygranej. Dzięki temu gra będzie bardziej elastyczna i można ją dostosować do różnych wariantów.

1. Opis zastosowanych algorytmów

**Algorytm Minimax** to metoda minimalizowania maksymalnych strat lub maksymalizacji minimalnego zysku w teorii gier o sumie zerowej. Jest używana do podejmowania decyzji w grach, zarówno tych, gdzie gracze wykonują ruchy naprzemiennie, jak i tych, gdzie wykonują ruchy jednocześnie.

**Algorytm Alfa-Beta** jest techniką przeszukiwania drzewa, która redukuje liczbę węzłów do analizy, przyspieszając tym samym obliczenia w algorytmie Minimax. Jest stosowany w grach dwuosobowych, takich jak kółko i krzyżyk, szachy czy go. Wykorzystuje on warunek stopu, który pozwala przerwać analizę gałęzi drzewa, jeśli znaleziono opcję ruchu gorszą od poprzednio zbadanych. To oszczędza czas, nie zmieniając wyniku algorytmu.

|  |  |  |
| --- | --- | --- |
| Algorytm | Złożoność obliczeniowa | Złożoność pamięciowa |
| MinMax |  |  |
| AlfaBeta |  |  |

Tabela 1 Złożoności algorytmów, gdzie b to średnia liczba możliwych ruchów w stanie gry, a d to głębokość drzewa gry

1. Zastosowanie techniki SI

W grach o sumie zerowej, powszechnie stosuje się **algorytm minmax**, aby umożliwić komputerowi podejmowanie optymalnych decyzji. Opiera się on na analizie drzewa gry, które przedstawia wszystkie możliwe stany planszy i ruchy graczy. Na początku gry korzeń drzewa reprezentuje bieżący stan planszy.

Algorytm przypisuje wartości do liści drzewa, które odpowiadają stanom planszy kończącym grę. W zależności od tego, czy taki stan jest wygraną, remisem czy porażką dla komputera, przypisywana jest odpowiednia wartość.

W węzłach wewnętrznych drzewa, wartość jest przypisywana na podstawie najlepszej ścieżki do zwycięstwa lub przegranej. Jeśli dany ruch prowadzi do przewagi gracza, wartość jest dodatnia, a jeśli prowadzi do przewagi przeciwnika, jest ujemna.

Następnie podejmuje decyzje, wybierając ruch, który maksymalizuje wartość dla gracza, minimalizując jednocześnie wartość dla przeciwnika. Przechodząc przez drzewo gry, algorytm naprzemiennie maksymalizuje i minimalizuje wartość dla kolejnych ruchów.

Aby zoptymalizować działanie algorytmu Minmax, wykorzystuje się przycinanie alfa-beta. Ta technika pozwala na pominięcie analizy niektórych gałęzi drzewa, które nie mają wpływu na ostateczną decyzję, przyspieszając tym samym obliczenia.

1. Interfejs kółka i krzyżyk

Obraz zawierający tekst, zrzut ekranu

Opis wygenerowany automatycznie

Rysunek 1 Interfejs konsolowy dla gry kółko i krzyżyk

1. Wnioski

Podczas implementacji gry w kółko i krzyżyk z algorytmem MinMax z alfa- beta cięciami, zaobserwowano kilka istotnych wniosków. Oto one:

* + Algorytm MinMax dla kółka i krzyżyk działa na zasadzie przewidywania ruchów przeciwnika i wybierania najlepszego możliwego ruchu dla danego stanu gry.
  + Wykorzystanie alfa-beta cięć w algorytmie MinMax pozwala na znaczące ograniczenie przestrzeni przeszukiwań, co prowadzi do efektywniejszego działania. Dzięki temu można uniknąć analizowania niepotrzebnych gałęzi drzewa gry.
  + Przy definiowaniu rozmiaru pola i ilości znaków w rzędzie, istotne jest odpowiednie dostosowanie algorytmu MinMax. Im większa plansza i/lub więcej znaków potrzebnych do wygranej, tym trudniejsze staje się przeszukiwanie wszystkich możliwych ruchów, co może wpływać na wydajność gry.
  + W przypadku większych plansz i bardziej złożonych układów, algorytm MinMax z alfa-beta cięciami może wymagać dużo czasu i zasobów obliczeniowych.