Projektowanie algorytmów i metody sztucznej inteligencji

Przemysław Erbert 258964

wt 15:15

1 Link do repozytorium

https://github.com/Przemoerb/Pamsi3

2 Wprowadzenie

Celem projektu było stworzenie programu z graficznym interfejsem umożliwiajacego gre w kółko i krzyżyk i zawierajacy prosty algorytm sztucznej inteligencji.

3 Opis poszczególnych funkcji

- czysc_xo(): Funkcja ta ma za zadanie wyczyścić plansze gry. Przechodzi przez wszystkie elementy tablicy xo i ustawia ich wartość na spacje (' '). Dzieki temu wszystkie pola planszy sa przygotowane do rozpoczecia nowej gry.
- "plansza()": Funkcja ta jest odpowiedzialna za wyświetlanie planszy gry na ekranie. Najpierw wypisuje numeracje kolumn, a nastepnie przechodzi przez wszystkie wiersze i kolumny tablicy xo, wyświetlajac zawartość poszczególnych pól w odpowiednim formacie. Plansza jest przedstawiana za pomoca znaków '—', '-', i '+', które tworza siatke planszy.
- "ruch_cz()": Funkcja ta pozwala użytkownikowi na wykonanie ruchu w grze. Najpierw proszone jest o podanie numeru wiersza i kolumny, a nastepnie sprawdzane jest, czy dane pole na planszy jest puste. Jeśli tak, na tym polu umieszczany jest znak 'o', oznaczajacy ruch użytkownika. Jeśli pole jest już zajete, użytkownik jest proszony o podanie innych współrzednych.
- "wygrana(char gracz)": Funkcja ta sprawdza, czy dany gracz (o lub x) osiagnał warunek wygranej w grze. Przechodzi przez wszystkie wiersze, kolumny oraz przekatne planszy i sprawdza, czy w którymś z tych przypadków wystepuje sekwencja znaków danego gracza o długości _w. Jeśli

tak, to zwracana jest wartość true, co oznacza wygrana. W przeciwnym przypadku zwracana jest wartość false.

- "wolne()": Funkcja ta sprawdza, czy na planszy sa jeszcze wolne pola, czyli czy gra może być kontynuowana. Przechodzi przez wszystkie elementy tablicy xo i sprawdza, czy któryś z nich ma wartość ' '. Jeśli tak, to oznacza, że jest jeszcze wolne pole na planszy, i funkcja zwraca wartość true. Jeśli wszystkie pola sa już zajete, zwracana jest wartość false, co oznacza remis.

- "minimax(char gracz, int poziom, int glebokosc)": Funkcja ta implementuje algorytm minimax z cieciem alfa-beta. Jest wykorzystywana do sztucznej inteligencji komputera w grze. Wykorzystuje rekurencje, aby analizować wszystkie możliwe ruchy i oceniać plansze dla obu graczy. W zależności od poziomu rekursji i głebokości analizy, funkcja podejmuje decyzje o ruchu komputera. Zwraca wartość oceny planszy dla danego stanu.

4 Szczegółowy opis minimax

Algorytm minimax z cieciem alfa-beta, zaimplementowany w funkcji minimax, jest wykorzystywany do sztucznej inteligencji w grze. Jest to zaawansowany algorytm przeszukiwania drzewa gry, który pozwala na wybór optymalnego ruchu dla komputera.

Funkcja minimax przyjmuje trzy argumenty: gracz, który określa aktualnego gracza (o lub x), poziom, który reprezentuje głebokość rekursji, oraz glebokosc, która oznacza aktualna głebokość analizy planszy. Na podstawie tych argumentów funkcja podejmuje decyzje o kolejnych ruchach komputera.

Działa to na zasadzie przeszukiwania drzewa gry, gdzie wierzchołkami sa kolejne stany planszy, a krawedziami sa możliwe ruchy graczy. Przeszukiwanie odbywa sie rekurencyjnie, zaczynajac od korzenia drzewa (poczatkowego stanu planszy) i analizujac kolejne możliwe ruchy graczy.

W przypadku algorytmu minimax, założeniem jest to, że zarówno gracz o, jak i gracz x, daża do maksymalizacji swojej korzyści. Dlatego w każdym wierzchołku drzewa gry, w zależności od aktualnego gracza, funkcja wybiera ruch, który prowadzi do maksymalnej wartości (dla gracza x) lub minimalnej wartości (dla gracza o).

Wykorzystuje on ocene stanu planszy do podejmowania decyzji. W momencie osiagniecia liścia drzewa (czyli końcowego stanu gry, w którym wystepuje wygrana, remis lub przegrana), funkcja przypisuje ocene danego stanu planszy. Dla wygranej gracza x przypisywana jest wartość 1, dla wygranej gracza o wartość -1, a dla remisu wartość 0.

Aby zoptymalizować działanie algorytmu, zostało wprowadzone ciecie alfa-beta. Polega to na przerywaniu analizy niektórych gałezi drzewa, gdy zostaje osiagnieta pewna wartość graniczna. Dzieki temu algorytm może pominać nieinteresujace ruchy i skupić sie na tych, które moga prowadzić do lepszych wyników.

Wartość graniczna jest reprezentowana przez zmienne vmax i v, które przechowuja wartości ocen dla gracza x.

Przeszukuje on drzewo gry w sposób rekurencyjny, analizujac wszystkie możliwe ruchy obu graczy, aż do osiagniecia końcowego stanu gry. Nastepnie na podstawie ocen stanów planszy podejmuje decyzje o optymalnym ruchu dla komputera.

Ostatecznie funkcja minimax zwraca wartość vmax, która reprezentuje ocene najlepszego ruchu dla komputera, uwzgledniajac wszystkie możliwe odpowiedzi przeciwnika.