Adversarial Search

Problema aleasa pentru utilizarea algoritmului de Adversarial Search este problema X si O. Aceasta problema este reprezentata printr-o matrice de 3x3 elemente. Scopul jocului este aranjarea a 3 elemente de acelasi tip pe o linie, o coloanal sau o diagonala. Exista 3 rezultate pentru joc:

1. Castig – linie / coloanal / diagonala cu elementele jucatorului
2. Remiza – nicio linie / coloanal / diagonala cu elementele jucatorului sau inamicului
3. Infrangere – linie / coloanal / diagonala cu elementele inamicului

Pentru aceasta problema s-a creat o clasa care reprezinta jocul si care contine:

1. Dimensiunea tablei de joc (n = 3)
2. Tabla de joc care initial va avea doar valori de 0 (nimeni nu a facut vre-o miscare)
3. Miscari posibile – exista un numar limitat de miscari si acestea se cunosc. O miscare pentru a putea fii facuta trebuie sa fie in lista de miscari posibile
4. O functie de miscare car va face miscarea data ca parametru (daca se poate) si va calcula daca jocul s-a terminat

Pentru aceasta problema s-a folosit algoritmul Alpha Beta Search si s-a testat impotriva unui bot care face miscari random.

Pasii algoritmului sunt:

1. Se ia actiunea care sa maximizeze functia utilitara (1 daca a castigat, 0 daca este remiza si -1 daca a pierdut)
2. Algoritmul pleaca de la alpha = - infinit si beta = infinit
3. Daca jocul s-a terminat atunci algoritmul se opreste si returneaza valoarea functiei utilitare
4. Altfel, pentru fiecare actiune din cele posibile
   1. Se face o copie din table de joc
   2. Se face miscarea pe copia tablei
   3. Daca se calculeaza valoarea maxima atunci se cauta cea mai mica valoare (pentru ca se vrea minimizarea valorii pentru inamic)
   4. Daca se calculeaza valoarea minima atunci se cauta cea mai mare valoare (pentru ca se vrea maximizarea valorii pentru jucator)
   5. Daca valoarea gasita este cea mai buna de pana acum atunci seteaza ca fiind cea mai buna valoare si cea mai buna actiune gasita pana acum
   6. Daca se cauta valoarea maxima si aceasta valoare este mai mare sau egala cu beta atunci am gasit o valoare suficient de buna si se returneaza cea mai buna valoare si cea mai buna actiune
   7. Daca se cauta valoarea minima si aceasta valoarea este mai mica sau egala cu alpha atunci am gasit o valoare suficient de buna si se returneaza cea mai buna valoare si cea mai buna actiune
   8. Se seteaza alpha ca fiind maximul dintre alpha si cea mai buna valoare (in cazul cautarii valorii maxime) resprectiv beta ca fiind minimul dintre beta si cea mai buna valoarea (in cazul cautarii valorii minime)
   9. Se repeta algoritmul pentru toate actiunile care se pot lua in joc