GUIÓN DE PRÁCTICAS 9

BÚSQUEDA ENTRE ADVERSARIOS: FUNCIONES HEURÍSTICAS

Continuando con el juego TicTacToe:

1. Implementa la poda alfa-beta para que funcione tanto si se desarrolla el árbol completo de búsqueda como si se establece un límite de profundidad y se aplican funciones heurísticas en los nodos finales.

```
tNodo: función poda_ab(E/S tNodo: nodo, E entero: jugador)
var
  entero:max_actual, jugada, mejorJugada, prof, v
  tNodo: intento
inicio
      alfa ← -infinito    beta← +infinito       prof ← 1
      jugada ←1
      mientras  jugada <=N  hacer
        si esValida(nodo, jugada) entonces
                    intento ← aplicaJugada(nodo,jugador,jugada)
                    v ← valorMin_ab(intento, opuesto(jugador)
                                                prof+1, alfa, beta)
                    si v > alfa  entonces
                          alfa ← v
                          mejorJugada ← jugada
                    fin_si
                fin_si
           jugada←jugada+1
      fin_mientras
si esValida(nodo, mejorJugada) entonces
          nodo=aplicaJugada(nodo,jugador,mejorJugada); fin_si
     devolver nodo
fin_función
```

GUIÓN DE PRÁCTICAS 9

```
entero: función valorMin_ab(E tNodo: nodo, E entero: jugador, E entero: prof,
                            E entero: alfa, E entero: beta)
var
  entero: vmin, jugada   tNodo: intento
inicio
    si terminal(nodo) entonces
        vmin← utilidad(nodo)
    si_no si prof=LIMITE entonces
        vmin← heuristica(nodo)
    si_no
        jugada ←1
        mientras  jugada <=N  Y alfa<beta hacer
                si esValida(nodo, jugada) entonces
                 intento←aplicaJugada(nodo, jugador, jugada))
                 beta ← minimo(beta, valorMax_ab(intento, opuesto(jugador),
                                                 prof+1, alfa, beta)
                fin_si
                jugada← jugada+1
        fin_mientras
        vmin←beta
   fin_si
   devuelve  vmin
fin_función
```

```
entero: función valorMax_ab(E tNodo: nodo, E entero: jugador, E entero: prof,
                            E entero: alfa, E entero: beta)
var
  entero: vmax, jugada   tNodo: intento
inicio
    si terminal(nodo) entonces
        vmax← utilidad(nodo)
    si_no si prof=LIMITE entonces
        vmax← heuristica(nodo)
    si_no
        jugada ←1
        mientras  jugada <=N  Y alfa<beta hacer
                si esValida(nodo, jugada) entonces
                 intento←aplicaJugada(nodo, jugador, jugada))
                 alfa ← maximo(alfa, valorMin_ab(intento, opuesto(jugador),
                                                 prof+1, alfa, beta)
                fin_si
                jugada← jugada+1
        fin_mientras
        vmax← alfa
   fin_si
   devuelve  vmax
fin_función
```