Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI5437 Inteligencia Artificial I
29 de mayo de 2014

OTHELLO

Prof. Blai Bonet

Elaborado por:
Gamar Azuaje 10-10051
Wilmer Bandres 10-10055
Juan A. Escalante 10-10227

Un árbol de juego, es un árbol dirigido en donde cada nodo representa un estado de algún juego en particular, y cada arco define un movimiento desde el estado anterior a un nuevo estado.

Existen juegos de dos personas, en donde cada uno tiene un turno y el objetivo de cada persona es jugar lo mejor posible. Este tipo de juegos se pueden representar con un árbol llamado “And-Or”, que está constituido en cada nivel del árbol solamente tiene nodos tipo “And” o solamente tiene nodos tipo “Or”.

La computación entra en juego, a la hora de buscar jugadas óptimas (o menos dañinas), o buenas jugadas para los dos jugadores. En el caso de un juego de ajedrez, puede ser que se busque hacer el jaque mate lo más rápido posible, o simplemente asegurar un jaque mate en una jugada futura. En el juego de damas, se puede ser ganar comiendo todas las fichas a la larga, o ganar lo más rápido posible. La intuición se presta a que buscar una solución óptima es mucho más costoso, computacionalmente hablando, que buscar una solución que se acerque al óptimo. Cuanto más complejo sea el juego (mientras más jugadas posibles en un turno se tenga), el espacio de búsqueda tiende a ser mucho más grande, que en juegos con una cantidad de movimientos posibles más limitada.

Existen diversos algoritmos que pueden ser utilizados, para la búsqueda de las mejores jugadas en un estado específico de un juego, entre los que se encuentran:

- Minimax y Negamax: son algoritmos que buscan la mejor jugada, similar a la forma de backtracking, esto es, prueban todas las posibles jugadas para un jugador en un estado dado, teniendo en cuenta que la mejor jugada para un jugador, se basa en escoger la peor jugada para su contrario.

- Poda alpha-beta con Minimax o Negamax: son algoritmos similares a los anteriores, con la diferencia que recorta el espacio de búsqueda, comparando el mejor resultado (buscando la mejor jugada) que se ha obtenido hasta ahora con las demás posibles jugadas y recortando el espacio de búsqueda cuando se sabe que por alguna rama del árbol de juego es imposible llegar a un mejor escenario, lo que permite podaresa rama y ahorrar tiempo.

- Scout: es un algoritmo que busca recortar el espacio de búsqueda haciendo uso de una función auxiliar llamada TEST, y recordando los parientes de un nodo del árbol para su beneficio.

- Negascout: es un algoritmo que busca un equilibrio entre poda alpha-beta y scout, uniendo lo mejor de ambos mundo y que en la práctica es el que resulta mejor.

A continuación, se presentan resultados obtenidos utilizando los algoritmos mencionados, utilizados para la búsqueda de una solución óptima de un juego de Othello de 6 x 6, luego de haber hecho varios pasos por la variante principal del juego (y tomando en cuenta que la profundidad de la raíz del árbol de juego es la profundidad 0):

- MiniMax:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	5	Indeterminado (el tiempo de culminación tiende a 0)	0.00
28	13	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	176	Indeterminado (el tiempo de culminación tiende a 0)	0.00
24	4497	Indeterminado (el tiempo de culminación tiende a 0)	0.00
22	76825	1920620	0.04
20	3478734	1911390	1.82
18	90647894	2005930	45.19
16	No termina	No termina	No termina
14	No termina	No termina	No termina
12	No termina	No termina	No termina

- NegaMax:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	5	Indeterminado (el tiempo de culminación tiende a 0)	0.00
28	13	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	176	Indeterminado (el tiempo de culminación tiende a 0)	0.00
24	4497	Indeterminado (el tiempo de culminación tiende a 0)	0.00
22	76825	1920620	0.04
20	3478734	1900950	1.83
18	90647894	1993580	45.37
16	No termina	No termina	No termina
14	No termina	No termina	No termina
12	No termina	No termina	No termina

- Poda alpha-beta con minimax:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	5	Indeterminado (el tiempo de culminación tiende a 0)	0.00

28	13	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	81	Indeterminado (el tiempo de culminación tiende a 0)	0.00
24	1002	Indeterminado (el tiempo de culminación tiende a 0)	0.00
22	4067	Indeterminado (el tiempo de culminación tiende a 0)	0.00
20	98754	1410770	0.07
18	267603	1486680	0.18
16	2031923	1551090	1.31
14	43574642	1580510	27.57
12	415909955	1622240	256.38

- Poda alpha-beta con Negamax:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	5	Indeterminado (el tiempo de culminación tiende a 0)	0.00
28	13	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	81	Indeterminado (el tiempo de culminación tiende a 0)	0.00

24	1002	Indeterminado (el tiempo de culminación tiende a 0)	0.00
22	4067	Indeterminado (el tiempo de culminación tiende a 0)	0.00
20	98754	1975080	0.05
18	267603	1911450	0.14
16	2031923	1847200	1.1
14	43574642	1815610	24.00
12	415909955	1939060	214.49

- Scout:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	2	Indeterminado (el tiempo de culminación tiende a 0)	0.00
28	9	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	67	Indeterminado (el tiempo de culminación tiende a 0)	0.00
24	1550	Indeterminado (el tiempo de culminación tiende a 0)	0.00

22	3832	Indeterminado (el tiempo de culminación tiende a 0)	0.00
20	74170	1483400	0.05
18	304674	1523370	0.2
16	1387259	1541400	0.9
14	42564855	1546130	27.53
12	323787340	1588830	203.79

- Negascout:

Profundidad	Nodos generados	Nodos generados por segundo	Tiempo
30	5	Indeterminado (el tiempo de culminación tiende a 0)	0.00
28	18	Indeterminado (el tiempo de culminación tiende a 0)	0.00
26	81	Indeterminado (el tiempo de culminación tiende a 0)	0.00
24	1326	Indeterminado (el tiempo de culminación tiende a 0)	0.00
22	3418	Indeterminado (el tiempo de culminación tiende a 0)	0.00

20	46998	1566600	0.03
18	179735	1633950	0.11
16	1126473	1586580	0.71
14	25670767	1586570	16.18
12	242962224	1646420	147.57

- Conclusiones:

El espacio de búsqueda es demasiado grande para buscar las mejores jugadas a profundidades más bajas. El límite establecido de tiempo fue de 15 minutos y se quedó corto con la cantidad de tiempo necesaria para la búsqueda en otras profundidades.

Un algoritmo que posea más información (Minimax con poda alpha-beta, Scout o Negascout) de lo que ha sucedido en la búsqueda de la mejor jugada logra podar, significativamente, el espacio de búsqueda a diferencia de los que no la posean (Minimax o Negamax). A esto se le refiere como "qué tan inteligente o astuto es un algoritmo de búsqueda en un árbol de juego".

El mejor algoritmo, de los utilizados, fue Negascout, seguido de Alpha-beta Minimax (y Negamax) al igual que Scout, y por último los algoritmos menos eficientes para recorrer el espacio de búsqueda fueron Minimax y Negamax.

Finalmente, el secreto de Negascout radica en que asume que el nodo actual que está abierto es el mejor y que se encuentra en la variación principal, buscando así si es cierto con una ventana nula de alpha-beta, lo que termina siendo mucho más rápido que una busqueda con alpha-beta normal.