ÁRBOLES DE JUEGOS

*Nelson Becerra Correa**

En el presente artículo se hace una descripción sobre los árboles de juegos, presentando los conceptos básicos y reportando los resultados de un trabajo de investigación sobre el procedimiento Minimax con corte alfa-beta aplicado a dichos árboles

Introducción

La Inteligencia Artificial (IA) es una de las ramas de la ciencia de la computación de mayor auge en los últimos lustros. Su principal objetivo no es reproducir la inteligencia humana sino la investigación de procesos simbólicos, el razonamiento no algorítmico y la representación del conocimiento.

El tema tratado en este artículo tiene que ver con dos aspectos que han dado lugar a la mayor investigación sobre IA en las dos últimas décadas: los juegos y los algoritmos de búsqueda. Para su desarrollo se va a caracterizar un proceso de juego entre dos adversarios, que discurre sobre un tablero reglamentario con piezas que se denominarán blancas, para el jugador que comienza el juego, y negras para su adversario.

Los dos jugadores alternan sus movimientos sobre el tablero, de modo que en cada oportunidad sólo se mueve una pieza (salvo excepciones como el enroque en el ajedrez), estando los movimientos regulados por las leyes que excluyen el azar. El juego finaliza cuando se alcanza alguna situación prevista en las leyes.

Las características mencionadas se aplican a juegos como ajedrez, damas, go y mancala, entre otros. El planteamiento que se hace conduce a posibilitar la construcción de programas que tomen el lugar de cualquiera de los adversarios.

Para el desarrollo de estos programas se utilizará el procedimiento Minimax, y una mejora de éste conocida como corte alfa-beta, el cual permite generar un árbol de búsqueda para representar el juego y elegir la mejor jugada en un momento determinado, sin tener que examinar todas las alternativas posibles.

En la segunda sección se representa un juego mediante un árbol de juegos; posteriormente se trabaja el procedimiento minimax y luego se desarrolla el procedimiento minimax con corte alfa-beta. Finalmente se presenta el desarrollo de una investigación sobre el procedimiento alfa-beta.

* Ingeniero de Sistemas, Magister en Ingeniería de Sistemas Universidad Nacional de Colombia, Profesor adscrito a la Facultad Tecnológica de la Universidad Distrital Francisco José de Caldas

Representación arborescente

La manera natural de representar lo que puede suceder en un juego es mediante lo que se conoce como árbol de juego[1], que es un tipo especial de árbol semántico en el que los «nodos» representan configuraciones de tablero y las «ramas» indican cómo una configuración puede transformarse en otra mediante un sólo movimiento.

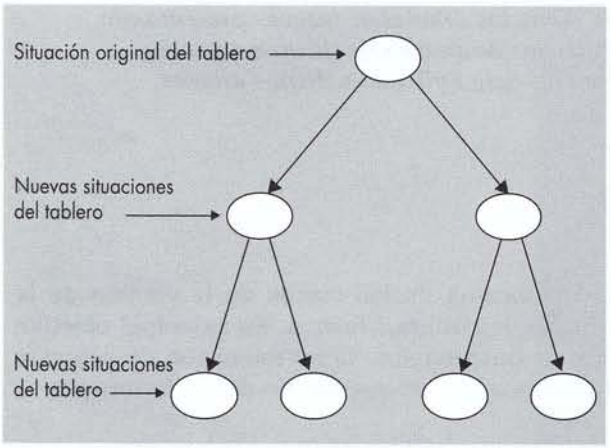

Figura 1. Arbol de Juego

En la Figura 1 se hace una representación en forma de árbol de juegos de un juego cualquiera (deterministico, de dos oponentes, con información perfecta y de suma cero), hasta un segundo nivel de profundidad. A continuación se definen unos conceptos básicos sobre este tipo de árboles[2].

- Un árbol es una estructura jerárquica aplicada sobre una colección de elementos u objetos llamados nodos, uno de los cuales es llamado nodo raíz.
- Un nodo X es descendiente directo de un nodo Y, si el nodo X es apuntado por el nodo Y. En este caso es común utilizar la expresión X es hijo de Y.
- Un nodo X es antecesor directo de un nodo Y, si el nodo X apunta al nodo Y. En este caso es común utilizar la expresión X es padre de Y.
- Se dice que todos los nodos que son descendientes directos (hijos) de un mismo padre, son hermanos.
- Todo nodo que no tiene ramificaciones (hijos), se conoce con el nombre de terminal u hoja.
- Grado es el número de descendientes directos de un determinado nodo. Grado del árbol es el máximo grado de todos los nodos del árbol.
- Nivel es el número de arcos que deben ser recorridos para llegar a un determinado nodo. Por definición la raíz tiene nivel 1.
- Altura del árbol es el máximo número de niveles de todos los nodos del árbol.

Procedimiento Minimax

Para ilustrar los árboles de juegos y a manera de ejemplo se seguirá la convención de que jugarán las fichas blancas (ver Figura 2), como una situación originada por una jugada inmediatamente anterior de las negras, y viceversa (ver Figura 3). En un momento cualquiera le corresponde jugar a las blancas. La situación es representada por la Figura 2.

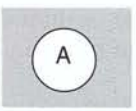

Figura 2. Momento en que Juegan las Fichas Blancas

Lo natural en este momento es que las blancas identifiquen todos los movimientos legales que pueden ejecutar. Esto dará, por ejemplo, el árbol de la Figura 3, en el cual se ilustra que existen sólo tres jugadas legales permitidas para este jugador.

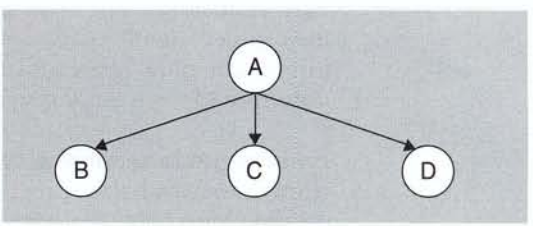

Figura 3. Alternativas de Juego para las Fichas Blancas

1. El concepto árbol de juego se toma de Rusell.
2. Los conceptos de arboles están tomados de Cairó.

Siguiendo el esquema natural de análisis, ahora el jugador de fichas blancas preveerá su jugada posterior en cada caso (Ver Figura 4); la reflexión se realiza asumiendo la perspectiva de las fichas negras.

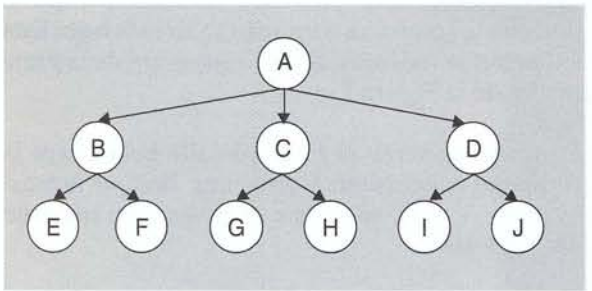

Figura 4. Segunda Jugada de Fichas Blancas

La exploración del jugador de fichas blancas (Figura 4) da como resultado los nodos E, F, G, H, I, J, correspondientes a un tercer nivel de profundidad; estos nodos plantean una jugada en el tablero A. El análisis podría continuar tanto como lo permita la concentración del jugador, pero supóngase aquí una detención.

A esta altura, el jugador de fichas blancas quiere saber cuál de los tableros o nodos es el que conviene alcanzar. Se recurre entonces al uso de la denominada función de evaluación estática[3], la cual deberá proveer un valor único para cada tablero, que sea indicativo de su valor para uno u otro jugador.

Una primera forma de imaginar tal función es recurrir a un juego concreto, por ejemplo el ajedrez:

Pieza	Blancas	Negras	
Peón	+1	-1	En un tablero se hace acumulación
Caballo	+3	-3	de valores dando un resultado positivo,
Torre	+5	-5	negativo o cero, e.t.c.

Los valores que aparecen (+1, -1, +3, +3, +5, -5[4]), son tomados de la teoría desarrollada para este juego específico.

Además del valor intrínseco de las piezas (el valor de cada pieza, independientemente de la posición), se pueden adicionar valores acordes con su buena posición en el tablero, el grado de seguridad de la posición, etc.

Supóngase que f representa la función de evaluación estática, y que la evaluación de los tableros-hoja en el árbol anterior, (Ver Figura 4), es: f(E) =-1, f(F) = +2, f(G) = +1, f(H) = +3, f(I) = -2, f(J) = +4 (Figura 5).

Ahora las blancas reflexionan: "si juego de modo que la posición para las negras sea B, el adversario podrá escoger entre jugadas que produzcan los tableros E o F".

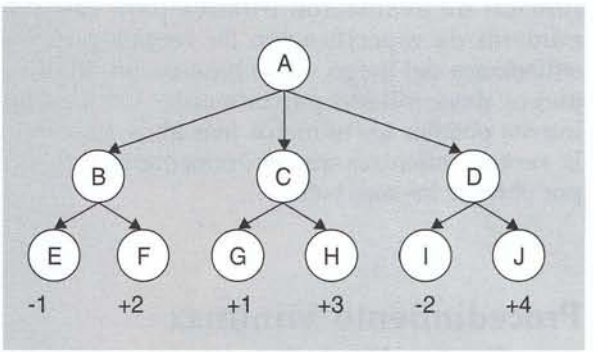

Figura 5. Reflexión del Jugador de Fichas Blancas

"El tablero F me favorece y será mi turno, pero es de suponer que el adversario juega bien, de modo que él me dejará en la situación E, y allí tendré un tablero desfavorable que defender".

Por consiguiente, «asocio con B, el mínimo de los valores de B. Así estaré indicando lo que sucederá si llego a B». En forma análoga se analizará C y D, obteniendo una valoración subsiguiente del árbol, como aparece en la Figura 6.

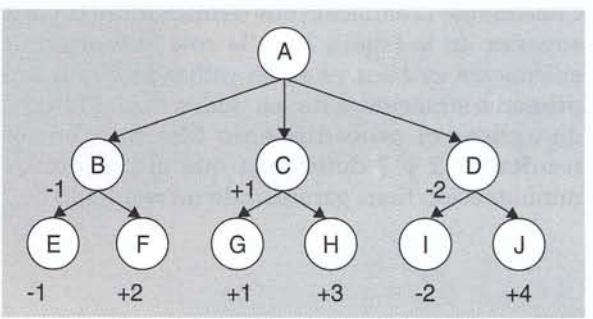

Figura 6. Valoración del Arbol de Decisión

3. El concepto función de evaluación estática se toma de Winston.
4. Véase por ejemplo Norving.

Ahora el blanco establecerá lo que debe jugar:

- Si juega a B, el tablero vale -1 (favorece al adversario).
- Si juega a C, el tablero vale +1 (me favorece)
- Si juega a D, el tablero vale -2 (favorece al adversario).

"Cuando es el turno de las blancas, se toma el máximo de los valores asociados a sus hijos".

La anterior descripción del juego corresponde al procedimiento de búsqueda Minimax[5]. Este es un método para determinar movimientos; emplea una función de evaluación estática para calcular números de especificación de ventaja para las situaciones del juego, en la base de un árbol de juegos desarrollado parcialmente. Un jugador intenta obtener los números más altos, buscando la ventaja, mientras que el oponente se esfuerza por obtener los más bajos.

Procedimiento Minimax con Corte alfa-beta

En principio puede parecer que la función de evaluación estática deberá utilizarse en cada nodo hoja de la base del árbol de juegos, pero por fortuna no sucede así. Existe un procedimiento que reduce el número de ramas que deben generarse y las evaluaciones estáticas por realizar, simplificando así el trabajo global necesario. Es algo similar a la idea de ramificación y poda, en el sentido de que algunas trayectorias son malas aún cuando no se sigan hasta el límite previsto.

Considérese la situación que se muestra en la parte superior de la Figura 7, en la cual la función de evaluación estática ya se ha utilizado en las dos primeras situaciones de los nodos hoja. El hecho de aplicar el procedimiento Minimax en los resultados 2 y 7 determina que el jugador de minimización tiene garantizado un resultado de 2 si el maximizador toma la rama izquierda del nodo de la cima. Este movimiento a su vez asegura que el maximizador tiene garantizado un resultado de por lo menos 2 en la cima. Esta garantía resulta evidente aún antes de realizar otras evaluaciones estáticas, ya que el maximizador puede elegir indudablemente la rama izquierda siempre que la derecha le conduzca a un resultado más bajo. Esta situación se indica en el nodo superior de la parte media de la Figura 7.

En forma general, el principio alfa-beta ofrece la siguiente conclusión: *si tiene una idea que indudablemente es mala, no se tome el tiempo para constatar qué tan mala es.*

Reporte de una Investigación acerca del Procedimiento Minimax

El contenido de este apartado hará la presentación de un trabajo de investigación que se desarrolló como tesis de Maestría en Ingeniería de Sistemas en la Universidad Nacional de Colombia.

Se trabajó sobre el juego de mancala clasificado dentro de la teoría de juegos como:

- Determinístico
- De dos oponentes
- De suma cero
- Con información perfecta.

Para poder implementar eficientemente el procedimiento minimax con corte alfa-beta a este juego fue necesaria la creación de un nuevo algoritmo iterativo. Se trabajó a partir de un primer algoritmo de este tipo desarrollado por Lauriére para el procedimiento minimax; este algoritmo se corrigió y incorporándole el procedimiento alfa-beta, el cual está disponible para la creación de juegos con las características anteriormente descritas.

5. Método descrito por Winston.

Algoritmo minimax original

La versión del algoritmo, escrito en francés, es la siguiente:

```
MINIMAX :  DONNEES:  (echiquier, camp)
           RESULTAT: (valeur de la position = minimax)
           Prof = 1; camp =1; E(1) = pilocoups (echiquier, camp);
           Eval(1) = -oo

REPETER TANT QUE  porf > 1
        REPETER TANT QUE E(prof) ¡= vide
                Copu  = sommet – pile (E(prof))
                Echiquier = jouer (echiquier, coup)
                Si prof ¡= (profmax – 1)
                   ALORS  prof = prof +1
                          Camp =  - camp
                          E (prof) = pilecoups (echiquier, campo)
                          eval(prof) = - oo
                   SINON  {profmax: evaluer la position
                          et comparer}
                          Eval (prof) = MAX [eval(prof),
                          camp*valuation (echl)]
                          {dejouer le dernier coup a profmax:}
                          coup = depile(E(prof))
                          echiquier = dejouer (echiquier, coup)
                          {rester a cette proffondeur tant
                          qu'il reste des coups}
                FSI
        FR sur E
  {retour arriere}
  SI prof=1 ALORS minimax = eval(1): fin FSI
  Camp = - camp
  Prof   = prof-1
  {Dejouer le dernier coup a cette profondeur }
  coup = depile ( E (prof))
  echiquier = dejouere (echiquier,copu)
  {Remonter L'evaluation par minimax}
  eval (prof) = MAX [eval (prof) – ( eval (prof +1))]
FR sur prof
FIN minimax.
```

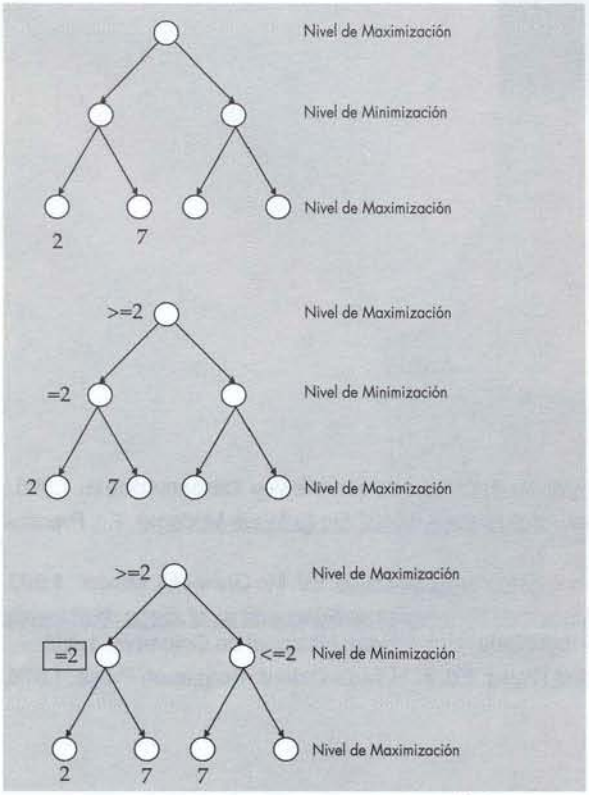

Figura 7. Procedimiento Minimax con corte alfa-beta

Algoritmo Modificado

El siguiente algoritmo iterativo tiene las modificaciones necesarias hechas sobre el propuesto por Lauriére, para que opere con corte alfa-beta.

```
Entrada: (tablero,lado)
Salida: Valor del la Posición.
Inicializacion: prof1; E(1) = listadejugadas(tablero,lado);Eval(1)=-(lado)*alfa
(tomamos alfa como el signo para infinito)
mientras prof >=1
      hacer
      mientras E(prof) <> vacio
            hacer
            jugada = primera (E(prof);
            tablero = Jugada(tablero,jugada);
            si prof <> (profmaxima –1);
            entonces
                  hacer
                  prof = prof +1;
                  lado = -lado;
                  E(prof) = lista de jugadas(tablero,lado);
                  Eval(prof) = -(lado) * alfa;
                  Fin_hacer
            Sino
                  Hacer
                  Sw=1;
                  Si lado=1
                  Entonces
                        Hacer
                              Eval(prof)=  MAX[eval(prof),valor(tablero)];
                              Si prof >1
                              Entonces
                                    hacer
                                    si eval(prof) >= eval(prof-1)
                                    entonces
                                          hacer
                                          eliminar(E(prof));
                                          SW=0;
                                          Fin_hacer
                                    Fin_hacer
                        Fin_hacer
                  Sino
                        Hacer
                        Eval(prof) = MIN[eval(prof),valor(tablero)]
                        Si prof > 1
                        Entonces
                              Hacer
                              Si eval(prof) <= eval(prof-1)
                              Entonces
                                    Hacer
                                    Eliminar(E(prof));
                                    Tablero = restaurar(tablero,jugada);
                                    Fin_hacer
                              Fin_hacer
                        Fin_hacer
                  Si SW=1
                  Entonces
                        Hacer
                        Jugada = sacarprimera(E(prof));
                        Tablero = restaurar(tablero,jugada);
                        Fin_hacer
                  Fin_hacer
            Fin_hacer
      Si prof = 1
      Entonces
            Hacer
            Retornar(eval(1))
            Terminar
            Fin_hacer
      Lado = -lado;
      Prof  = prof –1;
      Jugada = sacarprimera(E(prof));
      Tablero = restaurar(tablero,jugada);
      Si lado=1
      Entonces
            Hacer
            Eval(prof) = MAX[eval(prof),eval(prof + 1)];
            Si prof>1
            Entonces
                  Si eval(prof) >= eval(prof-1)
                  Si eval(prof) ¡= vacio()
                  Entonces eliminar(E(prof));
            Fin_hacer
      Sino
            Hacer
            Eval(prof) = MIN[eval(prof),eval(prof + 1)];
            Si prof >1
            Entonces
                  Si eval(prof) <= eval(prof-1)
                  Si eval(prof) ¡= vacia()
                  Entonces eliminar(E(prof));
            Fin_hacer
            Fin_hacer
```

Los algoritmos expuestos en los libros casi siempre son demasiado generales, e implican para su implementación un trabajo dispendioso. El desarrollo del anterior algoritmo iterativo es un aporte básico y está disponible para cualquier persona interesada en desarrollar juegos que reúnan las condiciones ya mencionadas.

Sin embargo, hasta el momento no se comprende suficientemente cómo combinar los algoritmos de búsqueda heurística (búsqueda preferente por lo mejor, y limitada por la capacidad de memoria y los algoritmos de mejoramiento iterativo), y los algoritmos de juegos (minimax, minimax con corte alfa-beta), en un sólo sistema robusto que permita un mejor aprovechamiento de las técnicas de Inteligencia Artificial. Un algoritmo que permita esta unificación será una herramienta de gran ayuda para el tratamiento de juegos.

El desarrollo del anterior algoritmo iterativo es un aporte básico y está disponible para cualquier persona interesada en desarrollar juegos que reúnan las condiciones ya mencionadas.

REFERENCIAS BIBLIOGRÁFICAS

- WINSTON, Patrick Henry. *Inteligencia Artificial*. Addison-Wesley Iberoamericana. 1.996.
- RUSELL Stuart, NORVING Peter. *Inteligencia Artificial. Un Enfoque Moderno*. Ed. Prentice Hall, 1.996
- CAIRO Osvaldo, GUARDATI Silva. *Estructuras de Datos*. Ed. Mc Graw-Hill, Mexico, 1.993.
- BECERRA Correa, Nelson. *Aplicación de Estrategias de Búsqueda en el Juego de Mancala, Tesis de Maestría*, Facultad de Ingeniería, Universidad Nacional de Colombia. 1.998.
- VON NEWMANN Jhon. *Collected Works*. Ed. A. H Taub-Oxford: Pergamon Press. 1.976.