

Poda Alfa-Beta y Monte Carlo Tree Search

Jorge Baier

Departamento de Ciencia de la Computación
Pontificia Universidad Católica de Chile

Santiago, Chile



- Conocer la técnica Poda Alfa-Beta como una forma de optimizar minimax
- Entender el funcionamiento del algoritmo MCTS
- Analizar la complejidad y eficiencia del algoritmo



Poda Alfa-Beta en el Algoritmo Minimax

- La cantidad de estados de juego es exponencial en la profundidad del árbol.
- La poda alfa-beta es una técnica para reducir la cantidad de estados evaluados por minimax al eliminar partes del árbol que no afectan el resultado final.
- En la poda alfa-beta, se mantienen dos valores: alfa (el valor mínimo que el jugador MAX ya ha encontrado) y beta (el valor máximo que el jugador MIN ya ha encontrado).
- La poda alfa-beta permite evitar evaluar ciertas ramas del árbol cuando ya se sabe que no conducirán a una mejor solución.



Función Alpha-Beta

Entrada: Estado S , alfa, beta, Maximizando

Salida: Valor de evaluación de S

1. **if** S es terminal **then**
2. **return** Función_de_evaluación(S)
3. **if** Maximizando = True **then**
4. $v \leftarrow -\infty$
5. **for each** hijo in S **do**
6. $v \leftarrow \text{máx}(v, \text{AlphaBeta}(\text{hijo}, \text{alfa}, \text{beta}, \text{False}))$
7. $\text{alfa} \leftarrow \text{máx}(\text{alfa}, v)$
8. **if** $\text{beta} \leq \text{alfa}$ **then**
9. **break** // *Podar el resto de los hijos*



Función Alpha-Beta

```
10. else
11.    $v \leftarrow +\infty$ 
12.   for each hijo in  $S$  do
13.      $v \leftarrow \text{mín}(v, \text{AlphaBeta}(\text{hijo}, \text{alfa}, \text{beta}, \text{True}))$ 
14.      $\text{beta} \leftarrow \text{mín}(\text{beta}, v)$ 
15.     if  $\text{beta} \leq \text{alfa}$  then
16.       break // Podar el resto de los hijos
17. return  $v$ 
```



Ejemplo

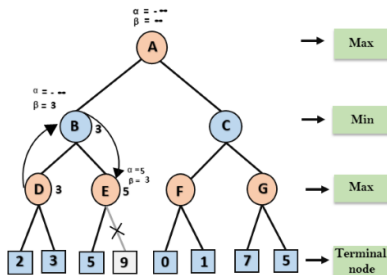


Figura: En el nodo E, Max tomará su turno y el valor de alfa cambiará. El valor actual de alfa se comparará con 5, por lo que $\max(-\infty, 5) = 5$, por lo tanto, en el nodo E $\alpha = 5$ y $\beta = 3$, donde $\alpha \geq \beta$, por lo que el sucesor derecho de E será podado y el valor en el nodo E será 5.



Ejemplo

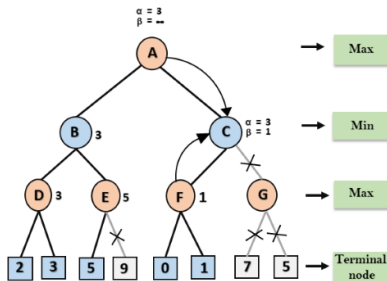


Figura: El nodo F devuelve el valor del nodo 1 al nodo C, en C $\alpha = 3$ y $\beta = +\infty$. Como estamos en un nivel MIN el valor de beta cambiará, ya que $\min(\infty, 1) = 1$. Ahora en C, $\alpha = 3$ y $\beta = 1$, y nuevamente satisface la condición $\alpha \geq \beta$, por lo que el siguiente hijo de C, que es G, será podado el algoritmo no calculará todo el subárbol G.



Monte Carlo Tree Search (MCTS)

- MCTS es una estrategia de búsqueda que se utiliza en juegos con alto factor de ramificación y para los cuales es difícil definir una buena función de evaluación.
- En MCTS, el valor de un estado se estima como la utilidad promedio sobre varias simulaciones de juegos completos a partir de ese estado.
- MCTS consta de cuatro etapas principales: Selección, Expansión, Simulación y Retropropagación.
- Utiliza una política de selección (como UCB1) para decidir qué movimientos explorar más a fondo.

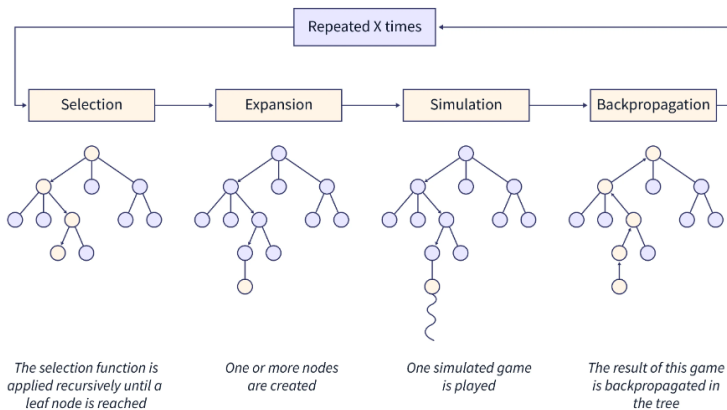


Algoritmo Monte Carlo Tree Search (MCTS)

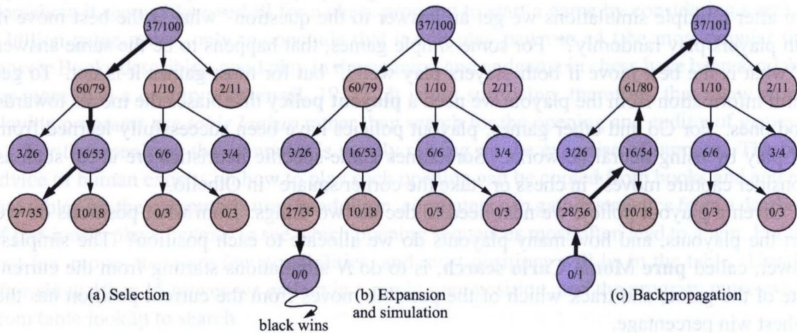
1. **Seleccionar:** Desde la raíz del árbol de búsqueda, elegir un movimiento guiado por la política de selección. Seguir haciendo esto hasta llegar a un nodo terminal.
2. **Expandir:** Añadir un nodo hijo al árbol para el nodo terminal seleccionado.
3. **Simular:** Realizar una simulación completa desde el nuevo nodo. Los movimientos de la simulación no son guardados en el árbol.
4. **Retropropagar:** Actualizar la información del árbol con el resultado de la simulación.



Ejemplo



Ejemplo



Un objetivo de la etapa de selección es balancear la exploración con la explotación. Una forma de lograr esto es elegir la siguiente acción de acuerdo a la regla:

$$a^* = \arg \max_{t \in \text{Succ}(s)} \left\{ \frac{U(t)}{N(t)} + C \sqrt{\frac{\log N(s)}{N(t)}} \right\},$$

donde:

1. $U(s)$ es la utilidad entregada por los juegos a partir de s .
2. $N(s)$ es el número de veces que hemos considerado jugar por s .



- Conocer la técnica Poda Alfa-Beta como una forma de optimizar minimax
- Entender el funcionamiento del algoritmo MCTS
- Analizar la complejidad y eficiencia del algoritmo

