Poda Alfa-Beta y Monte Carlo Tree Search

Jorge Baier

Departamento de Ciencia de la Computación
Pontificia Universidad Católica de Chile

Santiago, Chile

Objetivos

- 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.

Pseudocódigo de la Poda Alfa-Beta

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 ← −∞
5.     for each hijo in S do
6.         v ← máx(v, AlphaBeta(hijo, alfa, beta, False))
7.         alfa ← máx(alfa, v)
8.         if beta ≤ alfa then
9.             break // Podar el resto de los hijos
```

Pseudocódigo de la Poda Alfa-Beta

Función Alpha-Beta

```
10. else
11.     v ← +∞
12.     for each hijo in S do
13.         v ← mín(v, AlphaBeta(hijo, alfa, beta, True))
14.         beta ← mín(beta, v)
15.         if beta ≤ alfa then
16.             break // Podar el resto de los hijos
17. return v
```

Ejemplo

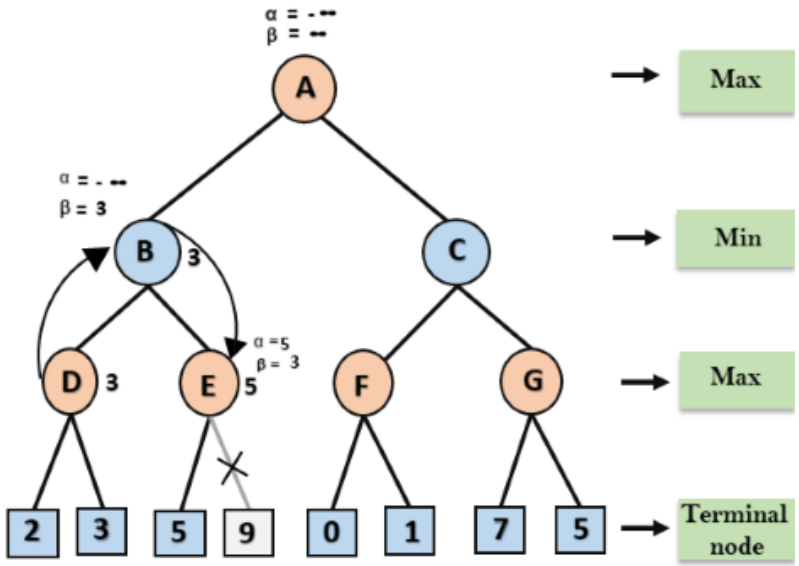


Figura: En el nodo E, Max tomará su turno y el valor de alfa cambiará. El valor actual de alfa se comparará con 5, por lo que max (-∞, 5) = 5, por lo tanto, en el nodo E $\alpha = 5$ y $\beta = 3$, donde $\alpha \geq \beta$, por lo que el sucesor derecho de E será podado y el valor en el nodo E será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

Ejemplo

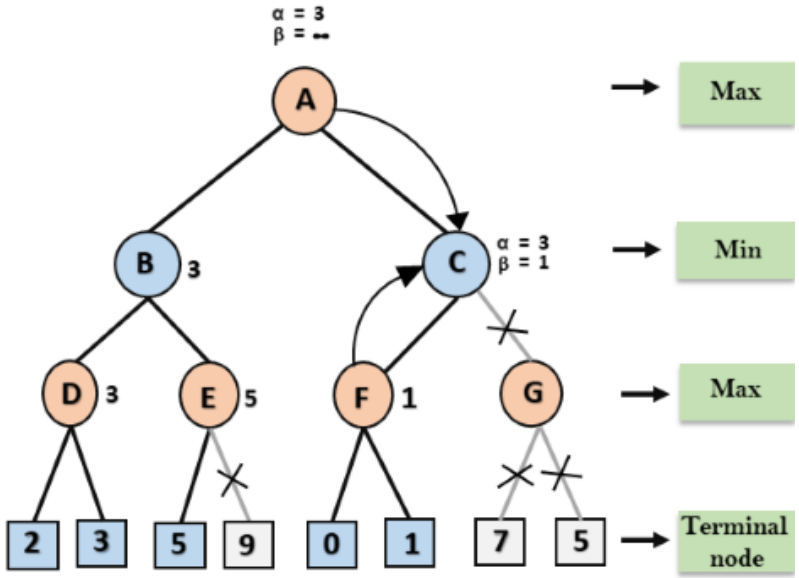


Figura: El nodo F devuelve el valor del nodo 1 al nodo C, en C $\alpha = 3$ y $\beta = +\infty$. Como estamos en un nivel MIN el valor de beta cambiará, ya que min $(\infty, 1) = 1$. Ahora en C, $\alpha = 3$ y $\beta = 1$, y nuevamente satisface la condición $\alpha \geq \beta$, por lo que el siguiente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

Monte Carlo Tree Search (MCTS)

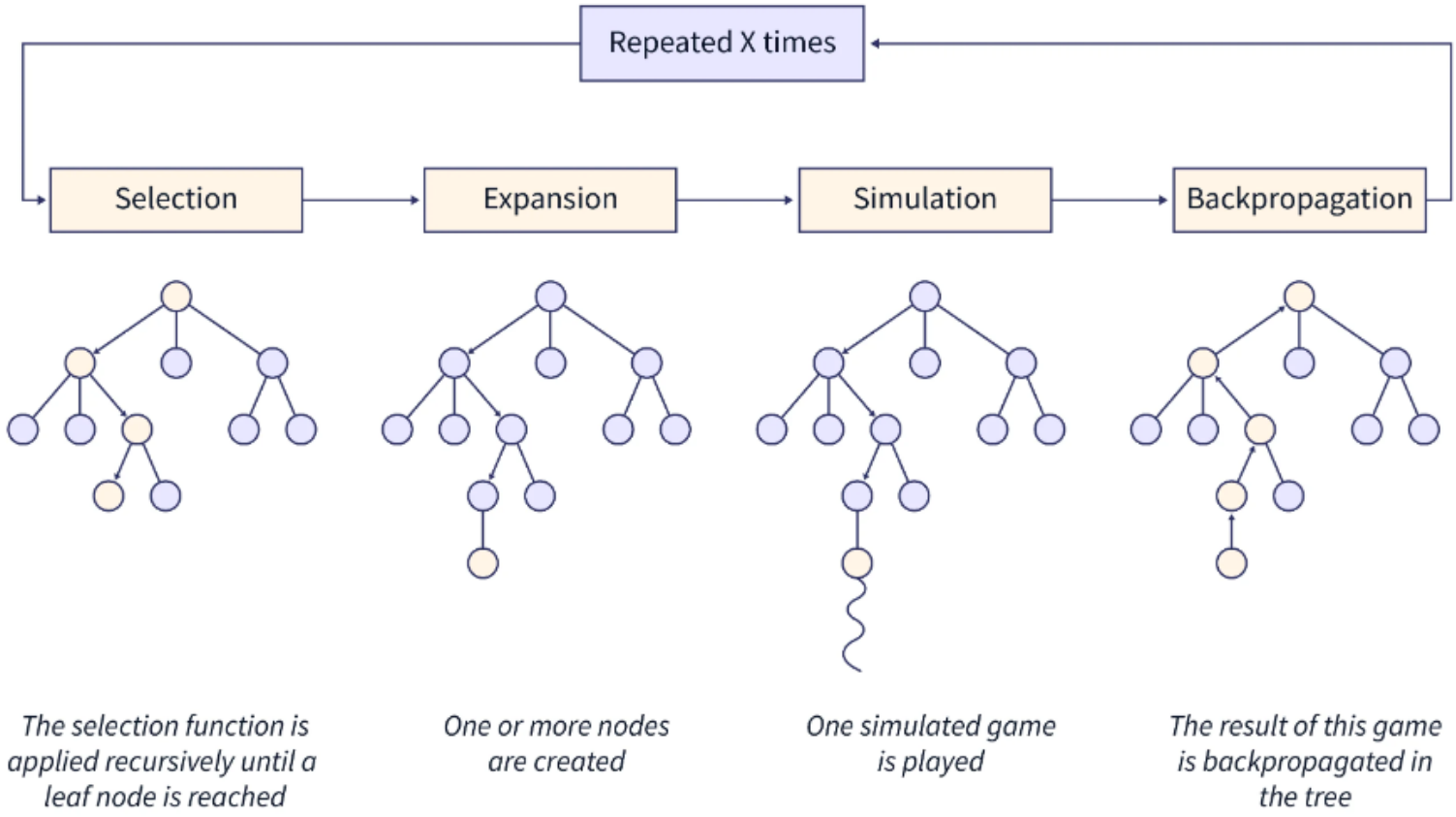
- MCTS es una estrategia de búsqueda que se utiliza en juegos con alto factor de ramificación y para los cuales es difícil definir una buena función de evaluación.
- En MCTS, el valor de un estado se estima como la utilidad promedio sobre varias simulaciones de juegos completos a partir de ese estado.
- 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.

Funcionamiento MCTS

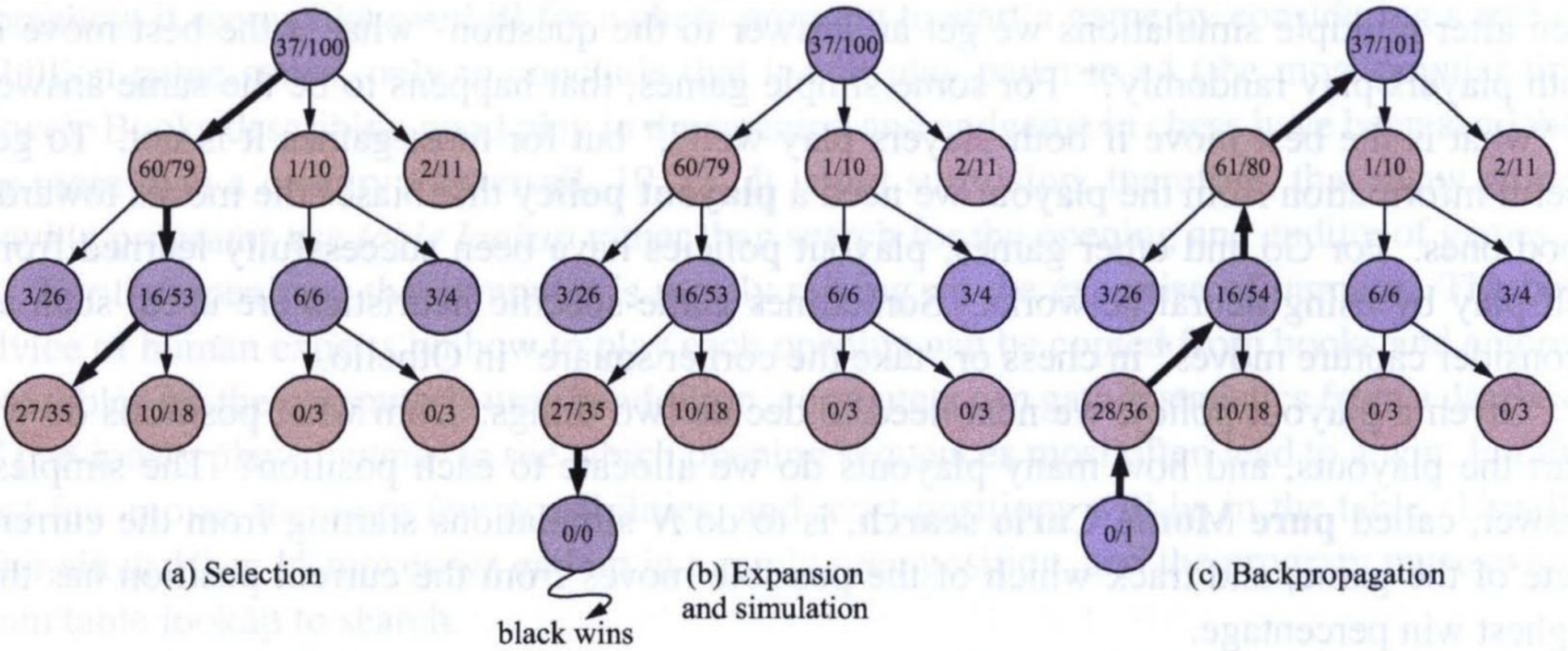
Algoritmo Monte Carlo Tree Search (MCTS)

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


Repeated X times
Selection
Expansion
Simulation
Backpropagation
The selection function is applied recursively until a leaf node is reached
One or more nodes are created
One simulated game is played
The result of this game is backpropagated in the tree

Ejemplo


37/100
60/79
1/10
2/11
3/26
16/53
6/6
3/4
27/35
10/18
0/3
0/3
(a) Selection
37/100
60/79
1/10
2/11
3/26
16/53
6/6
3/4
27/35
10/18
0/3
0/3
0/0
black wins
(b) Expansion
and simulation
37/101
61/80
1/10
2/11
3/26
16/54
6/6
3/4
28/36
10/18
0/3
0/3
0/1
(c) Backpropagation

Selección

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:

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

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.

Objetivos

- 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