Controlador para la generación de movimientos con grafos implícitos utilizando Monte Carlo Tree Search aplicado en Knight Chess

Introducción

A continuación describiremos cómo logramos crear un controlador para jugar una partida de Knight Chess, el cual recibe un tablero de tal juego y entrega una jugada en específico. Dentro de esto hablaremos de la definición de estado, la implementación del algoritmo Monte Carlo Tree Search y sus heurísticas y funciones utilizadas.

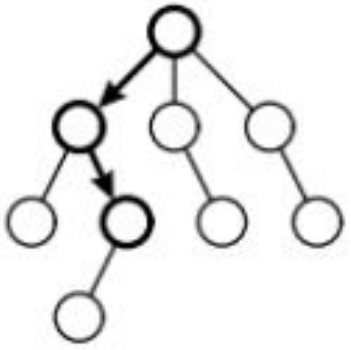
Definición del estado

Partiendo por nuestra definición de estado, encontramos pertinente tener atributos para el tablero, caballos aliados, caballos enemigos, acción que dio origen al estado actual, flag que indica un estado final y una variable delta que guarda los posibles movimientos para un caballo (independiente de su posición). Como se muestra en la figura.

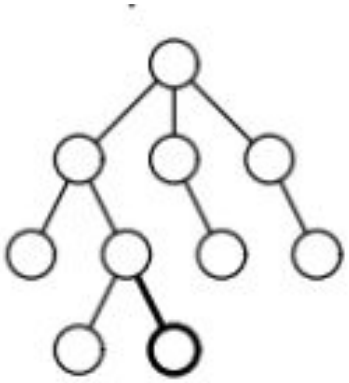
Algoritmo de búsqueda MCTS

Por parte de la creación de nodos, fue en dónde dejamos todos los atributos de decisión para el algoritmo MCTS, tales como simulaciones, recompensas, nodo padre, nodos hijos y una constante. Además, están todas las funciones para generar el gráfico implícito. De esta manera tendremos un acceso a las funciones fácilmente al momento de trabajar en el árbol.

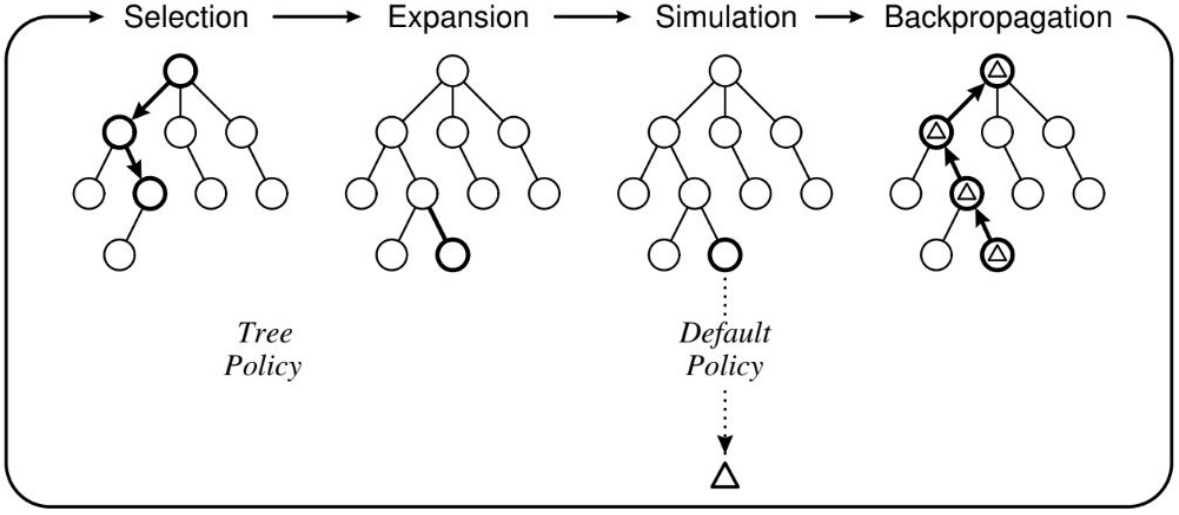
La **etapa de selección**, de nuestro algoritmo de búsqueda, es predominado por la política de búsqueda del árbol principal, la cual es denominada por un límite de expansiones y el descubrimiento de un nodo sin estar expandido completamente. Posterior a esto encontramos la **etapa de expansión**. Al encontrar un nodo que no está expandido



completamente generamos sus posibles hijos y seleccionamos el mejor, evaluado por la Política de selección UCT, la cual nos brinda una estimación optimista del valor que podría obtenerse al simular un nodo. En la **etapa de simulación,** veremos principalmente a la política de defecto de algoritmo, la cual a partir de un nodo específico genera sus hijos y selecciona uno al azar para seguir este procedimiento recursivamente, esta selección es evaluada aleatoriamente. El proceso se realiza hasta un límite de expansión o al encontrarse con nodo terminal. Finalmente llegamos a la **etapa de propagación hacia atrás**, la cual se encarga de evaluar la ruta de nodos por la que se llegó con la política por defecto. Esta evaluación es realizada en base a la recompensa dada al último nodo encontrado en la búsqueda de la política por defecto.



A continuación podemos ver una imagen que nos muestra los cuatro procedimientos involucrados continuamente.



Conclusiones

El algoritmo da buenos resultados, en el sentido de que está bien implementado y realiza sus funciones correspondientemente. A la hora de los resultados, nos dimos cuenta de que los parámetros de las heurísticas y políticas utilizadas deben ser reajustados, tales como el límite de expansión y valores de recompensas. En base a esto se plantea como desafío mejorar los parámetros mencionados y agregar una nueva heurística.