Un recorrido por Alpha Zero y una implementación de Montecarlo

A tour of Alpha Zero and an implementation of Monte Carlo

Autor 1: Kevin Leandro Bernal León, Leidy Carolina Arcila Cardona, Gustavo Alexander Dávila Ayala

*Facultad de Ingenierías. IS&C, Universidad Tecnológica de Pereira, Pereira, Colombia*

Correo-e: [kevinleon@utp.edu.co](mailto:kevinleon@utp.edu.co), [carolina.arcila@utp.edu.co](mailto:carolina.arcila@utp.edu.co) , [davila7@utp.edu.co](mailto:davila7@utp.edu.co)

***Resumen*— Alpha zero es un programa informático desarrollado por DeepMind, que utiliza el enfoque generalizado de Alpha Go Zero. El 5 de diciembre de 2017, el equipo DeepMind lanzó una preimpresión presentando Alpha Zero, que logro en 24 horas un nivel de juego sobrehumano en ajedrez, shogi y Go al derrotar a los campeones del mundo. El Método de Monte-Carlo es una técnica para analizar fenómenos por medio de algoritmos computacionales, que utilizan y dependen fundamentalmente de la generación de números aleatorios. Este es el uno de los algoritmos que se encuentran en el núcleo de Alpha Zero.**

***Palabras clave— Alpha Go, Alpha zero, DeepMind, programa, informático, logro, ajedrez, shogi.***

***Abstract*— Alpha zero is a computer program developed by DeepMind, which uses the generalized approach of Alpha Go Zero. On December 5, 2017, the DeepMind team suffered a prepress by presenting Alpha Zero, which achieved in 24 hours a superhuman game level in chess, shogi and Go by defeating the world champions. The Monte-Carlo Method is a technique to analyze phenomena through computational algorithms, which use and depend primarily on the generation of random numbers. This is the one of the algorithms found in the Alpha Zero core.**

***Key Word* — Alpha, zero, Go, DeepMind, program, computer, chess, shogi.**

1. INTRODUCCIÓN

El juego del ajedrez es el dominio más estudiado en la historia de la inteligencia artificial. Los programas más sólidos se basan en una combinación de técnicas de búsqueda sofisticadas, adaptaciones específicas de dominio y funciones de evaluación artesanales que han sido refinado por expertos humanos durante varias décadas.

En contraste, el programa AlphaGo Zero Recientemente logró un rendimiento sobrehumano en el juego de Go, mediante el aprendizaje de refuerzo tabula rasa de los juegos de auto-juego. También logró en 24 horas un nivel de juego sobrehumano en los juegos de ajedrez y shogi (ajedrez japonés), así como Go, y derrotaron convincentemente a un campeón mundial en cada caso.

El programa de inteligencia artificial AlphaZero ha borrado a Stockfish, el motor de ajedrez mejor calificado, el cual es la herramienta para la mayoría de los mejores jugadores, este motor ganó el Campeonato TCEC 2016 y el Campeonato Chess.com Computer Chess 2017, sin embargo, no tuvo ninguna oportunidad contra AlphaZero quien ganó el partido a puerta cerrada de 100 juegos con 28 victorias, 72 empates y cero pérdidas.

Ahora que ya vimos los hitos que se han logrado con respecto a este tópico, cabe resaltar que este documento se enfocará en hacer una descripción de los componentes de Alpha zero y explicar la implementación del algoritmo monte-carlo, dado que tiene una importancia alta dentro de todo el sistema.

1. CONTENIDO

AlphaZero calcula unas 60 mil posiciones por segundo, mientras que StockFish puede evaluar unos 700 millones de posiciones por segundo con el mejor hardware disponible. Pero tal vez la contribución de DeepMind a este tema no será el hacer un programa de ajedrez imbatible, sino que parecen haber encontrado un sistema para que cualquier juego de mesa pueda atacarse con el mismo algoritmo.

**1) Arquitectura**

Los programadores de AlphaZero, alojados dentro de la división DeepMind de Google usaron "aprendizaje automático", específicamente aprendizaje de refuerzo. Dicho más claramente, a AlphaZero no se le "enseñó" el juego en el sentido tradicional. Eso significa que no hay libro de apertura, no hay tablas de final de juego, y aparentemente no hay algoritmos complicados que realicen diferencias mínimas entre los peones centrales y los peones laterales.

**2) Componentes del sistema**

**Red Neuronal**

AlphaZero es un programa de computadora que usa una red neuronal para aprender habilidades como, por ejemplo, jugar ajedrez o cualquier juego para dos personas donde se pueda saber el estado de cada jugador. Solamente conoce las reglas de cómo se mueven las piezas y aprende jugando contra sí mismo, generalmente en un par de horas. Inicialmente cometía muchos errores, como los principiantes, pero después de un entrenamiento de 4 horas, batió a StockFish.

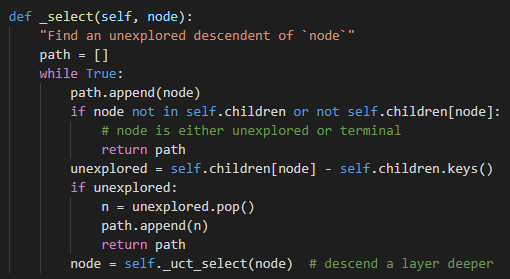
Únicamente hay que destacar el hecho de que el mismo algoritmo, sin cambios, conociendo exclusivamente las reglas del juego y con la misma técnica de autoaprendizaje, haya sido capaz de fulminar igualmente a los programas campeones mundiales de cada uno de estos dos juegos. En el caso del GO, ese campeón mundial era precisamente AlphaGo, también de DeepMind y en el caso de shogi ese programa es ELMO. Ambos han sido destrozados por Alphazero.

**3) Algoritmo Monte-Carlo**

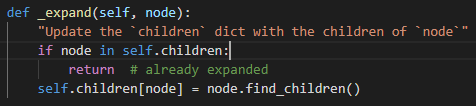
La importancia de este algoritmo parte de la poca cantidad de posiciones que explora Alpha zero por segundo con respecto a stockfish. Logra realizar 80000 búsquedas en ajedrez, en comparación con los 70 millones de búsquedas que despliega stockfish, aun con tan pocas exploraciones logra muy buenos resultados. Este es un algoritmo significativo que ha permitido realizar una mejor elección de jugadas sin comprometer el gasto computacional.

El algoritmo MCTS (monte carlo tree Search) construye un árbol, nodo por nodo, de acuerdo a los resultados de hacer la simulación. Este proceso puede ser explicado en 4 pasos:

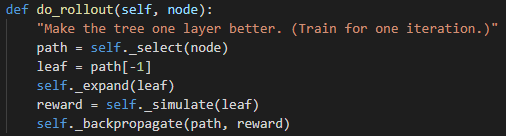
**Selección:** Seleccionar el mejor nodo hijo, comenzando desde la raíz, este representa el estado líder para mejorar el resultado en general.



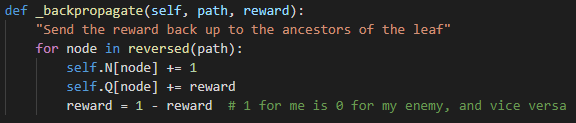
**Expansión:** si no se ha llegado a un nodo terminal (si no es el final del juego) entonces cree uno o más nodos hijos y seleccione uno.



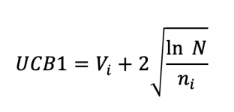
**Simulación(rollout):** correr una simulación hasta el final, partiendo desde el nodo seleccionado en el paso de Expansión hasta que se logre un resultado.



**Backpropagation:** actualizar la actual secuencia de movimientos con el resultado de la simulación. En otras palabras, cuando se obtenga un resultado debe devolverse informando el resultado obtenido por el camino que tomo para llegar allí, esto también se le denomina retropropagación.



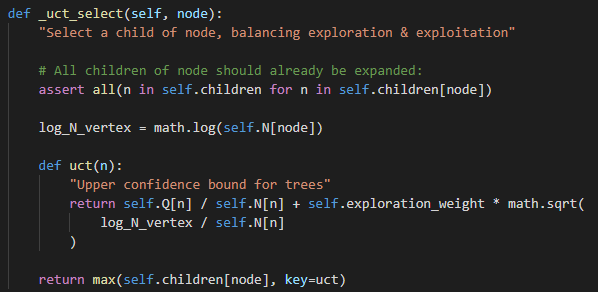
Para realizar la selección se hace con el valor UCB (límite de confianza superior para un nodo), esta dado por la siguiente formula.



Vi: es el promedio de la relación recompensa/valor de todos los nodos debajo del nodo actual.

N: es el número de veces que el nodo padre ha sido visitado

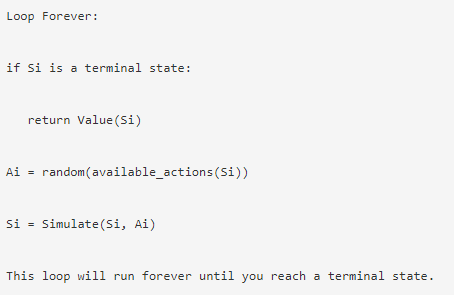
Ni: es el número de veces que el nodo hijo ha sido visitado.



Ahora vamos a ampliar la explicación de algunos puntos que son importantes para el entendimiento del algoritmo.

¿Qué significa hacer un lanzamiento? hasta que lleguemos al nodo hoja, se elige una acción al azar en cada paso y se simula esta acción para recibir una recompensa promedio cuando termina el juego. El problema/ventaja de algoritmos como el Minimax es que en cada nivel del árbol se escoge el movimiento más optimo, la desventaja se encuentra en juegos como el GO, donde hay millones de estados por recorrer y el tiempo de ejecución como el poder computacional se vuelve muy alto. El hecho que Montecarlo haga esta elección permite que la exploración sea más rápida.

**Pseudocodigo - Rollout**



¿Cómo se realiza la expansión? Se empieza con S0, que es el estado inicial. Si el nodo actual no es un nodo hoja, calculamos los valores para UCB1 y elegimos el nodo que maximiza el valor de UCB. Seguimos haciendo esto hasta llegar al nodo hoja.

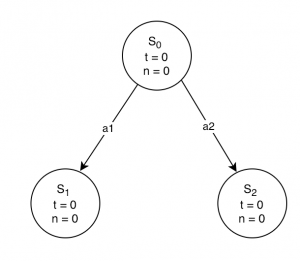
A continuación, preguntamos cuántas veces se muestreó este nodo hoja. Si nunca se ha muestreado antes, simplemente hacemos un lanzamiento (en lugar de expandirlo). Sin embargo, si se ha muestreado antes, entonces agregamos un nuevo nodo (estado) al árbol para cada acción disponible (que llamamos expansión aquí).

Su nodo actual es ahora este nodo recién creado. Luego hacemos un lanzamiento desde este paso.

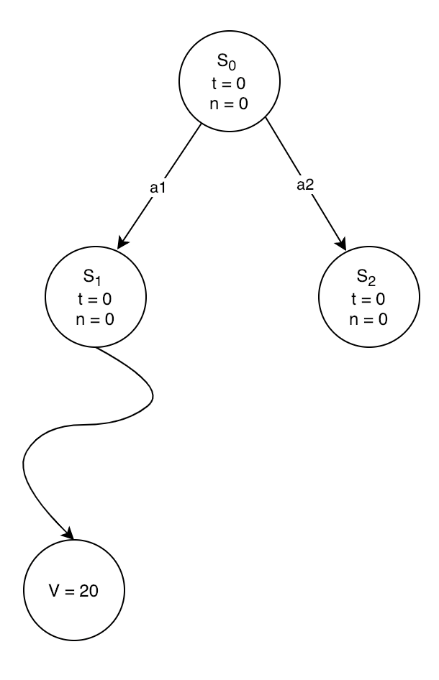
A continuación, se muestran todos los estados por los que pasa el algoritmo al correrlo, esto ocurre por el número de iteraciones que el programador defina.

**Iteración 1**

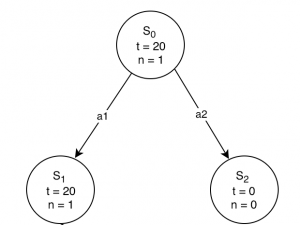
Comenzamos con un estado inicial S0. Aquí, tenemos acciones a1 y a2 que conducen a estados s1 y s2 con puntaje total t y número de visitas n. Pero, ¿cómo elegimos entre los 2 nodos secundarios?



* Aquí es donde calculamos los valores de UCB para los nodos secundarios y tomamos el nodo que maximiza ese valor. Como ninguno de los nodos ha sido visitado todavía, el segundo término es infinito para ambos. Por lo tanto, solo vamos a tomar el primer nodo
* Ahora estamos en un nodo hoja donde necesitamos verificar si lo hemos visitado. Como resultado, no lo hemos hecho. En este caso, sobre la base del algoritmo, hacemos un despliegue hasta el estado terminal. Digamos que el valor de este lanzamiento es 20



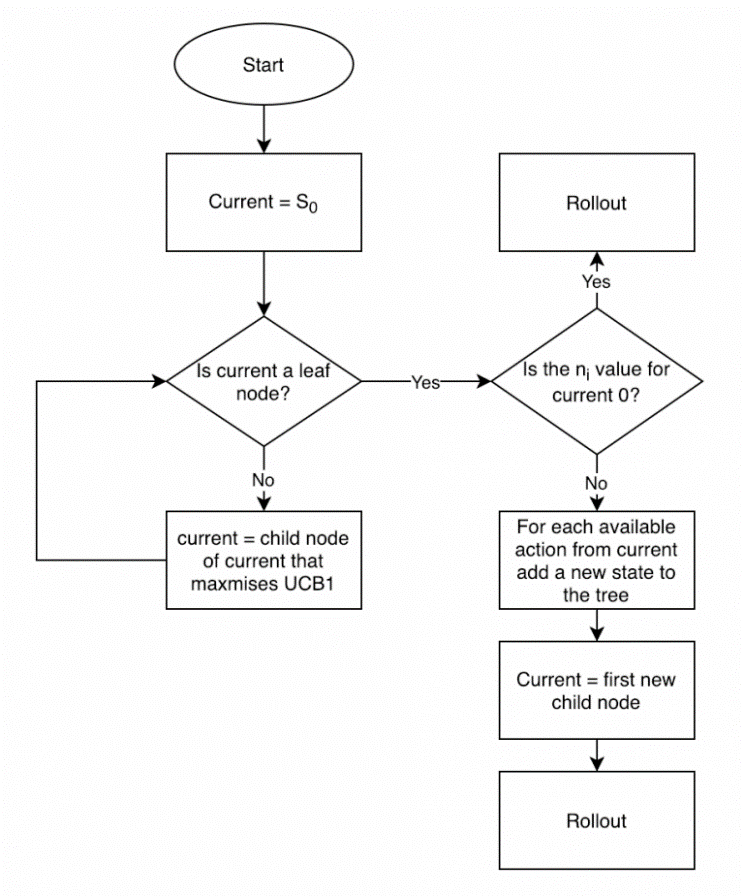
* Ahora viene la cuarta fase, o la fase de retropropagación. El valor del nodo hoja (20) se retroproduce hasta el nodo raíz. Entonces, t = 20 y n = 1 para los nodos S1 y S0



* Ese es el final de la primera iteración

La forma en que funciona MCTS es que lo ejecutamos durante un número definido de iteraciones o hasta que se nos acaba el tiempo. Esto nos dirá cuál es la mejor acción en cada paso que uno debe tomar para obtener el máximo rendimiento.

**Pseudo-Código**



III. CONCLUSIÓN

El desarrollo de este programa fue un gran avance para la inteligencia artificial y las redes neuronales, ya que solamente con información básica y con el mismo algoritmo se puede atacar diferentes problemas o situaciones.

En cuanto al algoritmo de Monte-Carlo, este también fue de gran relevancia puesto que permitió tomar decisiones más rápido y eficientemente, sin embargo, Alpha Zero va mucho más allá, con algoritmos más complejos y otras facetas como la red neuronal que lo compone, algo que esta fuera del alcance de este paper. Empero el algoritmo de búsqueda Monte Carlo sigue siendo el núcleo de esta revolución.

REFERENCIAS.

[1]<https://www.analyticsvidhya.com/blog/2019/01/monte-carlo-tree-search-introduction-algorithm-deepmind-alphago/>

[2] <https://chess24.com/es/informate/noticias/es-alphazero-realmente-un-gran-avance-cientifico-en-el-terreno-de-la-inteligencia-artificial>

[3]<https://vision.unipv.it/IA1/aa2009-2010/ProgrammingaComputerforPlayingChess.pdf>

[4] <https://www.unocero.com/software/alphazero-podria-cambiar-el-mundo-del-ajedrez/>