Universidad de la Habana, MATCOM
Ingeniería Artificial (2022)

# Proyecto de búsqueda adversarial

Miguel Alejandro Asin Barthelemy C511
José Carlos Hernández Piñera C511

## Resumen

Como parte del estudio de la ingeniería artificial, se define el problema de implementar un algoritmo capaz de tomar desiciones factibles en el juego conocido como Hex. Para ello se hará uso de la ya estudiada búsqueda adversarial, con la cual el jugador (artificial) deberá de ser capaz de realizar jugadas asumiendo en todo momento que el contrario u oponente realizará acciones que serán lo menos favorables posibles para el objetivo que se desea alcanzar, que en este caso, es ganar el juego. Este es un claro ejemplo de minimax, sin embargo debido a la gran variedad de opciones que presenta cada estado en el juego, es necesario realizar optimizaciones en el mismo, realizar podas ramas innecesarias y definir que tan favorable es un estado del juego. Finalmente se obtendrá un jugador que será capaz de definir un conjunto de operaciones necesarias para lograr conseguir la victoria en el juego.

## Introducción

La tarea fundamental del jugador artificial es la de ser capaz de definir un conjunto de pasos o algoritmo encaminados a alcanzar (o acercarse en cierta medida) la victoria en el juego. Sin embargo, debido a las dimensiones que podría tener el tablero de juego, las cantidad de opciones posibles que tiene un jugador en cada turno y el corto lapso de tiempo que es neceario para dar una respuesta suficientemente efectiva, descartan opciones muy conocinas y muy utilizadas como la fuerza bruta o depth first search (DFS). Es necesario entonces inclinarse por nuevas técnicas que minimicen los problemas anteriores, pero que aun permitan conducir a la victoria al jugador, una de ellas es la búsqueda adversarial.

## Problema

El hex es un juego entre dos jugadores que van colocando por turnos fichas sobre un tablero romboidal, compuesto de casillas hexagonales ( generalmente de 10 por 10, 11 por 11 hexágonos, o mayores tamaños). Las fichas se distinguen por su color, asociándose uno a cada jugador, y gana quien consigue formar una línea de sus fichas que conecte dos laterales opuestos del tablero previamente asignados. Se clasifica como juego de tablero abstracto.

Reglas:

- Inicialmente el tablero está vacío. A cada jugador se le asigna un color de fichas y dos laterales opuestos del tablero que tendrá que intentar conectar con sus fichas siguiendo las reglas del juego.
- Los jugadores van colocando fichas por turnos sobre el tablero en casillas desocupadas.

- Gana el primer jugador que consigue formar una línea de sus fichas que conecte sus dos laterales. No son posibles los empates.
- Como el primer jugador tiene ventaja, se jugarán 2 partidas una con cada color de fichas.

# Algoritmo

## Modelación

Para implementar al jugador artificial, se usará el algoritmo de minimax. A partir de este algoritmo se crea un árbol en el que cada nodo representa un estado del juego cuyo valor indica que tan favorable o desfavorable es el desenvolvimiento de la partida a partir del estado actual y las aristas indican la acción a realizar para obtner el siguiente estado del juego. Se define un camino del nodo raíz del árbol hacia alguno de sus descendientes como el conjunto de jugadas que realiza cada jugador hasta llegar al estado que representa el nodo del extremo del camino. Una jugada consiste es colocar una pieza en una casilla, sin importar el color de la misma y los jugadores son quienes colocan la pieza en dicha casilla.

Una búsqueda minimax es una búsqueda exhaustiva en profundidad del árbol que se menciona arriba que devuelve un valor "score". Una búsqueda minimax tiene dos fases llamadas fase de maximización y fase de minimización, respectivamente. La fase de maximización ocurre en todas las posiciones del tablero donde el primer jugador tiene el turno y la fase de minimización ocurre en todas las posiciones del tablero donde el segundo jugador tiene el turno. La fase de maximización devuelve el mayor score asignado a los sucesores mientras que la fase de minimización devuelve el valor más pequeño asignado a los sucesores.

Al buscar el árbol completo se crea un jugador que nuca pierde pues conoce completa todos los resultados posibles. Sin embargo, debido a la cantidad de nodos que serían necesario analizar ($n^{2n^2}$), se desecha inmediatamente la idea anterior y se opta por implementar una poda alpha-beta en el árbol minimax que permita eliminar ramas de desición innecesarias.

El algoritmo de poda alpha-beta es una extensión de la búsqueda por minimax que tiene dos valores alpha y beta que atan el 'score' en cada nodo o estado del juego. El valor de alpha es el menor valor atado al nodo el cual es maximizado en la búsqueda mientras que beta es el mayor y es minimizado en la búsqueda.

## Heurística

Al construir el árbol de minimax, es necesario definir la altura del mismo (o profundidad), pues la cantidad de jugadas necesarias para finalizar una partida no siempre es la misma. Al establecer una altura, las hojas del mismo denben devolver un valor asociado a la puntación del estado del tablero que se alcanza realizando el conjunto de jugadas correspondientes. Para ello es necesario definir entonces una función heurística que permita valorar que tan favorable es para un jugador un estado dado del juego.

Una idea inicial para afrontar el problema de construir la heurística consiste en contar la cantidad de piezas de un mismo jugador que están conectadas; para ello se almacena en un *set* las casillas ocupadas por el jugador, luego para cada una, se buscan los vecinos que también sean ocupadas por el jugador.

Despues de obtener la cantidad de casillas conectadas pertenecientes a cada jugador, se define entonces la función heurística como la diferencia entre las casillas conectadas por ambos jugadores (como dicho valor puede tener un valor absoluto mayor que 1, se normaliza dividiendo entre el máximo de ambas cantidades).

Con dicha heurística se establece una función que permite valorar un estado dado del tablero en cada hoja del árbol de desición para posteriormente establecer un valor más exacto que el minimax para cada nodo de dicho árbol.

## Conclusiones

Al desarrollar el algoritmo para la inteligencia artificial, es posible apreciar aspectos importantes: existen situaciones para las cuales independientemente de la heurística empleada, el jugador artificial no estará preparado. La desventaja que presenta la búsqueda adversarial es que asume en todo momento que el jugador contrario optará por la opción más favorable para él o menos favorable para nosotros. De ahí que jugadores con movimientos "random" pueden tener opoortunidad de derrotar al jugador artificial que utilice búsqueda adversarial. Aún así los resultados obtenidos por el algoritmo implementado son bastante favorables mejorables aún más con valores suficientementes grandes para la altura del árbol minimax (dentro de los parámetros adecuados).

## Recomendaciones

Existen criterios que son posibles modificar con el fin de obtener mejores resultados o mejores tiempos. Uno de ellos es la forma de calcular la puntuación para un estado del juego (Heurística). En ella se calcula el número de piezas conectadas por jugador, sin embargo una variante de la misma podría ser tomar la distancia entre el menor y mayor índice de dos piezas que estén conectadas, tal que la mejor puntuación fuese $n$, es decir la longitud entre los lados a conectar.