Proyecto Análisis Algoritmos: Diseño y Análisis

Gustavo Andrés Méndez Hernandez
Daniel Hamilton-Smith Santa Cruz
Juan Diego Osorio Hernández
Juan Felipe marin florez

Abril 2019

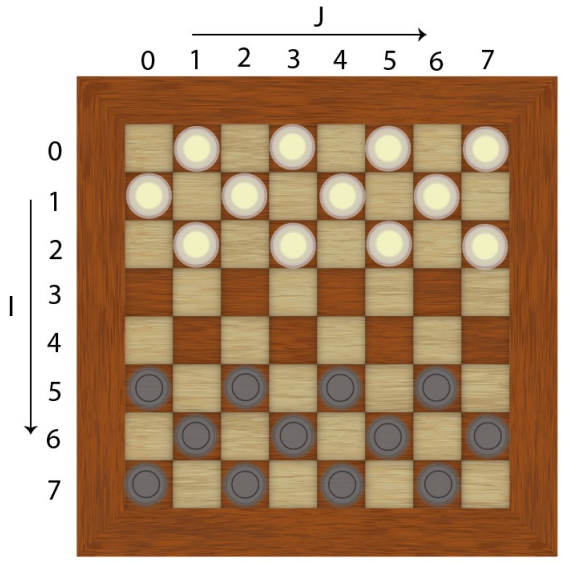
1. Introducción

El proyecto a ser presentado consistirá en el diseño e implementación de un algoritmo que encuentre la mejor jugada para realizar en un momento determinado de un juego de damas.

2. Reglas Del juego

2.1. Inicio Del juego

- Se dispone de un tablero de casillas cuadradas de 8x8 con filas y columnas enumeradas de 0 a 7.
- Se disponen de fichas blancas en todas las casillas cuyas coordenadas en el tablero sumen un numero par, y cuya fila se encuentre entre 0 y 2 (incluidos).
- Se disponen de fichas negras en todas las casillas cuyas coordenadas en el tablero sumen un numero par, y cuya fila se encuentre entre 5 y 7 (incluidos).
- Al inicio del juego todas las fichas son sencillas.
- A cada jugador se le asigna aleatoriamente un color de fichas para jugar.
- El jugador de las fichas blancas realiza el primer movimiento.



2.2. Movimientos

Cada jugador realiza un movimiento por turno

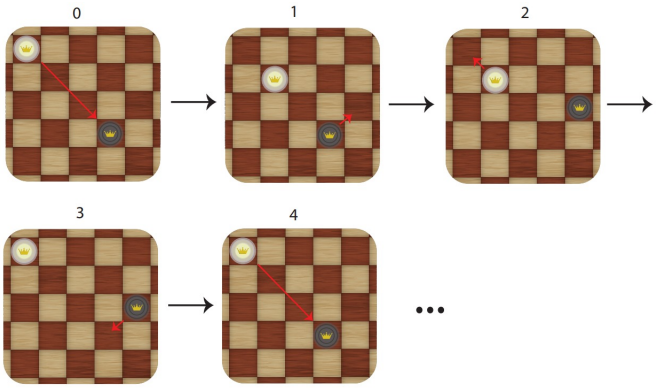
- Para las fichas simples solo se permiten movimientos diagonales a una casilla en dirección al oponente.
- Las fichas coronadas pueden moverse en diagonal a una casilla, en cualquier dirección.
- Una ficha puede moverse en una diagonal de dos casillas si la casilla final a la que se mueve esta libre y si en la casilla intermedia se encuentra una ficha del oponente. Cuando esto sucede se considera un movimiento de captura y la ficha del oponente se retira del tablero.
- Si se puede realizar un movimiento de captura se debe realizar un movimiento de captura(si hay múltiples movimientos de captura el jugador elige cual realizar).
- Una ficha puede realizar múltiples movimientos de captura si al terminar un primer movimiento de captura se encuentra en posición para realizar un segundo. Si se da la oportunidad de que esto se haga el jugador debe hacerlo(si se tienen múltiples movimientos de captura el jugador elige cual realizar).

2.3. Coronación

Sucede cuando una ficha simple llega al extremo opuesto del tablero, en este caso esta ficha pasa de ser simple a ser coronada.

2.4. Escenarios de victoria

- Si al finalizar un turno el oponente no tiene fichas sobre el tablero, este se considera derrotado.
- Si en el turno de un jugador este no puede realizar ninguna jugada valida este se considera derrotado.
- Con el fin de reducir la cantidad de empates: Si se repite un mismo ciclo de jugadas mas de 5 veces gana el jugador con mas fichas. Si ambos jugadores tienen el mismo numero de fichas, es empate. Un ejemplo de ciclo es el siguiente:



- Realizar cualquier jugada no valida resultara en una derrota y la victoria para el adversario.
- Si un jugador no juega en el tiempo establecido(determinado de antemano) este se considera derrotado y se le concede la victoria al oponente.

3. Definición del problema y entregable

3.1. Definición del problema

El problema, informalmente, se define como: Dado el estado de un tablero, las jugadas posibles y el turno de un jugador determinar cual de las jugadas incrementa la probabilidad de obtener la victoria para el jugador cuyo turno se juega.

Formalmente, se dice que: Dada una matriz de 8x8 que define el estado del tablero, cuyos contenidos son fichas validas o espacios libres, jugadas que se define como un vector de posiciones en el tablero que comienza con la posición inicial de la pieza y sigue con las posiciones que tomara en la jugada, y por ultimo el jugador que esta jugando deberá encontrar una jugada que maximice sus posibilidades de ganar. Ahora, la definición del contrato sería:

- **Entradas**:
  - Una matriz M de elementos $a_{i,j} \in \mathbb{E}^{8*8}$, donde:
    $\mathbb{E} = \{libre, Negra, Blanca, NegraCoronada, BlancaCoronada\}$
  - Un vector de jugadas P donde:
    $P = \langle p_1, p_2, \cdots, p_n \rangle$ y $p_k = \langle (i_1, j_1), (i_2, j_2), \cdots, (i_m, j_m) \rangle$
  - Un jugador $Q \in \{Negras, Blancas\}$
- **Salidas**: Una jugada $p = \langle (i_1, j_1), (i_2, j_2), \cdots, (i_m, j_m) \rangle \in P$ tal que la probabilidad de victoria se maximice.

3.2. Entregable

Para la entrega del proyecto se proveerá un programa python "checkers.py" que sera el arbitro entre jugadores, para cuyo funcionamiento se debe crear una librería que contenga una función "player". Dos de estas librerías se usaran para ejecutar el siguiente comando en terminal

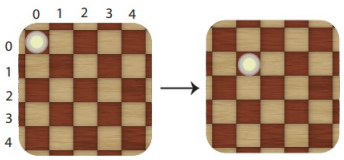
```
python checkers.py <libreria1.py> <libreria2.py> [tiempo]
```

Esto resultara en la simulación de un juego entre los jugadores que describen las librerías. El ultimo parámetro "tiempo" determina de cuanto tiempo disponen los jugadores(en segundos) para realizar sus jugadas, si no se da este parámetro, los jugadores dispondrán de tiempo ilimitado.

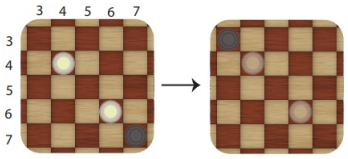
Para la entrega del proyecto se debe realizar dicha función en python esta debe recibir como parámetros:

- **E(estado)**:una matriz de caracteres de 8x8 con posibles valores 'x', 'X', 'o', 'O', '-'. Donde 'x' y 'X' representan las fichas blancas, y 'o' y 'O' las negras.
- **M(jugadas)**: un vector de jugadas. Donde una jugada es un vector de posiciones en el tablero y una posición en el tablero una tupla de enteros entre 0 y 7. Por ejemplo: M=[[(0,0), (0,1)] , [(7,7), (5,5), (3,3)]], lo que significa que la pieza en (0,0) puede moverse a la posición (0,1), y que la ficha en (7,7) puede moverse a (5,5) y luego a (3,3), tomando dos piezas. Ejemplos de jugada:

```
[(0,0),(0,1)]
```




```
[(7,7), (5,5), (3,3)]
```



- **J(jugador)**: que puede ser 'X' o 'O', dependiendo de que fichas esta usando el jugador.

Y retornar la jugada que se considera óptima según el algoritmo implementado. Esto acabaría en el siguiente prototipo de función:

```
def player(E,M,J):
    ...
    return m #donde m es una jugada
```

3.3. Programa

El programa "checkers.py" espera recibir como parámetro dos archivos de python que definan el comportamiento de los jugadores mediante la firma de función descrita anteriormente. Es importante saber que el programa considera el jugador 'X' como las fichas blancas. Además de esto este programa servirá como el arbitro entre ambos jugadores.

3.4. Condiciones de entrega

- El programa deberá correr en Ubuntu, sobre los computadores de la sala B.
- Las librerías deben tener un nombre único entre grupos, para asegurar esto, estas deben tener como nombre de archivo el apellido de todos los integrantes del grupo separados por '_' seguido de la extensión .py.
- Debe ser compatible con python 2.7.
- El tiempo del que disponen los jugadores para jugar sera determinado por el profesor.