Universidad Simón Bolívar
CI-5437 Inteligencia Artificial I
Septiembre-Diciembre 2016

# Proyecto 2: Implementación y evaluación de algoritmos de juegos para una versión reducida del juego de othello.

## Actividad 1.

Al correr la variación principal del juego se notó que ocuría un error debido a que la función *outflank* no podía determinar una posición donde jugar ya que no se encontraba la verificación correspondiente a las diagonales del tablero, esto porque el método *move* no contemplaba dichas posiciones.

Como solución, se agregaron los recorridos en ambos sentidos para las diagonales en los métodos mencionados. Estos cambios fueron añadidos al archivo *othello_cut.h*.

## Actividad 2.

Haciendo uso del conjunto de métodos del archivo *othello_cut.h*, se implementaron en el archivo *algorithmos_juego.hpp* los siguientes algoritmos de decisión, sin implementar tablas de transposición:

- Minmax/maxmin doblemente recursivo para calcular el valor del juego.
- Versión negamax de minmax/maxmin.
- Versión negamax de minmax/maxmin con poda alpha-beta.
- Scout.
- Negascout = negamax con poda alpha-beta + scout.

**Actividad 3.**

Se realizaron las corridas de los 5 algoritmos durante 10 minutos cada uno obteniendo los resultados por niveles para cada algoritmo, estos datos se encuantran reflejados en la carpeta *Resultados*.

A continuación, se muestra una tabla con los resultados relevantes de las corridas:

| **Algoritmo** | **Nivel Alcanzado** | **Tiempo (Segundos)** | **Nodos Generados por Segundo** | **Nodos Expandidos** |
|---|---|---|---|---|
| Minimax-Maxmin | 18 | 257.652 | 2.57129e+06 | 522894086 |
| Negamax (Minimax-Maxmin) | 18 | 298.76 | 2.56959e+06 | 625084814 |
| Negamax (Poda Alpha-Beta) | 13 | 154.568 | 2.44894e+06 | 315074162 |
| Scout | 12 | 282.156 | 4082.93 | 693495 |
| Negascout | 12 | 239.06 | 2.36379e+06 | 477003110 |

Con estos resultados, se determinó que el mejor algoritmo de decisión para grafos de juegos es el Negascout, ya que logró llegar más lejos sobre la variación principal, alcanzando el nivel 12 de profundidad, seguido por los algoritmos Scout, que tambi'en alcanzó el nivel 12 pero en un tiempo mayor, por otro lado, el algoritmo Negamax con Poda Alpha-Beta, llegó al nivel 13 de profundidad. Por últimos en el orden de mejor a peor, se encuentran los algoritmos Negamax de Minimax-Maxmin y Minimax-Maxmin que lograron alcanzar sólo el nivel 18 de profundidad.

Estos resultado se deben al orden de complejidad de estos algoritmos, pues los algoritmos Scout, Negascout y Negamax con poda alpha-beta tienen un orden de complejidad de O(b^(d/2)), mientras que Negamax para Minimax- Maxmin y Minimax-Maxmin tiene un orden es de O(b^d), donde b es el factor de ramificación, t d es la profundidad.