Lenguajes de Programación 2 – Profesor: Sergio Aranda

Trabajo: BACKGAMMON

## IMPLEMENTACIÓN DEL TABLERO Y LAS FICHAS

La representación del tablero la hicimos en un vector de 26 char, más un vector de 2 char para la barra.

El primer vector contiene al tablero, con todas las pilas, y las dos casas en los extremos (el índice determina la pila). El segundo vector corresponde a la barra para cada jugador. Esos vectores adquieren valores correspondientes a las fichas

#define BLACK -1

#define WHITE 1

La manera de representar las fichas es mediante dos números, el 1 y el -1, eso permite contar fácilmente la cantidad de fichas de una pila del tablero, y facilita mucho el trabajo de añadir o quitar fichas (desplazarlas)

Las funciones asociadas al tablero están ligadas a su vez a la impresión en la pantalla del mismo.

## INTERFAZ

La interfaz gráfica del programa fue hecha en allegro, utilizando C++ como lenguaje base (sólo para la interfaz, el resto del programa fue programado en C).

Cada ficha es un objeto con la capacidad de reconocer el click del mouse en caso de ser necesario, y con las propiedades de impresión necesarias (imagen que debe imprimirse y posición en la pantalla).

# INTELIGENCIA ARTIFICIAL

Sin duda el reto más grande para nosotros fue la implementación de la inteligencia artificial del programa, que cuenta con dos partes principales: la función de evaluación de tableros, y la función de selección de jugadas. Además de las partes secundarias: funciones de traducción y funciones de valorización extra a la función de evaluación.

### Selección de Jugadas

Nuestro equipo NO utiliza el corte alfa-beta que fue propuesto en clases, ya que luego de conocer bien las reglas del juego y jugar un número importante de partidas tanto contra inteligencias especiales variadas como contra otros seres humanos, decidimos que el corte alfa-beta en casos con mucho azar, tiene una complejidad que (dado el límite de tiempo de 15 segundos establecido en el trabajo) no genera una ventaja muy apreciable en el estado del juego, ya que la profundidad alcanzable no aprovecha los cortes de la implementación alfa-beta.

Inicialmente planteamos una solución recursiva a este problema, ya que la implementación era más simple, pero luego decantamos por la utilización de la solución iterativa, ya que nos permitía mayor control sobre las situaciones especiales (turnos dobles, fichas en la barra, un solo dado disponible, etc).

El código adjunto a este informe cuenta con la función ai\_selection completamente comentada y con el razonamiento claramente explicado.

De manera simple podemos decir que se generan todas las jugadas posibles que tiene nuestro programa, se aplica esa jugada a un tablero auxiliar, y se evalúa con la función E de evaluación de tableros, seleccionando el mayor puntaje mediante una simple competencia de puntajes en la que va quedando el mayor.

### Función de Evaluación

La función de evaluación de este programa se basa en la estrategia Priming de Backgammon, que consiste en tratar de formar una barrera al enemigo, dejando sus fichas retrasadas sin posibilidad de escapar, y obteniendo una ventaja doble a la hora de comer una ficha, ya que retrasamos al oponente y lo obligamos a perder jugadas sin poder escapar del prime formado.

A la vez se implementa otra estrategia, que trata en lo posible de no dejar blots inseguros (pilas con una sola ficha nuestra, que el enemigo puede atacar sin mucha dificultad).

En caso de que las fichas se hayan cruzado completamente (ningún jugador puede comer fichas del oponente), simplemente se pasa al modo carrera, en el cual las fichas que terminan en nuestra casa tienen un importante puntaje mayor al resto.

### Otras funciones de la IA

Una característica de nuestra inteligencia artificial es el hecho de que asume que controla las fichas negras a la hora de evaluar los tableros. Esto lo hicimos así para simplificar la programación de la inteligencia artificial. Es por eso que en lugar de dos versiones de la inteligencia artificial, decidimos implementar dos funciones de traducción, una de entrada y otra de salida, que en caso de que a la inteligencia artificial le toque el turno opuesto, simplemente se traduce el tablero de la pantalla a su versión opuesta, al igual que luego de seleccionarse la mejor jugada, ésta se traduce en caso de ser necesario, a una jugada aplicable al tablero real.