INTART

Práctica 4: Juegos

Grupo 2361, Pareja 10 Javier Muñoz y Guillermo Rodríguez

**Introducción**

Para esta práctica se nos pide mejorar una ‘bot’ que juega de forma muy básica al juego de mesa ‘Conecta 4’. El bot que se nos da consta de varias partes principales, las cuales le permiten jugar.

En un primer lugar tiene sensores cardinales, esto es que le permite saber que es lo que ocurre alrededor de la casilla donde está mirando, sabiendo en esa casilla la cantidad de fichas que hay por debajo suya, a su izquierda a su derecha, y en las diagonales arriba derecha, arriba izquierda, abajo derecha y abajo izquierda.

El método de la heurística del bot, es el que decide cual de las siete posibles inserciones de ficha (una por cada columna) es la solución mas viable para aumentar la ventaja en la partida.

Este método calcula para cada una de estas siete posiciones la “puntuación” de la casilla. Esta puntuación puede ser positiva (favorable) o negativa (desfavorable). Y, a la hora de insertar la ficha, escogerá la columna con la casilla con mayor puntuación.

En el método de heurística que nos facilitan, la puntuación se calcula utilizando los sensores cardinales nombrados anteriormente. Para cada una de las siete posibles casillas, primero se obtienen los números de fichas alineadas en todas las direcciones con los sensores. Estos resultados se ponderarían de manera que al tener mas fichas en linea, mas ponderación. Y estos resultados ponderados se guardaran en “puntuación actual”. Esta es la puntuación que expresaría la viabilidad de la casilla a favor de nuestro bot.

Lo mismo sucedería con la “puntuación oponente”. Se volverían a utilizar los sensores pero ahora con las fichas del oponente, y se ponderarían de la misma manera.

Finalmente, la “puntuación” seria el resultado de la resta de “puntuación actual” menos la “puntuación oponente”. Siendo directamente proporcional la puntuación con la ventaja que nos otorgaría la inserción de la ficha en la casilla.

A lo largo de la práctica hemos creado diferentes bots, a los cuales íbamos cambiándoles las heurísticas, haciéndolas a nuestro parecer cada vez mas convenientes para ganar en el juego.

**Heurística 1**

La primera heurística que tuvimos de nuestros jugadores es la básica. En ella se tiene un jugador que va analizando columna por columna del tablero y mapeando cuales son las que mejor le vienen según la heurística definida.

Tanto para ataque como para defensa, la heurística es la misma, cuando encuentra una ficha del mismo color, le da una puntuación de 10, si encuentra 2 del mismo color, 100 y si son 3 se le da una puntuación de 1000, siendo esta la mayor preferencia de nuestra IA, ya que si ha encontrado una secuencia de 3 casillas cuyas fichas son iguales, se intentará colocar la cuarta y así ganar la partida.

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila)))**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000))**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000)))))**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila)))**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

(cond ((= abajo 0) 0)

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000))**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000))))))**

**(- puntuacion-actual puntuacion-oponente)))))**

**Heurística 2**

Para la segunda heurística cambiamos las puntuaciones en el caso de la defensa, para ello aumentamos a 5000 en la parte de defensa, cuando ve que en el caso de que el jugador contrario está a punto de hacer 4 en raya, teniendo ya 3. Esto hace que el jugador tenga mas en cuenta defender los posibles ataques de otros jugadores antes que ser ofensivo y buscar las 4 en raya.

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila)))**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000)))))**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila)))**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

(cond ((= abajo 0) 0)

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 5000))))))**

**(- puntuacion-actual puntuacion-oponente)))))**

Como podemos observar, en sí, el jugador sigue siendo el mismo, pero con esos pequeños cambios previamente mencionados obteníamos una mejora sustancial en nuestro jugador. Debemos mencionar que esta idea nos la proporcionó nuestro profesor de prácticas.

**Heurística 3**

A medida que íbamos buscando maneras de mejorar nuestro bot nos topamos con un “fallo” en los sensores que nos daban de base en el jugador, y es que este no contemplaba las diagonales que se pueden formar en el Conecta-4, para arriba derecha, arriba izquierda, abajo derecha y abajo izquierda.

Para ello utilizamos las funciones que ya venían predefinidas:

* contar-abajo-derecha
* contar-abajo-izquierda
* contar-arriba-derecha
* contar-arriba-izquierda

Como sensores de las diagonales, de tal manera que también se contaban las fichas en las diferentes direcciones.

Este cambio nos proporcionó una nueva visión de como orientar la táctica de nuestro jugador. Las diagonales eran formaciones que si no se ganaba el centro del tablero eran difíciles de obtener, al igual que en algunas otras agrupaciones, es por ello, que empezamos a considerar el hecho de ganar el centro del tablero para así tener mas probabilidades de conexiones con diferentes fichas.

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila)))**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000))**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000))**

**)))**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila)))**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

(cond ((= abajo 0) 0)

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 5000))**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000))**

**))))**

**(- puntuacion-actual puntuacion-oponente)))))**

Como podemos observar, la densidad de código empieza a aumentar considerablemente.

**Heurística 4**

Nuestro profesor nos comentó como funcionaban algunos algoritmos que usaban los programas de IAs para ajedrez. Estos tenían diversos algoritmos en función de el estado de la partida en el que estaban, de tal manera que no jugaban bajo los mismos algoritmos en todas las fases de la partida, interactuando así de forma diferente en la partida.

Después de realizar diferentes pruebas consideramos que dividiríamos en dos tipos las heurísticas, las de apertura, que serían antes del movimiento 12, y las de juego, que eran después. Las heurísticas asociadas a la apertura tenían como funcionamiento principal, ser muy ofensivas, mientras que las heurísticas de juego tenían como fin protegerse de ataques del oponente.

**(let\* ((tablero (estado-tablero estado))**

**(ficha-actual (estado-turno estado))**

**(ficha-oponente (siguiente-jugador ficha-actual)))**

**(if (juego-terminado-p estado)**

**(let ((ganador (ganador estado)))**

**(cond ((not ganador) 0)**

**((eql ganador ficha-actual) +val-max+)**

**(t +val-min+)))**

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador.;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila)))**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(if (< (contar-fichas tablero) 12)**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000000))**

**))**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000000))**

**)))**

**) ;; let-altura**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador oponente. ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila)))**

**(if (< (contar-fichas tablero) 12)**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 5000))**

**))**

**;; end-primer-if**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 5000))**

**))))**

**); let-loop-for**

**(- puntuacion-actual puntuacion-oponente)))))**

Para ello, tuvimos que implementar la función contador. Esta funcion cuanta el numero de fichas de la partida (numero de turnos), y sera llamada antes de calcular las puntuaciones. No obstante, en este metodo se usa la misma heuristica para la apertura, como para el resto del juego (en las siguientes heuristicas implementaremos ponderaciones extra para la apertura).

**(defun contar-fichas (tablero)**

**(let ((contador 0))**

**(loop for fila from 0 below (tablero-alto tablero) do**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(if (not (null (aref (tablero-casillas tablero) fila columna)))**

**(setf contador (+ 1 contador))**

**)**

**)**

**)**

**contador**

**)**

**)**

**Heurística 5**

Como hemos mencionado previamente, a medida que íbamos descubriendo diferentes maneras de detectar situaciones de ofensiva y defensiva, nos dimos cuenta de que si nuestro jugador quería controlar el centro, tenía mas posibilidades de ganar. Esto es por que hay mas probabilidad de que las conexiones que se pueden generar sean con fichas que estén en el centro.

Fue entonces cuando pensamos en realizar las ponderaciones, dando una mayor puntuación a las casillas mas centradas (a nivel de columna y no de fila) en la apertura.

Para ello, a la hora de usar los sensores cardinales para cada una de las columnas, asignábamos a una variable “ponderación” dependiendo de la columna de la casilla a evaluar.

Finalmente, multiplicaríamos tanto nuestra puntuacion como la del oponente por la ponderación, en el caso de que el turno sea inferior a 12 (una vez alcanzado el turno 12, estas ponderaciones desaparecerían ya que el centro del tablero estaría mas lleno y las columnas del centro dejarían de tener mas “valor”.

**(let\* ((tablero (estado-tablero estado))**

**(ficha-actual (estado-turno estado))**

**(ficha-oponente (siguiente-jugador ficha-actual)))**

**(if (juego-terminado-p estado)**

**(let ((ganador (ganador estado)))**

**(cond ((not ganador) 0)**

**((eql ganador ficha-actual) +val-max+)**

**(t +val-min+)))**

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador.;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila))**

**(ponderacion (cond ((or (= columna 0) (= columna 6)) 1)**

**((or (= columna 1) (= columna 5)) 1.1)**

**((or (= columna 2) (= columna 4)) 1.3)**

**((= columna 3) 1.5))))**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(if (< (contar-fichas tablero) 12)**

**(setf puntuacion-actual**

**(\* ponderacion (+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 10000000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 10000000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 10000000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 10000000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 10000000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 10000000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 10000000))**

**)))**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000000))**

**)))**

**) ;; let-altura**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador oponente. ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila))**

**(ponderacion (cond ((or (= columna 0) (= columna 6)) 1)**

**((or (= columna 1) (= columna 5)) 1.1)**

**((or (= columna 2) (= columna 4)) 1.3)**

**((= columna 3) 1.5))))**

**(if (< (contar-fichas tablero) 12)**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(setf puntuacion-oponente**

**(\* ponderacion (+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 15)**

**((= abajo 2) 150)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 15)**

**((= der 2) 150)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 15)**

**((= izq 2) 150)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 15)**

**((= abajo-izq 2) 150)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 15)**

**((= abajo-der 2) 150)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 15)**

**((= arriba-izq 2) 150)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 15)**

**((= arriba-der 2) 150)**

**((= arriba-der 3) 5000))**

**)))**

**;; end-primer-if**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 15)**

**((= abajo 2) 150)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 15)**

**((= der 2) 150)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 15)**

**((= izq 2) 150)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 15)**

**((= abajo-izq 2) 150)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 15)**

**((= abajo-der 2) 150)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 15)**

**((= arriba-izq 2) 150)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 15)**

**((= arriba-der 2) 150)**

**((= arriba-der 3) 5000))**

**))))**

**); let-loop-for**

**(- puntuacion-actual puntuacion-oponente)))))**

**(defun contar-fichas (tablero)**

**(let ((contador 0))**

**(loop for fila from 0 below (tablero-alto tablero) do**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(if (not (null (aref (tablero-casillas tablero) fila columna)))**

**(setf contador (+ 1 contador))**

**)**

**)**

**)**

**contador**

**)**

**)**

Ademas, como se puede observar, cambiamos los valores a la hora de calcular las puntuaciones; tanto las favorables a nuestro bot, como las del oponente.

Dimos un valor de 1000000 (valor suficiente para que si se da, sea la máxima puntuación de las siete casillas posibles) en el caso en el que un sensor cardinal detectase tres fichas en linea a nuestro favor. De esta manera, en el momento en el que nuestro bot detectase tres fichas en linea desde nuestra casilla a evaluar, siempre la rellenaría para ganar la partida.

Por otra parte, también cambiamos los valores del oponente, siendo mas significativos que los nuestros excepto. Por ejemplo, si hubiera dos fichas alineadas en una casilla para nosotros y otras dos en otra casilla para nuestro oponente, rellenaríamos la del oponente ya que sus valores son mayores. Sin embargo, si esto pasara con tres fichas alineadas, rellenaría la que tiene la alineación con nuestras fichas, ya que ganaríamos la partida automáticamente.

**Heurística 6**

Una vez que las ponderaciones fueron implementadas, empezamos a ponderar las distintas acciones que se podían hacer con las ponderaciones, optamos por dos tipos de tácticas, una muy ofensiva y una muy defensiva, la ofensiva tenía como prioridad número uno encontrar las 3 fichas juntas para obtener una puntuación muy alta y por lo tanto realizar esa acción. Por otra parte tenía una ponderación muy alta para colocar las fichas que estuviesen en el centro, de tal manera que su estrategia fuese muchas fichas al centro, buscando las 3 seguidas siempre que pudiese.

El problema que encontramos con esta heurística es que siempre atacaba y nunca defendía, al tener facetas de su juego muy marcadas, no era capaz de reaccionar a lo que el jugador oponente jugaba, de tal manera que, si el jugador oponente era algo equilibrado, las probabilidades de que este bot ganase eran pocas.

**Heurística 7**

Con lo previamente explicado, llegamos al que fue nuestro bot final. Este jugador fue el resultado de diferentes pruebas entre los jugadores que íbamos creando. Modificamos el código para que las diferentes versiones que íbamos creando, tocando valores de heurísticas y ponderaciones, compitiesen entre sí a un modo “rey de la pista” de tal manera que si un jugador ganaba, se quedaba a la espera de que un nuevo contendiente apareciese, y así sucesivamente. Esta fue la fase que mas tiempo nos tomó de la práctica.

El bot “agresivo” que explicamos previamente fue nuestro punto de partida, intentamos que ganase contra jugadores que fuesen equilibrados, equilibrándolo a él también. Para ello diferenciamos mas las dos fases en la que nuestro jugador jugaba, en la primera fase (apertura, la cual ampliamos a 15 movimientos), le hacíamos muy agresivo y que siempre buscase el centro, de tal manera que se apoderase de las zonas donde mas conexiones podía crear. Pese a esto, también contemplaba el caso en el que el rival hiciese una apertura en las columnas mas abiertas, dándole mucha puntuación a las columnas abiertas o a las conexiones de mas de 2 fichas (puntuación de 150) y 3 fichas (puntuación de 5000 frente a las 1000 de la táctica ofensiva). Por otra parte, cuando la parte de apertura finalizaba, el jugador se volvía equilibrado, y su prioridad, junto con las ponderaciones de seguir buscando el centro, era encontrar las 3 fichas y así encajar la 4, ganando así la partida.

**(let\* ((tablero (estado-tablero estado))**

**(ficha-actual (estado-turno estado))**

**(ficha-oponente (siguiente-jugador ficha-actual)))**

**(if (juego-terminado-p estado)**

**(let ((ganador (ganador estado)))**

**(cond ((not ganador) 0)**

**((eql ganador ficha-actual) +val-max+)**

**(t +val-min+)))**

**(let ((puntuacion-actual 0)**

**(puntuacion-oponente 0))**

**(loop for columna from 0 below (tablero-ancho tablero) do**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador.;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-actual columna fila))**

**(der (contar-derecha tablero ficha-actual columna fila))**

**(izq (contar-izquierda tablero ficha-actual columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-actual columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-actual columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-actual columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-actual columna fila))**

**(ponderacion1 (cond ((or (= columna 0) (= columna 6)) 1)**

**((or (= columna 1) (= columna 5)) 1.1)**

**((or (= columna 2) (= columna 4)) 1.3)**

**((= columna 3) 1.5))))**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(if (< (contar-fichas tablero) 15)**

**(setf puntuacion-actual**

**(\* ponderacion1 (+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 1000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 1000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 1000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 1000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 1000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 1000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 1000))**

**)))**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-actual**

**(+ puntuacion-actual**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 10)**

**((= abajo 2) 100)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 10)**

**((= der 2) 100)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 10)**

**((= izq 2) 100)**

**((= izq 3) 5000))**

**;; contemplamos las diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 10)**

**((= abajo-izq 2) 100)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 10)**

**((= abajo-der 2) 100)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 10)**

**((= arriba-izq 2) 100)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 10)**

**((= arriba-der 2) 100)**

**((= arriba-der 3) 5000))**

**)))**

**) ;; let-altura**

**(let\* ((altura (altura-columna tablero columna))**

**(fila (1- altura))**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;; sensores cardinales del jugador oponente. ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;**

**(abajo (contar-abajo tablero ficha-oponente columna fila))**

**(der (contar-derecha tablero ficha-oponente columna fila))**

**(izq (contar-izquierda tablero ficha-oponente columna fila))**

**(abajo-der (contar-abajo-derecha tablero ficha-oponente columna fila))**

**(arriba-izq (contar-arriba-izquierda tablero ficha-oponente columna fila))**

**(abajo-izq (contar-abajo-izquierda tablero ficha-oponente columna fila))**

**(arriba-der (contar-arriba-derecha tablero ficha-oponente columna fila))**

**(ponderacion3 (cond ((or (= columna 0) (= columna 6)) 1)**

**((or (= columna 1) (= columna 5)) 1.1)**

**((or (= columna 2) (= columna 4)) 1.3)**

**((= columna 3) 1.5))))**

**(if (< (contar-fichas tablero) 18)**

**;; caso menor que 12 (heuristica ponderada para apertura de partida)**

**(setf puntuacion-oponente**

**(\* ponderacion3 (+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 15)**

**((= abajo 2) 150)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 15)**

**((= der 2) 150)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 15)**

**((= izq 2) 150)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 15)**

**((= abajo-izq 2) 150)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 15)**

**((= abajo-der 2) 150)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 15)**

**((= arriba-izq 2) 150)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 15)**

**((= arriba-der 2) 150)**

**((= arriba-der 3) 5000))**

**)))**

**;; end-primer-if**

**;; segunda parte de la partida evaluando heuristicas sin ponderar, una vez la apertura termina**

**(setf puntuacion-oponente**

**(+ puntuacion-oponente**

**(cond ((= abajo 0) 0)**

**((= abajo 1) 15)**

**((= abajo 2) 150)**

**((= abajo 3) 5000))**

**(cond ((= der 0) 0)**

**((= der 1) 15)**

**((= der 2) 150)**

**((= der 3) 5000))**

**(cond ((= izq 0) 0)**

**((= izq 1) 15)**

**((= izq 2) 150)**

**((= izq 3) 5000))**

**;; contamplamos diagonales**

**(cond ((= abajo-izq 0) 0)**

**((= abajo-izq 1) 15)**

**((= abajo-izq 2) 150)**

**((= abajo-izq 3) 5000))**

**(cond ((= abajo-der 0) 0)**

**((= abajo-der 1) 15)**

**((= abajo-der 2) 150)**

**((= abajo-der 3) 5000))**

**(cond ((= arriba-izq 0) 0)**

**((= arriba-izq 1) 15)**

**((= arriba-izq 2) 150)**

**((= arriba-izq 3) 5000))**

**(cond ((= arriba-der 0) 0)**

**((= arriba-der 1) 15)**

**((= arriba-der 2) 150)**

**((= arriba-der 3) 5000))**

**))))**

**); let-loop-for**

**(- puntuacion-actual puntuacion-oponente)))))**