Memoria Práctica 3

Pareja 4 Grupo 2302

Santiago Valderrábano Zamorano	Martín de las Heras Moreno
santiago.valderrabano@estudiante.uam.es	martin.delasheras@estudiante.uam.es

---

Índice

1. ab29a
2. a2b3d
3. fc259

---

1. ab29a

En esta heuristica, hemos optado por una version simple de la inicial centrada fundamentalmente en la defensa, como en todas las heuristicas, empieza por comprobar si hay ganador, dando la maxima puntuacion en caso de ganemos nosotros y la minima si gana el oponente. Despues, para el jugador comprueba si tiene 3 fichas seguidas en algun lado suma 3000, en caso contrario no suma nada. En cambio para el oponente se computa que tenga 1, 2 o 3 fichas seguidas, para luego restar las puntuaciones.

```
(defun heuristica (estado)
  ; current player standpoint
  (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))
                     (abajo
                      (contar-abajo
                       tablero ficha-actual columna fila))
                     (arriba
                      (contar-arriba
                       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))
                     (horizontal
                      (+ der izq))
                     (vertical
                      (+ abajo arriba))
                     (diag-des
                      (+ abajo-izq arriba-der))
                     (diag-asc
                      (+ abajo-der arriba-izq)))
                (setf puntuacion-actual
                  (+ puntuacion-actual
                    (if (= vertical 3)
                      3000
                      0)
                    (if (= horizontal 3)
                      3000
                      0)
                    (if (= diag-des 3)
                      3000
                      0)
                    (if (= diag-asc 3)
                      3000
                      0))))
              (let* ((altura (altura-columna tablero columna))
                     (fila (1- altura))
                     (abajo
                      (contar-abajo
                       tablero ficha-oponente columna fila))
                     (arriba
```

```
                    (contar-arriba
                     tablero ficha-actual 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))
                   (horizontal
                    (+ der izq))
                   (vertical
                    (+ abajo arriba))
                   (diag-des
                    (+ abajo-izq arriba-der))
                   (diag-asc
                    (+ abajo-der arriba-izq)))
             (setf puntuacion-oponente
               (+ puntuacion-oponente
                 (cond ((= vertical 0) 0)
                       ((= vertical 1) 20)
                       ((= vertical 2) 200)
                       ((= vertical 3) 7000)
                       (t 0))
                 (cond ((= horizontal 0) 0)
                       ((= horizontal 1) 20)
                       ((= horizontal 2) 200)
                       ((= horizontal 3) 7000)
                       (t 0))
                 (cond ((= diag-des 0) 0)
                       ((= diag-des 1) 20)
                       ((= diag-des 2) 200)
                       ((= diag-des 3) 7000)
                       (t 0))
                 (cond ((= diag-asc 0) 0)
                       ((= diag-asc 1) 20)
                       ((= diag-asc 2) 200)
                       ((= diag-asc 3) 7000)
                       (t 0))))))
       (- puntuacion-actual puntuacion-oponente)))))
```

2. a2b3d

Para esta heuristica, nos hemos basado en el funcionamiento basico del juego. Para ello, hemos ido cogiendo grupos de 4 fichas seguidas en horizontal, vertical, diagonal ascendente y diagonal descendente y guardandolos en 4 variables, una por ficha. Para coger estos grupos hemos practicamente el mismo metodo para todos cambiando los limites de los bucles. Y cada vez que obtenemos un grupo de 4 fichas lo mandamos a evaluar a la funcion secundaria puntuacion-grupo-4.

Esta primera funcion es igual para el fichero con codigo FC259.

```
(defun heuristica (estado)
  (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 0))
        (loop for filaCentro from 0 below (1- (tablero-alto tablero)) do
              (setf puntuacion
                (+ puntuacion
                   (if (eql
                        (obtener-ficha tablero 3 filaCentro) ficha-actual)
                       3
                       0))))
        (loop for columnaV from 0 below (1- (tablero-ancho tablero)) do
              (loop for filaV from 0 to 2 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaV filaV))
                           (ficha2
                            (obtener-ficha
                             tablero columnaV (+ 1 filaV)))
                           (ficha3
                            (obtener-ficha
                             tablero columnaV (+ 2 filaV)))
                           (ficha4
                            (obtener-ficha
                             tablero columnaV (+ 3 filaV))))
                      (setf puntuacion
                        (+ puntuacion
                           (puntuacion-grupo-4
                            ficha1
                            ficha2
                            ficha3
                            ficha4
                            ficha-actual
                            ficha-oponente))))))
        (loop for filaH from 0 below (1- (tablero-alto tablero)) do
              (loop for columnaH from 0 to 3 do
                    (let* ((ficha1
```

```
                        (obtener-ficha
                         tablero columnaH filaH))
                       (ficha2
                        (obtener-ficha
                         tablero (+ 1 columnaH) filaH))
                       (ficha3
                        (obtener-ficha
                         tablero (+ 2 columnaH) filaH))
                       (ficha4
                        (obtener-ficha
                         tablero (+ 3 columnaH) filaH)))
                  (setf puntuacion
                    (+ puntuacion
                       (puntuacion-grupo-4
                        ficha1
                        ficha2
                        ficha3
                        ficha4
                        ficha-actual
                        ficha-oponente))))))
        (loop for filaDD from 3 to 5 do
              (loop for columnaDD from 0 to 3 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaDD filaDD))
                           (ficha2
                            (obtener-ficha
                             tablero (+ 1 columnaDD) (- filaDD 1)))
                           (ficha3
                            (obtener-ficha
                             tablero (+ 2 columnaDD) (- filaDD 2)))
                           (ficha4
                            (obtener-ficha
                             tablero (+ 3 columnaDD) (- filaDD 3))))
                      (setf puntuacion
                        (+ puntuacion
                           (puntuacion-grupo-4
                            ficha1
                            ficha2
                            ficha3
                            ficha4
                            ficha-actual
                            ficha-oponente))))))
        (loop for filaDA from 0 to 2 do
              (loop for columnaDA from 0 to 3 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaDA filaDA))
                           (ficha2
                            (obtener-ficha
                             tablero (+ 1 columnaDA) (+ 1 filaDA)))
                           (ficha3
                            (obtener-ficha
                             tablero (+ 2 columnaDA) (+ 2 filaDA)))
                           (ficha4
                            (obtener-ficha
```

```
                              tablero (+ 3 columnaDA) (+ 3 filaDA))))
                        (setf puntuacion
                          (+ puntuacion
                             (puntuacion-grupo-4
                              ficha1
                              ficha2
                              ficha3
                              ficha4
                              ficha-actual
                              ficha-oponente))))))
          puntuacion))))
```

Esta funcion se encarga de analizar los grupos de 4 fichas le que pasa la funcion principal. La idea es dependiendo de la posicion de las fichas calcular cuan bueno o malo es ese movimiento. Para ello, contamos primero cuantas fichas de cada tipo hay (nuestras, oponente, vacias).

Consideramos despues los siguientes 4 casos:

1. Que haya 4 fichas nuestras -> Sumamos 100
2. Que haya 3 fichas nuestras y 1 vacia -> Sumamos 5
3. Que haya 2 fichas nuestras y 2 vacia -> Sumamos 2
4. Que haya 3 fichas del oponente y 1 vacia -> Restamos 4

Se puede observar que el fc259 y este son variaciones de la forma de valorar los bloques aunque la base es practicamente la misma.

```
(defun puntuacion-grupo-4
  (ficha1 ficha2 ficha3 ficha4 ficha-nuestra ficha-oponente)
  (let ((puntuacion 0)
        (total-vacias 0)
        (total-nuestra 0)
        (total-oponente 0))
    (setf total-vacias
      (+ total-vacias
         (cond ((null ficha1)
                1)
               (t 0))
         (cond ((null ficha2)
                1)
               (t 0))
         (cond ((null ficha3)
                1)
               (t 0))
         (cond ((null ficha4)
                1)
               (t 0))))
    (setf total-nuestra
      (+ total-nuestra
```

```
          (cond ((eql
                  ficha-nuestra ficha1)
                 1)
                (t 0))
          (cond ((eql
                  ficha-nuestra ficha2)
                 1)
                (t 0))
          (cond ((eql
                  ficha-nuestra ficha3)
                 1)
                (t 0))
          (cond ((eql
                  ficha-nuestra ficha4)
                 1)
                (t 0))))
      (setf total-oponente
        (+ total-oponente
          (cond ((eql
                  ficha-oponente ficha1)
                 1)
                (t 0))
          (cond ((eql
                  ficha-oponente ficha2)
                 1)
                (t 0))
          (cond ((eql
                  ficha-oponente ficha3)
                 1)
                (t 0))
          (cond ((eql
                  ficha-oponente ficha4)
                 1)
                (t 0))))
      (setf puntuacion
        (+ puntuacion
          (cond ((= total-nuestra 4)
                 100)
                ((and
                  (= total-nuestra 3) (= total-vacias 1))
                 5)
                ((and
                  (= total-nuestra 2) (= total-vacias 2))
                 2)
                (t 0))
          (if (and
               (= total-oponente 3) (= total-vacias 1))
              -4
            0)))
      puntuacion))
```

3. fc259

Para esta heuristica, nos hemos basado en el funcionamiento basico del juego. Para ello, hemos ido cogiendo grupos de 4 fichas seguidas en horizontal, vertical, diagonal ascendente y diagonal descendente y guardandolos en 4 variables, una por ficha. Para coger estos grupos hemos practicamente el mismo metodo para todos cambiando los limites de los bucles. Y cada vez que obtenemos un grupo de 4 fichas lo mandamos a evaluar a la funcion secundaria puntuacion-grupo-4.

Esta primera funcion es igual para el fichero con codigo A2B3D.

```
(defun heuristica (estado)
  (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 0))
        (loop for filaCentro from 0 below (1- (tablero-alto tablero)) do
              (setf puntuacion
                (+ puntuacion
                   (if (eql
                        (obtener-ficha tablero 3 filaCentro) ficha-actual)
                       3
                       0))))
        (loop for columnaV from 0 below (1- (tablero-ancho tablero)) do
              (loop for filaV from 0 to 2 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaV filaV))
                           (ficha2
                            (obtener-ficha
                             tablero columnaV (+ 1 filaV)))
                           (ficha3
                            (obtener-ficha
                             tablero columnaV (+ 2 filaV)))
                           (ficha4
                            (obtener-ficha
                             tablero columnaV (+ 3 filaV))))
                      (setf puntuacion
                        (+ puntuacion
                           (puntuacion-grupo-4
                            ficha1
                            ficha2
                            ficha3
                            ficha4
                            ficha-actual
                            ficha-oponente))))))
        (loop for filaH from 0 below (1- (tablero-alto tablero)) do
              (loop for columnaH from 0 to 3 do
                    (let* ((ficha1
```

```
                        (obtener-ficha
                         tablero columnaH filaH))
                       (ficha2
                        (obtener-ficha
                         tablero (+ 1 columnaH) filaH))
                       (ficha3
                        (obtener-ficha
                         tablero (+ 2 columnaH) filaH))
                       (ficha4
                        (obtener-ficha
                         tablero (+ 3 columnaH) filaH)))
                  (setf puntuacion
                    (+ puntuacion
                       (puntuacion-grupo-4
                        ficha1
                        ficha2
                        ficha3
                        ficha4
                        ficha-actual
                        ficha-oponente))))))
        (loop for filaDD from 3 to 5 do
              (loop for columnaDD from 0 to 3 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaDD filaDD))
                           (ficha2
                            (obtener-ficha
                             tablero (+ 1 columnaDD) (- filaDD 1)))
                           (ficha3
                            (obtener-ficha
                             tablero (+ 2 columnaDD) (- filaDD 2)))
                           (ficha4
                            (obtener-ficha
                             tablero (+ 3 columnaDD) (- filaDD 3))))
                      (setf puntuacion
                        (+ puntuacion
                           (puntuacion-grupo-4
                            ficha1
                            ficha2
                            ficha3
                            ficha4
                            ficha-actual
                            ficha-oponente))))))
        (loop for filaDA from 0 to 2 do
              (loop for columnaDA from 0 to 3 do
                    (let* ((ficha1
                            (obtener-ficha
                             tablero columnaDA filaDA))
                           (ficha2
                            (obtener-ficha
                             tablero (+ 1 columnaDA) (+ 1 filaDA)))
                           (ficha3
                            (obtener-ficha
                             tablero (+ 2 columnaDA) (+ 2 filaDA)))
                           (ficha4
                            (obtener-ficha
```

```
                              tablero (+ 3 columnaDA) (+ 3 filaDA))))
                        (setf puntuacion
                          (+ puntuacion
                             (puntuacion-grupo-4
                              ficha1
                              ficha2
                              ficha3
                              ficha4
                              ficha-actual
                              ficha-oponente))))))
          puntuacion))))
```

Esta funcion se encarga de analizar los grupos de 4 fichas que le pasa la funcion principal. La idea es dependiendo de la posicion de las fichas calcular cuan bueno o malo es ese movimiento. Para ello, contamos primero cuantas fichas de cada tipo hay (nuestras, oponente, vacias).

Consideramos parcialmente buenas las jugadas en las que haya mas de una ficha de ambos jugadores ya que es un bloque que el otro no va a poder ganar. Por otro lado consideramos muy buenas jugadas aquellas en las que haya fichas nuestras y no del oponente y por ultimo consideramos muy malas las jugadas en las que solo hay fichas del oponente.

Tras muchas pruebas y simulaciones hemos visto que cuando se da un bloqueo, es decir, que hay fichas de ambos jugadores, si es un bloqueo total (2 fichas nuestras y 2 del otro jugador) entonces sumamos 6 puntos en otro caso 11.

Por otro lado si solo hay fichas nuestras en el bloque hacemos la siguiente operacion: #fichas-nuestras#*fichas-nuestras*factorBuenaPos donde el factorBuenaPos = 50/((#fichasAContectar -1)/(#fichasAContectar)) El resultado de esta calculo se lo sumamos a la puntuacion que devolvemos.

Por ultimo, si solo hay fichas del contrario en el bloque calculamos la puntuacion devuelta de la siguiente forma: (#fichas-oponente/#fichasAContectar)*factorMalaPos donde el factorMalaPos = 100/((#fichasAContectar -1)/(#fichasAContectar))

El que el factorBuenaPos y factorMalaPos se diferencien en la constante que dividimos es debido a que creemos que tiene que primar la defensa sobre el ataque aunque luego lo compensemos un poco al multiplicar 2 veces el # de fichas nuestas.

```
(defun puntuacion-grupo-4
  (ficha1 ficha2 ficha3 ficha4 ficha-nuestra ficha-oponente)
  (let ((puntuacion 0)
        (total-nuestra 0)
        (total-oponente 0)
        (total-vacias 0))
```

```
      (setf total-nuestra
        (+ total-nuestra
           (if (eql ficha-nuestra ficha1)
               1
             0)
           (if (eql ficha-nuestra ficha2)
               1
             0)
           (if (eql ficha-nuestra ficha3)
               1
             0)
           (if (eql ficha-nuestra ficha4)
               1
             0)))
      (setf total-oponente
        (+ total-oponente
           (if (eql ficha-oponente ficha1)
               1
             0)
           (if (eql ficha-oponente ficha2)
               1
             0)
           (if (eql ficha-oponente ficha3)
               1
             0)
           (if (eql ficha-oponente ficha4)
               1
             0)))
      (setf total-vacias
        (+ total-vacias
           (if (null ficha1)
               1
             0)
           (if (null ficha2)
               1
             0)
           (if (null ficha3)
               1
             0)
           (if (null ficha4)
               1
             0)))
      (setf puntuacion
        (+ puntuacion
           (if (and (> total-nuestra 0) (> total-oponente 0))
               (if (and (= total-nuestra 2) (= total-oponente 2))
                   6
                 11)
             (if (> total-nuestra 0)
                 (* (* total-nuestra total-nuestra)(/ 50 0.75))
               (if (> total-oponente 0)
                   (* (* (/ total-oponente 4) (/ 100 0.75)) -1)
                 0)))))
      puntuacion))
```