Práctica 4: Conecta 4

Inteligencia Artificial – Curso 2018-2019

Fecha de publicación: 10-04-2019 (revisión: 29-14-2019)
Fecha del torneo final: 03-05-2019, 23:55
Fecha de entrega en Moodle:

- Grupos de los jueves: miércoles 08-05-2019, 23:55
- Grupos de los viernes: jueves 09-05-2019, 23:55

---

1. El juego del Conecta 4

Conecta 4, también conocido como 4 en Línea, es un juego de estrategia abstracta donde los contrincantes disponen de información perfecta.

Las reglas del Conecta 4 son:

1. Dos jugadores introducen fichas en un tablero vertical formado por seis filas y siete columnas con el objetivo de alinear cuatro consecutivas de un mismo color.

2. Cada jugador dispone de 21 fichas de un color, que nosotros representaremos con un `0` o un `1`.

3. Por turnos, los jugadores deben introducir una ficha en la columna que prefieran, siempre que no esté completa, y ésta caerá a la posición más baja.

4. Gana la partida el primero que consiga alinear cuatro fichas consecutivas de un mismo color en horizontal, vertical o diagonal.

5. Si todas las columnas están llenas pero nadie ha hecho una fila, columna o diagonal válida, hay empate.

Por norma general, el primer jugador tiene más posibilidades de ganar si introduce la primera ficha en la columna central. Si lo hace en las contiguas se puede forzar un empate, mientras que si la mete en las más alejadas del centro su rival puede vencerle con mayor facilidad.

2. Implementación de Conecta 4

Tablero

La estructura `tablero` contiene una representación del tablero:

```
(defstruct tablero
   alto
   ancho
   casillas)
```

Las siguientes funciones permiten visualizar, modificar o consultar distinta información de los tableros:

`muestra-tablero` *`tablero`*
`poner-ficha` *`tablero columna ficha`*
`obtener-ficha` *`tablero columna fila`*
`columnas-jugables` *`tablero`*
`altura-columna` *`tablero columna`*
`ganador-tablero` *`tablero`* (suponiendo que se invoque cuando la partida esté terminada)
`dentro-del-tablero-p` *`tablero columna fila`*

El siguiente grupo de funciones permite contar cuántas fichas hay iguales que *`ficha`* en una determinada dirección a partir de unas coordenadas del tablero.

`contar-abajo` *`tablero ficha columna fila`*
`contar-arriba` *`tablero ficha columna fila`*
`contar-derecha` *`tablero ficha columna fila`*
`contar-izquierda` *`tablero ficha columna fila`*
`contar-abajo-derecha` *`tablero ficha columna fila`*
`contar-abajo-izquierda` *`tablero ficha columna fila`*
`contar-arriba-derecha` *`tablero ficha columna fila`*
`contar-arriba-izquierda` *`tablero ficha columna fila`*

Estado

La estructura `estado` representa el estado de la partida, y consta de un turno y un tablero:

```
(defstruct estado
   turno
   tablero)
```

Los estados tienen asociadas las siguientes funciones:

```
acciones-posibles estado
ejecutar-accion estado acción
generar-sucesores estado
juego-terminado-p estado
tablas-p estado
ganador estado
```

Jugador

La estructura `jugador` representa a un jugador de la partida. Su definición es:

```
(defstruct jugador
   nombre
   f-jugador
   f-eval)
```

y consta de los siguientes campos:

1. un nombre, p.ej.:

   ```
   'mi-jugador
   ```

2. una función con el jugador:

   ```
   f-jugador estado &optional profundidad-max f-eval
   ```

3. una función de evaluación:

   ```
   f-eval estado
   ```

   que recibe un estado de la partida y devuelve un valor numérico. Este valor es positivo o negativo dependiendo de si el estado de la partida es favorable o desfavorable **al jugador que tiene el turno**. Existen dos valores predefinidos que pueden usarse en las funciones de evaluación: `+val-min+` y `+val-max+`.

La siguiente función devuelve la ficha siguiente a una ficha dada (`0` → `1` y `1` → `0`):

```
siguiente-jugador ficha-jugador
```

Se proporcionan, además, varias funciones de jugadores:

```
f-jugador-aleatorio estado &optional profundidad-max f-eval
f-jugador-negamax estado &optional profundidad-max f-eval
```

```
f-jugador-humano estado &optional profundidad-max f-eval
```

Una vez definidos unos jugadores como los siguientes:

```
(defvar *jugador-aleatorio* (make-jugador :nombre 'Jugador-aleatorio
                                          :f-jugador #'f-jugador-aleatorio
                                          :f-eval  #'f-eval-aleatoria))

(defvar *jugador-bueno* (make-jugador :nombre 'Jugador-bueno
                                      :f-jugador #'f-jugador-negamax
                                      :f-eval  #'f-eval-bueno))

(defvar *jugador-humano* (make-jugador :nombre 'Jugador-humano
                                       :f-jugador #'f-jugador-humano
                                       :f-eval  #'f-no-eval))
```

se pueden usar en los parámetros `jugador0` y `jugador1` de la siguiente función para que jueguen una partida:

```
partida jugador0 jugador1 &optional (profundidad-max 4)
```

3. Función de evaluación

La función de evaluación heurística tiene como argumento un estado de juego y devuelve un valor numérico que indica la confianza de que el jugador que tiene el turno gane la partida:

```
(defun funcion-de-evaluacion (estado) ...)
```

El valor devuelto puede ser positivo o negativo.

Se dan implementadas dos funciones de evaluación: una que asigna un valor aleatorio a un estado y otra que tiene en consideración algunas configuraciones de fichas en línea y que puede servir de base para que el alumnos desarrolle sus propias funciones de evaluación:

```
f-eval-aleatoria estado
f-eval-bueno estado
```

La función de evaluación no debe generar estados sucesores, puesto que de eso se encarga el algoritmo MiniMax. No se permite usar funciones que pretendan modificar la mecánica del juego. Se tendrá en cuenta la eficiencia de la función de evaluación.

4. Torneo

Formato del fichero para subir al torneo:

```
(defpackage :3012_P01_5e4rf ; se declara un paquete con el grupo, la pareja y
                            ; el código
  (:use :common-lisp :conecta4)  ; el paquete usa common-lisp y conecta4
  (:export :heuristica :*alias*)) ; exporta la función de evaluación y un alias
```

```
(in-package 3012_P01_5e4rf)

(defvar *alias* '|Soy el ganador|)  ; alias que aparece en el ranking

(defun heuristica (estado) ...)      ; función de evaluación heurística

(defun ...) ; funciones auxiliares usadas por la función heuristica
```

Los códigos serán proporcionados por los profesores de prácticas. La página para subir ficheros al torneo es http://150.244.56.96/intart.

IMPORTANTE: El tornero se ejecutará en SBCL, los archivos que se suban tendrán que estar en UTF-8.

5. Evaluación de la práctica

Esta práctica está compuesta por dos partes.

Parte 1 (7 puntos)

Consiste en la implementación y presentación de jugadores a un torneo entre todas las parejas de la asignatura.

Como resultado de los enfrentamientos entre los jugadores se publicará una clasificación, accesible mediante un enlace que se mostrará en la página del torneo. Esta clasificación será actualizada de manera regular con el fin reflejar la incorporación al torneo de nuevos jugadores. En cierto momento se introducirán de manera anónima jugadores suministrados por los profesores, cuyo objetivo es servir de indicadores de nivel de juego.

La nota de la parte 1 se calculará de la siguiente manera:

- Participación anticipada en el torneo: 3 puntos
- En función del resultado del mejor de los 3 jugadores presentados por cada pareja en el momento de cierre del torneo en relación con los jugadores de referencia, se puntuará:
    - jugador que supera a 5 de los jugadores de referencia: 4 puntos
    - jugador que supera a 4 de los jugadores de referencia: 3 puntos
    - jugador que supera a 3 de los jugadores de referencia: 2 puntos
    - jugador que supera a 2 de los jugadores de referencia: 1 punto
    - jugador que supera a 1 de los jugadores de referencia: 1/2 punto
- Se concederá un premio adicional de 1 punto a los 5 jugadores que ocupen las primeras posiciones de la clasificación final.
- Es obligatoria la participación en el torneo con al menos un jugador. No presentar jugadores, haber sido descalificados por errores de forma o compilación, o por no reunir los mínimos requisitos de calidad o por cualquier otro motivo: 0 puntos

Para obtener los 3 puntos por participación anticipada habrá que subir:

- tres jugadores el 11 de abril los grupos del jueves y el 12 de abril los grupos del viernes
- al menos dos jugadores los días 22, 23, 24, 29 y 30 de abril.

~~Durante los días por participación anticipada se habilitan subidas hasta las 9:00 de la mañana del día siguiente, y los resultados se publican a las 13:00 de ese mismo día. Se podrán subir hasta tres funciones en cada torneo. El cierre del torneo se realizará el 03-05-2019 a las 13:00. Las clasificaciones diarias serán publicadas en un enlace accesible desde la página de entrega.~~Los jugadores para el torneo final habrá que subirlos el 03-05-2019 antes de las 23:55.

Parte 2 (3 puntos)

Consiste en una memoria con una descripción de las heurísticas implementadas que se subirá a Moodle junto con el código de la parte 1.

Para la entrega en formato electrónico, se creará un archivo ZIP que contenga todo el material de la entrega, cuyo nombre, todo él en minúsculas y sin acentos, tildes, o caracteres especiales, tendrá la siguiente estructura:

[gggg]_p4_[mm]_[apellido1]_[apellido2].zip

donde

[gggg] : Número de grupo: (2301, 2311, 2312, etc.)
[mm] : Número de orden de la pareja con dos dígitos (01, 02, 03, etc.)
[apellido1] : Primer apellido del miembro 1 de la pareja
[apellido2] : Primer apellido del miembro 2 de la pareja

Los miembros de la pareja aparecen en orden alfabético. Ejemplos :

- 2311_p3_01_delval_sanchez.zip (práct. 3 de la pareja 01 en el grupo de prácticas 2311)
- 2362_p3_18_salabert_suarez.zip (práctica 3 de la pareja 18 en el grupo de prácticas 2362)

[]