Práctica final: conecta4
Estructura de Datos

Antonio Coín Castro Miguel Lentisco Ballesteros
José María Martín Luque

26 de enero de 2017

Índice

1. Consideraciones de diseño

1.1. Modificaciones realizadas al código proporcionado

Hemos modificado la clase `tablero` añadiendo la siguiente funcionalidad:

- Una variable en tablero llamada `ult_col` que almacena la última columna en la que se insertó una ficha.
- Una función llamada `int get_elemento(int i, int j)` que devuelve qué es lo que hay en la posición indicada.

1.2. Funciones de ayuda

Hemos diseñado una función, `cantidadAlineadas`, que cuenta cuántas alineaciones de n fichas hay en el tablero que se le proporcione, para el jugador que insertó la ficha en último lugar. Como nota, decir que esta función ha sido añadida de última hora y somos conscientes de que podría facilitar la programación de otros métodos. Sin embargo, estos métodos estaban ya implementados cuando decidimos incorporar esta función y no disponemos del tiempo necesario para reescribir el código.

1.3. Código descartado

En el código final, la función `generarHijos`, que se encarga de crear los hijos de un tablero hasta la profundidad deseada, era una función recursiva en vez de la versión iterativa que utilizamos ahora.

```
JugadorAuto::generarHijos(ArbolGeneral<Tablero>::Nodo& n, int profundidad)
{
  if (profundidad)
  {
    Tablero original = partida.etiqueta(n);
    int num_cols = original.GetColumnas();

    for (int col = 0; col < num_cols; ++col)
    {
      if ((original.hayHueco(col) > -1) && !original.quienGana())
      {
        // Crear nuevo tablero y hacer movimiento
        Tablero nuevo(original);
        nuevo.colocarFicha(col);
        nuevo.cambiarTurno();
        ArbolGeneral<Tablero> hijo(nuevo);

        // Crear hijos del nuevo nodo
        ArbolGeneral<Tablero>::Nodo raiz = hijo.raiz();
        generarHijos(raiz, profundidad - 1);
        // Enganchar el nodo al árbol
        partida.insertar_hijomasizquierda(n, hijo);
      }
    }
  }
```

```
}
```

Finalmente se desechó este código porque nos daba algunos problemas, aunque la versión iterativa parece ser más lenta (desconocemos el por qué).

1.4. Profundidad del árbol generado

Hemos considerado que la profundidad más adecuada del árbol de tableros es 5, puesto que con profundidades mayores el programa tarda más tiempo en generar el árbol, ralentizando el inicio y el desarrollo del juego.

2. Métricas utilizadas

2.1. Métrica 4

Bajo esta métrica, el jugador automático insertará la ficha en una columna que tenga hueco, elegida aleatoriamente.

2.2. Métrica 3

Utilizando esta métrica, el jugador automático seguirá los siguientes pasos para elegir la columna en la que insertar:

1. Primero intentará buscar una columna en la que insertando su ficha puede ganar.

2. En caso de que no exista dicha columna, buscará columnas de tal forma que al insertar su ficha, se generen tableros en los que el jugador humano no pueda ganar en ese turno, insertándola en una comlumna aleatoria que cumpla dichas condiciones.

3. Si no existe ninguna columna como la descrita en el paso anterior, el jugador humano podrá ganar sin importar dónde insertemos la ficha, por lo que insertamos la ficha en la primera columna en la que sea posible.

Exploramos un segundo nivel del árbol con el fin de evitar programar una función que compruebe si el jugador humano tiene tres fichas alineadas y un hueco libre donde insertar una cuarta que le haga ganar la partida. Estamos primando la **simplicidad** frente a la **eficiencia**.

2.3. Métrica 2.

En esta métrica, el jugador automático se encarga de asignar a cada subárbol que cuelga de cada uno de los hijos del tablero actual una puntuación. Las puntuaciones se asignan de la siguiente forma:

- Por cada tablero en el que el jugador automático pierde, la puntuación disminuye 2 puntos.
- Por cada tablero en el que el jugador automático empata, la puntuación aumenta 1 punto.
- Por cada tablero en el que el jugador automático gana, la puntuación aumenta 2 puntos.

Se tiene en cuenta también el nivel en el que se encuentran cada uno de estos tableros para ajustar la puntuación.

El jugador automático introducirá la ficha en la columna correspondiente al tablero cuyo subárbol tenga más puntuación.