



**DECSAI**

**Departamento de Ciencias de la Computación e I.A.**

Universidad de Granada



# **INTELIGENCIA ARTIFICIAL e INGENIERIA DEL CONOCIMIENTO**

E.T.S. de Ingenierías Informática y de  
Telecomunicación

## **Práctica 1**

**"RESOLUCIÓN DE PROBLEMAS MEDIANTE  
BÚSQUEDA  
EN UN ESPACIO DE ESTADOS"**

**DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN E INTELIGENCIA  
ARTIFICIAL  
UNIVERSIDAD DE GRANADA  
Curso 2010-2011**



# DECSAI

## Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



## 1. Introducción

La primera práctica de la asignatura INTELIGENCIA ARTIFICIAL E INGENIERIA DEL CONOCIMIENTO consiste en la implementación y experimentación con diferentes algoritmos de búsqueda ante un problema simple en su definición pero complejo a nivel de resolución como es el “Clikomania”.

## 2. Aspectos generales del juego “Clikomania”.

En este apartado se resumen brevemente los aspectos más importantes del juego.

### 2.1. Clikomania.

Clikomania es un juego para un jugador con información completa (puzzle) con las siguientes reglas. El tablero es una rejilla rectangular de  $n$  filas y  $m$  columnas. Inicialmente el tablero tiene bloques ocupando todas las casillas, cada bloque con un color de entre  $k$  colores posibles. Dentro del tablero se encuentra *grupos* que son polígonos monocromáticos conectados horizontal o verticalmente, es decir, un grupo está compuesto por un conjunto de casillas que son todas del mismo color y que tienen al menos una casilla adyacente (sólo vertical u horizontal) que es del mismo color.

Un **movimiento** o **jugada** consiste en seleccionar uno de los grupos del tablero que contenga al menos dos casillas. El grupo seleccionado desaparece del tablero y los bloques que se encontraban situados encima de él caen para ocupar sus posiciones, de manera que nunca pueden aparecer agujeros internos en la estructura de bloques.

Una situación adicional que se produce en el juego es cuando desaparece una columna completa del tablero. En este caso, las columnas que se encuentra a la derecha de dicha columnas se desplazan hacia la izquierda para ocupar el espacio.

El juego termina cuando se eliminan todos lo bloques o bien cuando todos los grupos que quedan en el tablero son de tamaño uno.



El **objetivo** básico del juego consiste en determinar la secuencia de jugadas o movimientos que permiten obtener la máxima puntuación del tablero de partida. La puntuación de un tablero se calcula como la suma de la valoración de cada movimiento o jugada. El valor de una jugada se calcula como  $r \cdot (r-1)$  siendo  $r$  el tamaño del grupo eliminado del tablero.

## 2.2. Un ejemplo

Supongamos que el tablero del juego es el que se muestra en la siguiente figura 1, es decir un tablero con 5 filas, 6 columnas y 3 colores.

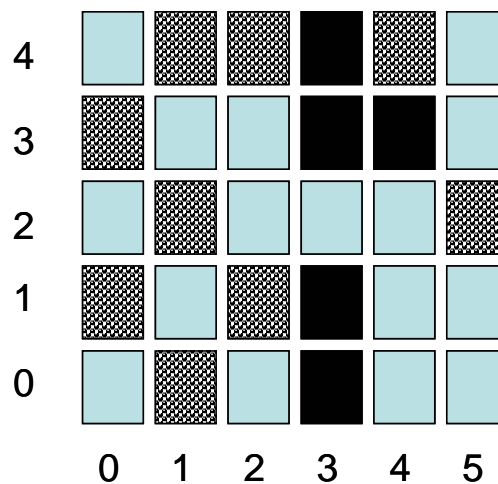


Figura 1 Tablero inicial

Como criterio general estableceremos que el sistema de coordenadas para referenciar cada posición en el tablero sitúa su punto de inicio en la esquina inferior izquierda, haciéndola corresponder con la posición (0,0), siendo la primera coordenada la  $x$  (*columna*) y la segunda la  $y$  (*fila*).

En este tablero inicial detectamos que existen 5 grupos con dos o más bloques del mismo color. Los grupos se muestran en la figura 2 marcados con flechas.

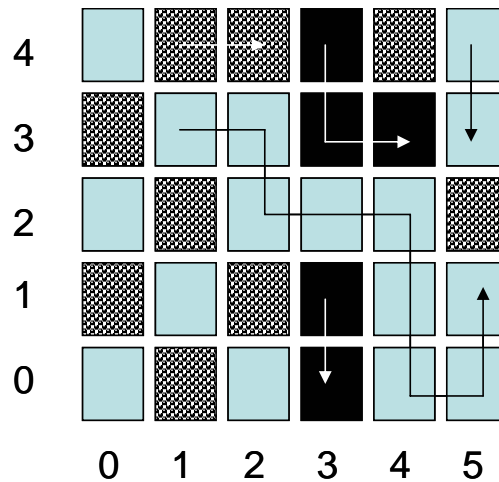


Figura 2 Grupos detectados en el tablero

De los 5 grupos, hay uno de tamaño 9, otro de tamaño 3 y el resto de tamaño 2. Cada uno de los 5 grupos implica un movimiento posible. Describiremos un **movimiento** o **jugada** con un par de coordenadas: las que corresponden a la casilla que formando parte del grupo se encuentra más a la izquierda (es decir, con valor menor de su componente  $x$ ). Si hubiera varias con el mismo valor, tomaremos la de la casilla que se encuentra en la posición más baja del tablero (es decir, aquella con valor menor en su componente  $y$ ). Así, los movimientos posibles son:

1. (1,3) de 9 casillas.
2. (1,4) de 2 casillas.
3. (3,0) de 2 casillas.
4. (3,3) de 3 casillas.
5. (5,3) de 2 casillas.

Supongamos que se selecciona como primer movimiento el grupo (1,3). La valoración de dicho movimiento es de  $9 \cdot 8 = 72$ , por ser 9 el número de casillas que compone el grupo seleccionado. La recomposición del tablero se realiza eliminando los bloques del grupo como muestra la figura 3, y desplazando los bloques para ocupar los espacios dejados por dicho grupo, obteniéndose el nuevo tablero que muestra la figura 4.

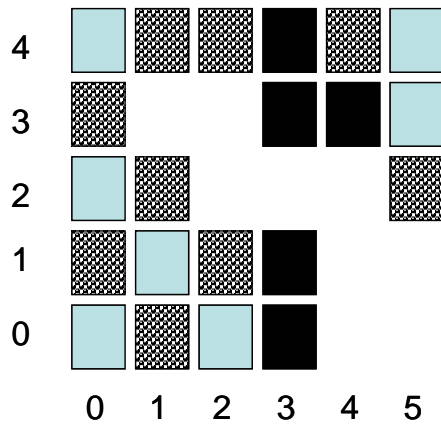


Figura 3 Eliminación de los bloques del grupo seleccionado

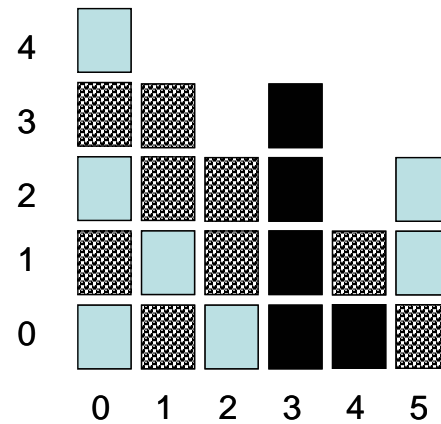
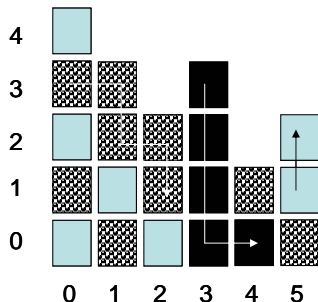


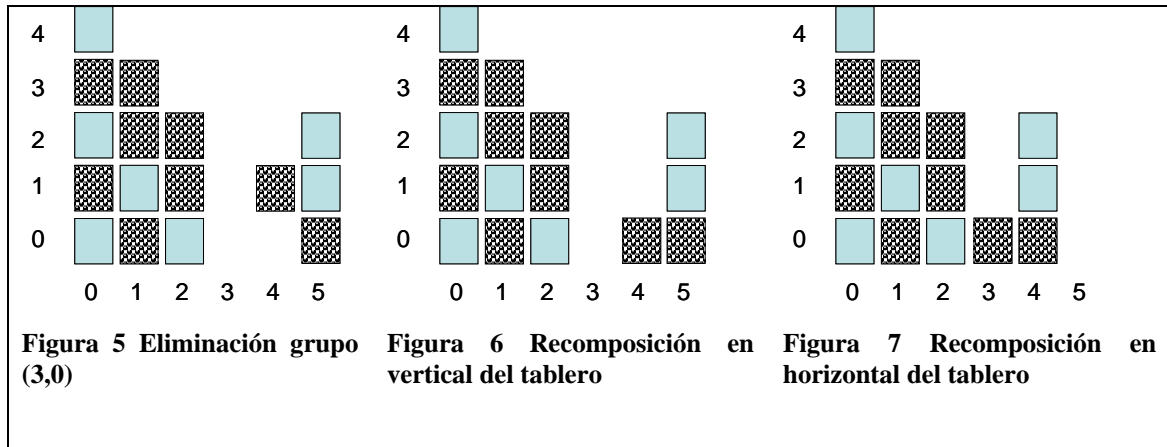
Figura 4 Desplazamiento de los bloques para ocupar las posiciones eliminados en el tablero.

En el nuevo tablero, podemos detectar 3 grupos: (0,3) con 5 bloques, (3,0) con 5 bloques y (5,1) con 2 bloques.

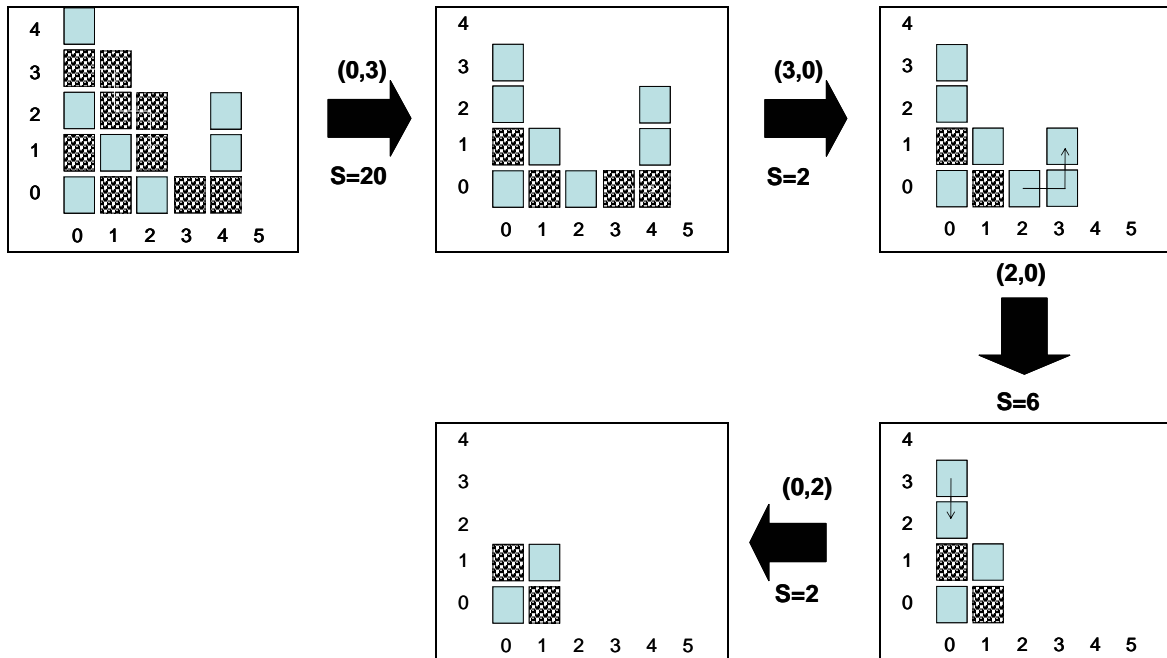


Supongamos que el siguiente movimiento a seleccionar es (3,0). El movimiento tiene una valoración de  $5 \cdot 4 = 20$ , que acumulado con el obtenido en el movimiento anterior, da una valoración a esta solución parcial de  $72 + 20 = 92$ .

Esta jugada conlleva que la columna 3 del tablero quede sin bloques (como muestra la figura 5). El proceso correcto para recomponer el tablero consiste, primero en realizar un desplazamiento hacia debajo de los bloques que no están sustentados por un bloque en su parte inferior, al igual que se hizo en el movimiento anterior. En este caso, como muestra la figura 6, sólo implica desplazar el único bloque de la columna 4. El segundo paso, consiste en un desplazamiento hacia la izquierda todas las columnas que se encuentran más allá de la columna eliminada (figura 7).



Una secuencia de movimientos desde este tablero hasta un tablero final podría ser la siguiente:



El último de los tableros es un tablero final porque no se pueden encontrar grupos de al menos tamaño 2, todos son de tamaño 1.



DECSAI

Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



Tomando los dos movimientos iniciales y esta relación final de movimientos, la secuencia del juego puede describirse con la siguiente lista ordenada de jugadas:

(1,3) (3,0) (0,3) (3,0) (2,0) (0,2)

La valoración de una partida completa es la suma de la valoración de cada uno de los movimientos realizados. En el caso anterior, la valoración de la partida es  $72+20+20+2+6+2=122$ .

### 3. Objetivo de la práctica.

La práctica tiene como objetivo estudiar el comportamiento de los diferentes algoritmos de búsqueda en espacio de estados frente al problema que representa la resolución de una serie de tableros concretos del juego “clikomania”.

La práctica se va a centrar en dos casos de estudio:

1. Conocer el valor óptimo de cada uno de los tableros que se adjuntan en esta práctica, así como conocer la secuencia de movimientos para que llevan a obtener esa máxima puntuación.
2. Proponer **una heurística y un algoritmo de búsqueda con información** para este juego que proporcionen secuencias de movimientos, que sin obtener el valor óptimo, resuelven más rápidamente el problema dando soluciones *semi-óptimas*. Entenderemos que una solución es semi-óptima ***si dista como máximo un 20% de la puntuación de la solución óptima***. Es decir, una secuencia de movimientos sobre un tablero que tenga como valoración óptima 100, se considerará que ha obtenido una solución semi-óptima si la solución obtenida tiene valoración mayor o igual que 80.

Para el primer caso de estudio, el alumno deberá implementar un algoritmo de búsqueda que permita obtener la máxima puntuación que es alcanzable





**DECSAI**

**Departamento de Ciencias de la Computación e I.A.**

Universidad de Granada



en cada uno de los tableros propuestos. Esta implementación **SI** formará parte del material que se entrega.

Para el segundo caso de estudio, el alumno deberá implementar un algoritmo de búsqueda con información que permita obtener soluciones semi-óptimas para los tableros propuestos. Esta implementación **SI** formará parte del material a entregar.

### 3.1 ¿Qué algoritmos de búsqueda se pueden utilizar?

La elección del algoritmo de búsqueda solo está restringida a que sea uno de los vistos en clase, o una variación del mismo diseñada o consultada por el alumno. Pero de entre todos los vistos en clase, cada uno puede implementar el/los que crea conveniente.

### 3.2. ¿Qué es necesario entregar?

La entrega se realizará a través de la web del Departamento de Ciencias de la Computación e I.A. (<http://decsai.ugr.es>) que tiene destinada para la entrega de prácticas de la asignatura. El material es un fichero comprimido **zip**, que debe ser llamado “practica1.zip” (sin tilde y todo junto).

La estructura interna de ese fichero comprimido **DEBE SER OBLIGATORIAMENTE** la siguiente:

- Un fichero llamado **docu.pdf** que contenga la documentación de la práctica.
- Una carpeta llamada **software1** que contendrá los ficheros fuente así como un **Makefile** para contruir un fichero ejecutable que resuelva el caso de estudio 1. Dentro de la carpeta **software1** no puede haber subcarpetas.
- Una carpeta llamado **software2** que contendrá los ficheros fuente así como un **Makefile** para construir el fichero ejecutable





que resuelva el caso de estudio 2. Dentro de la carpeta **software2** no puede haber subcarpetas.

### ***3.2.1. El fichero docu.pdf***

En el fichero de documentación docu.pdf se distinguirán dos partes, cada una destinada a cada uno de estudios propuestos en la práctica:

Estudio 1: En esta parte se consignará los valores óptimos, las secuencias de movimientos asociadas a esos óptimos y una medida del tiempo de ejecución de los tableros propuestos<sup>1</sup>.

También se debe indicar el algoritmo de búsqueda implementado para obtener los resultados anteriores y sus características si la implementación realizada presenta alguna modificación sobre el algoritmo estándar.

Estudio 2: En esta parte se describe la heurística propuesta, así como el algoritmo de búsqueda con información usado para la resolución, y cuya implementación se encontrará en la carpeta **programa**.

El tamaño de esta documentación no debe superar las tres páginas.

### ***3.2.2. Las carpetas software1 y software2***

En las carpetas **software1** y **software2** aparecerán los ficheros necesarios para construir el ejecutable que implementa los algoritmos de búsqueda para encontrar la máxima puntuación en un tablero y la más rápida que encuentra soluciones semi-óptimas respectivamente. El lenguaje de programación ha de ser ANSI C++ y ejecutable en un sistema operativo LINUX. Además, debe aparecer un fichero el fichero Makefile, en cada una de las carpetas, que permite realizar la compilación de los archivos fuente.

---

<sup>1</sup> Una tabla con los resultados anteriores sería suficiente para describir esta parte



# DECSAI

## Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



### 3.3. ¿Cómo se evaluará la práctica?

La calificación final de la práctica será la suma de la valoración de cada uno de los casos de estudio. El primero de los casos de estudio (encontrar los óptimos de los tableros) tendrá una valoración de hasta 3 puntos. El segundo (encontrar soluciones semi-óptimas vía algoritmos de búsqueda con información) tendrá una valoración de hasta 7 puntos.

Para obtener la calificación en el segundo de los casos, se realizará una competición entre todas las prácticas entregadas (carpeta **software2**). La máxima puntuación (7 puntos) la obtendrá aquella práctica que obtenga el mayor número de soluciones semi-óptimas en el menor tiempo. La puntuación del resto de las prácticas se realizará de forma proporcional a la mejor práctica, teniendo en cuenta las soluciones semi-óptimas encontradas y el tiempo utilizado.

En la competición se usaran los 10 tableros que se adjuntan a esta práctica y se dará un máximo de 1 minuto para resolver cada uno de ellos. Pasado ese minuto, si no se encuentra solución, se considerará un problema no resuelto y 1 minuto de tiempo consumido. Lo mismo ocurre si la solución encontrada no verifica ser semi-óptima.

### 3.4 Restricciones del software a entregar y representación.

Se pide desarrollar dos programas (en ANSI C++ usando el compilador C++ de GNU (llamado g++) que pueda ser compilado y ejecutado en una plataforma Linux. La invocación a cada programa debe respetar la siguiente sintaxis:

*buscaOptimo FichPuzzle FichSalida*  
*buscaRapido FichPuzzle FichSalida*

siendo **FichPuzzle** un fichero de texto (fichero ASCII) que contiene la descripción de la situación de partida del tablero, y **FichSalida**, otro fichero



**DECSAI**

**Departamento de Ciencias de la Computación e I.A.**

Universidad de Granada

































de texto con la secuencia de movimientos que obtendrá el programa para resolver el puzzle.

*buscaOptimo* es el nombre del programa que se utilizará para encontrar las soluciones óptimas a cada tablero y *buscaRapido* el de la **Competición de Rapidez**.

La estructura del fichero de entrada ***FichPuzzle*** será la siguiente:

<num filas>  
<num columnas>  
<num colores>  
<fila 0>  
<fila 1>  
.....  
<fila n>

Por ejemplo, para el tablero de partida es el que se mostró en la sección 2.2.

4						
3						
2						
1						
0						
	0	1	2	3	4	5

El fichero ***FichPuzzle*** contendrá la siguiente información:

5  
6  
3  
1 2 1 3 1 1  
2 1 2 3 1 1  
1 2 1 1 1 2  
2 1 1 3 3 1  
1 2 2 3 2 1



# DECSAI

## Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



El valor 0 de color se reserva para los bloques vacíos. Obsérvese además que la representación gráfica de la matriz y la representación que aparece en el fichero no coinciden. En realidad, es como si la segunda fuera el reflejo en un espejo de la primera.

La estructura del fichero *FichSalida* es la siguiente:

<valor de la solución>

<num. movimientos>

<jug 1 >

...

<jug m>

Recordar que cada jugada viene definida por el par de coordenadas (x,y) que representan a un grupo. Así, el *FichSalida* que representa la solución propuesta en el ejemplo de la sección 2.2. sería el siguiente:

122

6

1 3

3 0

0 3

3 0

2 0

0 2

Los pares de coordinas se encuentran separadas por un espacio.

### 3.4 Recomendaciones para la experimentación.

Es **muy recomendable** realizar varios experimentos con la intención de validar mejor la bondad del algoritmo y de la heurística implementada para la competición.



# DECSAI

## Departamento de Ciencias de la Computación e I.A.

Universidad de Granada



Si el alumno lo desea, aunque no es obligatorio en la práctica, puede definirse tableros adicionales de forma aleatoria (bien manual, bien automáticamente, lo que se deja a libre elección del alumno) y pasarlos como entrada (como **estado inicial**) al programa implementado.

**IMPORTANTE:** Los algoritmos de búsqueda que se describen en clase de teoría asumen que el óptimo es el valor mínimo de la función objetivo. En el problema que se plantea aquí, el óptimo es encontrar el máximo. Por consiguiente, es necesario adaptar los algoritmos que se implementen para tener en cuenta esta consideración.

La duración de esta práctica es de 3 semanas. Se recomienda que el alumno siga la siguiente secuencia temporal para su realización:

- Semana 1:
  - o Definición de una clase en C++ para establecer una estructura de datos que permita manejar los tableros del problema. Sobre esa clase definir entre otras operaciones, una que dado un tablero devuelva una lista con todos los movimientos posibles y el número de casillas implicadas, y otra que dado un tablero y un movimiento, devuelva el tablero resultante.
  - o Implementar un algoritmo de búsqueda sin información para obtener el valor óptimo de los 10 tableros.
- Semana 2 y 3: Implementar varios algoritmos de búsqueda con información y definir varias heurísticas para agilizar el proceso de encontrar soluciones semi-óptimas.

**Fecha de entrega de la práctica:** 5 de Noviembre del 2010, HASTA LAS 15:00 horas.