Técnicas de diseño de algoritmos
Búsqueda exhaustiva
Ejercicios (Bloque 1): con soluciones

Luis Javier Rodríguez Fuentes
Amparo Varona Fernández

Departamento de Electricidad y Electrónica
Facultad de Ciencia y Tecnología, UPV/EHU
luisjavier.rodriguez@ehu.es
amparo.varona@ehu.es

OpenCourseWare 2015
Campus Virtual UPV/EHU

Búsqueda exhaustiva – Ejercicios (Bloque 1)

(B1.1) El problema del **caballo de ajedrez** consiste en encontrar un recorrido de un tablero de ajedrez de $n \times n$ casillas que incluya exactamente una vez cada casilla, utilizando los movimientos del caballo a partir de una casilla inicial arbitraria.

Escribir en lenguaje Python un algoritmo de backtracking que obtenga una solución al problema. Escribir a continuación una versión mejorada que utilice el *heurístico de Warnsdorff* para ordenar las casillas a recorrer a partir de una dada (véase http://en.wikipedia.org/wiki/Knight's_tour). ¿Se nota la diferencia?

(B1.2) Considerese un grafo $G = (V, E)$ no dirigido. Dado un entero $k \leq n$, con $n = |V|$, el problema del **coloreado de grafos** consiste en encontrar una función $f : V \to \{1, \ldots, k\}$ tal que $f(u) \neq f(v)\ \forall (u, v) \in E$.

Asociado a este problema, se define el problema de optimización que trata de encontrar el menor k para el que se cumple la condición anterior (conocido como **número cromático** del grafo).

Escribir en lenguaje Python un algoritmo de backtracking que retorne el número cromático de un grafo G. Para ello, deberá definirse un algoritmo auxiliar que trate de encontrar un k-coloreado del grafo.

Búsqueda exhaustiva – Ejercicios (Bloque 1)

(B1.3) Un laberinto puede representarse mediante una matriz de booleanos de tamaño $n \times m$ donde cada valor booleano indica si se puede (`True`) o no (`False`) pasar por una casilla. Desde una casilla (i, j) sólo se puede transitar a una casilla adyacente en la misma fila o columna, es decir, a $(i + 1, j)$, $(i - 1, j)$, $(i, j + 1)$ o $(i, j - 1)$ (siempre que estén dentro del laberinto y sean accesibles). Usando el esquema de búsqueda exhaustiva, escribir en lenguaje Python una función que tome como entrada un laberinto L y, partiendo de la casilla $(0, 0)$, encuentre un camino hasta la casilla $(n - 1, m - 1)$ (ambas a `True`), retornando una lista con las casillas del recorrido o `None` en caso de que no exista solución.

Búsqueda exhaustiva – Ejercicios (Bloque 1)

(B1.4) Considerese un tablero lineal formado por $2n + 1$ casillas, tal que las n primeras casillas contienen piezas negras, a continuación hay una casilla vacía y por último n piezas blancas:

Se trata de realizar los movimientos necesarios para alcanzar el siguiente estado, en el que las piezas blancas y negras han intercambiado sus posiciones:

Las piezas negras sólo pueden moverse hacia la derecha y las blancas sólo hacia la izquierda. Una pieza puede avanzar sólo si la casilla adyacente está vacía, o saltar una pieza (blanca o negra) sólo si la casilla siguiente está vacía. Usando el esquema de búsqueda exhaustiva, escribir en lenguaje Python una función que retorne la lista de movimientos necesarios para pasar de la situación inicial a la final.