Práctica 5

Búsqueda exhaustiva

ALGORÍTMICA Y COMPLEJIDAD
Grados en Ing. Informática y Matemáticas
Universidad de Cantabria

Camilo Palazuelos Calderón

Objetivos de la práctica

1. Resolver un problema mediante búsqueda exhaustiva
2. Profundizar en el análisis de algoritmos recursivos
3. Comparar los costes temporales teórico y empírico

Duración de la práctica y fecha de entrega

- Las dos sesiones destinadas a la práctica son las de los días 14/15 y 21/22 de mayo
- La práctica debe entregarse, a través de la plataforma Moodle, antes del 26 de mayo a las 23:59

Qué debes entregar

- Memoria con respuestas a las preguntas formuladas en este documento
- Código desarrollado y material suplementario que se considere oportuno

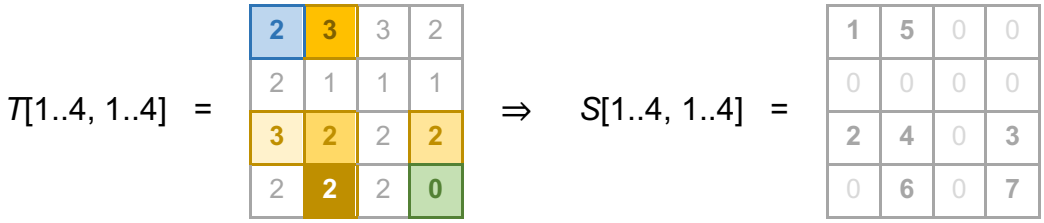
---

Laberinto numérico

Al laberinto numérico se juega con un tablero de $n \times n$ casillas y una ficha. Cada casilla tiene grabado un número entre 1 y $n - 1$, salvo la de la esquina inferior derecha (**la salida** del laberinto), que tiene grabado el 0. Para jugar, se empieza colocando la ficha en la casilla de la esquina superior izquierda (**la entrada** del laberinto); el objetivo es moverla hasta la salida del laberinto. En cada turno, se puede mover la ficha a la izquierda, a la derecha, arriba o abajo, siempre que no se salga del tablero; la distancia en número de casillas que se debe recorrer la marca el número grabado en la casilla actual.

No todas las configuraciones de números grabados en el tablero hacen alcanzable la salida del laberinto. Diseñad un algoritmo que indique si con un tablero dado es posible jugar al laberinto numérico hasta alcanzar la salida. Si lo es, el

algoritmo deberá devolver uno de los posibles caminos hasta la salida del laberinto, escribiendo en cada casilla que sea parte del camino el número de movimientos realizados hasta el momento; si no lo es, deberá devolver NULL. Por ejemplo, para el tablero *T*[1..4, 1..4], el algoritmo podría devolver *S*[1..4, 1..4]:



Preguntas en grupos de 2 o 3 alumnos

Pregunta 1 [3 PUNTOS]. Diseñad un algoritmo que, siguiendo la estrategia de vuelta atrás, resuelva laberintos numéricos como indica el enunciado.

Pregunta 2 [1 PUNTO]. Calculad el coste temporal de vuestro diseño de la pregunta 1 en el peor caso.

Preguntas en grupos de 4 o 5 alumnos

Pregunta 3 [3 PUNTOS]. Implementad vuestro diseño de la pregunta 1 en Python. Mostrad que la implementación es correcta proporcionando 4 o más ejemplos de entrada junto con sus salidas correspondientes.

Pregunta 4 [3 PUNTOS]. Dibujad una gráfica con los tiempos de ejecución de vuestra implementación con laberintos numéricos de peor caso en función de *n* (para, por ejemplo, *n* entre 2 y 6 de 1 en 1). ¿Es coherente la curva obtenida con el coste temporal calculado en la pregunta 2?