Práctica 3.1: Algoritmos voraces

Algoritmia y optimización

Curso 2025–26

1. Introducción

En esta práctica plantearemos la resolución de problemas mediante estrategias voraces.

2. Objetivos

- Entender en qué consiste una estrategia voraz.
- Practicar la implementación de estrategias voraces.

3. Ejercicio

Implementa estrategias voraces para los problemas presentados en las siguientes secciones. Haz que tu código calcule **tanto el valor óptimo como la solución completa**.

Además, para cada estrategia plantéate:

- Cómo vas a codificar una posible solución para el correspondiente problema.
- Si la estrategia encuentra siempre una solución al problema.
- Si la estrategia encuentra siempre la solución **óptima** al problema.

3.1. El cambio de monedas

El problema consiste en formar una suma M con el número mínimo de monedas escogidas (con posibilidad de repetición) de un conjunto finito $C = \{c_1, c_2, \ldots\}$.

El problema se puede formalizar como sigue. Una solución es una secuencia de decisiones $S = (s_1, s_2, \ldots, s_n)$ tal que $s_i \in C$. El objetivo es minimizar $|S|$, con la la restricción de que $\sum_{i=1}^{n} \text{valor}(s_i) = M$.

3.2. El fontanero diligente

Un fontanero necesita hacer n reparaciones urgentes y sabe de antemano el tiempo que le va a llevar cada una de ellas: en la tarea i-ésima tardará t_i minutos. Como en su empresa le pagan dependiendo de la satisfacción del cliente, necesita decidir el orden en el que atenderá los avisos para minimizar el tiempo medio de espera de los clientes.

En otras palabras, si llamamos E_i a lo que espera el cliente i-ésimo hasta ver reparada su avería por completo, necesita minimizar la expresión:

$$E = \sum_{i=1}^{n} E_i \quad \text{con} \quad E_i = \sum_{j=1}^{i} t_j$$

3.3. El coloreado de grafos

Dado un grafo no dirigido, el problema consiste en encontrar el menor número de colores con el que se pueden colorear sus vértices de forma que no haya dos vértices adyacentes con el mismo color. Por ejemplo, dado el grafo de la Fig. 1, el mínimo número de colores sería 3.

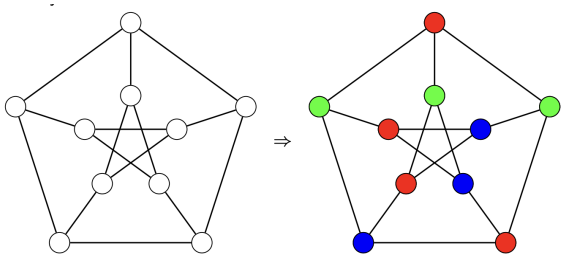

Figura 1: Ejemplo de grafo cuyo coloreado mínimo requiere 3 colores.

Para este problema puedes definir un grafo mediante una matriz de adyacencia $\{0,1\}^{|V|\times|V|}$.

3.4. El laberinto con cuatro movimientos

Se dispone de una cuadrícula $n \times m$ de valores $\{0,1\}$ que representa un laberinto. Un valor 0 en una casilla cualquiera de la cuadrícula indica una posición inaccesible; por el contrario, con el valor 1 se simbolizan las casillas accesibles.

Encontrar un camino que permita ir de la posición $(0,0)$ a la posición $(n-1, m-1)$ con cuatro tipos de movimiento (arriba, abajo, derecha, izquierda).

3.5. La asignación de tareas

Supongamos que disponemos de n trabajadores y n tareas. Sea $b_{ij} > 0$ el coste de asignarle el trabajo j al trabajador i.

Una asignación de tareas puede ser expresada como una asignación de los valores 0 ó 1 a las variables x_{ij}, donde $x_{ij} = 0$ significa que al trabajador i no le han asignado la tarea j, y $x_{ij} = 1$ indica que sí.

Una asignación válida es aquella en la que a cada trabajador sólo le corresponde una tarea y cada tarea está asignada a un trabajador.

Dada una asignación válida, definimos el coste de dicha asignación como:

$$\sum_{i=1}^{n}\sum_{j=1}^{n} x_{ij}b_{ij}$$

El problema consiste en encontrar una asignación óptima, es decir, de mínimo coste.

Cuestión: ¿cambia el algoritmo si añadimos una matriz $W = \{0,1\}^{n\times n}$ tal que un trabajador i solo puede asignarse a la tarea j si $W_{ij} = 1$?

3.6. El viajante de comercio

Imagina que un vendedor debe visitar varias ciudades, partiendo de una ciudad inicial, y regresar a ella al final. Su objetivo es encontrar la ruta más corta que le permita visitar todas las ciudades una sola vez.

Formalmente, el problema se reduce a encontrar un recorrido en un grafo ponderado, con pesos no negativos, que recorre todos los vértices sólo una vez y regresa al de partida con el mínimo coste (es decir, que minimice la suma del peso de las aristas). Se puede asumir que el vértice de partida y regreso siempre es el numerado como primero.

Para este problema puedes definir un grafo mediante una matriz de adyacencia ponderada $\mathbb{R}^{+|V|\times|V|}$. Puedes usar un valor especial (-1) para indicar que no hay arista entre dos vértices.

Cuestión: ¿Cambia el algoritmo dependiendo de si el grafo es completo?