

Métodos algorítmicos en resolución de problemas II

Grado en Ingeniería Informática
Doble Grado en Ingeniería Informática y Matemáticas

CONVOCATORIA ORDINARIA DE JUNIO

Curso 2020/2021

1. Programación dinámica [2,75 puntos]

Tenemos un array de n naturales S que en cada posición i ($1 \leq i \leq n$) contiene el máximo salto que puede darse hacia el final del array desde la posición i y un 0 en caso de que no se pueda saltar a dicha posición. Un salto k permite avanzar de golpe de la posición i a la posición $i + k$ siempre que en dicha posición no haya un 0. Se desea calcular cuál es el menor número de saltos necesarios para llegar desde la primera posición del array a la última, así como conocer la secuencia de posiciones del array por las que se ha pasado. Por ejemplo, si el array es $[1, 3, 8, 0, 1, 1, 1]$, el menor número de saltos es 3 y la secuencia de posiciones es 1, 2, 3, 7.

Se pide desarrollar un algoritmo de programación dinámica para resolver este problema. Se valorarán todos los pasos: definición de la recurrencia, implementación del algoritmo y análisis de los costes en tiempo y espacio.

2. Ramificación y poda [2,75 puntos]

Un ratón de laboratorio tiene disponibles en su jaula n teclas para pulsar. Cuando pulsa una tecla por primera vez pone en marcha un mecanismo de recompensas y castigos. Cada vez que pulsa de nuevo una tecla recibe una recompensa o un castigo dependiendo de la última tecla que pulsó y la actual. Esta información viene dada en una matriz T en la que para cada par de teclas i, j ($1 \leq i, j \leq n$) $T[i][j]$ indica la recompensa (un valor ≥ 0) o el castigo (un valor < 0) que recibe el ratón al pulsar la tecla j inmediatamente después de la i .

Se pide diseñar e implementar un algoritmo de ramificación y poda que resuelva el problema de encontrar una secuencia de teclas de longitud m que maximice la suma de recompensas obtenidas por el ratón, teniendo en cuenta que la suma de los castigos en valor absoluto no puede ser superior a un determinado valor dado $C \geq 0$.

3. NP-completitud [1,5 puntos]

El problema de decisión SPARSE-SUBGRAPH consiste en: dado un grafo no dirigido $G = (V_G, A_G)$ y dos números naturales a y b , determinar si G tiene un subgrafo G' con a vértices y a lo sumo b aristas. Se pide demostrar que SPARSE-SUBGRAPH es NP-completo utilizando alguno de los problemas NP-completos vistos en clase.

Nota aclaratoria: Que G' es subgrafo de G quiere decir que $G' = (V_{G'}, A_{G'})$ tal que:

- $V_{G'} \subseteq V_G$
- $A_{G'} = A_G \cap (V_{G'} \times V_{G'})$