UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la Computación e Inteligencia Artificial

Algorítmica (2023/24)
2º Grado en Ingeniería Informática
Ejemplo de examen

◁ Ejercicio 1 ▷ [2 puntos]

Calcular el orden de eficiencia en notación $\mathrm{O}(\cdot)$ de un algoritmo recursivo cuya expresión de tiempo es:

$$T(n) = 3 \cdot T\left(\frac{n}{2}\right) + 4 \cdot T\left(\frac{n}{4}\right) + n^2$$

◁ Ejercicio 2 ▷ [2 puntos]

Dado un conjunto de n números enteros positivos $C = \{c_1, c_2, \ldots, c_n\}$ y dado un entero positivo N, se quiere encontrar un subconjunto de C, $\{y_1, \ldots, y_m\}$ que maximice $\prod_{i^m} y_i$ (se maximiza el producto de los números del subconjunto), sujeto a la restricción $\prod_{i=1}^{m} y_i \leq N$. Nota: m NO es un parámetro de entrada.

Diseñad un algoritmo de vuelta atrás para resolver este problema.

Por ejemplo, para $n = 4$, $N = 37$ y $C = \{4, 4, 12, 3\}$, la solución es emplear el subconjunto $\{12, 3\}$ con un valor de producto igual a 36.

◁ Ejercicio 3 ▷ [2 puntos]

Disponemos de n tipos de monedas. Sabiendo que las monedas de tipo i tienen un valor de $v[i]$ y que la cantidad de monedas disponibles de tipo i es igual a $c[i]$, se quiere devolver una cantidad exacta M utilizando el menor número posible de monedas. Diseñad un algoritmo de programación dinámica que determine el número óptimo de monedas a usar. Aplicadlo para resolver el siguiente caso del problema, con $n = 3$, construyendo la tabla correspondiente:

$$v[] = (1, 4, 6) \qquad c[] = (10, 1, 2) \qquad M = 8$$

◁ Ejercicio 4 ▷ [2 puntos]

Sea un conjunto C conteniendo n actividades de un calendario. Cada actividad i tiene un instante de tiempo s_i de comienzo, y un instante de tiempo f_i donde debe haber ya finalizado. Diseñad un algoritmo voraz que nos permita encontrar el subconjunto que contenga el máximo número de actividades compatibles. Dos actividades i, j se dicen compatibles si no se solapan en el tiempo, es decir, si $[s_i, f_i) \cap [s_j, f_j) = \emptyset$.

◁ Ejercicio 5 ▷ [2 puntos]

Sea $v[1..n]$ un vector ordenado de forma creciente de enteros positivos sin repetir. Diseñe un algoritmo «divide y vencerás» que compruebe si existe algún elemento tal que $v[i] = n - i$.