UNIVERSIDAD DE JAÉN

DISEÑO DE ALGORITMOS

Grado en Ingeniería Informática
Departamento de Informática
Teoría de Algoritmos
CURSO 2021/22

RELACIÓN DE PROBLEMAS de Programación Dinámica

Problema 1: Cálculo de funciones

Implementa un algoritmo Basado en Programación dinámica que calcule la siguiente función:

$$f(n) = \begin{cases} 0 & si\ n = 0 \\ n + \frac{2}{n}\sum_{i=0}^{n-1} f(i) & si\ n > 0 \end{cases}$$

Problema 2: Cálculo de funciones

Implementa un algoritmo Basado en Programación dinámica que calcule la siguiente función:

$$f(n,m) = \begin{cases} 0 & si\ m = 0 \\ n & si\ m = 1 \\ f\left(n, \left\lfloor \frac{m}{2} \right\rfloor\right) + f\left(n, \left\lceil \frac{m}{2} \right\rceil\right) & si\ m > 1 \end{cases}$$

Problema 3: Cálculo de funciones

Implementa un algoritmo basado en PD que, dados dos valores enteros m y n, y un vector V que como mínimo tenga m+1 elementos, calcule el valor de Q(m, n) según la siguiente función:

$$Q(m,n) = \begin{cases} 1 & si\ m = 0 \\ 1 & si\ n = 0 \\ Q(m-1,n) & si\ n < v_m \\ \sum_{i=1}^{v_m - 1} Q(m, n-i) & e.o.c. \end{cases}$$

Problema 4: Cambio de Monedas

Vivimos en un país donde el sistema monetario está formado por una serie de monedas, cada una de ellas con un valor facial.

$$M=\{m1, m2, \ldots, mn\}$$

Diseñar un algoritmo que permita devolver una cierta cantidad de dinero C utilizando el menor número de monedas posible.

Guía: Para resolver este problema podemos considerar su similitud con el problema de la mochila. Así, si $x_1, x_2, \ldots, x_{n-1}, x_n$ representa las monedas en un cambio óptimo para una cantidad C, entonces $x_1, x_2, \ldots, x_{n-1}$ debe ser también un cambio óptimo para la cantidad C-Valor_Facial (x_n) y de forma obvia x_n es un cambio óptimo para la cantidad Valor_Facial (x_n).

Definir una función: *monedas (n, C)* que represente el mínimo de monedas necesarias para pagar la cantidad C considerando los tipos de monedas del 1 al n.

A partir se plantea la recurrencia en el caso general, cuando se consideran los tipos de monedas del 1 al i y nos queda pagar una cantidad j: *monedas(i,j)*. (Utilizar un enfoque hacia atrás). Una vez hecho esto habrá que determinar cuáles son los casos base (soluciones triviales) y hallar sus respectivos valores.

Problema 5: El problema del Botellón

Dos amigos, *Agonioso* y *Listillo*, salen de botellón. La nueva moda es poner una fila con n vasos (n es par). Cada vaso i, entre 1 y n contiene una cantidad de líquido c_i distinta. Los amigos beben por turnos. Cada uno, en su turno debe elegir el vaso de uno de los extremos y beberse su contenido. El vaso se retira y el turno pasa al otro amigo. La persona que comienza bebiendo se determina a priori por un procedimiento cualquiera. El objetivo de ambos amigos es beber la mayor cantidad posible de líquido.

La estrategia de *Agonioso* consiste en pensar poco y coger el vaso de los extremos que esté más lleno. En cambio *Listillo* prefiere pensárselo un poco más.

a) Demostrar, con un contraejemplo que la estrategia de *Agonioso* no es óptima, incluso cuando le toca escoger primero.
b) *Listillo* tiene unos amigos que cursan la asignatura de Diseño de Algoritmos y les pide que le diseñen un algoritmo, basado en Programación Dinámica, para que le ayude a conseguir su objetivo, suponiendo que es él quien empieza a escoger.

Guía: Definir una función *botellón(i,j)* que represente la cantidad máxima que bebe *Listillo* con los vasos desde el i hasta el j cuando le toca empezar a beber. La solución final será *botellón(1,n)*

Problema 6: Los puentes que une un grupo de islas

Un archipiélago está formado por varias islas. Existen una serie de puentes de dirección única que unen ciertos pares de islas entre sí. Para cada puente se conoce su anchura, que siempre es mayor que 0.

La *anchura de un camino* formado por una sucesión de puentes es la anchura mínima entre las anchuras de todos los puentes que lo forman.

Diseñar un algoritmo basado en Programación Dinámica para saber, si existe, cuál es el camino de anchura máxima entre todo par de islas.

Guía: Para resolverlo pensad en una estrategia parecida a la del Algoritmo de Floyd.