Metodología y tecnología de la programación

Ingeniería Informática (UCM)

Hoja de ejercicios 4 Curso 2007/2008

Ejercicios de programación dinámica

Ejercicio 1 El país de Fanfanisflán emite n sellos diferentes de valores $s_1, s_2, \ldots, s_n$. Jaffar tiene que enviar una carta a Faragonescia y sabe que la correspondiente tarifa postal es T. Jaffar tiene interés en saber de *cuántas* formas diferentes se puede franquear exactamente la carta, si el orden de los sellos no importa. Desarrollar un algoritmo eficiente para obtener este número y calcular su coste de tiempo y espacio adicional en términos de T y n.

Ejercicio 2 Sobre un río hay N embarcaderos en cada uno de los cuales se puede alquilar un bote para ir a cualquier otro embarcadero río abajo (es casi imposible remontar la corriente). La tarifa indica el coste del viaje desde i hasta j para cualquier punto de partida i y cualquier punto de llegada j más abajo en el río. Puede suceder que un viaje de i a j sea más caro que una sucesión de viajes más cortos, en cuyo caso se tomaría un primer bote hasta k y otro bote para continuar el viaje a partir de k (no hay coste adicional por cambiar de bote). Escribir un algoritmo eficiente para determinar el coste mínimo (así como las correspondientes paradas) para ir desde i hasta j y calcular el tiempo en función de N.

Ejercicio 3 Tenemos n varillas distintas de longitudes enteras $l_1, \ldots, l_n$ y precios reales $c_1, \ldots, c_n$, respectivamente. Las varillas *no* se pueden cortar. Se desea soldar algunas de ellas para obtener una varilla de longitud total L (también entera).

Se pide escribir algoritmos de programación dinámica para resolver directamente cada uno de los siguientes problemas:

1. Indicar si es posible o no obtener la varilla deseada soldando algunas de las varillas dadas.
2. Calcular el número total de maneras de obtener la varilla deseada soldando algunas de las varillas dadas, sin que importe el orden de soldadura.
3. Minimizar el número de varillas utilizadas.
4. Minimizar el precio total de las varillas utilizadas.

Cada algoritmo tiene que resolver exactamente el problema pedido en cada caso y *no* hay que calcular en ningún caso las varillas necesarias. Por supuesto, hay que minimizar los costes en espacio y tiempo.

Ejercicio 4 Se tienen dos almacenes A_1 y A_2 en los que se dispone respectivamente de a_1 y a_2 unidades (indivisibles) de un determinado producto. Desde estos dos almacenes se ha de abastecer a n comercios $C_1, \ldots, C_n$, en cada uno de los cuales se precisa una determinada cantidad c_j (para j entre 1 y n) de unidades del producto, de forma que $\sum_{j=1}^{n} c_j = a_1 + a_2$. El coste del transporte desde A_i hasta C_j viene dado por una función $t_{ij}(x)$ donde x es el número de unidades de producto transportadas. Se desea determinar las cantidades de unidades x_{ij} que cada almacén debe enviar a cada comercio de forma que el coste total del transporte $\sum_{ij} t_{ij}(x_{ij})$ sea mínimo.

Ejercicio 5 Considérese el alfabeto $\Sigma = \{a, b, c\}$. Los elementos de Σ tienen la siguiente tabla de multiplicar, donde las filas representan el símbolo izquierdo y las columnas el derecho.

	a	b	c
a	b	b	a
b	c	b	a
c	a	c	c

Así, $ab = b, ba = c$, etcétera. Nótese que la multiplicación definida por esta tabla no es conmutativa ni asociativa.

Se pide encontrar un algoritmo eficiente que examine una cadena $x = x_1x_2 \cdots x_n$ de caracteres de Σ y devuelva el número de formas diferentes en que se pueden poner paréntesis en la expresión para obtener a. Por ejemplo, si $x = acc$, el algoritmo debería devolver 2 porque $(ac)(c) = a$ y $(a)(cc) = a$. Estudiar su coste en términos de n, la longitud de la cadena x.

Ejercicio 6 Escribir todos los detalles del algoritmo de programación dinámica para resolver el problema de la mochila entera, como se ha visto en clase. ¿Cuál es el coste en tiempo y en espacio adicional?

Ejercicio 7 Tenemos n programas $P_1, \ldots, P_n$ para almacenar en un disco. Cada programa P_i requiere s_i kilobytes de espacio y la capacidad total del disco es D kilobytes, con $D < \sum_{i=1}^{n} s_i$. Escribir dos algoritmos que resuelvan las siguientes tareas:

1. Maximizar el número de programas almacenados en el disco.
2. Utilizar la mayor capacidad posible del disco.