Algoritmos y Estructuras de Datos II - 18 de junio de 2014

Segundo Parcial

Alumno: .....

Siempre se debe explicar la solución, una respuesta correcta no es suficiente sino viene acompañada de una justificación que demuestre que la misma ha sido comprendida. Las explicaciones deben ser completas. La utilización de código c o de nivel de abstracción excesivamente bajo influirá negativamente. **Por favor, resolvé cada ejercicio en una hoja aparte,** para acelerar el proceso de corrección. **¡No olvides escribir claramente tu nombre en cada una de las hojas!** En los ejercicios con varios incisos, por favor, no resuelvas varios incisos simultáneamente, sino cada uno por separado (pero no hace falta que sea en hojas aparte).

1. Para el problema de la mochila, mostrá con ejemplos las siguientes afirmaciones:
   a) el criterio de elegir en cada paso el objeto de mayor valor aún no seleccionado no garantiza que el algoritmo voraz obtenga la solución óptima,
   b) el criterio de elegir en cada paso el objeto de menor peso aún no seleccionado no garantiza que el algoritmo voraz obtenga la solución óptima,
   c) el criterio de elegir en cada paso el objeto de mayor cociente valor/peso aún no seleccionado no garantiza que el algoritmo voraz obtenga la solución óptima cuando no está permitido fraccionar objetos.

2. Respondé separadamente cada una de las siguientes preguntas, y proporcioná el ejemplo requerido
   a) ¿qué es un árbol generador?
   b) ¿qué es un árbol generador de costo mínimo?
   c) dá un ejemplo de un grafo con 7 vértices que tenga 3 diferentes árboles generadores de costo mínimo, señalá cuáles serían esos árboles.

3. Considerá una cuadrícula de n filas por k columnas donde cada celda tiene un costo $c_{i,j}$. Se comienza en la posición $(1,1)$ (celda superior izquierda) de la cuadrícula y se debe viajar hasta la celda (n,k) (inferior derecha). Sólo hay dos movimientos permitidos: hacia abajo (de (i,j) a $(i+1,j)$) o hacia la derecha (de (i,j) a $(i,j+1)$). Estos movimientos se permiten siempre que no se salga de la cuadrícula. El costo de un viaje es la suma de los costos de las celdas por las que se pasa, incluyendo $c_{1,1}$ y $c_{n,k}$. Escribí un algoritmo que utilice backtracking para calcular el costo del viaje de menor costo desde $(1,1)$ hasta (n,k). (Ayuda: si lo desea, puede utilizar la siguiente definición $m(i,j)$ = "costo del viaje de menor costo desde $(1,1)$ hasta (i,j) con movimientos permitidos".)

   Justificá con detalle cada una de las ecuaciones.

4. Para el problema de la moneda, se propone la siguiente variante: se cuenta con monedas de denominación $d_1, d_2, \ldots, d_n$. No se asume que están ordenadas. A diferencia del problema original, en que se asumían infinitas monedas de cada denominación se tiene una cierta cantidad de monedas $c_1, c_2, \ldots, c_n$ de cada denominación. Se debe pagar de manera exacta el monto M.

   Se cuenta con una solución que utiliza backtracking. La misma consiste en definir $m(i,j)$ = "menor número de monedas necesarias para pagar el monto j eligiendo entre las monedas disponibles de denominación $d_1, d_2, \ldots, d_i$" que dan lugar a la siguiente definición recursiva:

$$m(i,j) = \begin{cases} 0 & \text{si } j = 0 \\ \infty & \text{si } j > 0 \wedge i = 0 \\ \text{mín}\{k + m(i-1, j - k * d_i) | k \geq 0 \wedge k \leq c_i \wedge k * d_i \leq j\} & \text{si } j > 0 \wedge i > 0 \end{cases}$$

   Otra manera equivalente de escribir la última ecuación sería:

$$m(i,j) = \text{mín}\{m(i-1,j),\ 1 + m(i-1, j-d_i),\ 2 + m(i-1, j-2*d_i),\ \ldots\}$$

   Observá que cada elemento del conjunto es de la forma $k + m(i-1, j-k*d_i)$, donde k no puede superar el número de monedas de denominación d_i disponibles (es decir $k \leq c_i$), y tampoco se debe exceder, con las k monedas de denominación d_i el monto j (es decir $k * d_i \leq j$).

   El ejercicio consiste en transformar **esta** solución en una que utilice programación dinámica. **No es válido recurrir a otra** solución al mismo problema.