Examen 2

Análisis y Diseño de Algoritmos

12 de julio de 2021

Duración: 3 horas
No se permite ningún tipo de apunte

Ejercicio 1 (2 ptos). Muestre la tabla para el problema SUBSET-SUM visto en clase para los items $[2, 2, 1]$ con $W = 5$. Solo es necesario que muestre la tabla resultante final.

Ejercicio 2 (2 ptos). Escriba la recurrencia para el problema MIN-PARTITION visto en clase: dado un arreglo A de números enteros no negativos, encontrar una partición de tamaño k con peso mínimo. El peso de una partición es el mayor valor de la suma de los elementos de un bloque de la partición.

Ejercicio 3 (4 ptos). Considere el siguiente problema. Dado un rectángulo $n \times m$, queremos cortarlo en cuadrados. En cada movimiento puedes seleccionar un rectángulo y cortarlo en dos rectángulos, de manera tal que todos los lados siguen siendo números enteros. ¿Cual es el mínimo número posible de movimientos? Por ejemplo, si $n = 3$ y $m = 5$, la respuesta es 3.

Sea $OPT(n, m)$ el valor de la solución óptima para un rectángulo $n \times m$. Describa una recurrencia para $OPT(n, m)$ (no necesita probar su correctitud). A partir de ello, diseñe un algoritmo de programación dinámica (haga el pseudocódigo) con tiempo de ejecución $O(nm(n + m))$.

Ejercicio 4 (3 ptos). Queremos codificar el archivo cuyo contenido es la siguiente cadena:

$$aaabbbbcccccccdddddddeeefffggggghhhhhhh.$$

(a) Muestre el resultado **final** (el árbol y las codificaciones asociadas) luego de correr la implementación $O(n \lg n)$ de Huffman (no necesita mostrar pasos intermedios).

(b) Indique cuál es el peso del archivo codificado y compárelo con el peso de un archivo codificado usando una cantidad fija de bits para cada caracter. ¿Existe una mejora?

Ejercicio 5 (2 ptos). Indique la salida para el problema de mochila fraccionaria luego de ejecutar el algoritmo visto en clase con la siguiente entrada: $v = [10, 22, 10, 40, 90, 56]$, $w = [1, 2, 5, 8, 10, 7], W = 16$.

Ejercicio 6 (4 ptos). Sea A un conjunto de n pares ordenados en el eje X. Sea B un conjunto de n pares ordenados en el eje Y. Un *matching* para A y B es un conjunto de asociaciones de los elementos de A con los de B, de manera que un mismo par ordenado no puede estar en dos asociaciones. El costo de un matching es la suma de las distancias entre los puntos asociados. Por ejemplo, si $A = \{(0, 4), (0, 3)\}$ y $B = \{(3, 0), (4, 0)\}$, el matching $\{\{(0, 4), (3, 0)\}, \{(0, 3), (4, 0)\}\}$ tiene costo 5+5=10.

Queremos diseñar un algoritmo voraz para encontrar el costo mínimo de un matching.

(a) Mencione cual es la elección voraz

(b) Escriba el pseudocódigo (no código) de su algoritmo voraz. Indique claramente qué recibe y qué devuelve su algoritmo. El algoritmo debe ser recursivo.

(c) Demuestre que su elección voraz es correcta. Enuncie a modo de lema y demuestre que su lema es correcto.

Deberá utilizar la siguiente notación.

Para los conjuntos: $A = \{(0, a_1), (0, a_2), \ldots, (0, a_n)\}$, $B = \{(b_1, 0), (b_2, 0), \ldots, (b_n, 0)\}$. Para soluciones devueltas por el algoritmo o utilizadas en las demostraciones: variables X, X'.

Pista: ordene previamente su entrada según algún criterio (o considere que recibe la entrada ordenada según ese criterio).

Ejercicio 7 (3 ptos). Tareas de evaluación continua