Diseño y análisis de algoritmos

Tarea 5

Profesores Armando Castañeda y Sergio Rajsbaum
Ayudantes Luis Gómez y Diego Velázquez

Martes 8 de octubre de 2019

Fecha de entrega: Jueves 17 de octubre de 2019

INSTRUCCIONES:

- Se puede hacer en equipos de 2 o 3 personas, pero hay que entregarla individualmente. La resuelven entre todos, pero cada quien la escribe en sus palabras. Anotar en cada tarea el nombre de todos los miembros del equipo.
- Las respuestas deben estar escritas con claridad, todos los enunciados demostrados.
- No se aceptan tareas después de la fecha límite.
- No escribas la implementación de tus algoritmos.
- Si el ejercicio dice "prueba", "demuestra" o "muestra", no debes dejar ningún hecho sin justificar; esto significa que debes decir por qué lo que escribes es verdad. Si lo que se pide es una explicación, es suficiente que enuncies los hechos que explican lo que se pide sin decir por qué son verdaderos pero tienes que manifestarlos completamente.

PROBLEMAS:

1. Presenta algoritmos que corran en tiempo $O(n \log n)$ basados en colas de prioridades para los siguientes problemas vistos en clase:
   - Encontrar un conjunto de cardinalidad máxima con trabajos compatibles.
   - Calendarizar clases usando el mínimo número de salones de clases posibles.
   - Calendarización para minimizar la latencia máxima.

   Prueba la complejidad de tiempo de tus algoritmos.

2. Da un algoritmo de tipo *greedy* para alguno de los siguientes problemas:
   - Encontrar un conjunto independiente maximal de una gráfica no dirigida G. Un conjunto independiente es un conjunto de vértices que no tienen aristas entre ellos, y es maximal si no hay otro conjunto independiente que lo contenga propiamente.

- Encontrar una $\Delta + 1$ coloración de los vértices de una gráfica no dirigida G, donde Δ denota el grado máximo en G. Una coloración es una función que mapea los vértices a un conjunto de colores C tal que vértices adyacentes reciben colores diferentes; decimos que es una $\Delta + 1$ coloración si $|C| = \Delta + 1$.

Explica claramente cual es la entrada y salida de tu algoritmo, así como por qué tu solución puede ser catalogada como del tipo *greedy*. Demuestra que tu algoritmo es correcto, y también demuestra su complejidad de tiempo y espacio (cantidad de memoria adicional a la entrada).

Además, prueba que tu algoritmo, a pesar de ser correcto, no es óptimo. Es decir, da una gráfica G para la que existe una mejor solución que la que tu algoritmo produce, ya sea un conjunto independiente de mayor cardinalidad o una coloración que use menos colores.

3. Diseña un algoritmo que dada una cadena S encuentre y regrese el primer símbolo que no se repite (leyendo S de izquierda derecha); si dicho símbolo no existe, entonces la salida debe ser #. Tu algoritmo debe de recorrer una sola vez la cadena S y utilizar $O(1)$ en memoria (adicional a la entrada). La cadena S contiene solo letras minúsculas del alfabeto Español.

4. Diseña un algoritmo tipo *greedy* para el siguiente problema de optimización. La entrada está compuesta de un conjunto con n pares de enteros positivos, $\{(w_1, v_1), \ldots, (w_n, v_n)\}$, y un entero positivo W; cada (w_i, v_i) representa una clase de objetos con peso w_i y valor v_i, y W representa una caja que puede cargar esa cantidad de peso. El objetivo es seleccionar algunos objetos de los n posibles a meter en la caja de forma tal que se maximice la suma del valor de los objetos seleccionados, permitiendo meter fracciones de objetos; por ejemplo, tu algoritmo podría decidir meter la unidad completa de un objeto y solo una fracción de otro. Formalmente, para el i-ésimo objeto, (w_i, v_i), tu algoritmo debe decidir meter una fracción $x_i \in [0, 1]$ de él, tal que $\mathsf{weight} = \Sigma_{i=1}^{n} x_i w_i \leq W$ y el valor de la solución, $\mathsf{val} = \Sigma_{i=1}^{n} x_i v_i$, sea el máximo posible.

   Explica por qué tu solución puede ser catalogada como del tipo *greedy*, y demuestra que tu algoritmo es correcto, y también demuestra su complejidad de tiempo y espacio (cantidad de memoria adicional a la entrada).