Proyecto 3

Análisis de Algoritmos

Primer Semestre 2020, Prof. Cecilia Hernández

5 de agosto de 2020

Fecha Inicio: Miércoles 5 de Agosto 2020. Fecha Entrega: Miércoles 19 de Agosto 2020 (23:59 hrs).

1. Descripción
   Este proyecto consiste en aplicar las técnicas de programación dinámica y aproximación para resolver el problema de la mochila 0-1. Este problema es NP-Complete en general, aunque usando programación dinámica se considera pseudo-polinomial.

   Considere una mochila 0-1 con capacidad máxima C con n objetos posibles de agregar los cuales son identificados mediante un número $i \in N = \{1, 2, 3, ..., n\}$. Además, cada objeto tiene un peso p_i y un valor v_i. El problema de la mochila consiste en elegir el subconjunto de los n objetos que maximiza el valor acumulado o ganancia de los objetos cuyo peso acumulado no supera la capacidad C de la mochila, tal como se expresa a continuación:

   $M(C) = max \sum_{i \in N} x_i v_i$
   tal que $\sum_{i \in N} x_i p_i \leq C$
   $x_i \in \{0, 1\} \; \forall \; i \in N$

   Asuma que $v_i \in N$ y $p_i \in N$

   a) Describa y proporcione una solución usando programación dinámica y establezca su complejidad asintótica del tiempo de ejecución.

   b) Qué puede decir respecto al tiempo de ejecución? Observa alguna diferencia a la forma en la cual hemos analizado el tiempo de ejecución en unidades anteriores? Establezca de que manera la complejidad obtenida para el problema puede incidir en la complejidad en términos de tiempo de ejecución. La complejidad asociada a la solución obtenida normalmente se conoce como pseudo polinomial. Investigue y comente por qué la solución para la mochila 0-1 se dice que es pseudo polinomial.

   c) Implemente el algoritmo usando el enfoque bottom-up mediante tabulación.

   d) Ahora, resuelva el problema de programación dinámica usando tabulación, pero esta vez en lugar de usar el peso en las columnas use los valores de los objetos. Describa la solución usando programación dinámica para este enfoque de solución.

e) Considere el mismo algoritmo anterior, pero ahora asuma que puede aceptar algo de error y no requiere obtener el óptimo. En este caso analice el caso en el cual los objetos se pueden agrupar por valor $\lfloor \frac{v_i}{x} \rfloor$, donde $x = (1-\beta)\frac{V}{n}$ y $0 < \beta < 1$, y $V = \sum_i^n v_i$. Esta solución conlleva a un algoritmo aproximado. Para este caso, implemente el algoritmo, analice el tiempo de ejecución asintótico del algoritmo en el peor caso, el error absoluto máximo y el factor de aproximación $\rho(n) = \frac{V^*}{V_a}$, donde V^* es la solución óptima y V_a la solución aproximada.

f) Considere que la entrada se proporciona en un archivo de entrada donde cada linea contiene el siguiente formato:
<*n*> <*C*> <*lista* p_i v_i>
Y la salida asociada a cada instancia de entrada debe ser:
<*n*> <*C*> <*lista* x_i>
Donde n es el número de objetos, C es la capacidad de la mochila, p_i peso de objeto i, v_i valor de objeto i, y $x_i \in \{0, 1\}$.
Ejemplo:

```
Entrada
4 100 34 169 23 152 62 44 2 224

Salida
4 545 1 1 0 1
```

2. Informe
Debe incluir el desarrollo para cada una de los ítemes descritos en punto anterior y su implementación en C o C++. Asegúrese de incluir un readme.txt que indique como compilar y ejecutar su proyecto.