Teoría de Algoritmos I

Trabajo práctico n.º 3

El trabajo práctico consiste en las dos partes que se listan a continuación.

Lineamientos básicos:

- el trabajo se realizará en grupos de tres o cuatro personas.
- la fecha de entrega es el **viernes 2 de diciembre de 2016**. Se debe entregar en el horario de clase en papel (informe + `código en monoespacio`), más una entrega en digital de código (.zip) e informe (.pdf) al correo de entregas del curso: `tps.7529rw@gmail.com`.

Contenidos

Clases de complejidad

Escribir el pseudocódigo de un algoritmo que resuelva cada uno de los siguientes problemas en tiempo polinomial, o bien demostrar que son NP-Completos.

1. Se tiene un grafo dirigido y pesado G, cuyas aristas tienen pesos que pueden ser negativos. Se pide devolver si el grafo tiene algún ciclo con peso negativo.

2. Se tiene un grafo dirigido y pesado G, cuyas aristas tienen pesos que pueden ser negativos. Se pide devolver si el grafo tiene algún ciclo con peso exactamente igual a cero.

3. Se tiene un conjunto de n tareas, cada una con un tiempo de ejecución $t_i \in R_+$, una fecha límite de finalización $d_i \in R_+$ y una ganancia $v_i \in R_+$ que será otorgada si se

finaliza antes que su tiempo límite. Se pide devolver si existe alguna planificación que obtenga una ganancia total mayor o igual a $k \in R_+$ sabiendo que no se pueden ejecutar dos tareas a la vez.

4. Se tiene un conjunto de n tareas, cada una con un tiempo de ejecución igual a 1, una fecha límite de finalización $d_i \in N$ y una ganancia $v_i \in R_+$ que será otorgada si se finaliza antes que su tiempo límite. Se pide devolver si existe alguna planificación que obtenga una ganancia total $k \in R_+$ sabiendo que no se pueden ejecutar dos tareas a la vez.

Algoritmos de aproximación

En el trabajo práctico número 2 hemos podido verificar empíricamente que obtener la solución exacta usando la técnica de programación dinámica podía resultar muy costoso para algoritmos no polinómicos.

El objetivo de esta sección del trabajo es obtener aproximaciones a estos problemas. Para los dos problemas se pide:

1. la implementación del algoritmo de aproximación,
2. un breve informe con los tiempos de ejecución sobre distintas instancias del problema, usando los conjuntos de datos usados en el trabajo práctico 2,
3. un resumen de las características de los algoritmos implementados, incluyendo el cálculo teórico de qué tan bien funcionan y explicando en qué situaciones su comportamiento es mejor o peor.

A continuación se especifica formalmente cada algoritmo.

El problema de la mochila

El algoritmo de aproximación de la **versión 0-1 del problema de la mochila** que usaremos está descripto en el capítulo 11.8 de Kleinberg y Tardos (2006).[1]

Dado un conjunto de elementos $S = \{1, 2, \ldots, n\}$ el problema consiste encontrar la sequencia $X = (x_1, x_2, \ldots, x_n)$ que maximice:

$$\sum_{i \in S} v_i x_i$$

con la restricción:

$$\sum_{i \in S} w_i x_i \leq W$$

donde $v_i, w_i, W \geq 0$ y $x_i \in \{0, 1\}$.

En esta ocasión, el algoritmo a usar para aproximar eficientemente una solución al problema usa la siguiente ecuación de recurrencia:

$$P(i, v) = \begin{cases} \text{si } v > \sum_{i=1}^{i-1} v_i : & w_i + P(i-1, v - v_i) \\ \text{si no} : & \min \begin{bmatrix} P(i-1, v) \\ w_i + P(i-1, max\{0, v - v_i\}) \end{bmatrix} \end{cases}$$

La solución X se obtiene buscando el valor máximo de v tal que $P(i, v) \leq W$.

El problema del viajante de comercio

Para aproximar el **problema del viajante de comercio** usaremos el algoritmo descripto en el capítulo 35.2.1 de Cormen.[2] Recordemos que el problema plantea que dado un grafo dirigido, *completo* y pesado, se encuentre un tour o ciclo hamiltoniano de costo mínimo que comience y termine en un vértice v_0 dado. (El costo se define como la suma de pesos de las aristas recorridas).

Para implementar este método se debe reutilizar el grafo implementado en el trabajo práctico 1, agregándole una primitiva para encontrar un árbol de tendido mínimo usando los algoritmos de Prim o Kruskal.

1. Otra referencia para este problema es el capítulo 9.2.4 de “Algorithms” por Dasgupta, Papadimitriou y Vazirani (1ra ed.) Ed. McGraw-Hill Education. ↩
2. Otra referencia para este problema es el capítulo 9.2.3 de “Algorithms” por Dasgupta, Papadimitriou y Vazirani (1ra ed.) Ed. McGraw-Hill Education. ↩