Práctico ALGABO: Introducción a la programación dinámica.

Mauricio Velasco

1. Se G_n el grafo de cadena no dirigido con vértices $1, 2\ldots, n$ (es decir el que tiene aristas $E(G) = \{(j, j+1) : j = 1, \ldots, n-1\}$ y sus reversas). Suponga que G tiene pesos en los vértices dados por la función $w : V(G_n) \to \mathbb{R}$ especificada mediante la fórmula $w(j) = j$ para $j = 1, 2, \ldots, n$

   a) Cuál cree usted que es el peso máximo posible de un subconjunto independiente de vértices de G_n? La respuesta debe depender del índice n.

   b) Utilice inducción y la formulación del problema mediante programación dinámica para *demostrar* que la respuesta que propuso en la parte (a) es correcta.

2. **Máximo conjunto independiente para grafos generales.** Para un grafo G y un subconjunto $W \subseteq V(G)$ *definimos* el grafo $G \setminus W$ como aquel que tiene vértices $V(G) \setminus W$ y cuyas aristas son las aristas de G que tienen ambos extremos en $V(G) \setminus W$.

   Si G es un grafo no dirigido finito con vértices $v_1, \ldots, v_n$ dotado de una función de pesos positivos en los vértices w defina el problema de optimización

   $$\Lambda(G) := \max\{w(T) : T \subseteq V(G) \text{ y } T \text{ es independiente en } G\}.$$

   donde $w(T) := \sum_{t\in T} w(t)$.

   a) Demuestre que la siguiente ecuación de Bellman se cumple

   $$\Lambda(G) = \max\left(\Lambda(G \setminus \{v_n\}), w(v_n) + \Lambda(G \setminus \mathrm{star}(v_n))\right)$$

   donde $\mathrm{star}(v_n)$ es el conjunto que consiste de v_n y de todos los vértices adyacentes a v_n.

b) Proponga un algoritmo de programación dinámica para calcular $\Lambda(G)$ usando la ecuación del renglón anterior.

c) Verdadero ó Falso? El algoritmo propuesto permite calcular $\Lambda(G)$ en tiempo lineal $O(n)$ para *cualquier* grafo?

3. **Knapsack**. Considere la siguiente instancia del problema de Knapsack para un saco con capacidad $C = 9$

Item	Valor	Tamaño
1	1	1
2	2	3
3	3	2
4	4	5
5	5	4

a) Escriba el código en Python de una implementación de la solución del problema de Knapsack dada la capacidad, los items y los valores.

b) Usando su programa encuentre el valor óptimo del problema de arriba y los items que constituyen una solución de máximo valor.

c) Escriba la sucesión de subproblemas que su implementación resuelve con sus respectivos valores óptimos.

4. Proponga un algoritmo de programación dinámica y encuentre la ecuación de Bellman correspondiente para resolver el siguiente problema: *Dadas capacidades enteras positivas C_1 y C_2 y una colección de n items con sus tamaños y capacidades (enteras, positivas), encuentre dos subconjuntos* **disjuntos** *de items S_1 y S_2 con valor máximo total posible, entre aquellos que pueden meterse en los sacos. Es decir (S_1, S_2) es un maximizador de la función $W(T_1, T_2) := \sum_{i \in T_1} w(i) + \sum_{i \in T_2} w(i)$ entre las parejas (T_1, T_2) de conjuntos de $[n]$ tales que $T_1 \cap T_2 = \emptyset$ y $\sum_{v \in T_i} w(v) \leq C_i$ para $i = 1, 2$.*