

Introducción a la programación dinámica

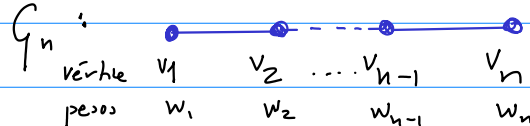
La Programación Dinámica es un paradigma para resolver problemas de optimización en familias, la explicaremos mediante un ejemplo

PROBLEMA: ^{WIS} [Weighted independent set] Sea G un grafo no dirigido con pesos $w \geq 0$ en los vértices.

$$WIS(G) = \max \left\{ \bar{w}(S) : \begin{array}{l} S \subseteq V(G) \\ S \text{ independiente} \end{array} \right\}$$

$$\text{donde } \bar{w}(S) := \sum_{s \in S} w(s).$$

Para explicar programación dinámica nos enfocaremos en "Cadenas" G_n :



usando programación dinámica construiremos un algoritmo muy eficiente (de tiempo $O(m+n)$) para resolver WIS en cadenas (aunque G_n tiene una cantidad exponencial de conjuntos independientes)

IDEA CLAVE: Preguntamos: Si tuviéramos una solución óptima, ¿qué propiedades tendría?
¿Cómo se relaciona con soluciones óptimas de problemas más pequeños?

Suponga que S^* es una solución óptima, es decir $\bar{w}(S^*) = WIS(G_n)$ y nos preguntamos si el último vértice v_n pertenece o no a S^*

$S^* \subseteq G_n$ independiente con $\bar{w}(S^*) = \text{WIS}(G_n)$.

$\overset{\circ}{\cap} v_n \in S^* ?$

NO:

Todos los v rtices de S^*
pertenecen a G_{n-1}
as  que S^*
es  ptimo para G_{n-1}

(de lo contrario, si hubiera
 $T \subseteq G_{n-1}$ T independiente
con
 $\bar{w}(T) > \bar{w}(S^*)$
 $T \subseteq G_n$ lo que
contradice la optimalidad
de S^*)

$$\bar{w}(S^*) = \text{WIS}(G_{n-1})$$

SI

$v_{n-1} \notin S^*$ (porque S^* es indep.)

luego $S' := S^* \setminus \{v_n\} \subseteq G_{n-2}$
y S' es  ptimo en G_{n-2}

(de lo contrario $\exists T \subseteq G_{n-2}$
independiente con $\bar{w}(T) > \bar{w}(S')$)

luego $T \cup \{v_n\}$ es independiente en
 G_n y $\bar{w}(T \cup \{v_n\}) = w(v_n) + \bar{w}(T)$

contradice la optimalidad de S^* $\bar{w}(S^*) = w(v_n) + \bar{w}(S')$

$$\bar{w}(S^*) = w(v_n) + \text{WIS}(G_{n-2})$$

Como $\bar{w}(S^*) = \text{WIS}(G_n)$ concluimos que los
valores  ptimos de los problemas WIS en cadenas
tienen la siguiente relaci n, llamada **ECUACI N**

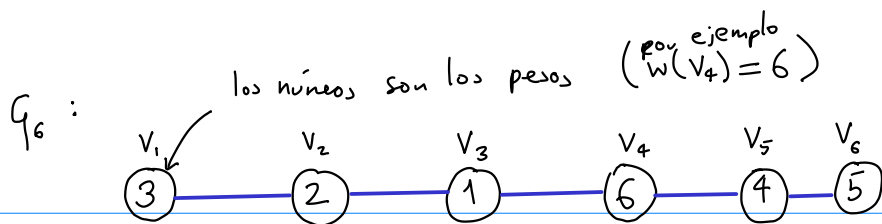
DE BELLMAN: (*)

Lema: Si $n \geq 2$ entonces

$$(*) \text{WIS}(G_n) = \max \{ \text{WIS}(G_{n-1}), w(v_n) + \text{WIS}(G_{n-2}) \}$$

Este lema es muy  til porque si sabemos $\text{WIS}(G_1)$ y
 $\text{WIS}(G_2)$ (f cil), la f rmula (*) nos dice como
calcular $\text{WIS}(G_3), \text{WIS}(G_4), \dots$

Ve moslo en un ejemplo concreto (p gina
siguiente).



$$WIS(G_1) = 3$$

$$WIS(G_2) = 3$$

$$WIS(G_3) = \max(WIS(G_2), 1 + WIS(G_1)) = \max(3, 4) = 4$$

$$WIS(G_4) = \max(4, 6 + WIS(G_2)) = \max(4, 9) = 9$$

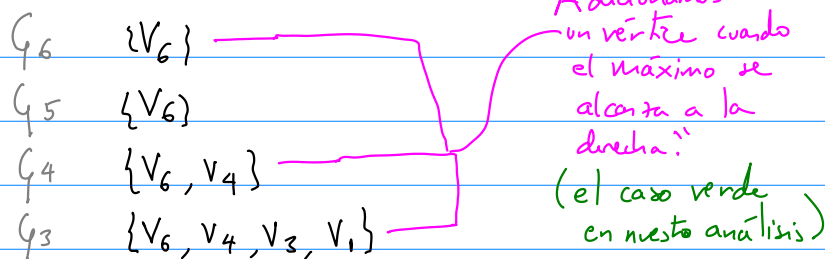
$$WIS(G_5) = \max(9, 4 + 4) = 9$$

$$WIS(G_6) = \max(9, 5 + 9) = 14$$

Concluimos que el máximo peso de un subconjunto independiente es **14**. ¡Cómo encontrar un conjunto indep con ese peso?

Marcamos en — arriba en cuál de las dos alternativas se alcanza el máximo y eso nos dice qué hacer, empezando por V_n y caminando hacia atrás...

Concretamente: S^* :



En general, tenemos la siguiente conclusión:

Para cualquier grafo cadena con pesos positivos el siguiente algoritmo produce un conjunto independiente de máximo peso: **WIS-Cadena es fácil!**

(1) Usamos la recursión de Bellman para calcular un vector

$$A[i] = WIS(G_i) \quad i=1, \dots, n$$

(2) Reconstruimos el arreglo A de adelante hacia atrás seleccionando los vértices del conjunto independiente maximal.

Tiempo $O(n)$, muy rápido.

Este primer ejemplo funcionó MUY BIEN porque los subproblemas venían ordenados, lo que simplificó fuertemente la aplicación del método

$$q_1 \subseteq q_2 \subseteq \dots \subseteq q_n$$

La programación dinámica se puede aplicar a muchos, muchos otros problemas.

PROBLEMA 2: [Knapsack o "de la maleta"]

Tenemos una tabla de ítems que tienen un cierto volumen y un valor

item	volumen	valor
1	h_1	p_1
2	h_2	p_2
\vdots	\vdots	\vdots
n	h_n	p_n

y tenemos una maleta con capacidad total C .

¿Cuál es la colección de ítems más valiosa que cabe en la maleta?

$$\text{Knapsack}(n, C) = \max \left\{ \bar{p}(S) : S \subseteq [n] \atop \left[\sum_{s \in S} h_s \leq C \right] \right\}$$

con $\bar{p}(S) = \sum_{s \in S} p_s$

$\{1, 2, \dots, n\}$
 $!!$
 los ítems caben en la maleta

Aplicamos la idea clave:

¿Si $S^* \subseteq [n]$ es una selección óptima contiene al ítem n -ésimo?

NO \rightarrow SI

Lema: [Ecuación de Bellman para Knapsack]

$$K(n, C) = \max \left(K(n-1, C), p_n + K(n-1, C - h_n) \right)$$