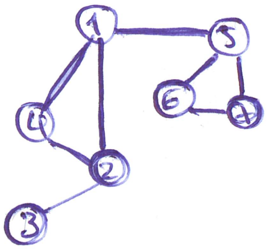
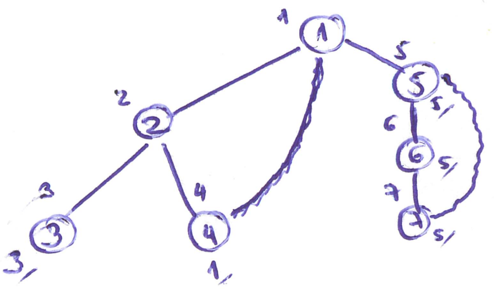
Feb. 12 | PREDA - Sem 1 MoH

① Aplicacion del algoritmo de calculo de los ptos de articulación (y de su arbol de recubrimiento asociado) da el vector bajo[]



a) [1, 1, 3, 1, 5, 5, 5]
b) [1, 2, 3, 4, 1, 6, 7]
c) [1, 3, 3, 1, 1, 3, 3]
d) Ninguno

Se realiza un recorrido en profundidad por el orden de los indices de c/u nodos, queda:



N Valor de Orden
N/ Vector bajo[]
~~ Arista de retroceso

Realizamos entonces el recorrido del grafo empezando por el nodo más a la izquierda, el nodo 3.

Se toma en cuenta el nº orden de dicho nodo, el nº orden de cualquiera de sus aristas de retroceso, y el valor del vector bajo[v]

El valor para bajo[] para el nodo 3 es el valor comprendido entre su nº de orden (3), el nº orden de cualquier arista de retroceso que tenga (este caso no tiene) y el valor de bajo[] para alguno de sus hijos, por lo que el valor de bajo[] para el nodo 3 es 3.

Para el nodo 4 cuyo valor de bajo[] es el valor entre su número de orden (4) el nº orden de su arista de retroceso (1) y el valor de bajo[] de sus hijos (que no tiene) por lo que bajo[] para el nodo 4 es 1.

bajo[2] { nº orden: 2, bajo[] aristas retroceso: 0, bajo[] nodos hijos: 3 y 1 } menor valor es 1

bajo[5] { nº orden: 5, bajo[] arist. retroc: 0, bajo[] nodos hijos: 5 y 5 } menor valor es 5

bajo[6] { nº orden: 6, bajo[] arist. retroceso: 0, bajo[] nodos hijos: 5 } menor valor es 5

bajo[7] { nº orden: 7, bajo aristas[] retroceso: 5, bajo[] nodos hijos: 5 } menor valor es 5

Feb. 12 | PRED/A | Sem + Mod. A

1) 2ª parte.

Este grafo posee 3 ptos articulacion (PDA)

Los PDA o Vertices de Corte (VDC) son un vertice de un grafo, tal que si se elimina del grafo, se produce un incremento en el nº de componentes conexos.

Los PDA que hay en este grafo son 1, 2, 5. Calculamos el valor del vector bajo comenzando en el primer PDA y por número de orden.

Nodo	1	2	3	4	5	6	7
bajo[ ]	1	1	3	1	5	5	5

Respuesta: A

2) Prob. mochila versión: objetos no fraccionados, solucionado con programación dinámica

5 objetos con pesos: 1, 3, 4, 5 y 7 y beneficios: 2, 5, 10, 14 y 15

Volumen maximo = 8

Tabla de resultados parciales en la fila correspondiente al objeto del peso 5, si dichos objetos se consideran en orden creciente de pesos.

a) 0 2 2 5 10 12 12 15 15
b) 0 2 2 5 10 14 16 16 19
c) 0 2 2 5 10 12 14 16 19
d) Ninguna

Lo que se pretende es maximizar el beneficio. Al ser objetos no fraccionables, podemos hacer el calculo mediante dos algoritmos distintos pero nos dice que con prog. dinámica.

PROBLEMA DE LA MOCHILA
- → Obj. fraccionables → Algoritmos Voraces (Alg. de optimización)
- → Obj no fraccionables → Prog. Dinamica (mejor coste espacial O(nC))
  - ↳ Ramificacion y Poda (mayor coste Exponencial)

Valor maximo entre:
- Valor fila anterior en la misma columna
- Beneficio del volumen + X

X: fila anterior, Columna Y

Y: Capacidad − Volumen

Caso base

CAPACIDADES

Pesos	0	1	2	3	4	5	6	7	8	Beneficios
0	0	0	0	0	0	0	0	0	0	0
1	0	2	2	2	2	2	2	2	2	2
3	0	2	2	5	5+2 7	7	7	7	7	5
4	0	2	2	5	10	10+2 12	12	10+5 15	15	10
$\boxed{5}$	0	2	2	5	10	14	14+2 16	16	14+5 19	14
7	6	2	2	5	10	14	16	16	19	15

5+3=8 → 19>17

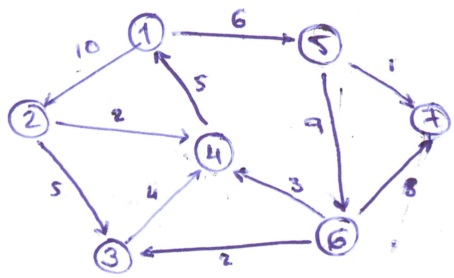
7+1=8 →

Respuesta: B

2

feb 12 | PIREDA   Semest. …

③ Grafo dirigido. Detalle el valor del vector distancias especial[] en el paso del algoritmo de Dijkstra en el nodo v=7 tomando como origen el nodo 1



a) [10, 15, 12, 6, 15, 7]
b) [10, ∞, ∞, 6, 15, 7]
c) [10, ∞, ∞, 6, ∞, ∞]
d) Ninguna

Respuesta = B

Nodos que han salido.	Nodos que no han salido	Vector Distancia especial[]						Predecesores					
		2	3	4	5	6	7	2	3	4	5	6	7
1	2, 3, 4, 5, 6, 7	10	∞	∞	6	∞	∞	1	1	1	1	1	1
1, 5	2, 3, 4, 6, 7	10	∞	∞	6	15	7	1	1	1	1	5	5
1, 5, 7	2, 3, 4, 6	10	∞	∞	6	15	7	1	1	1	1	5	5
1, 5, 7, 2	3, 4, 6	10	15	12	6	15	7	1	2	2	1	5	5
1, 5, 7, 2, 4	3, 6	10	15	12	6	15	7	1	2	2	1	5	5
1, 5, 7, 2, 4, 3	6	10	15	12	6	15	7	1	2	2	1	5	5
~~1, 5, 7, 2, 4, 3, 6~~		10	15	12	6	15	~~7~~	~~1~~	2	2	~~1~~	~~5~~	~~5~~

④ Prob. mochila con objetos fraccionables, tenemos

n = 8 objetos

pesos ⇒ W = 8, 7, 6, 5, 4, 3, 2, 1

beneficios ⇒ V = 10, 9, 8, 7, 6, 5, 4, 3 la capacidad de la mochila es M=9?

¿Beneficio óptimo suponiendo que la capacidad de la mochila es M=9?

a) 12 b) 15 c) 16'5 d) Ninguna

Resolvemos con un alg. voraz. La fórmula a aplicar sería:

$$\frac{\text{Beneficio}}{\text{peso}} = \frac{V_i}{W_i}$$

Se hace la sumatoria de mayor a menor beneficio obtenido

$\Sigma_v = 3 + 4 + 5 + \left(\frac{3}{4} \cdot 6\right) = 16'5$

$\frac{3}{4}$ 4

$\Sigma_w = 1 + 2 + 3 + \left(\frac{3}{\cancel{4}} \cdot \cancel{4}\right) = 9$

5

Peso (w)	Beneficio (v)	Vi/Wi
8	10	10/8 = 1'25
7	9	9/7 = 1'286
6	8	8/6 = 4/3 = 1'3
5	7	7/5 = 1'4
4	6	6/4 = 3/2 = 1'5
3	5	5/3 = 1'6
2	4	4/2 = 2
1	3	3/1 = 3

Respuesta: C

~~Sept~~ | Feb 12 | PREDA SIM. A

~~Parte 3~~

(5) Vector $v[1..n] = [6, 5, 5, 2, 5, 5, 3, 2, 2]$ ¿Cuál es cierta?

a) V es un montículo de maximos

b) v no es un montículo de maximos porque $v[4]=2$ debe ser flotado

c) v no es un montículo de máximos porque $v[4]=2$ debe ser hundido

d) Ninguna

Los montículos son un tipo especial de árbol binario que se implementan sobre vectores con las siguientes propiedades.

- Es un arbol balanceado y completo (los nodos internos tienen siempre 2 hijos) con la posible excepcion de un único nodo cuando el nº elementos es par.
- Cada nodo contiene un valor mayor o igual que el de sus nodos hijos (montículo de máximos) o menor o igual (montículo de mínimos).

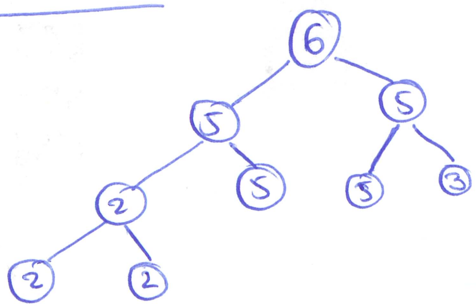
Los nodos de profundidad K estan situados en las posiciones 2^K y siguientes del vector hasta la posición $2^{K+1}-1$.

La inserción y borrado de elementos es muy eficiente, tiene un coste $O(\log n)$

las operaciones que podemos realizar son:

- flotar. Intercambiar con el nodo inmediatamente a un nivel superior
- hundir. Intercambiar con el nodo hijo de menor valor (montí. mínimos)
- Insertar. Añadir el nodo al final, y flotar a su sitio correspondiente.
- Primero. Lleva asociado un coste constante $O(1)$
- Obtener Cima y Borrarla. Situar el último elemento en la cima y hundir esta.

Representación:



Respuesta: A

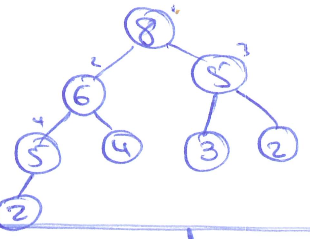
B no lo es, ya que si intercambio v[4]- por v[2]=5, esto hace que el montículo no sea máximo.

⑥ V almacena nºs enteros en orden creciente. Se desea averiguar si existe algun elemento que cumpla V[i]=i ¿Cual sería la estrategia más eficiente para resolver el problema?

a) Algoritmo voraz
b) Divide y vencerás
c) Programación dinámica
d) Ninguna

2021.02.02

① $v = [8, 6, 5, 5, 4, 3, 2, 2]$



D) Vector v es un montículo de maximos.

② Tabla hash $M = 11$ elementos
funcion $h1(k) = k \text{ MOD } m$
claves $= 22, 13, 11, 24, 34, 12$
hashing cerrado con recorrido lineal usando $h(k,i) = (h1(k) + 2i) \text{ MOD } m$

A) Ninguna.

③ Grafo dirigido
Especial[ ] algoritmo Dijkstra
selecciona $v = 5$, nodo origen $v = 1$.

④ Cuál es cierta

c) La eficiencia del algoritmo de ordenación rapida (quicksort), cuando se

⑤ Mochila - objetos no fraccionables con progr. dinámica

Capacidades → Beneficios

Pesos	0	1	2	3	4	5	6	7	8	9	10	Beneficios
0	0	0	0	0	0	0	0	0	0	0	0	
2	0	0	5	5	5	5	5	5	5	5	5	15
3	0	0	5	10	10	10	10	10	10	10	10	10
5	0	0	5	10	10	16	16	21	26	26	31	16
10	0	0	5	10	10	16	16	21	26	26	40	40

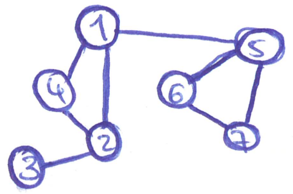
d) Para el peso 5 es:
0 0 5 10 10 16 16 21 26 26 31

Feb. 2012 - S.1 (A)

Tiempo:

Ayudas: S☐ N☐

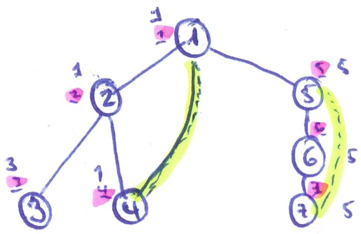
1. Grafo no dirigido
Ap. cálculo de ptos de articulación. ¿Vector bajo[]?



a) [1, 1, 3, 1, 5, 5, 5]
b) [1, 2, 3, 4, 1, 6, 7]
c) [1, 3, 3, 1, 1, 3, 3]
d) Ninguna.

---

1. Se realiza un recubrimiento en profundidad, por orden de los índices de cada nodo, quedando.



Aristas de retroceso
Valor de orden
Vector bajo[]

2. Se dibujan las aristas de retroceso, son aristas que conectan nodos descendientes con nodos antecesores en el árbol.
En nuestro caso existía una de estas aristas de retroceso, desde 1 al nodo 4, y otra de ellas desde en nodo 5 al nodo 7.
Realizamos entonces el recorrido del grafo empezando por el nodo más a la izquierda de todos, es decir, el nodo 3. Se toma en cuenta el nº de orden de dicho nodo, el nº de orden de cualquiera ~~(en este caso no tiene))~~ ~~y el~~ de sus aristas de retroceso, ~~(que tenga (en este caso ... el valor de bajo[v]~~ valor de bajo[v] ~~(para alguno de sus hijos, por lo que~~ comprendido

El valor de bajo[] para el nodo 3 es el valor de cualquier entre el número de orden (3), el número de orden de cualquier arista de retroceso que tenga (que en este caso no tiene), y el valor de bajo[] el valor bajo[] para alguno de sus hijos, por lo que el valor de bajo[] para el nodo 3 es: 3.
El siguiente es el nodo 4, cuyo valor de bajo[] es el valor entre su número de orden (4), el nº de orden de cualquiera de sus aristas de retroceso (1), y el valor de bajo[] de sus hijos (que no tiene), por lo que el valor de bajo[] para el nodo 4 es: 1.

bajo[2] { nº orden: 2; bajo[] arista retroceso: 0; bajo[] nodos hijos: 3 y 1 } el menor valor es 1.

bajo[5] { Nº ord: 5; Bajo[] arist: 0; bajo[] hijos: 5 y 5 } menor valor es 5

bajo[6] { Nº orden: 6; bajo[] arist: 0; Bajo[] hijos: 5 } menor valor es 5

bajo[7] { Nº ord. 7; bajo[] ari: 5; bajo[] hijos: 0 } me... va...

bajo[1] { nº ord: 1; bajo[] arist: 0; bajo[] hijos: 1, } menor valor 1

Este grafo posee 3 ptos articulacion (PDA)
Los PDA o Vertices de Corte (VDC) son un vertice del grafo, tal que, si se elimine del grafo, se produce un incremento del nº de componentes ~~c~~ conexos.

Los PDA que hay son 1, 2, 5. Calculamos el valor del vector bajo, comenzando en el 1º PDA y por nº de orden:

Nodo	bajo[v]
1	1
2	1
3	3
4	1
5	5
6	5
7	5

② Mochila con obj. no fraccionados, solucionado con programación dinámica

5 obj. con pesos	1	3	4	5	7
con beneficio	2	5	10	14	15

Volumen max: 8

→ ¿Contenido de la tabla de result. parciales en la fila al objeto de peso 5, si dicho obj. se consideran en orden creciente de pesos

a) 0 2 2 5 10 12 12 15 15
b) 0 2 2 5 10 14 16 16 19
c) 0 2 2 5 10 12 14 16 19
d) Ninguna

Dibujamos una tabla de dos dimensiones. Una Volumen/Peso (objetos) y otra columna con las capacidades/Volumen de la propia mochila.
Las capacidades se comprenden desde 0 hasta la capacidad maxima de la mochila (8).

La primera fila y columna son el caso base, por lo que el valor de cada celda en estos casos es 0. El valor en cada una de las celdas será el valor maximo entre:

- Valor Fila Anterior en la misma columna
- Beneficio del Volumen + X

X = Fila anterior, Columna Y
Y = Capacidad - Volumen

Capacidades

Pesos	0	1	2	3	4	5	6	7	8	Beneficios
0	0	0	0	0	0	0	0	0	0	
1	0	2	2	2	2	2	2	2	2	2
3	0	2	2	5	7	7	7	7	7	5
4	0	2	2	5	10	12	12	15	15	10
5	0	2	2	5	10	14	16	16	$\boxed{19}$	14
7	0	2	2	5	10	14	16	16	19	15

$\underline{2+5}$

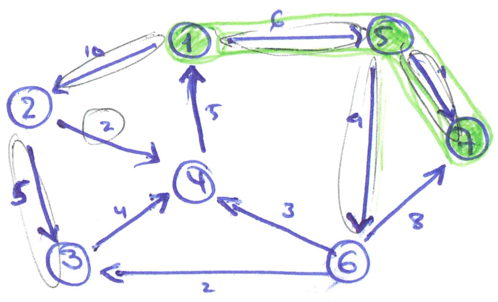
Capacidades.

Peso	0	1	2	3	4	5	6	7	8
5	0	2	2	5	10	14	16	16	19

6 = 5 + 1
14 + 2 = 16
8 = 7 + 1
~~15 + 2 = 17~~
5 + 14 = 19
8 = 5 + 3

2/

③ Grafo dirigido



Valor del vector especial[] con el algoritmo de Dijkstra en el nodo v=7 tomando origen el nodo 1.

a) [10, 15, 12, 6, 15, 7]
b) [10, ∞, ∞, 6, 15, 7]
c) [10, ∞, ∞, 6, ∞, ∞]
d) Ninguna

Nodos que han salido	nodos que no han salido	Vector distancia especial[]						Predecedores					
		2	3	4	5	6	7	2	3	4	5	6	7
1	2, 3, 4, 5, 6, 7	10	∞	∞	6	∞	∞	1	1	1	1	1	1
1, 5	2, 3, 4, 6, 7	10	∞	∞	6	6+9 15	6+1 7	1	1	1	1	5	5
1, 5, 7	2, 3, 4, 6	10	∞	∞	6	15	7	1	1	1	1	5	5
1, 5, 7, 2	3, 4, 6	10	15	12	6	15	7	1	2	2	1	5	5
1, 5, 7, 2, 4	3, 6	10	15	12	6	15	7	1	2	2	1	5	5
1, 5, 7, 2, 4, 3	6	10	15	12	6	15	7	1	2	2	1	5	5