Algoritmos y Estructuras de Datos II
Clase 13

Carreras:

Licenciatura en Informática

Ingeniería en Informática

2024

Unidad III

Técnicas de diseño de algoritmos

1

1 1

1 2 1

1 3 3 1

1 4 6 4 1

1 5 10 10 5 1

1 6 15 20 15 6 1

1 7 21 35 35 21 7 1

1 8 28 56 70 56 28 8 1

1 9 36 84 126 126 84 36 9 1

1 10 45 120 210 252 210 120 45 10 1

1 11 55 165 330 462 462 330 165 55 11 1

1 12 66 220 495 792 924 792 495 220 66 12 1

1 13 78 286 715 1287 1716 1716 1287 715 286 78 13 1

1 14 91 364 1001 2002 3003 3432 3003 2002 1001 364 91 14 1

1 15 105 455 1365 3003 5005 6435 6435 5005 3003 1365 455 105 15 1

Programación Dinámica(2)

Problema de la mochila 0/1

Datos:
Se tienen n objetos y una mochila para llevar en una mochila.
Cada objeto tiene un peso: p_i y un beneficio asociado: b_i
La mochila puede cargar un peso máximo dado: M

Objetivo: *llenar la mochila de tal manera que se* ***maximice el beneficio*** *de los objetos transportados, respetando la limitación de la capacidad impuesta.*

- Se ha mostrado que el problema de la mochila no obtiene la solución óptima con la técnica greedy cuando los objetos no se pueden fragmentar.

- Se va a plantear ahora una solución con programación dinámica para ese caso.

Problema de la mochila 0/1

Solución: vector X

$x_i = 0$ si el objeto i no se carga en la mochila

$x_i = 1$ si el objeto i si se carga en la mochila

Maximizar la cantidad: $\sum_{i=1}^{n} b_i x_i$ Restricción: $\sum_{i=1}^{n} p_i x_i \leq M$

- Para resolver este problema con programación dinámica se prepara una tabla ***V(1..n, 0..M)*** que tiene una *fila* por cada *objeto* disponible (desde 0 a n) y una *columna* para cada *peso* (desde 0 a M).

Objeto / Limite de peso	0	1	2	3	***j***	…	M
Objeto 1: p1 , b1							
Objeto 2: p2 , b2							
Objeto ***i***					***V(i,j)***		
…							
Objeto n: pn bn							

Problema de la mochila 0/1

- En esa matriz, ***V(i,j)*** será el beneficio máximo de los objetos que se puede transportar si el límite de peso es ***j*** incluyendo objetos numerados de ***1*** a ***i***.

Objeto / Limite de peso	0	1	2	3	*j*	…	**M**
Objeto 1: p1 , b1							
Objeto 2: p2 , b2							
Objeto i					***V(i,j)***		
…							
Objeto n: pn bn							***V(n,M)***

- La solución del problema se encuentra entonces en ***V(n,M)*.**

- Se puede llenar la matriz fila por fila o columna por columna.

Problema de la mochila 0/1

En cada paso (i,j) hay que decidir entre 2 opciones para asignar al elemento de la tabla $V(i,j)$:

1) no agregar el objeto i a la carga:

$$V(i,j) = V(i-1,j)$$

2) agregar el objeto i , incrementando el beneficio en ***bi***, pero reduciendo la capacidad de carga en su peso ***pi***:

$$V(i,j) = V(i-1, j-pi)+bi$$

De las 2 opciones: elegir la mayor, de modo que se define:

$$V(i,j) = \text{Maximo}(\ V(i-1, j), V(i-1, j-pi)+bi\ )$$

Para las entradas fuera de límites se define:

- $V(0,j)=0$ cuando $j\geq 0$
- $V(i,j) = -\infty$ para todo i , cuando $j<0$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1												
Objeto 2: p2=2 b2=6												
Objeto 3: p3=5 b3=18												
Objeto 4: p4=6 b4=22												
Objeto 5: p5=7 b5=28												

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0											
Objeto 2: p2=2 b2=6	0											
Objeto 3: p3=5 b3=18	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j) = \mathrm{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0											
Objeto 3: p3=5 b3=18	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j)= \mathrm{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0											

$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i-1,j-pi)+bi)$$

- La entrada **V(5,11)=40** da el beneficio máximo de la carga de 11 unidades de peso con objetos de tipo 1 al 5.

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

- Cuales son los objetos que se cargan en la mochila para obtener ese beneficio de 40 ?

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

- Cuales son los objetos que se cargan en la mochila para obtener ese beneficio de 40 ?
- No se carga ningún objeto de tipo 5.

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

- Cuales son los objetos que se cargan en la mochila para obtener ese beneficio de 40 ?
- Se carga un objeto del tipo 4 que aporta beneficio 22.

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

- Cuales son los objetos que se cargan en la mochila para obtener ese beneficio de 40 ?
- Se carga un objeto del tipo 3 que aporta beneficio 18.

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

- Cuales son los objetos que se cargan en la mochila para obtener ese beneficio de 40 ?
- Ningún objeto mas.

Problema de la mochila 0/1

Ejemplo: n=5 objetos: pi=(1,2,5,6,7) bi=(1,6,18,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	1	1	1	1	1	1	1	1	1	1
Objeto 2: p2=2 b2=6	0	1	6	7	7	7	7	7	7	7	7	7
Objeto 3: p3=5 b3=18	0	1	6	7	7	18	19	24	25	25	25	25
Objeto 4: p4=6 b4=22	0	1	6	7	7	18	22	24	28	29	29	40
Objeto 5: p5=7 b5=28	0	1	6	7	7	18	22	28	29	34	35	40

SOLUCION DEL PROBLEMA :

- Máximo beneficio que se puede obtener de la carga: B = 40
- Objetos que se cargan: X = (0,0,1,1,0)

Problema de la mochila 0/1

Algoritmo

- Ejercitación

Costo del algoritmo

- Para construir la tabla se necesita O(n*M) en almacenamiento y en tiempo de ejecución.

- La composición de la carga de beneficio máximo se puede realizar agregando un costo adicional de O(n+M) en tiempo de ejecución.

Problema de la mochila múltiple

Datos:

- Se tienen ***n*** tipos de objetos y una mochila para llevarlos.
- De un objeto de tipo ***i*** se pueden elegir tantas unidades como se quiera para cargar en la mochila.
- Cada objeto de tipo ***i*** tiene un peso: ***pi*** y un beneficio asociado: ***bi***.
- La mochila puede cargar un peso máximo dado: ***M***.

Objetivo: *llenar la mochila de tal manera que se **maximice el beneficio** de los objetos transportados, respetando la limitación de la capacidad impuesta.*

Problema de la mochila múltiple

Solución : un vector **X**

x_i = cantidad de objetos de tipo i que se cargan en la mochila

Maximizar la cantidad: $\sum_{i=1}^{n} b_i x_i$ Restricción: $\sum_{i=1}^{n} p_i x_i \leq M$

- Se almacena la solución en una tabla ***V(1..n, 0..M)*** que tiene una fila por cada tipo de objeto disponible y una columna para cada peso desde 0 a M.

- En esa matriz, ***V(i,j)*** será el valor máximo de los objetos que se puede transportar si el límite de peso es ***j*** incluyendo objetos de los tipos de 1 a ***i***.

- La solución del problema se encuentra entonces en ***V(n,M).***

- Se puede llenar la matriz fila por fila o columna por columna.

Problema de la mochila múltiple

En cada paso *(i,j)* hay que decidir entre 2 opciones para asignar a la entrada en la tabla *V(i,j):*

1) no agregar una unidad del objeto de tipo *i* a la carga, entonces:

$$V(i,j) = V(i-1,j)$$

2) agregar una unidad más del objeto del tipo *i* ,incrementando el beneficio en ***bi***, pero reduciendo la capacidad de carga en su peso ***pi***, entonces:

$$V(i,j) = V(i, j-pi) + bi$$

De las 2 opciones: elegir la mayor, de modo que se define:

$$V(i,j) = \mathbf{Maximo}(\ V(i-1, j), V(i, j-pi)+bi\ )$$

Problema de la mochila múltiple

Relación de recurrencia:

$V(i,j)=0$	**si** $j=0$
$V(i,j)=(j$ **div** $p1)\ b1$	**si** $i=1$
$V(i,j)= V(i-1, j)$	**si** $pi>j$
$V(i,j)= Max\ (V(i-1, j), V(i,j-pi)+bi)$	**en otro caso**

Costo del algoritmo:

Para construir la tabla se necesita O(n*M) en almacenamiento y en tiempo de ejecución.

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1												
Objeto 2: p2=2 b2=6												
Objeto 3: p3=5 b3=21												
Objeto 4: p4=6 b4=22												
Objeto 5: p5=7 b5=28												

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i,j-pi)+bi)$$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0											
Objeto 2: p2=2 b2=6	0											
Objeto 3: p3=5 b3=21	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$V(i,j)=$ Maximo$(V(i-1,j), V(i,j-pi)+bi)$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 2: p2=2 b2=6	0											
Objeto 3: p3=5 b3=21	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$V(i,j)$= Maximo$(V(i-1,j), V(i,j-pi)+bi)$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 2: p2=2 b2=6	0	1	6	7	12	13	18	19	24	25	30	31
Objeto 3: p3=5 b3=21	0											
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i,j-pi)+bi)$$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 2: p2=2 b2=6	0	1	6	7	12	13	18	19	24	25	30	31
Objeto 3: p3=5 b3=21	0	1	6	7	12	21	22	27	28	33	42	43
Objeto 4: p4=6 b4=22	0											
Objeto 5: p5=7 b5=28	0											

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i,j-pi)+bi)$$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Objeto 1: p1=1 b1=1	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**
Objeto 2: p2=2 b2=6	**0**	**1**	**6**	**7**	**12**	**13**	**18**	**19**	**24**	**25**	**30**	**31**
Objeto 3: p3=5 b3=21	**0**	**1**	**6**	**7**	**12**	**21**	**22**	**27**	**28**	**33**	**42**	**43**
Objeto 4: p4=6 b4=22	**0**	**1**	**6**	**7**	**12**	**21**	**22**	**27**	**28**	**33**	**42**	**43**
Objeto 5: p5=7 b5=28	**0**											

$$V(i,j) = \text{Maximo}(V(i-1,j),\ V(i,j-pi)+bi)$$

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Tabla V (1..5,0..11)

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 2: p2=2 b2=6	0	1	6	7	12	13	18	19	24	25	30	31
Objeto 3: p3=5 b3=21	0	1	6	7	12	21	22	27	28	33	42	43
Objeto 4: p4=6 b4=22	0	1	6	7	12	21	22	27	28	33	42	43
Objeto 5: p5=7 b5=28	0	1	6	7	12	21	22	28	29	34	42	43

- La entrada V(5,11)=43 da el beneficio máximo de la carga de 11 unidades de peso.
- SOLUCION DEL PROBLEMA, beneficio máximo: B=43

Problema de la mochila múltiple

Ejemplo: 5 objetos: pi=(1,2,5,6,7) bi=(1,6,21,22,28) M=11

Objeto / Limite de peso	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 1: p1=1 b1=1	0	1	2	3	4	5	6	7	8	9	10	11
Objeto 2: p2=2 b2=6	0	1	6	7	12	13	18	19	24	25	30	31
Objeto 3: p3=5 b3=21	0	1	6	7	12	21	22	27	28	33	42	43
Objeto 4: p4=6 b4=22	0	1	6	7	12	21	22	27	28	33	42	43
Objeto 5: p5=7 b5=28	0	1	6	7	12	21	22	28	29	34	42	43

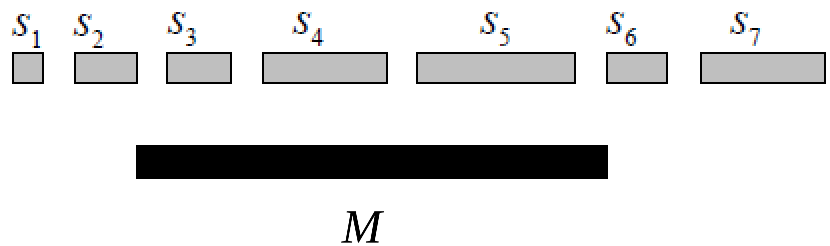
- Cuales son los objetos que carga en la mochila para obtener ese beneficio de 43 ?
- Existe una única solución del problema?

Problema de la suma exacta

Dado un vector de *n* números: $S=(s1, s2, s3,...sn)$ determinar si existe algún subconjunto que sume exactamente M.

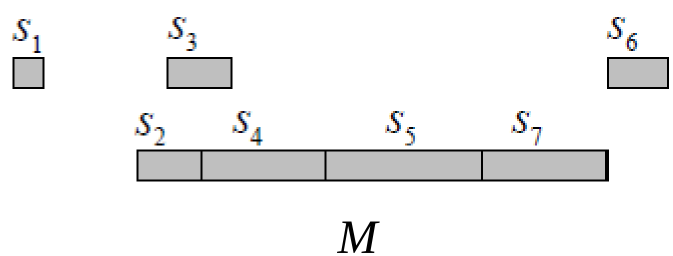
Ejemplo: n=7, S=(1,2,2,4,5,2,4) M=15

- Datos:



- Solución del problema: Si existe un subconjunto de suma 15

Una Solución:



Problema de la suma exacta

Dado un vector de *n* números: $S=(s1, s2, s3,\ldots sn)$ determinar si existe algún subconjunto que sume exactamente *M*.

Solución:

- Se almacena la solución en una tabla ***V(0..n, 0..M)*** que tiene una fila por cada numero disponible y una columna para cada suma desde 0 a *M*.

- Inicialmente la matriz ***V*** se inicializa en 0.
- En esa matriz, ***V(i,j)*** son valores booleanos (0,1) y será:

  V(i,j) **=** 1 si con los números desde ***s1*** a ***si*** se logra sumar la cantidad ***j***
  V(i,j) **=** 0 si con los números desde ***s1*** a ***si*** no se suma la cantidad ***j***
- La solución del problema se encuentra entonces en ***V(n,M).***
- Se puede llenar la matriz fila por fila o columna por columna.

Problema de la suma exacta

En cada paso (i,j) del llenado de la matriz hay que considerar entre 2 opciones para asignar un ***1*** a la entrada en la tabla $V(i,j)$:

1) La suma ya es ***j*** sin agregar el numero ***si*** a la suma, entonces si:

$$V(i\text{-}1,j) = 1$$

2) La suma exacta de valor ***j*** se logra agregando el numero ***si*** al subtotal de la fila anterior ***(i-1)*** y columna ***j-si***, entonces si:

$$V(i\text{-}1, j\text{-}si) = 1$$

Si es que existe la columna ***j-si***

De las 2 opciones:

$$V(i,j) = V(i\text{-}1, j) \text{ OR } V(i\text{-}1, j\text{-}si)$$

Problema de la suma exacta

Función **SumaExacta** **(S,n,M)**→Vector x entero x entero → bool
Auxiliar: matriz V(0..n, 0..M) que almacena la solución

```
    V(0,0) ← 1
    Para j=1,M hacer  V(0,j) ← 0
    Para i=1,n hacer
            Para j=1,M hacer
                        V(i,j) ← V(i-1,j)
                        Si j-s(i) ≥ 0 entonces
                                    V(i,j)←V(i,j) OR V(i-1,j-s(i))
     Retorna V(n,M)
Fin
```

Costo del algoritmo:

Para construir la tabla se necesita $\Theta(n*M)$ en almacenamiento y en tiempo de ejecución.

Problema de la suma exacta

Ejemplo

n=7, S=(1,2,2,4,5,2,4) M=15

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0:	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1: s1=1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2: s2=2	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
3: S3=2	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
4: S4=4	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
5: S5=5	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
6: s6=2	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
7: s7=4	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

V(i,j) = V(i-1, j) OR V(i-1, j-si)

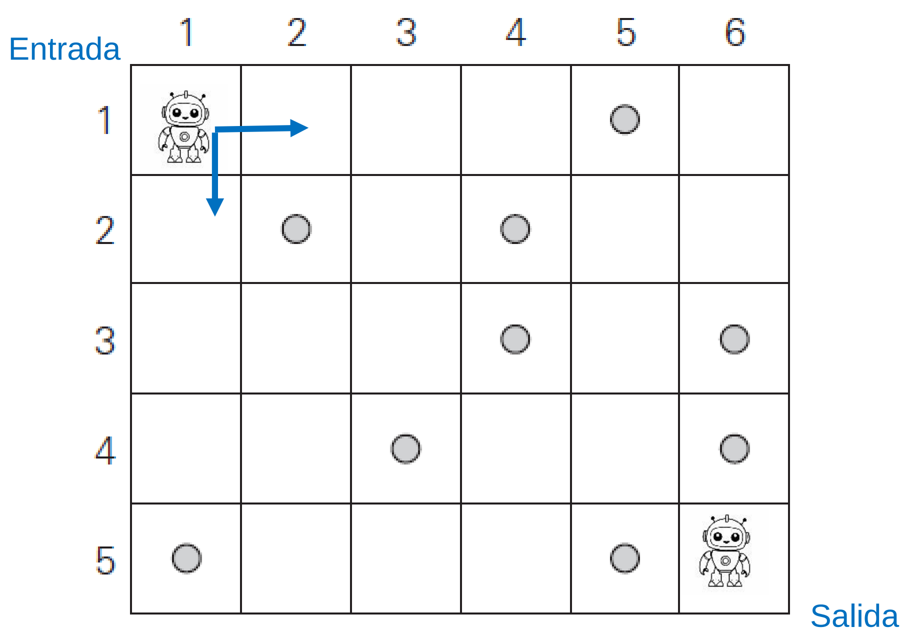
Entonces si existe un subconjunto de S que tiene suma exacta 15.
Cómo se determina cuáles son los números de la suma exacta?

Problema del robot

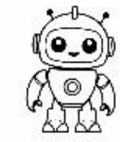
Se tiene un tablero de mxn casillas y algunas fichas en cada una de las celdas. Un robot inicia su recorrido desde la celda superior izquierda hasta la celda inferior derecha tratando de juntar la mayor cantidad de fichas posibles. En cada paso el robot puede moverse a la celda a su derecha o a la que está debajo. Cuando llega a una celda que tiene una ficha la puede levantar.

Diseñar un algoritmo para determinar el numero máximo de fichas que el robot pueda coleccionar durante su recorrido.

Ejemplo:



Problema del robot

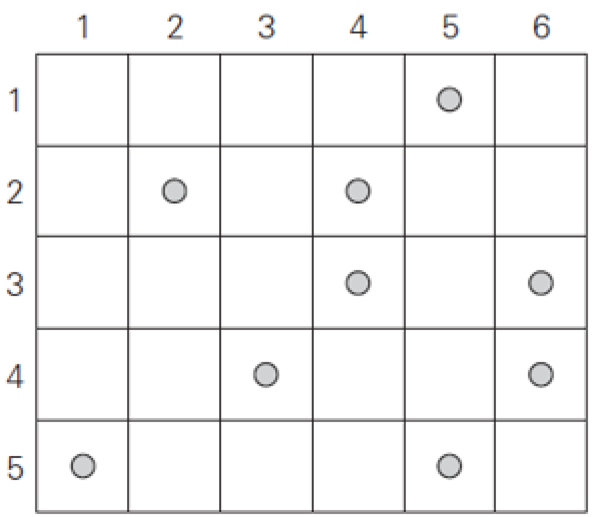


Datos:

Tabla T de dimensión mxn con valores booleanos

$T(i,j) = 0$ si en la celda (i,j) no hay una ficha

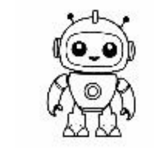
$T(i,j) = 1$ si en la celda (i,j) hay una ficha



T:

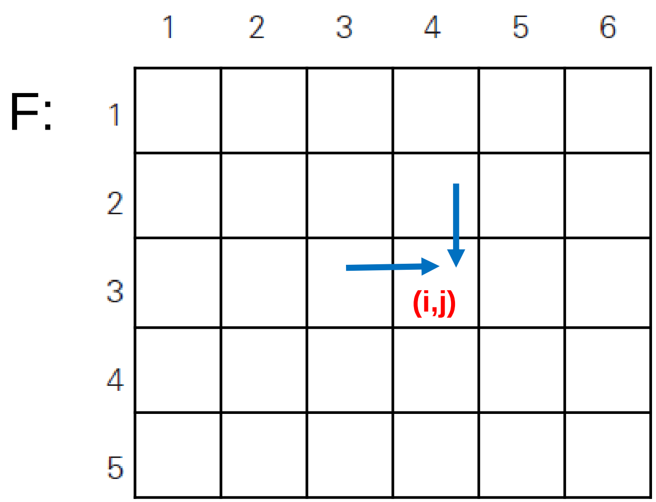
	1	2	3	4	5	6
1	0	0	0	0	1	0
2	0	1	0	1	0	0
3	0	0	0	1	0	1
4	0	0	1	0	0	1
5	1	0	0	0	1	0

Problema del robot



Solución:

- Se almacena la solución en una nueva matriz ***F(1..n, 1..m)***.
- Inicialmente la matriz ***F*** se inicializa en *0*.
- En esa matriz, ***F(i,j)*** será el máximo numero de monedas que el robot puede juntar hasta la posición ***(i,j)*** de la tabla.
- A la posición ***(i,j)*** de la tabla se puede llegar desde la posición superior ***(i-1,j)*** o desde la celda a la izquierda ***(i,j-1)*** **.** Si no existen esas posiciones se consideran de valor *0*.
- En cada celda si hay una ficha se la agrega a las que ya se van juntando.



Problema del robot

De este modo la máxima cantidad de fichas que el robot puede juntar es**:**

Para $2 \leq i \leq n,\ 2 \leq j \leq m$

$$F(i,j) = \text{Maximo}\ \{\ F(i-1,j)\ ,\ F(i,j-1)\ \} + T(i,j)$$

Para los bordes:

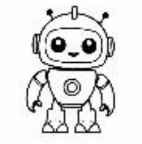
Primer elemento: $F(1,1) = T(1,1)$

Primera fila: $F(1,j) = F(1,j-1)+T(1,j) \quad 2 \leq j \leq m$

Primera columna: $F(i,1) = F(i-1,1)+T(i,1) \quad 2 \leq i \leq n$

- La solución optima del problema se encuentra entonces en $F(n,m)$.
- Se puede llenar la matriz fila por fila o columna por columna.

Problema del robot

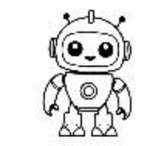


```
Función Robot (T,n,m)→Matriz x entero x entero → entero
Auxiliar: F, matriz mxn   F(1..n,1..m)
    F(1,1) ← T(1,1)
    Para j=2,m hacer  // primera fila
          F(1,j) ← F(1,j-1)+T(1,j)
    Para i=2,n hacer  // otras filas
          F(i,1) ← F(i-1,1)+T(i,1)   // primer elemento de la fila
          Para j=2,m hacer
                    F(i,j) ← Maximo { F(i-1,j) , F(i,j-1) } + T(i,j)
     Retorna F(n,m)
Fin
```

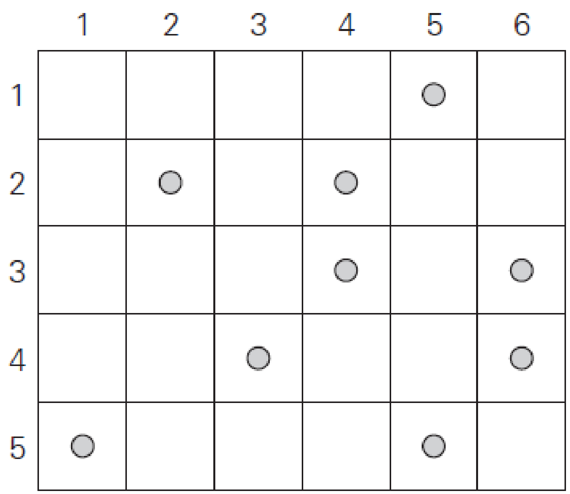
Costo del algoritmo:

Para construir la tabla se necesita $\Theta(n*M)$ en almacenamiento y en tiempo de ejecución.

Problema del robot



Ejemplo:



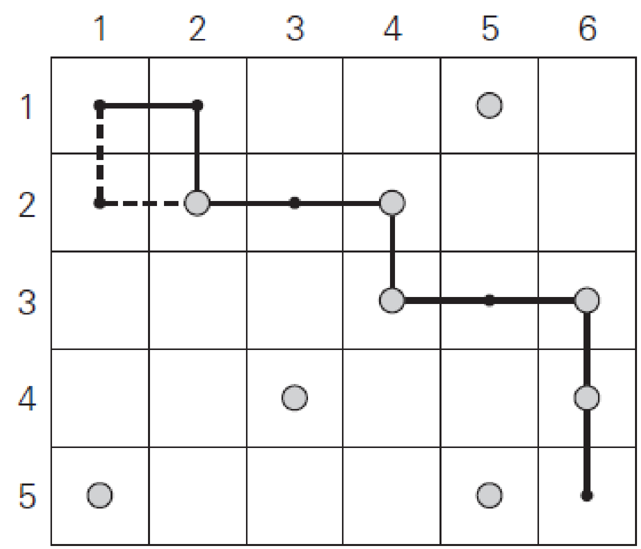
T:

	1	2	3	4	5	6
1	0	0	0	0	1	0
2	0	1	0	1	0	0
3	0	0	0	1	0	1
4	0	0	1	0	0	1
5	1	0	0	0	1	0

F:

	1	2	3	4	5	6
1	0	0	0	0	1	1
2	0	1	1	2	2	2
3	0	1	1	3	3	4
4	0	1	2	3	3	5
5	1	1	2	3	4	5

Camino:



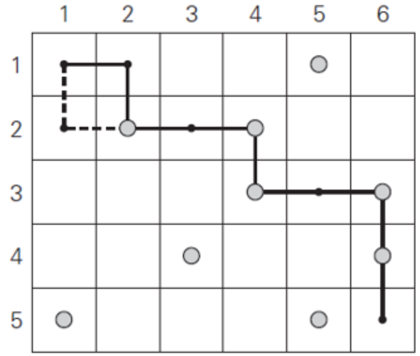
Problema del robot

Para recuperar el camino:

El camino de la solución optima del problema que se encuentra en ***F(i,j)*** , puede venir desde la celda superior o celda izquierda según:

- ***F(i-1,j) > F(i,j-1)*** : desde la celda de arriba
- ***F(i-1,j) < F(i,j-1)*** : desde la celda de la izquierda.
- ***F(i-1,j) =F(i,j-1)*** : desde la celda de arriba o desde la celda de la izquierda. Lo que da la posibilidad de tener 2 caminos óptimos.

Comenzando con la solución optima del problema que se encuentra en ***F(n,m)*** , se vuelve con el camino hasta la posición ***F(1,1)*** .



El costo del algoritmo que recupera el camino de la solución optima será de orden Θ(n+M) en tiempo de ejecución.

Problema del robot con restricción

Se tiene un tablero de mxn casillas y algunas fichas en cada una de las celdas. Un robot inicia su recorrido desde la celda superior izquierda hasta la celda inferior derecha tratando de juntar la mayor cantidad de fichas posibles. En cada paso el robot puede moverse a la celda a su derecha o a la que está debajo si es que está accesible. Cuando llega a una celda que tiene una moneda la puede levantar. Diseñar un algoritmo para determinar el numero máximo de fichas que el robot pueda coleccionar durante su recorrido.

La X *indica que la celda es inaccesible*