.UBAfiuba
FACULTAD DE INGENIERÍA

Set cover

Teoría de Algoritmos I (75.29 / 95.06)
Ing. Víctor Daniel Podberezski
vpodberezski@fi.uba.ar

Problema de set cover

Sea

Set X de n elementos

Una lista F={ S_1,. . .,S_m} de subsets de X

Un set cover

es un un subconjunto de F cuya unión es X

Objetivo

Encontrar el set cover S con menor cantidad de subsets

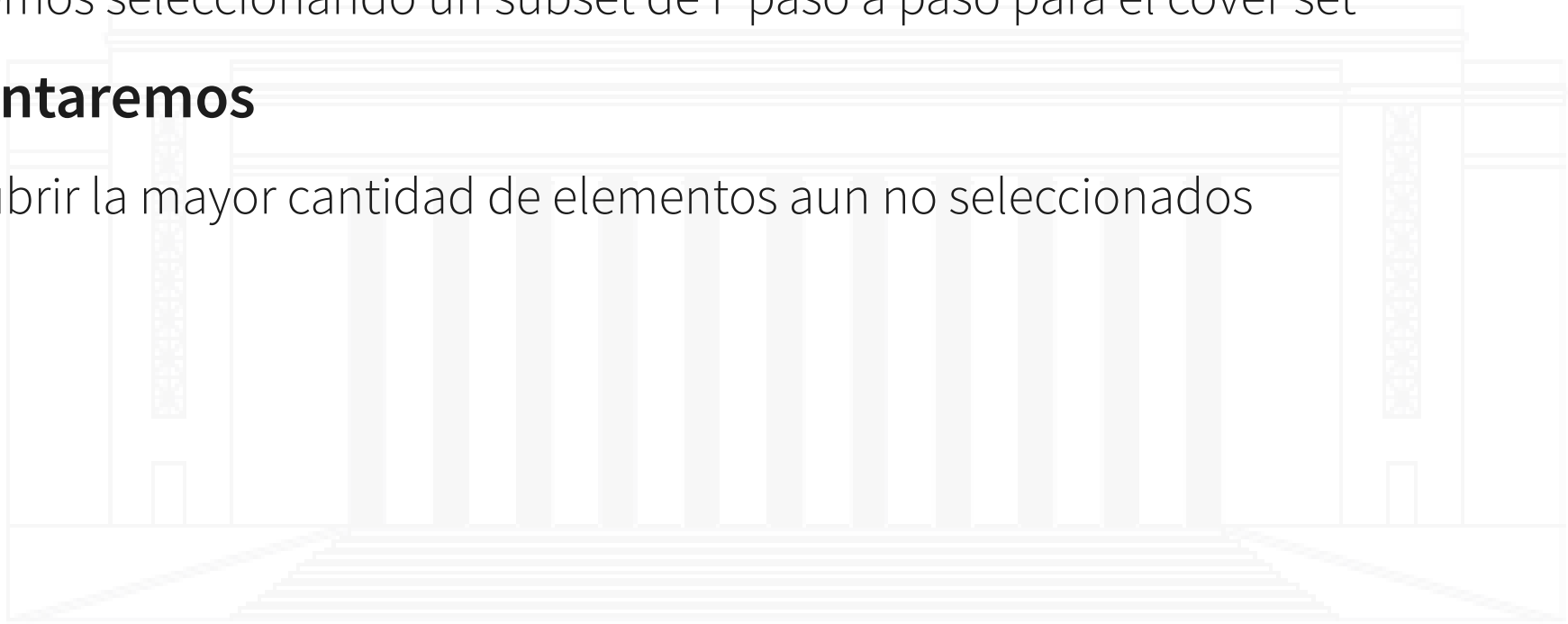
Tratando de construir un buen algoritmo

Construiremos un algoritmo greedy

Iremos seleccionando un subset de F paso a paso para el cover set

Intentaremos

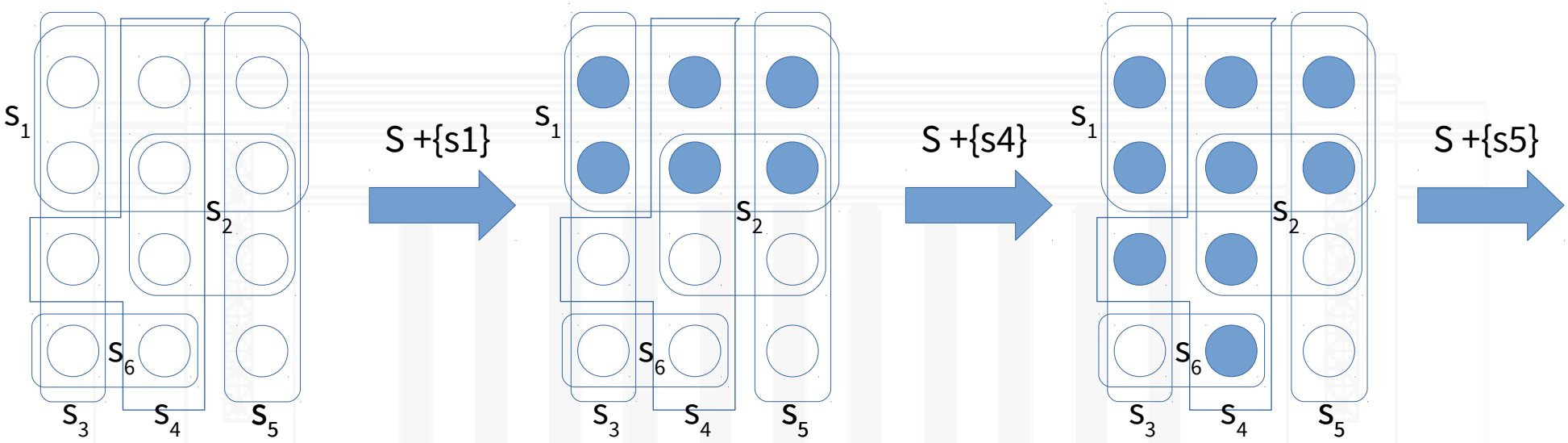
Cubrir la mayor cantidad de elementos aun no seleccionados



Algoritmo propuesto

```
R = X y S=Ø (sin sets seleccionados)
Mientras R ≠ Ø
   Seleccionar set Si con mayor Si ∩ R
   Agregar set Si a S
   Quitar elementos de Si de R
Retornar S
```

Ejemplo

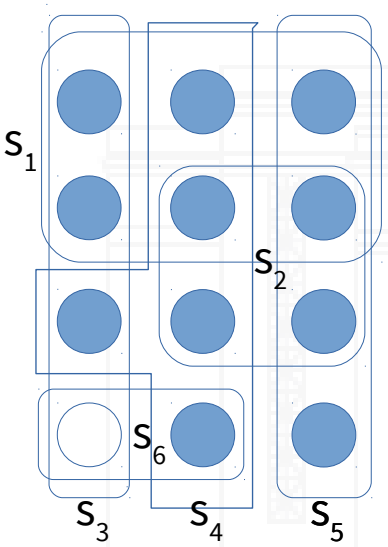


Inicialmente s1 tiene 6 elementos a agregar

s4 tiene 3 elementos a agregar

s5 tiene 2 elementos a agregar

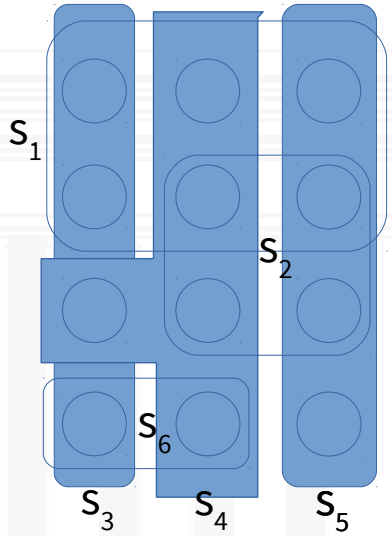
Ejemplo (cont.)



s3 tiene 1 elementos a agregar

S +{s3}

S ={s1, s4, s5,s3}



El elemento se puede lograr con solo 3 subconjuntos {s3,s4,s5}

Análisis del algoritmo

El tiempo de ejecución del algoritmos

Esta acotado por la cantidad de subsets y elementos

Y se puede implementar en tiempo polinomial

¿Qué tan diferente es el costo óptimo del Generado en el caso general?

Análisis del algoritmo

Asignamos un costo de 1

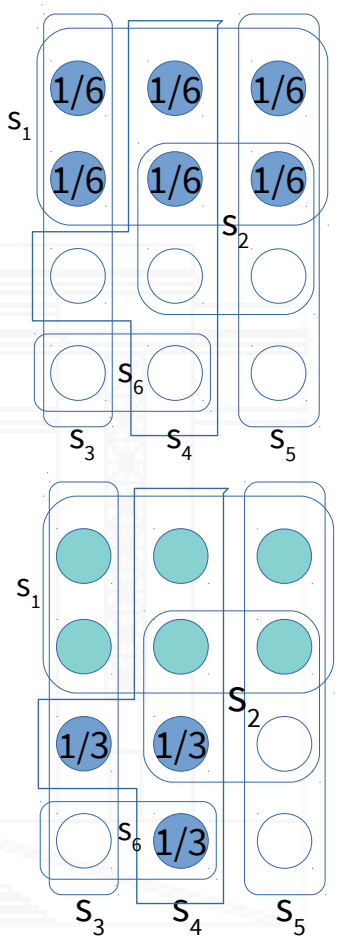
A cada set seleccionado por el algoritmo

Distribuimos el costo

Entre los elementos cubiertos por primera vez

Utilizaremos este costo para derivar las relación

Entre el tamaño del resultado optimo C* y el tamaño del resultado retornado por el algoritmo greedy C



Análisis del algoritmo (cont.)

Llamemos

Si al i-esimo set seleccionado por el algoritmo greedy

C_x el costo asignado al elemento x ,

Sabemos que

A cada elemento x se le asigna el costo solo 1 vez

Si x es cubierto por por Si

$$c_x = \frac{1}{|S_i - (S_1 \cup S_2 \cup \ldots \cup S_{i-1})|}$$

Análisis del algoritmo (cont.)

Podemos ver que

$$|C| = \sum_{x \in X} c_x$$

Ademas, cada elemento x

se encuentra en al menos un set del resultado optimo. entonces

$$\sum_{s \in C^*} \sum_{x \in S} c_x \geq \sum_{x \in X} c_x$$

En definitiva

$$|C| \leq \sum_{s \in C^*} \sum_{x \in S} c_x$$

Análisis del algoritmo (cont.)

Sea S cualquier set de F

Llamamos $u_i = |S - (S_1 \cup S_2 \cup ... \cup S_i)|$

a la cantidad de elementos aun no cubiertos luego del paso i en S

Con

$U_o = |S|$

Sea k el menor indice

Donde $u_k = 0$ ← no quedan elementos sin cubrir en S

Y donde $u_{k-1} > 0$ ← aun quedaban elementos sin cubrir en $S - (S_1 \cup S_2 \cup \ldots \cup S_{k-1})$

Se puede ver que

$u_{i-1} \geq u_i$

$u_{i-1} - u_i$ es la cantidad de elementos que se cubren por Si

Análisis del algoritmo (cont.)

Por lo que podemos expresar

$$\sum_{x \in S} c_x = \sum_{i=1}^{k} (u_{i-1} - u_i) \frac{1}{|S_i - (S_1 \cup S_2 \cup \ldots \cup S_{i-1})|}$$

Observar que

$$|S_i - (S_1 \cup S_2 \cup \ldots \cup S_{i-1})| \geq |S - (S_1 \cup S_2 \cup \ldots \cup S_{i-1})| = u_{i-1}$$

Por la elección greedy, sino en ese paso debería seleccionar a S en vez de Si

Por lo tanto

$$\sum_{x \in S} c_x \leq \sum_{i=1}^{k} (u_{i-1} - u_i) \frac{1}{u_{i-1}}$$

Que podemos reescribir como

$$\sum_{i=1}^{k} (u_{i-1} - u_i) \frac{1}{u_{i-1}} = \sum_{i=1}^{k} \sum_{j=u_i+1}^{u_{i-1}} \frac{1}{u_{i-1}}$$

Análisis del algoritmo (cont.)

Entonces

$j \le u_{i-1}$

$$\sum_{x \in S} c_x \le \sum_{i=1}^{k} \sum_{j=u_i+1}^{u_{i-1}} \frac{1}{u_{i-1}} \le \sum_{i=1}^{k} \sum_{j=u_i+1}^{u_{i-1}} \frac{1}{j} = \sum_{i=1}^{k} \left(\sum_{j=1}^{u_{i-1}} \frac{1}{j} - \sum_{j=1}^{u_i} \frac{1}{j} \right)$$

Y

Se cancelan las sumas intermedias

$H(0)=0$

$$\sum_{i=1}^{k} \left(\sum_{j=1}^{u_{i-1}} \frac{1}{j} - \sum_{j=1}^{u_i} \frac{1}{j} \right) = \sum_{i=1}^{k} \left(H(u_{i-1}) - H(u_i) \right) = H(u_0) - H(u_{u_k}) = H(u_0) - H(0) = H(u_0)$$

Por lo tanto

$$\sum_{x \in S} c_x \le H(u_0) = H(|S|)$$

Análisis del algoritmo (cont.)

Como

$$|C| \leq \sum_{s \in C^*} \sum_{x \in S} c_x \quad \text{y} \quad \sum_{x \in s} c_x \leq H(|S|)$$

Entonces

$$|C| \leq \sum_{s \in C^*} H(|S|) \leq |C^*| H(max\{|S| : S \in F\})$$

Podemos expresar |S|

Por la cantidad n de elementos del set original por lo tanto

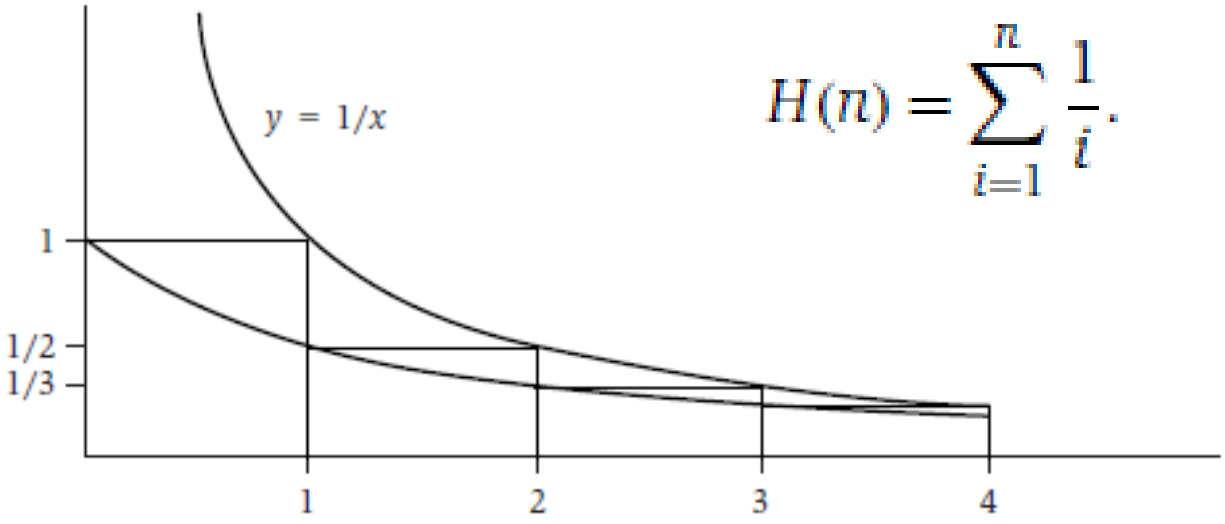
$$|C| \leq |C^*| * \log(n)$$

(como mucho un set tiene los n elementos del conjunto)

Función armónica

Dada la funcion armónica

Vemos que la misma puede ser acotada por 2 funciones logarítmicas



$$H(n) = \sum_{i=1}^{n} \frac{1}{i}.$$

$$H(n) = \Theta(\ln n).$$

Conclusión

Podemos expresar |S|

Por la cantidad n de elementos del set original por lo tanto $|C| \leq |C^*| * \log(n)$

Por todo lo anterior

Nuestro algoritmo es un (1+logn)-algoritmo de aproximación

.UBAfiuba

FACULTAD DE INGENIERÍA

Presentación realizada en Julio de 2020