Auxiliar #7.

"Algoritmos Online"

P1 Online Set Covering.

Universo U

Secuencia S, $|S| = m$, $x \in S \Rightarrow x \subseteq U$

Secuencia : $e_1, e_2, \ldots, e_n$

$n = 2^d$, $U = \{0,1\}^d$

ⓐ $S_0 = \{0\}^d$, $S_i = \{$ Cadenas con un 1 en el iésimo dígito $\}$ con $i = 1, \ldots, d$.

Sea un adversario que entrega $e_1, \ldots, e_n$

- $e_0 = 1 \ldots 1 \ldots 1 \Rightarrow$ A elige un S_{i_1}, $i_1 \geq 0$
- e_1 es la cadena con un 0 en i_1 y i_2 $\rightarrow$ $1 \ldots \underset{i_1}{0} \ldots 1 \ldots 1$
  $\Rightarrow$ A responde con S_{i_2} con $i_2 \neq i_1$
- e_2 tiene un 0 en i_1 y i_2

$\Rightarrow$ e_j tiene 0's en $i_1, \ldots, i_j$, donde $S_{i_1} \ldots S_{i_j}$ son los conjuntos que A va agregando. Cuando queda un dígito en 1 (i^*) paramos.

$\therefore$ El óptimo usa S_{i^*} $\Rightarrow$ uso 1 conjunto

$\Rightarrow$ competitividad $\log_2(n)$ (?)

(b) $\Omega = \left(\frac{\log m}{\log n}\right)$

$U = [n]$ $(\{1, 2, \ldots, n\})$

S: subconjunto de U de tamaño $\sqrt{n}$

$\Rightarrow m = |S| = \binom{n}{\sqrt{n}}$

El adversario es sencillo. Para cada paso de A, elijo un x_i no cubierto.

- El adversario puede hacer esto al menos $\sqrt{n}$ veces

$\Rightarrow$ Usa $\sqrt{n}$ conjuntos

- El Opt usa $(\{x_i\})$

$\Rightarrow$ Es $\sqrt{n}$-competitivo al menos.

Queda ver que en este caso $\sqrt{n} \in \Omega\left(\frac{\log m}{\log n}\right)$

Es decir $\sqrt{n} \in \Omega\left(\frac{\log \binom{n}{\sqrt{n}}}{\log(n)}\right)$ Usar stirling en la combinatoria. $(n! \sim n\log n - n)$

P2

- Tenemos k servidores
- $d(x,y) \geq 0$, $d(x,y) = d(y,x)$

$d(x,y) + d(y,z) \geq d(x,z)$

(1) Peticiones en forma de puntos del espacio.
Usaremos un espacio de $k+1$ puntos: $1, 2, \ldots, k-1$

Pueden haber varios en un mismo punto.

→ Si tenemos k-servidores, hay al menos un punto desocupado

→ Vamos a tener k adversarios

5

Mostraremos que:

$\sum_i Adv_i \leq A \Rightarrow$ debe haber uno tal que $Adv_i \leq \frac{A}{k}$

Sin pérdida de generalidad, "A" parte con los servidores $1, 2, \ldots, k$
Además supondremos que A es lazy (sólo mueve el servidor que llegará la petición).
↳ Todo algoritmo no lazy puede ser lazyficado sin costo.

Adv: Tiene servidores en todos los puntos excepto i
Invariantes del equipo de adversarios:
* Cada adversario siempre tiene un servidor listo en la siguiente request.

- Para cada punto cubierto por A, $\exists$ un Adv que lo tiene libre.

Operación de los adversarios:

↳ Si A mueve un servidor de i_1 a i_2
$\Rightarrow Adv\, i_1$ mueve $i_2 \rightarrow i_1$ (y pasa a ser $Adv\, i_2$)

- La secuencia de peticiones es pedir siempre el punto que A no ocupa $\Rightarrow$ Todos los adv lo tienen cubierto.

- Si A mueve un servidor una distancia d, un único adversario hace lo mismo en la otra sección.

$\Rightarrow \sum_{j=1}^{k} Adv_j = A \Rightarrow \exists\, Adv_{j^*}$ tal que $Adv_{j^*} \leq \frac{A}{k}$

$\Rightarrow A \geq k \cdot Adv_{j^*} \geq k \cdot Opt \rightarrow$ A es al menos k-competitivo.

②

$\phi \geq 0$

1 op. de Opt $\to \Delta\phi \leq k \cdot C_{opt}$

1 op. de A $\to \Delta\phi \leq -C_A$

$\phi = k\Psi + \theta$ $\begin{cases} S_i: \text{servidor de A} \\ a_i: \text{Servidor de Opt} \end{cases}$

$\Psi = \sum_{i=1}^{k} d(s_i, a_i)$

$\theta = \sum_{i<j} d(s_i, s_j)$

La idea de este ϕ es que Ψ sirve para ① y θ para ②

- Si Opt mueve un servidor de $a_i \to a'$ $\to$ θ no cambia ü

$$\Delta\phi = k \cdot \Delta\Psi = k\left(d(a_i', s_i) - d(a_i, s_i)\right)$$

$$\leq d(a_i', a_i)\, k \quad (\text{por } \Delta)$$

$$= k \cdot C_{opt} \Rightarrow (1) \checkmark$$

- Ante una petición r:

  Si r está a la izquierda de todos los servidores, sólo mueve el más cercano una distancia x.

  ↳ Opt ya tiene un servidor ahí

  $\Rightarrow \Psi$ disminuye en x

θ crece en $(k-1)x \Rightarrow \Delta\phi = k\Delta\Psi + \Delta\theta$

$$= -kx + (k-1)x$$

$$= -x$$

$\Rightarrow$ Cumple (2) ✓

• Análogo si r está a la derecha de todos.

1) Si r está entre S_i y S_{i+1} y a_j es el servidor Opt en r_j

2) Si $j \leq i \Rightarrow S_i$ se acerca a a_i una distancia x
S_{i+1} se aleja de a_{i+1} una distancia x.
$\Rightarrow \Delta\Psi = \theta$.

θ? Sólo cambian los términos con S_i y S_{i+1}
$d(S_i, S_j) + d(S_{i+1}, S_j)$ es constante.
(Uno se aleja y otro se acerca)

↳ Sólo queda $d(S_i, S_{i+1})$ que disminuye en $2x$ el costo.
$\Rightarrow \Delta\phi = -2x \leq -C_A$ //