ALGORITMOS APROXIMADOS

No sólo nos interesan los problemas de decisión, sino también los de optimización".

∃ problemas de optimización NP-completos, que como requiere mucho tiempo resolverlos, tomaremos una solución aproximada.

→ Una máquina de Turing determinística requiere tiempo exponencial en el tamaño del input.

Así, decimos que un algoritmo A que resuelve un problema de optimización es una $\rho(n)$-aproximación si:

maximización → $\forall n, \max_{I, |I|=n} \frac{S_{\text{ópt}}(I)}{S_A(I)} \leq \rho(n)$

minimización → $\forall n, \max_{I, |I|=n} \frac{S_A(I)}{S_{\text{ópt}}(I)} \leq \rho(n)$

I: input.
$S_{\text{ópt}}$: solución óptima para I.
S_A: solución de A para I.

* ∃ problemas NP-completos que se pueden aproximar "mejor" que otros.
// $\rho(n) = 2$, $\rho(n) = \log(n)$, ...

→ **VERTEX COVER** prob. de decisión: ver si el grafo $G=(V,E)$ tiene un recubrimiento V' de tamaño k.
prob. de optimización: encontrar el recubrimiento V' mínimo

1) sin peso: cuál es la solución? Un conjunto V' de cierto tamaño
objetivo: encontrar una 2-aproximación

$S_A \leq 2 \cdot S_{\text{ópt}}$
$\|$ $\|$
$|V'_A| \leq 2 \cdot |V'_{\text{ópt}}|$

Por cada 2 nodos que se insertan en V', el algoritmo óptimo tiene al menos 1. Esto se aplica para cada par de nodos (u,v)

solución:
$V' \leftarrow \emptyset$
while $E \neq \emptyset$
← $(u,v) \leftarrow$ arista en E
$V' \leftarrow V' \cup \{u, v\}$
$E \leftarrow E \setminus$ {aristas que inciden en u o en v}
return V'

2) **Ponderado:** ahora minimizamos el peso total de $V' \hookrightarrow \min \sum_{v \in V'} c(v)$

objetivo: obtener una 2-aproximación

Definimos $x(v) = \begin{cases} 1 & \text{si } v \in V' \\ 0 & \sim \end{cases}$ } Programación entera

Y queremos minimizar $\sum_{v \in V} c(v) \cdot x(v)$ } Simplemente redefinimos el problema.

Se debe cumplir que toda arista esté cubierta, entonces:

$$x(u) + x(v) \geq 1 \quad \forall (u,v) \in \bar{E}$$
$$x(v) \in \{0,1\}, \quad \forall v \in V \quad (\heartsuit)$$

* La optimización de programación entera es NP-completa !!

Pero, se puede transformar fácilmente a un problema de programación lineal, lo que toma tiempo polinomial.

Si cambiamos la restricción ($\heartsuit$) por $\boxed{0 \leq x(v) \leq 1, \quad \forall v \in V}$

tenemos un problema de programación lineal !!

↳ se aproxima al problema del vertex cover

$\Rightarrow v \in V'$ sii $x(v) \geq 0.5$

$v \notin V'$ sii $x(v) < 0.5$

Luego, si $(u,v) \in E \rightarrow x(u) + x(v) \geq 1$

$\Rightarrow x(u) \geq \frac{1}{2}$ o $x(v) \geq \frac{1}{2}$

$\Rightarrow u \in V'$ o $v \in V'$ → Obtenemos una solución válida !! (sobre los vértices)

Sea x' el problema con: prog. lineal y x' la solución al problema real con prog. entera: → no es VERTEX COVER

Se transforma a esto en la solución de prog. entera

$$\sum_{v \in V} c(v) \cdot x(v)$$

si $x(v) < 0.5 \rightarrow x''(v) = 0$ (reduce el costo)

$x(v) \geq 0.5 \rightarrow x''(v) = 1$ (a lo sumo duplica el costo)

x'' : nuestra solución que "discretiza" la prog. lineal

$x''(v) \leq 2x(v) \rightarrow \sum c(v) \cdot x''(v)$

$\leq 2 \sum c(v) \cdot x(v)$

$\leq 2 \cdot \sum c(v) \cdot x'(v)$

$= 2 \cdot S_{OPT}$