MA-47A Optimización Combinatorial	**8 de Julio, 2003**

Pauta Examen

Prof. Cátedra: M. Kiwi — *Prof. Auxiliar: G. Espinoza*

PROBLEMA 1:

(i).- Considere el siguiente algoritmo glotón de aproximación para el problema MAX-CUT:

APROX-MAX-CUT
Entrada: $G = (V, E)$ grafo y $\omega : E \to \mathbb{R}$ función de peso
Código: $V' \longleftarrow \emptyset$
$V'' \longleftarrow \emptyset$
For $\exists v \in V$
  $S' \longleftarrow \{u \in V' \,:\, uv \in E\}$
  $S'' \longleftarrow \{u \in V'' \,:\, uv \in E\}$
  If $\sum_{u \in S'} \omega(uv) \geq \sum_{u \in S''} \omega(uv)$
    then $V' \longleftarrow V' \cup \{v\}$
    else $V'' \longleftarrow V'' \cup \{v\}$
  $V \longleftarrow V \setminus \{v\}$
Return V'

Veamos que el algoritmo es una 2–aproximación. Sean $v_1, \ldots, v_n$ el orden en que entran los nodos de V a V' o V''. Definimos $i_1 < i_2 < \ldots < i_{n'}$ y $j_1 < \ldots < j_{n''}$ de manera que $V' = \{v_{i_1}, \ldots, v_{i_{n'}}\}$ y $V'' = \{v_{j_1}, \ldots, v_{j_{n''}}\}$. Observar que

$$\begin{aligned}\sum_{e \in \delta(V')} \omega(e) &= \sum_{k \in \{1,\ldots,n'\}} \sum_{l \in \{1,\ldots,n''\} : j_l < i_k} \omega(v_{i_k} v_{j_l}) + \sum_{l \in \{1,\ldots,n''\}} \sum_{k \in \{1,\ldots,n'\} : i_k < j_l} \omega(v_{i_k} v_{j_l}) \\ &\geq \frac{1}{2} \left(\sum_{k \in \{1,\ldots,n'\}} \sum_{l=1}^{i_k - 1} \omega(v_{i_k} v_l) + \sum_{l \in \{1,\ldots,n''\}} \sum_{k=1}^{j_l - 1} \omega(v_k v_{j_l}) \right) = \frac{1}{2} \sum_{e \in E} \omega(e)\,.\end{aligned}$$

Como el óptimo, $\mathrm{OPT}(G, \omega)$, es a lo más $\sum_{e \in E} \omega(e)$ se tiene que APROX-SET-COVER entrega un resultado que es al menos la mitad del óptimo, i.e., APROX-SET-COVER es un algoritmo de 2–aproximación para MAX-CUT.

El tiempo que toma APROX-MAX-CUT es $O(|V| + |E|)$. En efecto, el **For** se ejecuta $O(|V|)$ veces y por cada nodo v el costo total de las operaciones dentro del **For** es $O(|\delta(v)|)$. Si el algortimo se implementa usando matrices de adyacencia, entonces el tiempo total de ejecución quedará en $O(|V| + |E|)$.

(ii).- En el problema del flujo máximo, para ser resuelto en forma eficiente, se asume que la función de capacidad es no–negativa. En el argumento sugerido, la función de capacidad será $-\omega$. Por lo tanto, el argumento no reduce el problema del corte máximo a un problema de flujo máximo.

<u>Nota</u>: El hecho que en MAX-CUT no hayan nodos distinguidos que juegen el rol de s y t en un problema de flujo máximo no es relevante. En efecto, siempre podemos agregar dos nodos distinguidos s y t a un grafo G, de manera que s y t esten conectados por arcos con peso 0 a cada uno de los nodos de G. Es facil ver que un corte máximo que separa a s y t induce un corte máximo en G, y viceversa.

(iii).- Ver cátedra.

(iv).- Observar que el problema del MAX-FC se puede plantear como un PL en las variables $(x_{uv} : uv \in E; x_{vu} : uv \in E)$ de la siguiente forma:

$$\begin{array}{lll} \text{(P)} & \text{MAX} & \alpha \\ & \text{s.a.} & f_{s_i}(x_i) \geq \alpha \cdot d_i \quad i = 1\ldots,k \\ & & f_v(x_i) = 0\,, \quad \forall v \in V \setminus \{s_i,t_i\}\,, \quad i = 1,\ldots k \\ & & \displaystyle\sum_{i=1}^{k}(x_i(uv) - x_i(vu)) \leq u(uv)\,, \quad \forall uv \in E \\ & & x_i \geq 0\,, \quad i = 1,\ldots,k, \end{array}$$

donde $f_v(x_i) = \displaystyle\sum_{v:uv\in E}(x_i(uv) - x_i(vu))$.

El PL planteado tiene $2k|E| + 1$ variables, $|E| + k$ desigualdades, $k(|V| - 2)$ igualdades y $2k|E|$ restricciones de signo. Es decir es un PL polinomial en el tamaño de la entrada (una codificación razonable de G y ω). Como un PL puede resolverse en tiempo polinomial, sigue que MAX-FC es a tiempo polinomial.

(v).- Primero probemos la desigualdad. Para ello, consideremos un flujo $x \geq 0$ en G y recordemos que cualquiera sea $S \subseteq V$, se tiene que $f_s(x) = x(\delta(S)) \leq u(\delta(S))$ si $\delta(S)$ es un (s,t)–corte. En el caso de flujo concurrente, esta desigualdad generaliza a

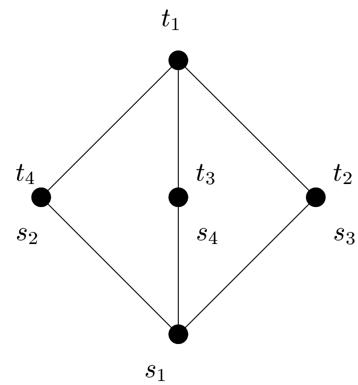
$$\sum_{i\in I(S)} f_{s_i}(x_i) \leq u(\delta(S))\,.$$

Por lo tanto, si $x_1,\ldots,x_k$ son una solución óptima del problema MAX-FC, entonces cualquiera sea $S \subseteq V$,

$$\alpha^* \sum_{i\in I(S)} d_i \leq \sum_{i\in I(S)} f_{s_i}(x_i) \leq u(\delta(S))\,.$$

Sigue que $\alpha^* \leq \beta^*$.

Veamos ahora el contraejemplo. Para ello, considere la situación descrita en la siguiente figura, donde todas las capacidades y demandas son iguales a 1.



Observar que cada tipo de mercancía debe recorrer al menos 2 arcos para llegar de s_i a t_i. Luego, para satisfacer una demanda de una unidad para cada mercancía, se requiere una red con "volumen" de al menos 8, siendo que la del ejemplo tiene un "volumen" igual a 6 (una unidad por arco). Por lo tanto, en el mejor de los casos se podrá satisfacer $8/6 = 3/4$ de la demandad para cada mercancía. En otras palabras, $\alpha^* \leq 3/4$ para el ejemplo de la figura. Para calcular β^*, nos referiremos a los nodos de la red por N, S, E, O y C por *norte*, *sur*, *este*, *oeste* y *centro*, respectivamente. Observar que por complementación basta considerar sólo los subconjuntos de nodos S de a lo más 2 elementos y por simetría sólo los siguientes conjuntos:

$$\{N\}\,, \quad \{O\}\,, \quad \{NO\}\,, \quad \{EO\}\,, \quad \{NS\}\,.$$

Los $u(\delta(S))$ correspondientes son 3, 2, 3, 4 y 6, respectivamente. Además, el valor de $\sum_{i \in I(S)} d_i$ para cada uno de estos subconjuntos es 1, 2, 3, 4 y 2, respectivamente. Sigue que $\beta^* = \min\{3, 1, 1, 1, 3\} = 1$. Luego, $\alpha^* < \beta^*$.

PROBLEMA 2:

(i).- En lo que sigue, abreviaremos VERTEX-COVER y SET-COVER por VC y SC cuando aparezcan como sub–índices.

Se pide probar que el lenguaje de decisión, L_{VC}, asociado a SET-COVER es NP–completo.

Veamos primero que L_{VC} es NP–duro. En efecto, dado una instancia $\langle G, k \rangle$ donde $G = (V, E)$ es un grafo y $k \in \mathbb{Z}_+$, construimos $\langle E, \mathcal{S}, \omega, k \rangle$ tal que que $\mathcal{S} = \{\delta(v) \,:\, v \in V\}$ y $\omega(S) = 1$ para todo $S \in \mathcal{S}$. Notar que $V' \subseteq V$ es un recubrimiento de nodos de G de cardinalidad a lo más k si y sólo si $\mathcal{C} = \{\delta(v') \,:\, v' \in V'\}$ es tal que $U \subseteq \cup_{C \in \mathcal{C}} C$ y $\sum_{C \in \mathcal{C}} \omega(C) \leq k$. En resumen,

$$\langle G, k \rangle \in L_{\mathrm{VC}} \iff \langle E, \mathcal{S}, \omega, k \rangle \in L_{\mathrm{SC}}\,.$$

Sigue que L_{SC} es NP–duro.

Para ver que L_{SC} esta en NP, basta observar que $\mathcal{C} \subseteq \mathcal{S}$ es un certificado de pertenencia de $\langle E, \mathcal{S}, \omega, k \rangle$ en L_{SC} cuyo tamaño esta acotado en el tamaño de $\langle E, \mathcal{S}, \omega, k \rangle$, i.e., es polinomial (especificamente, lineal) en el tamaño de la instancia. El proceso de verificación del certificado puede hacerse en forma eficiente, pues sólo consiste en comprobar que $U \subseteq \cup_{C \in \mathcal{C}} C$ y $\sum_{C \in \mathcal{C}} \omega(C) \leq k$.

Se concluye que L_{VC} es NP–completo.

(ii).- En efecto,

$$\begin{aligned} c^T x &= \sum_j c_j x_j = \sum_{j:x_j\neq 0} c_j x_j \leq \alpha \sum_{j:x_j\neq 0} \sum_i a_{i,j} y_i x_j \\ &= \alpha \sum_{i:y_i\neq 0} \sum_{j:x_j\neq 0} a_{i,j} x_j y_i \leq \alpha\beta \sum_{i:y_i\neq 0} b_i y_i = \alpha\beta \sum_i b_i y_i = \alpha\beta b^T y\,. \end{aligned}$$

(iii).- Primero plantearemos el problema de SET-COVER como un PL y determinaremos su dual. En efecto, SET-COVER puede verse como un PLI en las variables $x = (x_S : S \in \mathcal{S})$ de la manera siguiente:

$$\begin{array}{lll} \text{(P)} & \text{MIN} & \sum_{S\in\mathcal{S}} \omega(S) x_S \\ & \text{s.a.} & \sum_{S:u\in S} x_S \geq 1, \quad \forall u \in U \\ & & x_S \geq 0, \quad \forall S \in \mathcal{S} \\ & & x_S \in \mathbb{N}, \quad \forall S \in \mathcal{S}. \end{array}$$

Nota: Dado que ω es no–negativa, no es necesario agregar la restricción $x_S \leq 1$ para todo $S \in \mathcal{S}$.

El dual de la relajación de (P) queda:

$$\begin{array}{lll} \text{(D)} & \text{MAX} & \sum_{u\in U} y_u \\ & \text{s.a.} & \sum_{u\in S} y_u \leq \omega(S), \quad \forall S \in \mathcal{S} \\ & & y_u \geq 0, \quad \forall u \in U. \end{array}$$

Observar ahora que APROX-SET-COVER parte con una solución factible de (D) y satisface al comienzo de cada iteración la propiedad de:

Holgura complementaria dual: $y_u = 0$ o $1 \leq \sum_{S:u\in S} x_S \leq f$,

puesto que x_S es en todo momento un vector a coordenadas en $\{0,1\}$, f es por definición el máximo número de conjuntos $S \in \mathcal{S}$ a los que puede pertenecer un elemento $u \in U$, y si $y_u \neq 0$ entonces existe un S que contiene U tal que $\sum_{u\in S} y_u = \omega(S)$, i.e., $x_S = 1$.

Además APROX-SET-COVER se detiene al encontrar una solución factible de (P) que satisface:

Holgura complementaria primal: $x_S = 0$ o $\sum_{u\in S} y_u = \omega(S)$.

Luego, por (ii), con $\alpha = 1$ y $\beta = f$, se tiene que si x^* e y^* son los valores de x e y cuando APROX-SET-COVER se detiene y retorna $\mathcal{C}$, entonces

$$\sum_{C\in\mathcal{C}} \omega(C) = \sum_{S\in\mathcal{S}} \omega(S) x^*_S \leq f \sum_{u\in U} y^*_u\,.$$

Pero, el Teorema de Holgura Complementaria nos garantiza que $\sum_{u \in U} y_u^*$ es una cota inferior del óptimo de la relajación de (P), y por lo tanto también es una cota del óptmo de (P), i.e., del valor óptimo de la instancia dada de SET-COVER. En resumen, APROX-SET-COVER es una $f-$aproximación de SET-COVER.