Algoritmos de Aproximación

Clase 4
Cubrimientos de conjuntos

Pablo Romero

Jueves 18 de agosto de 2023, Montevideo, Uruguay.

Formulación del Problema

Cubrimiento de vértices de mínimo cardinal (MCVC)

Dado un grafo $G = (V, E)$, el MCVC consiste en hallar el subconjunto V' de V de mínimo cardinal tal que para cada arista e de E se cumple que $e \cap V' \neq \emptyset$.
Entonces, el MCVC es el problema $\Pi_1 = (\mathcal{G}, S_{\Pi_1}, f_1)$, donde

- $\mathcal{G}$ es la colección de todos los grafos simples.
- $S_{\Pi_1}(G) = \{V' : V' \subseteq V, \forall e \in E, V' \cap \{e\} \neq \emptyset\}$.
- $f_1(V') = |V'|$.
- $OPT_{\Pi_1}(G) = \min_{\{V' \in S_{\Pi_1}(G)\}} |V'|$.

Función de Emparejamiento Maximal $\mathcal{MM}$

A partir del vínculo que estudiamos entre emparejamientos de aristas y cubrimientos de vértices, surge el siguiente algoritmo que retorna a todos los vértices que son extremos de cada arista de un emparejamiento maximal.

Algoritmo 1 $V' = \mathcal{MM}(G)$

1: $V' \leftarrow \emptyset$
2: **while** $E(G) \neq \emptyset$ **do**
3: $\quad e \leftarrow RandomEdge(G)$
4: $\quad \{x, y\} \leftarrow Incident(e)$
5: $\quad V' \leftarrow V' \cup \{x, y\}$
6: $\quad G \leftarrow G - \{x, y\}$
7: **end while**
8: **return** V'

Algoritmo $\mathcal{MM}$

Lema 1.

La función $\mathcal{MM}$ es un algoritmo y $\mathcal{MM}(G) \in S_{\Pi_1}(G)$, $\forall G \in \mathcal{G}$.

Prueba. Primero veamos que $\mathcal{MM}$ termina. Sea $G \in \mathcal{G}$. En cada iteración del ciclo **while** el grafo obtenido tiene, al menos, una arista menos que al inicio de la iteración. Luego la función $\mathcal{MM}$ termina en una cantidad finita de pasos. Sean $\{x_1, y_1\}, \ldots, \{x_n, y_n\}$ los $2n$ vértices quitados a G en ese orden, tal que $V' = \mathcal{MM}(G) = \cup_{i=1}^{n}\{x_i, y_i\}$. Para cada $i \in \{1, 2, \ldots, n\}$, sea $G_i = G - \cup_{j=1}^{i}\{x_j, y_j\}$. Entonces, G_n no satisface la condición $E(G_n) \neq \emptyset$, y por lo tanto G_n no tiene aristas. Además, por construcción, $G_n = G - V'$. Por lo tanto, cada una de las aristas de G tiene al menos un extremo en V', y V' es un cubrimiento de vértices de G. Luego, es solución factible del MCVC. ■

Algoritmo de aproximación para el MCVC

Teorema 1.

$\mathcal{MM}$ es un algoritmo de aproximación de factor 2 para el MCVC.

Sea $G \in \mathcal{G}$. Por el Lema 1, $V' = \mathcal{MM}(G)$ cumple que $V' \in S_{\Pi_1}(G)$. Para cada $i \in \{1, 2, \ldots, n-1\}$ sea $V_i = \cup_{j=1}^{i} \{x_j, y_j\}$ y $G_i = G - V_i$. Sea $e_{i+1} = \{x_{i+1}, y_{i+1}\}$ la arista elegida en G_i. Tal arista e_{i+1} es incidente a dos vértices de G_i, por lo que e_{i+1} no tiene ningún extremo en común con las aristas $\{e_1, e_2, \ldots, e_i\}$. Por inducción finita se sigue que $M = \{e_1, e_2, \ldots, e_n\}$ es un emparejamiento de G. Sea ahora V^{opt} un cubrimiento de vértices de G tal que $|V^{opt}| = OPT_{\Pi_1}(G)$. Para cada $i \in \{1, 2, \ldots, n\}$, la arista e_i de M tiene un extremo que es vértice de V^{opt}, por lo que $|M| \leq |V^{opt}|$. Entonces, $2|M| \leq 2|V^{opt}|$, y como $2|M|$ es la cantidad de vértices de V' se sigue el resultado. ■

Preguntas sobre la aproximabilidad del MCVC

Preguntas

1. Es posible reducir el factor 2 para el MCVC con otro análisis?
2. Es posible elegir emparejamientos para reducir el factor 2?
3. Es posible construir otro algoritmo con mejor factor?

Defínción 1.

Sea $\Pi = (\mathcal{I}, S_\Pi, f)$ *un COP y* $\mathcal{A}$ *un algoritmo de aproximación de factor* δ *para* Π. *Una sucesión* $(I_n)_{n\in\mathbb{N}} \subseteq \mathcal{I}$ *es una familia justa para* $\mathcal{A}$ *si* $\forall\epsilon > 0 \exists n_0 : \forall n \geq n_0, |f(\mathcal{A}(I_n)) - \delta(|I_n|)OPT_\Pi(I_n)| < \epsilon$.

Respuestas

Las sucesiones $(K_{n,n})_{n\in\mathbb{N}}$ y $(K_{2n+1})_{n\in\mathbb{N}}$ son familias justas para $\mathcal{MM}$, y la respuesta a las dos primeras preguntas es negativa. La tercerca pregunta es un problema abierto.

Cubrimiento de conjuntos

Cubrimiento de conjuntos (SC)

Dado un universo U con n elementos, una colección $\mathcal{S} = \{S_1, S_2, \ldots, S_k\}$ de subconjuntos que cubren U y una función de costos $c : \mathcal{S} \to \mathbb{Q}^+$, seleccionar una subcolección $\mathcal{U} \subseteq \mathcal{S}$ que cubra U con costo mínimo.

Sea $\mathcal{X}$ el conjunto que contiene a todas las colecciones de cubrimientos $\mathcal{S}$ de U. Entonces, el SC es el problema $\Pi_2 = (\mathcal{I}_2, S_{\Pi_2}, f_2)$, donde

- $\mathcal{I}_2 = \{(\mathcal{S}, U, c) : \mathcal{S} \in \mathcal{X}, c : \mathcal{S} \to \mathbb{Q}^+\}$.
- $S_{\Pi_2}(\mathcal{S}, U) = \{\mathcal{U} : \mathcal{U} \subseteq \mathcal{S}, U \in \mathcal{X}\}$.
- $f_2((\mathcal{S}, U, c), \mathcal{U}) = \sum_{S_i \in \mathcal{U}} c(S_i)$.
- $OPT_{\Pi_2}(\mathcal{S}, U, c) = \min_{\{\mathcal{U} \in S_{\Pi_2}(\mathcal{S}, U, c)\}} \sum_{S_i \in \mathcal{S}} c(S_i)$.

Algoritmo Goloso para el Cubrimiento de conjuntos

Algoritmo 2 $\mathcal{U} = Greedy(\mathcal{S}, U, c)$

1: $C \leftarrow \emptyset$
2: **while** $C \neq U$ **do**
3: $\quad S \leftarrow \arg\min_{\{S \in \mathcal{S}\}} \left\{ \frac{c(S)}{|S - C|} \right\}$
4: $\quad \mathcal{U} \leftarrow \mathcal{U} \cup S$
5: $\quad C \leftarrow C \cup S$
6: **end while**
7: **return** $\mathcal{U}$

Algoritmo Goloso para el Cubrimiento de conjuntos

Lema 2.

La función Greedy es un algoritmo y $Greedy(I) \in S_{\Pi_2}(I), \forall I \in \mathcal{I}_2$.

Prueba. Primero veamos que *Greedy* termina. Sea $I = (\mathcal{S}, U, c) \in \mathcal{I}_2$, y sea $C \subseteq U$ tal que $C \neq U$. Como $\mathcal{S} \in \mathcal{X}$, en particular $\mathcal{S}$ cubre $U - C$, y existe algún S_i en $\mathcal{S}$ tal que $S_i \cap (U - C) \neq \emptyset$. Luego, en cada paso se agrega algún elemento de U al conjunto C. Como U es finito, el conjunto C va a finalizar igualando al conjunto U. Entonces, la función *Greedy* termina en una cantidad finita de pasos. Además, cuando termina obtendremos una colección de conjuntos $\mathcal{U}$ tal que $\mathcal{U} \subseteq \mathcal{S}$ y además $U \subseteq \cup_{S_i \subseteq \mathcal{U}} S_i$, por lo que $\mathcal{U} \in \mathcal{X}$, y en conclusión, $\mathcal{U} \in \Pi_2(\mathcal{S}, U, c)$. Por lo tanto, *Greedy* retorna una solución factible para cada instancia I de SC, como queríamos demostrar. ■

Factor armónico para el Cubrimiento de conjuntos

Para determinar el rendimiento del algoritmo *Greedy* nos serán de utilidad las dos definiciones siguientes.

Defición 2.

Sea n un entero positivo cualquiera. El n-ésimo armónico se denota H_n y vale $H_n = \sum_{i=1}^{n} \frac{1}{i}$.

Defición 3.

Sea (S, U, c) una instancia del SC y $\mathcal{U} = \{S_1, \ldots, S_t\}$ un cubrimiento de U. Si cubrimos a U usando a los conjuntos $S_1, S_2, \ldots, S_t$ y en ese mismo orden y definimos $A_1 = S_1$ y $A_{j+1} = S_{j+1} - A_j$, entonces, para cada $j \in \{1, 2, \ldots, t\}$ el precio de cada uno de los elementos e de A_j es idéntico, y vale $p(e) = \frac{c(S_j)}{|A_j|}$.

Factor armónico para el Cubrimiento de conjuntos

Lema 3.

Sea $I = (\mathcal{S}, U, c)$ una instancia del SC donde $|U| = n$. Si Greedy agrega los elementos de U en el orden $e_1, e_2, \ldots, e_n$, entonces para cada $k \in \{1, 2, \ldots, n\}$, $p(e_k) \leq \frac{OPT_{\Pi_2}(I)}{n-k+1}$

Prueba. Antes de cubrir e_k tenemos que $U - C \subseteq \{e_1, \ldots, e_{k-1}\}$, y que $|U - C| \geq n - k + 1$. Sea $\mathcal{V}$ un cubrimiento óptimo para I, y sean $V_1, V_2, \ldots, V_s \in \mathcal{V}$ tales que cubren a $U - C$. Tomemos la partición $A'_1, \ldots, A'_s$ tal que $A'_1 = V_1 - C$ y $A'_{j+1} = V_{j+1} - A'_j$. El precio de cada elemento $a_j \in A'_j$ es $p(a_j) = \frac{c(V_j)}{|A'_j|}$. Entonces, $\sum_{j=1}^{s} \sum_{i=1}^{|A_j|} \frac{c(V_j)}{|A'_j|} \leq OPT_{\Pi_2}(I)$. Como hay $|U - C|$ sumandos, uno de ellos cumple que $\frac{c(V_j)}{|A'_j|} \leq \frac{OPT_{\Pi_2}(I)}{|U-C|} \leq \frac{OPT_{\Pi_2}(I)}{n-k+1}$ para algún j.

Como *Greedy* alcanza el menor cociente, $p(e_k) \leq \frac{OPT_{\Pi_2}(I)}{n-k+1}$. ■

Factor armónico para el Cubrimiento de conjuntos

Teorema 2.

Sea $I = (\mathcal{S}, U, c)$ una instancia de SC tal que $|U| = n$. Entonces: $Greedy(I) \leq H_n OPT_{\Pi_2}(I)$.

Prueba. Sea $A_1 = S_1$ y para cada $j \in \{1, \ldots, t-1\}$, $A_{j+1} = S_{j+1} - A_j$. Como $\mathcal{U}$ cubre U entonces $A_1, A_2, \ldots, A_t$ es partición de U. Como S_j cubre a cada elemento de A_j a idéntico precio $c(S_j)/|A_j|$, resulta que:

$$c(\mathcal{U}) = \sum_{j=1}^{t} c(S_j) = \sum_{j=1}^{t} |A_j| \frac{c(S_j)}{|A_j|} = \sum_{j=1}^{t} \sum_{a_j \in A_j} \frac{c(S_j)}{|A_j|} = \sum_{j=1}^{t} \sum_{a_j \in A_j} p(a_j)$$

$$= \sum_{k=1}^{n} p(e_k) \leq \sum_{k=1}^{n} \frac{OPT_{\Pi_2}(I)}{n-k+1} = H_n OPT_{\Pi_2}(I),$$

donde la desigualdad vale gracias al Lema 3. ■

Familia justa para el algoritmo Goloso

Proposición 1.

Existe alguna familia justa para el algoritmo Greedy en SC.

Prueba. Para cada $\epsilon > 0$ construyamos la sucesión de instancias $I_n = (\mathcal{S}_n, U_n, c_n)$ tal que $U_n = \{1, 2, \ldots, n\}$, $\mathcal{S}_n = \{\{1\}, \{2\}, \ldots, \{n\}, U\}$ y $c_n(\{i\}) = \frac{i}{n}$ para cada $i \in \{1, 2, \ldots, n\}$, mientras que $c_n(U_n) = 1 + \frac{\epsilon}{H_n}$. El algoritmo *Greedy* va a seleccionar en orden los conjuntos $\{n\}, \{n-1\}, \ldots, \{1\}, U_n$ con costo total H_n, mientras que el óptimo consiste en elegir directamente U_n, con costo $OPT_{\Pi_2}(I_n) = 1 + \frac{\epsilon}{2H_n}$. Entonces, conseguimos que

$$|f(Greedy(I_n)) - H_n OPT_{\Pi}(I_n)| = |H_n - H_n(1 + \frac{\epsilon}{2H_n})| < \epsilon,$$

y por lo tanto $(I_n)_{n \geq 2}$ es una familia justa para *Greedy* en SC. ■

Comentarios

Comentarios

- El MCVC es un caso particular de SC ($\Pi_1 \subseteq \Pi_2$). De hecho, dada una instancia G de Π_1 basta con tomar la instancia $I = (\{E_v\}_{v \in V(G)}, E(G), c)$, donde $E_v = \{vy : y \in N_G(v)\}$ con costo $c(E_v) = 1$.
- Vamos a recuperar el factor armónico utilizando la teoría de dualidad en programación lineal.
- Sorprendentemente, el algoritmo *Greedy* es lo mejor que se dispone actualmente para el SC.