Escuela de algoritmos de aproximación

Módulo 2: Introducción a los algoritmos de aproximación

Francisco Javier Zaragoza Martínez

Universidad Autónoma Metropolitana Unidad Azcapotzalco

Universidad Autónoma Metropolitana
Casa abierta al tiempo Azcapotzalco

Posgrado en Optimización

ITAM, 14 de septiembre de 2016

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Decisiones

Optimización discreta

Muchos problemas de la sociedad moderna consisten en la toma de decisiones discretas para lograr un objetivo de la mejor manera posible.

- Inventarios.
- Ruteo de vehículos.
- Organización de datos.

Malas noticias

La mayoría de los problemas interesantes de optimización discreta son NP duros. En otras palabras, a menos que $P = NP$, no hay algoritmos de tiempo polinomial que resuelvan estos problemas.

¿Qué hacer?

Disyuntiva

Si P $\neq$ NP entonces no podemos tener simultáneamente:

- Soluciones óptimas.
- En tiempo polinomial.
- Para todas las instancias.

Relajación

Debemos relajar al menos uno de estos tres requerimientos.

Tres relajaciones posibles

Relajar: Para todas las instancias

En este caso se buscan algoritmos polinomiales que encuentren soluciones óptimas para instancias especiales.

Relajar: En tiempo polinomial

En este caso se buscan soluciones óptimas para las instancias que se presenten, aunque tome una gran cantidad de tiempo: investigación de operaciones, programación matemática, etc.

Relajar: Soluciones óptimas

En este caso se buscan algoritmos polinomiales que encuentren soluciones buenas: heurísticas, metaheurísticas, algoritmos de aproximación.

Algoritmos de aproximación

Definición

Un algoritmo de aproximación con garantía α para un problema de optimización es un algoritmo de tiempo polinomial que, para todas las instancias del problema, produce una solución cuyo valor está dentro de un factor α del valor de la solución óptima.

Detalles

- $\alpha \geq 1$ para los problemas de minimización.
- $\alpha \leq 1$ para los problemas de maximización.
- α no es necesariamente una constante.

Ejemplos de garantía

$\alpha = 2$

Un algoritmo de aproximación con garantía $\alpha = 2$ para un problema de minimización es un algoritmo de tiempo polinomial que, para todas las instancias del problema, produce una solución cuyo valor es a lo más el doble del valor de la solución óptima.

$\alpha = \frac{1}{2}$

Un algoritmo de aproximación con garantía $\alpha = \frac{1}{2}$ para un problema de maximización es un algoritmo de tiempo polinomial que, para todas las instancias del problema, produce una solución cuyo valor es al menos la mitad del valor de la solución óptima.

Límites a la aproximabilidad

PTAS

Un esquema de aproximación de tiempo polinomial (PTAS) es una familia de algoritmos A_ϵ, con un algoritmo para cada $\epsilon > 0$, tal que A_ϵ es un algoritmo de aproximación con garantía $1 \pm \epsilon$.

- Problema de la mochila.
- Agente viajero euclideano.

MAX SNP

Un problema de optimización es MAX SNP duro si no tiene PTAS a menos que $\mathrm{P} = \mathrm{NP}$.

- Clan.
- Conjunto independiente.

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Cubierta por conjuntos

Definición

Sea $E = \{e_1, \ldots, e_n\}$ un conjunto, $S_1, \ldots, S_m$ una familia de subconjuntos de E y $w_j \geq 0$ el costo del subconjunto S_j. Se desea encontrar una subfamilia de subconjuntos de costo total mínimo tal que contenga a todos los elementos de E. En otras palabras, se desea encontrar $I \subseteq \{1, \ldots, m\}$ que minimice $w(I) = \sum_{j \in I} w_j$ sujeto a que $\cup_{j \in I} S_j = E$.

Ejemplo

Sea $E = \{a, e, i, o, u\}$, con $S_1 = \{e, u\}$, $S_2 = \{a, e\}$, $S_3 = \{e, i, o\}$, $S_4 = \{e, u\}$, $S_5 = \{e, i\}$, $S_6 = \{a, o\}$, $S_7 = \{i, o\}$ y costos $w_1 = 5$, $w_2 = 6$, $w_3 = 9$, $w_4 = 6$, $w_5 = 7$, $w_6 = 6$, $w_7 = 7$.

- $I = \{2, 3, 4\}$ con $w(I) = w_2 + w_3 + w_4 = 21$.
- $I = \{1, 6, 7\}$ con $w(I) = w_1 + w_6 + w_7 = 18$.

Cubierta por vértices

Definición

Sea $G = (V, E)$ una gráfica con $w_v \geq 0$ el costo del vértice v. Se desea encontrar un subconjunto de vértices de costo total mínimo tal que cada arista tenga al menos un vértice en ese subconjunto.

Caso especial

Cubierta por vértices es un caso especial de cubierta por conjuntos en el que para $v \in V$ el subconjunto S_v contiene a las aristas incidentes con v.

Complejidad

Cubierta por vértices y cubierta por conjuntos son NP duros.

Modelo de programación entera

Variables de decisión

Para cada $j \in \{1, \ldots, m\}$ sea $x_j \in \{0, 1\}$ una variable tal que $x_j = 1$ si y sólo si el conjunto S_j se escoge como parte de la solución.

Modelo

$$
\begin{aligned}
z^*_{IP} &= \min \sum_{j=1}^{m} w_j x_j \\
& \quad \text{sujeto a} \\
\sum_{j : e_i \in S_j} x_j &\geq 1 \text{ para toda } 1 \leq i \leq n \\
x_j &\in \{0, 1\} \text{ para toda } 1 \leq j \leq m.
\end{aligned}
$$

Ejemplo

Ejemplo

Sea $E = \{a, e, i, o, u\}$, con $S_1 = \{e, u\}$, $S_2 = \{a, e\}$, $S_3 = \{e, i, o\}$, $S_4 = \{e, u\}$, $S_5 = \{e, i\}$, $S_6 = \{a, o\}$, $S_7 = \{i, o\}$ y costos $w_1 = 5$, $w_2 = 6$, $w_3 = 9$, $w_4 = 6$, $w_5 = 7$, $w_6 = 6$, $w_7 = 7$.

Modelo

$$
\begin{aligned}
z^*_{IP} &= \min 5x_1 + 6x_2 + 9x_3 + 6x_4 + 7x_5 + 6x_6 + 7x_7 \\
x_2 + x_6 &\geq 1 \text{ ya que } a \in S_2, S_6 \\
x_1 + x_2 + x_3 + x_4 + x_5 &\geq 1 \\
x_3 + x_5 + x_7 &\geq 1 \\
x_3 + x_6 + x_7 &\geq 1 \\
x_1 + x_4 &\geq 1 \\
x_j &\in \{0, 1\} \text{ para toda } 1 \leq j \leq 7.
\end{aligned}
$$

Relajación de programación lineal

Complejidad

Programación entera es NP duro, programación lineal está en P. Podemos obtener un programa lineal reemplazando $x_j \in \{0, 1\}$ por $x_j \geq 0$.

Relajación

$$
\begin{aligned}
z^*_{LP} &= \min \sum_{j=1}^{m} w_j x_j \\
& \quad \text{sujeto a} \\
\sum_{j : e_i \in S_j} x_j &\geq 1 \text{ para toda } 1 \leq i \leq n \\
x_j &\geq 0 \text{ para toda } 1 \leq j \leq m.
\end{aligned}
$$

Ejemplo

Relajación

$$
\begin{aligned}
z^*_{LP} &= \min 5x_1 + 6x_2 + 9x_3 + 6x_4 + 7x_5 + 6x_6 + 7x_7 \\
& \quad \text{sujeto a} \\
x_2 + x_6 &\geq 1 \\
x_1 + x_2 + x_3 + x_4 + x_5 &\geq 1 \\
x_3 + x_5 + x_7 &\geq 1 \\
x_3 + x_6 + x_7 &\geq 1 \\
x_1 + x_4 &\geq 1 \\
x_j &\geq 0 \text{ para toda } 1 \leq j \leq 7.
\end{aligned}
$$

Solución óptima de la relajación

$z^*_{LP} = 18$ con $x^*_1 = 1$, $x^*_2 = 1$, $x^*_3 = 0$, $x^*_4 = 0$, $x^*_5 = 0$, $x^*_6 = 0$, $x^*_7 = 1$.

Otro ejemplo

Relajación

$$
\begin{aligned}
z^*_{LP} &= \min x_1 + x_2 + x_3 + x_4 + x_5 \\
& \quad \text{sujeto a} \\
x_1 + x_2 + x_3 &\geq 1 \\
x_2 + x_3 + x_4 &\geq 1 \\
x_3 + x_4 + x_5 &\geq 1 \\
x_2 + x_4 + x_5 &\geq 1 \\
x_1 + x_2 + x_5 &\geq 1 \\
x_j &\geq 0 \text{ para toda } 1 \leq j \leq 5.
\end{aligned}
$$

Solución óptima de la relajación

$z^*_{LP} = \frac{3}{2}$ con $x^*_1 = 0$, $x^*_2 = \frac{1}{2}$, $x^*_3 = \frac{1}{2}$, $x^*_4 = 0$, $x^*_5 = \frac{1}{2}$.

Dos casos

Solución entera

A veces ocurre que el programa lineal tiene solución óptima entera. En este caso ya tenemos una solución óptima del programa entero.

Solución fraccionaria

Lo usual es que el programa lineal tenga una solución óptima fraccionaria. En este caso no tenemos ni siquiera una solución factible del programa entero. Pero tenemos una cota al valor óptimo.

- En un problema de minimización $z^*_{LP} \leq z^*_{IP}$.
- En un problema de maximización $z^*_{LP} \geq z^*_{IP}$.

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Recuperación de una solución

Redondeo simple

Sea x^* la solución óptima (fraccionaria) del programa lineal. Una primera idea es definir $\bar{x}_j = \lceil x_j^* \rceil$. Esta solución será factible, pero tal vez muy cara.

Algoritmo de redondeo (determinista)

Sea f la máxima cantidad de conjuntos en los que aparece cualquier elemento de E. Definamos $\bar{x}_j = 1$ si $x_j^* \geq \frac{1}{f}$ y $\bar{x}_j = 0$ en los demás casos[a].

[a] Hochbaum. *Approximation algorithms for the set covering and vertex cover problems*. SIAM Journal on Computing, 11(3), 555-556, 1982.

Ejemplo

En nuestro ejemplo $x_1^* = x_4^* = 0$ y $x_2^* = x_3^* = x_5^* = \frac{1}{2}$. Además $f = 3$, por lo que tomamos $\bar{x}_1 = \bar{x}_4 = 0$ y $\bar{x}_2 = \bar{x}_3 = \bar{x}_5 = 1$.

Algoritmo de aproximación con garantía f

Factibilidad

Sea $e_i \in E$. Como x^* es factible en el programa lineal, tenemos que $\sum_{j:e_i \in S_j} x_j^* \geq 1$. Como esta suma tiene f sumandos o menos, al menos uno de ellos cumple $x_j^* \geq \frac{1}{f}$. Por lo que $\bar{x}_j = 1$ con $e_i \in S_j$.

Garantía

Sea $I = \{j : \bar{x}_j = 1\}$. Como $1 \leq f \cdot x_j^*$ para $j \in I$, tenemos que

$$w(I) = \sum_{j \in I} w_j \leq \sum_{j \in I} w_j (f \cdot x_j^*) \leq \sum_{j=1}^{m} w_j (f \cdot x_j^*) = f \sum_{j=1}^{m} w_j x_j^* = f \cdot z_{LP}^*.$$

Como $z_{LP}^* \leq z_{IP}^*$, tenemos que

$$w(I) \leq f \cdot z_{IP}^*.$$

Caso especial

Cubierta por vértices

Como cada arista es adyacente a dos vértices, tenemos que $f = 2$ y por lo tanto, un algoritmo de aproximación de garantía 2 para este problema.
Este resultado es casi el mejor que se conoce para este problema.

Resultado negativo

No es aproximable con garantía ≈ 1.3606, a menos que $\mathrm{P} = \mathrm{NP}$[a].

[a] Dinur y Safra. *On the Hardness of Approximating Minimum Vertex Cover*. Annals of Mathematics. 2nd Series, 162(1), 439-485, 2005.

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Dualidad de programación lineal

Programas lineales primal y dual

Considere el programa lineal primal

$$z^*_{LP} = \min w^\top x \text{ sujeto a } Ax \geq b \text{ y } x \geq 0.$$

Entonces su programa lineal dual es

$$z^*_{DP} = \max b^\top y \text{ sujeto a } A^\top y \leq w \text{ y } y \geq 0.$$

Dualidad de programación lineal

Teorema débil de dualidad

Si x, y son primal y dual factibles, entonces $w^\top x \geq b^\top y$.

Teorema de holguras complementarias

Si x^*, y^* son primal y dual óptimas, entonces:

1. $x_j^* > 0 \Rightarrow$ la j-ésima restricción del dual se cumple como igualdad.
2. $y_j^* > 0 \Rightarrow$ la j-ésima restricción del primal se cumple como igualdad.

Teorema fuerte de dualidad

Si los programas lineales primal y dual son factibles, entonces $z_{LP}^* = z_{DP}^*$.

Dual de la relajación lineal

Dual

Para cada elemento $e_i \in E$, sea y_i su variable dual correspondiente.

$$\begin{aligned} z^*_{DP} &= \max \sum_{i=1}^{n} y_i \\ & \quad \text{sujeto a} \\ \sum_{i:e_i \in S_j} y_i &\leq w_j \text{ para toda } 1 \leq j \leq m \\ y_i &\geq 0 \text{ para toda } 1 \leq i \leq n. \end{aligned}$$

Algoritmo de redondeo dual

Sea y^* una solución dual óptima y sea $I = \{j : \sum_{i:e_i \in S_j} y_i^* = w_j\}$.

Ejemplo

Relajación dual

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
y_2 + y_5 &\leq 5 \\
y_1 + y_2 &\leq 6 \\
y_2 + y_3 + y_4 &\leq 9 \\
y_2 + y_5 &\leq 6 \\
y_2 + y_3 &\leq 7 \\
y_1 + y_4 &\leq 6 \\
y_3 + y_4 &\leq 7 \\
y_i &\geq 0 \text{ para toda } 1 \leq i \leq 5.
\end{aligned}
$$

Solución óptima de la relajación dual

$z^*_{DP} = 18$ con $y^*_1 = 6$, $y^*_2 = 0$, $y^*_3 = 7$, $y^*_4 = 0$, $y^*_5 = 5$.

Otro ejemplo

Relajación dual

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
& \quad \text{sujeto a} \\
y_1 + y_5 &\leq 1 \\
y_1 + y_2 + y_4 + y_5 &\leq 1 \\
y_1 + y_2 + y_3 &\leq 1 \\
y_2 + y_3 + y_4 &\leq 1 \\
y_3 + y_4 + y_5 &\leq 1 \\
y_i &\geq 0 \text{ para toda } 1 \leq i \leq 5.
\end{aligned}
$$

Solución óptima de la relajación dual

$z^*_{DP} = \frac{3}{2}$ con $y^*_1 = \frac{1}{2}$, $y^*_2 = 0$, $y^*_3 = \frac{1}{2}$, $y^*_4 = \frac{1}{2}$, $y^*_5 = 0$.

Factibilidad

I es una cubierta

Suponga que e_k no está cubierto. Entonces $\sum_{i:e_i \in S_j} y_i^* < w_j$ para cada S_j que contiene a e_k. Definamos

$$\epsilon = \min_{j:e_k \in S_j} \left(w_j - \sum_{i:e_i \in S_j} y_i^* \right) > 0.$$

Considere la solución dual y' con $y'_k = y_k + \epsilon$ y $y'_i = y_i$ para toda $i \neq k$. Entonces y' es dual factible y además

$$\sum_{i=1}^{n} y'_i > \sum_{i=1}^{n} y_i^*$$

contradiciendo la optimalidad de y^*.

Garantía

Garantía f

Ya que $j \in I$ sólo si $w_j = \sum_{i:e_i \in S_j} y_i^*$, por lo tanto

$$\begin{aligned} w(I) = \sum_{j \in I} w_j &= \sum_{j \in I} \sum_{i:e_i \in S_j} y_i^* \\ &= \sum_{i=1}^{n} |\{j \in I : e_i \in S_j\}| \cdot y_i^* \\ &\leq \sum_{i=1}^{n} f \cdot y_i^* \\ &= f \sum_{i=1}^{n} y_i^* \\ &\leq f \cdot z_{IP}^*. \end{aligned}$$

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Método primal dual

Programas lineales

Los programas lineales generales se pueden resolver eficientemente, es decir, en tiempo polinomial. Pero existen métodos más rápidos para programas lineales particulares.

Redondeo dual

Este método sólo usó dos propiedades de la solución óptima:

1. $\sum_{i=1}^{n} y_i \leq z^*_{IP}$, válido para cualquier solución dual factible.
2. $j \in I$ sólo si $w_j = \sum_{i:e_i \in S_j} y_i$.

No necesitamos resolver el programa lineal dual: comenzamos con una solución dual factible y la modificamos mientras no tengamos una cubierta.

Algoritmo primal dual

Algoritmo primal dual

Este algoritmo mantiene factibilidad dual[a].

1. $y \leftarrow 0$
2. $I \leftarrow \emptyset$
3. Mientras exista $e_k \notin \cup_{j \in I} S_j$
   1. Incrementa y_k hasta que $w_\ell = \sum_{j:e_j \in S_\ell}$ para alguna ℓ con $e_k \in S_\ell$
   2. $I \leftarrow I \cup \{\ell\}$
4. Emite I a la salida

[a] Bar-Yehuda y Even. *A linear-time approximation algorithm for the weighted vertex cover problem*. Journal of Algorithms. 2, 198-203, 1981.

Ejemplo

Paso 1

$y_1 = 0$, $y_2 = 0$, $y_3 = 0$, $y_4 = 0$, $y_5 = 0$, $I = \emptyset$, $\cup_{j \in I} S_j = \emptyset$.

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
y_2 + y_5 &\leq 5 \\
y_1 + y_2 &\leq 6 \\
y_2 + y_3 + y_4 &\leq 9 \\
y_2 + y_5 &\leq 6 \\
y_2 + y_3 &\leq 7 \\
y_1 + y_4 &\leq 6 \\
y_3 + y_4 &\leq 7
\end{aligned}
$$

Entonces $y_1 \leftarrow 6$, $\ell \leftarrow 2$, $I \leftarrow \{2\}$.

Ejemplo

Paso 2

$y_1 = 6$, $y_2 = 0$, $y_3 = 0$, $y_4 = 0$, $y_5 = 0$, $I = \{2\}$, $\cup_{j \in I} S_j = \{1, 2\}$.

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
y_2 + y_5 &\leq 5 \\
y_1 + y_2 &\leq 6 \\
y_2 + y_3 + y_4 &\leq 9 \\
y_2 + y_5 &\leq 6 \\
y_2 + y_3 &\leq 7 \\
y_1 + y_4 &\leq 6 \\
y_3 + y_4 &\leq 7
\end{aligned}
$$

Entonces $y_3 \leftarrow 7$, $\ell \leftarrow 5$, $I \leftarrow \{2, 5\}$.

Ejemplo

Paso 3

$y_1 = 6$, $y_2 = 0$, $y_3 = 7$, $y_4 = 0$, $y_5 = 0$, $I = \{2, 5\}$, $\cup_{j \in I} S_j = \{1, 2, 3\}$.

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
y_2 + y_5 &\leq 5 \\
y_1 + y_2 &\leq 6 \\
y_2 + y_3 + y_4 &\leq 9 \\
y_2 + y_5 &\leq 6 \\
y_2 + y_3 &\leq 7 \\
y_1 + y_4 &\leq 6 \\
y_3 + y_4 &\leq 7
\end{aligned}
$$

Entonces $y_4 \leftarrow 0$, $\ell \leftarrow 6$, $I \leftarrow \{2, 5, 6\}$.

Ejemplo

Paso 4

$y_1 = 6$, $y_2 = 0$, $y_3 = 7$, $y_4 = 0$, $y_5 = 0$, $I = \{2, 5, 6\}$, $\cup_{j \in I} S_j = \{1, 2, 3, 4\}$.

$$
\begin{aligned}
z^*_{LP} &= \max y_1 + y_2 + y_3 + y_4 + y_5 \\
y_2 + y_5 &\leq 5 \\
y_1 + y_2 &\leq 6 \\
y_2 + y_3 + y_4 &\leq 9 \\
y_2 + y_5 &\leq 6 \\
y_2 + y_3 &\leq 7 \\
y_1 + y_4 &\leq 6 \\
y_3 + y_4 &\leq 7
\end{aligned}
$$

Entonces $y_5 \leftarrow 5$, $\ell \leftarrow 1$, $I \leftarrow \{1, 2, 5, 6\}$.

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Algoritmo glotón

Algoritmos glotones

Toman decisiones en rondas, cada una optimizando una decisión, aunque la secuencia de decisiones no lleve a una solución óptima.

Ventajas y desventajas

- Son muy fáciles de implementar.
- Son muy rápidos.
- A veces entregan soluciones óptimas.
- A veces entregan soluciones aproximadas.
- Fallan con frecuencia.

Algoritmo glotón

Algoritmo glotón

Este algoritmo construye I de forma glotona[a].

1. $I \leftarrow \emptyset$
2. $F_j \leftarrow S_j$ para toda j
3. Mientras I no sea una cubierta
   1. Escoge $\ell \leftarrow j$ que minimice $\frac{w_j}{|F_j|}$ con $F_j \neq \emptyset$
   2. $I \leftarrow I \cup \{\ell\}$
   3. $F_j \leftarrow F_j \setminus S_\ell$ para toda j
4. Emite I a la salida

[a] Chvatal. *A greedy heuristic for the set-covering problem*. Mathematics of Operations Research. 4(3), 233-235, 1979.

Ejemplo

Ejemplo

Sea $E = \{a, e, i, o, u\}$, con $S_1 = \{e, u\}$, $S_2 = \{a, e\}$, $S_3 = \{e, i, o\}$, $S_4 = \{e, u\}$, $S_5 = \{e, i\}$, $S_6 = \{a, o\}$, $S_7 = \{i, o\}$ y costos $w_1 = 5$, $w_2 = 6$, $w_3 = 9$, $w_4 = 6$, $w_5 = 7$, $w_6 = 6$, $w_7 = 7$.

Algoritmo glotón

1. $I = \emptyset$, $F_1 = \{e, u\}$, $F_2 = \{a, e\}$, $F_3 = \{e, i, o\}$, $F_4 = \{e, u\}$, $F_5 = \{e, i\}$, $F_6 = \{a, o\}$, $F_7 = \{i, o\}$, $\frac{w_1}{|F_1|} = \frac{5}{2}$, $\frac{w_2}{|F_2|} = 3$, $\frac{w_3}{|F_3|} = 3$, $\frac{w_4}{|F_4|} = 3$, $\frac{w_5}{|F_5|} = \frac{7}{2}$, $\frac{w_6}{|F_6|} = 3$, $\frac{w_7}{|F_7|} = \frac{7}{2}$, $\ell = 1$.
2. $I = \{1\}$, $F_1 = \emptyset$, $F_2 = \{a\}$, $F_3 = \{i, o\}$, $F_4 = \emptyset$, $F_5 = \{i\}$, $F_6 = \{a, o\}$, $F_7 = \{i, o\}$, $\frac{w_2}{|F_2|} = 6$, $\frac{w_3}{|F_3|} = \frac{9}{2}$, $\frac{w_5}{|F_5|} = 7$, $\frac{w_6}{|F_6|} = 3$, $\frac{w_7}{|F_7|} = \frac{7}{2}$, $\ell = 6$.
3. $I = \{1, 6\}$, $F_1 = \emptyset$, $F_2 = \emptyset$, $F_3 = \{i\}$, $F_4 = \emptyset$, $F_5 = \{i\}$, $F_6 = \emptyset$, $F_7 = \{i\}$, $\frac{w_3}{|F_3|} = 9$, $\frac{w_5}{|F_5|} = 7$, $\frac{w_7}{|F_7|} = 7$, $\ell = 5$.

Un par de cosas antes

Números armónicos

Para $k \geq 1$ sea $H_k = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{k}$.

Constante de Euler

$\gamma = \lim_{k\to\infty}(H_k - \ln k) \approx 0.5772$.

Desigualdad

Para $k \geq 1$ sean $a_1, \ldots, a_k$ y $b_1, \ldots, b_k$ reales positivos. Entonces

$$\min_{1\leq i\leq k} \frac{a_i}{b_i} \leq \frac{\sum_{i=1}^{k} a_i}{\sum_{i=1}^{k} b_i} \leq \max_{1\leq i\leq k} \frac{a_i}{b_i}.$$

Algoritmo glotón

Idea

Sea I^* una solución óptima con costo z^*. Como I^* cubre los n elementos con costo promedio $\frac{z^*}{n}$, entonces algún S_j cubre sus elementos con costo promedio $\leq \frac{z^*}{n}$. En general, si ya se han cubierto k elementos, entonces I^* cubre los $n-k$ elementos faltantes con costo promedio $\frac{z^*}{n-k}$ y entonces algún S_j cubre sus elementos faltantes F_j con costo promedio $\leq \frac{z^*}{n-k}$.

Ejemplo

La solución óptima I^* tiene costo $z^* = 18$. En el algoritmo glotón:

1. S_1 cubre sus 2 elementos faltantes a costo $2.5 = \frac{5}{2} \leq \frac{18}{5} = 3.6$.
2. S_6 cubre sus 2 elementos faltantes a costo $3 = \frac{6}{2} \leq \frac{18}{3} = 6$.
3. S_5 cubre su 1 elemento faltante a costo $7 = \frac{7}{1} \leq \frac{18}{1} = 18$.

Con costo $2.5 + 2.5 + 3 + 3 + 7 \leq 18(\frac{1}{5} + \frac{1}{5} + \frac{1}{3} + \frac{1}{3} + 1) \leq 18H_5$.

Garantía (parte 1)

Algoritmo de aproximación con garantía H_n

Supongamos que el algoritmo toma r rondas y que al principio de la ronda k falta cubrir n_k elementos ($n_1 = n$ y $n_{r+1} = 0$). Escoja $1 \leq k \leq r$ y sea I_k el conjunto de los índices de los conjuntos elegidos en las rondas 1 a $k-1$. Para $1 \leq j \leq n$, ea F_j el conjunto de los elementos no cubiertos en S_j al inicio de la ronda k. Sea ℓ el índice elegido en la ronda k. Entonces

$$w_\ell \leq \frac{n_k - n_{k+1}}{n_k} z^*$$

que probaremos en un momento.

Garantía (parte 2

Algoritmo de aproximación con garantía H_n

De esto se sigue que

$$
\begin{aligned}
w(I) &\leq \sum_{k=1}^{\ell} \frac{n_k - n_{k+1}}{n_k} z^* \\
&\leq \sum_{k=1}^{\ell} \left(\frac{1}{n_k} + \frac{1}{n_k - 1} + \cdots + \frac{1}{n_{k+1} + 1} \right) z^* \\
&= \sum_{i=1}^{n} \frac{1}{i} z^* \\
&= H_n z^*.
\end{aligned}
$$

Esto es lo que queríamos demostrar.

Garantía (parte 3)

Algoritmo de aproximación con garantía H_n

Sólo nos falta probar la primera desigualdad. Primero observe que:

$$\frac{w_\ell}{|F_\ell|} = \min_{j:F_j \neq \emptyset} \frac{w_j}{|F_j|} \leq \min_{j \in I^*:F_j \neq \emptyset} \frac{w_j}{|F_j|}.$$

Ahora observe que

$$\min_{j \in I^*:F_j \neq \emptyset} \frac{w_j}{|F_j|} \leq \frac{\sum_{j \in I^*:F_j \neq \emptyset} w_j}{\sum_{j \in I^*:F_j \neq \emptyset} |F_j|} \leq \frac{z^*}{\sum_{j \in I^*} |F_j|} \leq \frac{z^*}{n_k}.$$

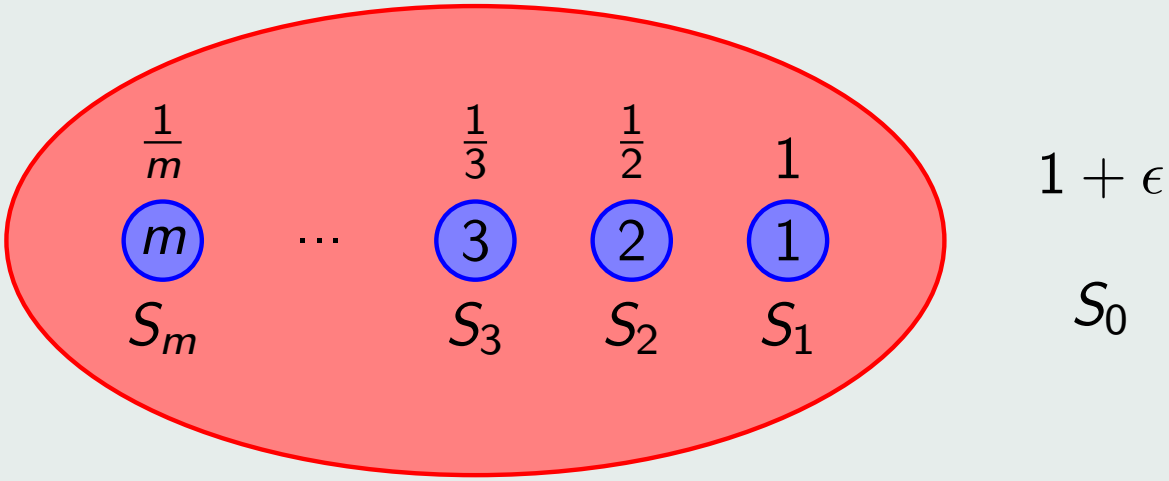
Así $\frac{w_\ell}{|F_\ell|} \leq \frac{z^*}{n_k}$ y como $n_{k+1} = n_k - |F_\ell|$ finalmente tenemos

$$w_\ell \leq \frac{|F_\ell|}{n_k} z^* = \frac{n_k - n_{k+1}}{n_k} z^*.$$

Optimalidad del resultado

Ejemplo justo

Para toda m considere la familia de $m+1$ subconjuntos $S_0 = \{1, \ldots, m\}$ con $c_0 = 1 + \epsilon$ y $S_i = \{i\}$ con $c_i = \frac{1}{i}$ para toda $1 \leq i \leq m$.



El algoritmo glotón escoge estos conjuntos en el orden $S_m, \ldots, S_1$ con costo $H(m)$ mientras que lo óptimo es escoger S_0.

Dureza de aproximación

Teorema

Si existe $c < 1$ para la cual existe un algoritmo de aproximación para cubierta por conjuntos con garantía $c \ln n$, entonces para cada problema NP completo existe un algoritmo determinista con tiempo de ejecución en $O(n^{O(\log \log n)})$[a].

[a] Lund y Yannakakis. *On the hardness of approximating minimization problems*. Journal of the ACM 41(5), 960-981, 1994.

Teorema

Existe una constante $c > 0$ tal que si existe un algoritmo de aproximación para cubierta por conjuntos con garantía $c \ln n$, entonces $\mathrm{P} = \mathrm{NP}$[a].

[a] Feige. *A threshold of ln n for approximating set cover*. Journal of the ACM 45(4), 634-652, 1998.

Contenido

1. Algoritmos de aproximación
2. Cubierta por conjuntos
3. Redondeo determinista
4. Redondeo dual
5. Método primal dual
6. Algoritmo glotón
7. Redondeo aleatorio

Redondeo aleatorio

Algoritmo de redondeo aleatorio

Sea x^* una solución óptima de la relajación lineal. Para cada $1 \leq j \leq m$ y de manera independiente, escoja S_j con probabilidad x_j^*.

Valor esperado

Sea X_j una variable aleatoria que vale 1 si escogemos a S_j y 0 si no. Entonces, usando la linealidad de la esperanza:

$$E[w(I)] = E\left[\sum_{j=1}^{m} w_j X_j\right] = \sum_{j=1}^{m} w_j \Pr[X_j = 1] = \sum_{j=1}^{m} w_j x_j^* = z_{LP}^*.$$

Es decir, el valor esperado de esta solución aleatoria es a lo mucho el del programa lineal. Sin embargo, nada asegura que será factible.

Probabilidad de falla

Probabilidad de que un elemento no esté cubierto

La probabilidad de que e_i no esté cubierto es la probabilidad de que ningún conjunto que lo contiene haya sido elegido. Por la independencia esto es $\prod_{j:e_i \in S_j}(1 - x_j^*)$. Ahora usaremos que $1 - x \leq e^{-x}$ para toda $x \geq 0$:

$$\begin{aligned}
\Pr[e_i \text{ no está cubierto}] &= \prod_{j:e_i \in S_j} (1 - x_j^*) \\
&\leq \prod_{j:e_i \in S_j} e^{-x_j^*} \\
&= e^{-\sum_{j:e_i \in S_j} x_j^*} \\
&\leq e^{-1}.
\end{aligned}$$

Esta probabilidad es alta (≈ 0.3679). ¿Cómo mejorar esto?

Probabilidad de falla

Intentos repetidos

Sea $c \geq 2$. Para cada conjunto S_j haremos $c \ln n$ intentos. Entonces, la probabilidad de que S_j no sea elegido es $(1 - x_j^*)^{c \ln n}$ y por lo tanto:

$$\begin{aligned}
\Pr[e_i \text{ no está cubierto}] &= \prod_{j : e_i \in S_j} (1 - x_j^*)^{c \ln n} \\
&\leq \prod_{j : e_i \in S_j} e^{-x_j^* (c \ln n)} \\
&= e^{-(c \ln n) \sum_{j : e_i \in S_j} x_j^*} \\
&\leq e^{-c \ln n} \\
&= n^{-c}.
\end{aligned}$$

Probabilidad de falla

Probabilidad de que algún elemento no esté cubierto

La probabilidad de que no tengamos una cubierta es la misma que la de que algún elemento no esté cubierto. Por la cota de la unión:

$$\Pr[\exists e_i \text{ no está cubierto}] \leq \sum_{i=1}^{n} \Pr[e_i \text{ no está cubierto}] \leq n^{1-c}.$$

En otras palabras

Tenemos un algoritmo aleatorizado de tiempo polinomial que entrega una cubierta con probabilidad al menos $1 - n^{1-c}$.

Algoritmo de aproximación aleatorizado

Teorema

El algoritmo de aproximación aleatorizado redondeo aleatorio con $c \ln n$ intentos por conjunto entrega una cubierta con probabilidad $\geq 1 - n^{1-c}$. En ese caso, el costo esperado de la cubierta es $\leq 2c(\ln n)z^*_{LP}$.

Esto es todo por hoy

¡Gracias!

Posgrado en Optimización

Posgrado en Optimización

UAM Azcapotzalco