Algoritmos aproximados

Tecnología Digital V: Diseño de Algoritmos
Universidad Torcuato Di Tella

El querido problema de la mochila

Knapsack-01 (KP-01)

Debemos llenar una mochila eligiendo entre varios objetos posibles. Cada producto tiene un peso, una medida de comfort (beneficio) y la mochila tolera un peso máximo de carga. **Los objetos no pueden ser fraccionados, y solo se puede elegir una unidad de cada objeto.**

Una instancia del KP-01 está dada por

- $N = \{1, \ldots, n\}$ el conjunto de objetos (o productos).
- $p_i \in \mathbb{Z}_+$ el peso del objeto i, para $i = 1, \ldots, n$.
- $b_i \in \mathbb{Z}_+$ el beneficio del objeto i, para $i = 1, \ldots, n$.
- Capacidad $C \in \mathbb{Z}_+$ de la mochila (peso máximo).

Problema

Determinar qué objetos debemos incluir en la mochila sin excedernos del peso máximo C, de modo tal de maximizar el beneficio total entre los objetos seleccionados.

Heurística golosa

Heurística golosa

- Ordenar los objetos según su beneficio por unidad de peso. b_i/p_i
- Elegir los objetos en orden decreciente de beneficio por unidad de peso, hasta que no se pueda agregar más.
- Mientras haya espacio en la mochila, agregar el objeto con mayor beneficio por unidad de peso.
- **Complejidad:** $O(n \log n)$.

- Recordemos que la complejidad de resolver el problema utilizando Programación Dinámica es $O(nC)$.
- El problema de la mochila es NP completo, por lo que no se espera que haya un algoritmo polinomial que lo resuelva.
- Si el tamaño de C es muy grande, la programación dinámica puede ser muy costosa aunque no haya tantos objetos a considerar.

Esquema de aproximación

Sea Π un problema de optimización NP-difícil con función objetivo f_Π. Un algoritmo A es un esquema de aproximación para Π si, dado un parámetro de error $\varepsilon > 0$, al recibir una instancia I de Π, devuelve una solución s tal que:

- $f_\Pi(I, s) \leq (1 + \varepsilon) \cdot \text{OPT}$ si Π es un problema de minimización.
- $f_\Pi(I, s) \geq (1 - \varepsilon) \cdot \text{OPT}$ si Π es un problema de maximización.

Si queremos un algoritmo aproximado al 10%, $\varepsilon = 0.1$ y $1/\varepsilon = 10$.

Si queremos un algoritmo aproximado al 5%, $\varepsilon = 0.05$ y $1/\varepsilon = 20$.

Si queremos un algoritmo aproximado al 1%, $\varepsilon = 0.01$ y $1/\varepsilon = 100$.

Esquema de aproximación polinomial

Un esquema de aproximación A se dice que es un esquema de aproximación polinomial (PTAS) si para cada $\varepsilon > 0$, su tiempo de ejecución está acotado por un polinomio en el tamaño de la instancia I.

Esto, sin embargo, significa que podría ser exponencial con respecto a $1/\varepsilon$, en cuyo caso acercarse a la solución óptima es increíblemente difícil. Por ejemplo un algoritmo $O(n^2 2^{\frac{1}{\varepsilon}})$ es polinomial con respecto a n.

Por lo tanto, un esquema de aproximación polinomial en tiempo completo, o FPTAS, es un esquema de aproximación para el cual el algoritmo está acotado polinómicamente tanto en el tamaño de la instancia I como en $1/\varepsilon$.

Un esquema de aproximación polinomial para KP-01

Una forma más inteligente de resolver el problema de la mochila es mediante haciendo fuerza bruta de parte de la solución y luego usar el algoritmo goloso para terminar el resto.

En particular, consideramos todos los $O(kn^k)$ subconjuntos posibles de objetos que tengan hasta k objetos, donde k es una constante fija.

Luego, para cada subconjunto, usar el algoritmo goloso para llenar el resto de la mochila en tiempo $O(n)$.

Elegir el subconjunto con mejor beneficio. El tiempo total de ejecución de este algoritmo es $O(kn^{k+1})$.

Teorema

Si S es el subconjunto óptimo, entonces la aproximación resultante $B(A)$ cumple:

$$B(S) \leq B(A)\left(1 + \frac{1}{k}\right)$$

Si el conjunto S óptimo tiene tamaño menor o igual a k, entonces este algoritmo devuelve la solución óptima porque S será considerado en el paso de fuerza bruta.

Si no, sea $H = \{a_1, \dots, a_k\}$ el conjunto de los k objetos con mejor beneficio en S.

Dado que todos los subconjuntos de tamaño k son considerados en el paso de fuerza bruta, H debe ser uno de ellos.

Queremos ver que pasa con los objetos que están en la solución y no son estos k objetos.

Sea $L_1 = O \setminus H = \{a_{k+1}, \ldots, a_x\}$, los objetos restantes en O ordenados en orden decreciente de beneficio por unidad de peso. Sea m el índice del primer objeto en L_1 que no es elegido por el algoritmo goloso después de elegir H en el paso de fuerza bruta. La razón por la que el objeto a_m no es elegido debe ser porque su tamaño es mayor que el espacio restante C_e.

Esto significa que el algoritmo goloso solo ha elegido objetos que tienen una densidad de beneficio de al menos b_m/p_m porque elige objetos en orden decreciente de beneficio por unidad de peso.

Hasta ahora, la mochila contiene $H, a_{k+1}, \ldots, a_{m-1}$ y algunos objetos no en O. Sea G los objetos elegidos por el algoritmo goloso hasta ahora, todos ellos tienen una densidad de beneficio de al menos b_m/p_m. Los objetos en G que no están en el conjunto óptimo O, tienen un tamaño total

$$D = C - C_e - \sum_{i=1}^{m-1} p_i$$

Como tenemos acotado la densidad de beneficio.

$$B(G) \geq \sum_{i=k+1}^{m-1} b_i + D\frac{b_m}{p_m}$$

Ahora podemos escribir el beneficio de S como:

$$
\begin{aligned}
B(S) &= \sum_{i=1}^{k} b_i + \sum_{i=k+1}^{m-1} b_i + \sum_{i=m}^{|S|} b_i \\
&\leq B(H) + \left(B(G) - D\frac{b_m}{p_m}\right) + \left(C - \sum_{i=1}^{m-1} p_i\right)\frac{b_m}{p_m} \\
&= B(H) + B(G) + C_e\frac{b_m}{p_m} < B(H \cup G) + b_m
\end{aligned}
$$

El mejor subconjunto A devuelto después de los pasos de fuerza bruta y goloso tendrá al menos tanto beneficio como H y G combinados, ya que sabemos que la unión de H y G debe haber sido uno de los subconjuntos considerados. Como $B(S) < B(H \cup G) + b_m$ y $B(A) \geq B(H \cup G)$, entonces $B(S) - B(A) < b_m$. Los k objetos en H tienen al menos tanto beneficio como el de a_m, entonces $b_m(k+1) \leq \sum_{i=1}^{m} b_i \leq B(S)$ y se deduce lo que queríamos.

El esquema de aproximación polinomial o PTAS con una aproximación de $(1-\varepsilon)$ donde $1/\varepsilon = k+1$. El tiempo de ejecución resultante es $O(\frac{1}{\varepsilon} n^{\frac{1}{\varepsilon}})$, por lo que el esquema de aproximación es polinomial en n pero no en $1/\varepsilon$.

Existen problemas que tienen un PTAS y no un FTPAS. En el problema de la mochila podemos encontrar un algoritmo (1-ε)-aproximado en tiempo polinomial en n y $1/\varepsilon$.

- Definir $K = \frac{\varepsilon b_{max}}{n}$ y $C' = C/K$.
- Definir $b'_i = \lfloor b_i/K \rfloor$.
- Resolver el problema de la mochila usando $O(nC')$

Observación

$C \leq b_{max} n$.

$C' = \frac{C}{K} \leq \frac{b_{max} n}{\varepsilon b_{max}/n} = \frac{n^2}{\varepsilon}$.

Entonces $O(nC') = O(n^3/\varepsilon)$

Esquema de aproximación polinomial completo para KP-01 - FTPAS

Sea O el conjunto óptimo que devuelve el máximo beneficio posible. Dado que escalamos por K y luego redondeamos hacia abajo, cualquier objeto a tendrá $K \cdot b'_i \leq b_i$ con una diferencia de a lo sumo K. Por lo tanto, lo más que el beneficio del conjunto óptimo O puede disminuir es a lo sumo nK:

$$B(O) - K \cdot B'(O) \leq nK$$

.

Sea S' la respuesta de programación dinámica:

$$B(S') \geq KB'(O) \geq B(O) - nK = OPT - \varepsilon b_{max} \geq (1 - \varepsilon) \cdot OPT$$