MA4701. Optimización Combinatorial. 2013.
Profesor: José Soto
Escriba(s): Hugo Carrillo , Martin Castillo y Camilo Iturra
Fecha: 2 de Diciembre de 2013.

Cátedra 26

1. Problema de la Mochila (Knapsack)

Entrada

- $A = \{a_1, ..., a_n\}$ un conjunto de objetos con tamaños $s(a_i) \in \mathbb{Z}^+$ y valores $v(a_i) \in \mathbb{Z}^+$.
- Una mochila de tamaño $B \in \mathbb{Z}^+$.

Salida

- Un subconjunto $S \subseteq A$ de valor $v(S)$ máximo que quepa en la mochila, es decir, que $s(S) = \sum_{a \in S} s(a) \leq B$.

Observación. *Este problema es NP-completo.*

Queremos encontrar un algoritmo de aproximación.
Supuestos

- $s(a_i) \leq B$, es decir, todo objeto cabe en la mochila.
- $s(A) > B$, es decir, el problema no es trivial.

Veamos el siguiente algoritmo glotón para el problema de Knapsack:

1. Ordenar los objetos por densidad
$$\frac{v(a_1)}{s(a_1)} \geq \cdots \geq \frac{v(a_n)}{s(a_n)}$$
2. Incluir uno a uno $a_1, a_2..., a_k$ en la mochila hasta que a_{k+1} no quepa.

El siguiente ejemplo es un mal caso para el algoritmo glotón.

Objeto	v	s	v/s
a_1	2	1	2
a_2	B	B	1

Glotón devuelve $\{a_1\}$ con valor 1 pero el óptimo es $\{a_2\}$ con valor B si $B > 1$.

En lo que sigue se utilizará el programa lineal entero del problema de Knapsack y el algoritmo glotón para encontrar mejores algoritmos de aproximación.

Programa Lineal Entero

$$\begin{aligned} \text{máx} \sum_{a_i \in A} x_i v(a_i) \\ \sum_{a_i \in A} x_i s(a_i) \leq B \\ x_i \in \{0, 1\} \end{aligned}$$

Donde x_i es la variable binaria que indica si a_i se introduce o no en la mochila.

Observación. *Cuando $v(a_i)$ y $s(a_i)$ son enteros el problema se comporta muy bien.*

Programa Relajado

$$\begin{aligned} \text{máx} \sum_{a_i \in A} & x_i v(a_i) \\ \sum_{a_i \in A} & x_i s(a_i) \leq B \\ & 0 \leq x_i \leq 1 \end{aligned}$$

Observación. *Sea $H = \{a_1, ..., a_k\}$ el conjunto de los objetos retornado por el algoritmo glotón, luego la solución del programa lineal relajado será:*

$$x_i = \begin{cases} 1 & si\ 1 \leq i \leq k \\ \frac{B - \sum_{i=1}^{k} s_i}{s_k + 1} & si\ i = k+1 \\ 0 & si\ i \geq k+2 \end{cases}$$

Notemos además que $x^\top v = \sum_{i=1}^{k} 1 \cdot v(a_i) + \frac{B - \sum_{i=1}^{k} s_i}{s_{k+1}} v(a_{k+1})$

Esto inspira el siguiente algoritmo:

1. Guardar el conjunto $S_1 = \{a_1, ..., a_k\}$ retornado por glotón.
2. $S_2 = \{a_{k+1}\}$.
3. Retornar el mejor S de los anteriores.

Veamos que tan buena aproximación realiza el algoritmo anterior:

$$\begin{aligned} v(S) &= \text{máx}\{v(S_1), v(S_2)\} \\ &\geq \frac{1}{2}(v(S_1) + v(S_2)) \\ &= \frac{1}{2}\left(\sum_{i=1}^{k} 1 \cdot v(a_i) + 1 \cdot v(a_{k+1})\right) \\ &\geq \frac{1}{2} x^\top v \quad (\star) \\ &\geq \frac{1}{2} v(OPT) \end{aligned}$$

Donde la desigualdad $(\star)$ se tiene por la observación anterior.
Por lo tanto es una $2-$aproximación.

El problema de la mochila se puede aproximar mucho mejor. Notemos que el algoritmo anterior es igual o peor que el siguiente:

1. Para todo $a \in A$ definir X_a como sigue
   - Agregar a a X_a.
   - Llenar el resto de la mochila usando gloton, es decir, hacer glotón con el problema de Knapsack auxiliar de datos $A' = A \setminus \{a\}$ y $B' = B - s(a)$, e incluir el conjunto retornado a X_a.
2. Devolver el mejor X_a.

¿Como mejorar aún más?
Sea k fijo :

1. Para todo $T \subseteq A$ con $|T| \leq k$ que quepa en la mochila definir X_T como sigue
   - Agregar T a X_T.
   - Borrar de A todos los elementos con valor mayor que $\min_{a \in T} v(a)$.
   - Ordenar los elementos de $A \setminus T$ de mayor a menor densidad y agregarlos en X_T uno a uno de manera glotona y parar cuando alguno no quepa.
2. Devolver el mejor X_T.

¿Cual es la complejidad con k fijo?
Ordenar toma $O(n \log(n))$. Hay $O(n + n^2 + ... + n^k) = O(kn^k)$ conjuntos de tamaño $\leq k$. Cada uno de ellos toma $O(n)$ trabajo. Luego el total es $O(kn^{k+1} + nlog(n)) = O(kn^{k+1})$. Si k es constante entonces el algoritmo es polinomial.

Lema 1. *El algoritmo anterior devuelve X_T tal que $v(X_T) \geq v(OPT)\left(1 - \frac{1}{k}\right)$.*

Demostración.
- Si $|OPT| \leq k$, entonces el algoritmo encuentra OPT, luego la garantía es 1.
- Si $|OPT| \geq k + 1$:
  Sea T el conjunto de los k elementos más valiosos de OPT. Ordenemos ahora $OPT \setminus T = \{o_1, \ldots, o_{\ell-k}\}$ de mayor a menor densidad. Queremos comparar $v(OPT)$ con $v(X_T)$. Para ello, sea m el primer índice tal que o_m no está en X_T.

  Observaciones

  1. Como o_m no está en X_T. Todos los objetos de $X_T \setminus T$ tienen densidad al menos $v(o_m)/s(o_m)$.
  2. Sea ahora $B' = B - s(X_T)$ el espacio vacío que queda en la mochila al considerar el conjunto X_T. Recordemos que cuando glotón estaba incluyendo elementos en X_T, o_m no fue incluido. Esto quiere decir que algún elemento, digamos a^* de densidad mayor o igual a la de o_m (que puede estar en $A \setminus OPT$) no cupo en B'. En otras palabras

$$B' < s(a) = \frac{s(a)}{v(a)} v(a) \leq \frac{s(o_m)}{v(o_m)} v(a^*) \leq \frac{s(o_m)}{v(o_m)} \min_{a \in T} v(a) \leq \frac{s(o_m)}{v(o_m)} \frac{v(OPT)}{k}.$$

  La penúltima desigualdad sale del hecho que eliminamos de A todos los objetos de valor mayor a $\min_{a \in T} v(a)$.

  De las observaciones anteriores y usando que $X_T \setminus T \subseteq X_T \setminus OPT$ tenemos:

$$\begin{aligned} v(X_T \setminus OPT) &\geq \frac{v(o_m)}{s(o_m)} \cdot s(X_T \setminus OPT) \\ &= \frac{v(o_m)}{s(o_m)} \cdot (B - s(X_T \cap OPT) - B') \\ &\geq \frac{v(o_m)}{s(o_m)} \cdot (s(OPT \setminus X_T) - B') \\ &\geq \frac{v(o_m)}{s(o_m)} \left(s(OPT \setminus X_T) - \frac{s(o_m)}{v(o_m)} \frac{v(OPT)}{k} \right) \\ &\geq \frac{v(o_m)}{s(o_m)} \cdot s(OPT \setminus X_T) - \frac{v(OPT)}{k}. \end{aligned} \tag{1}$$

  Usando que todos los elementos de $OPT \setminus X_T$ tienen densidad a lo más $v(o_m)/s(o_m)$ concluimos que:

$$\begin{aligned} v(OPT) &= v(X_T \cap OPT) + v(OPT \setminus X_T) \\ &= v(X_T) - v(X_T \setminus OPT) + \sum_{i=m}^{\ell} \frac{v(o_i)}{s(o_i)} s(o_i) \\ &\overset{(1)}{\leq} v(X_T) - \left(s(OPT \setminus X_T) \frac{v(o_m)}{s(o_m)} - \frac{v(OPT)}{k} \right) + s(OPT \setminus X_H) \cdot \frac{v(o_m)}{s(o_m)} \\ &= v(X_T) + \frac{v(OPT)}{k}. \end{aligned}$$

Esto termina la demostración del lema.

□

Si tomamos ε cualquiera y elegimos $k = \lceil \frac{1}{\varepsilon} \rceil$, obtenemos una $\frac{1}{1-\varepsilon}$ aproximación.

Definición (PTAS)**.** Esquema de aproximaciíon polinomial para el problema de la mochila. Es un algoritmo que dado $\varepsilon > 0$ fijo, encuentra una $1 - \varepsilon$ (o $1 + \varepsilon$) aproximación en tiempo polinomial donde el polinomio puede depender de ε.

Observación. *En en la parte anterior el algoritmo es* $O(\varepsilon^{-1} n^{\varepsilon^{-1}} + 1)$.

Definición (FPTAS)**.** Esquema de aproximación totalmente polinomial. Algoritmo que dado $\varepsilon > 0$, encuentra una $1 + \varepsilon$ (o $1 - \varepsilon$) aproximación en tiempo polinomial en n y en $\frac{1}{\varepsilon}$.

Para Knapsack:
Si todos los tamanõs de los objetos son chicos (entre $\{1, ..., k\}$) se puede crear un Programa Dinámico para Mochila que corra en tiempo $O(n^2 k)$, o si no, redondear tamaños a un rango $\{1, ..., k\}$ con k dependiendo de ε, y luego usar el Programa Dinámico.