LEC016 Knapsack Problem

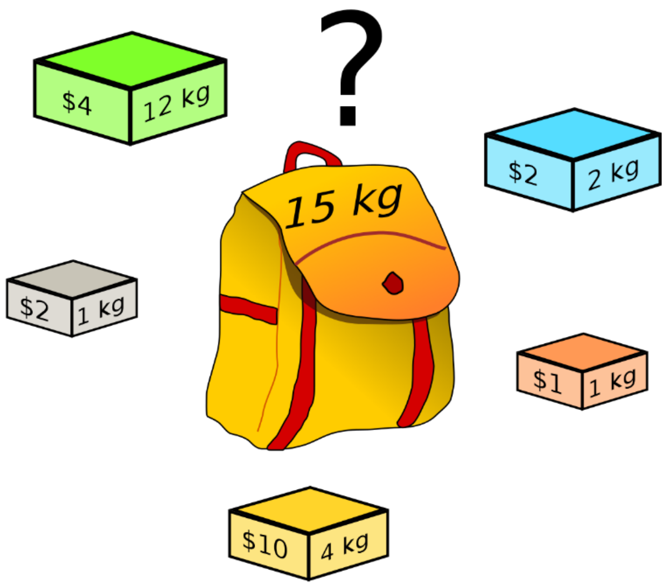
VG441 SS2020

Cong Shi
Industrial & Operations Engineering
University of Michigan

Knapsack

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes)

Objective: We wish to pick a maximium-value subset S from n items (without exceeding the capacity constraint).



MILP Formulation

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes)
- Find a max-value subset S from n items

Let x_i be a binary decision variable of whether to include the item i

$$
\begin{aligned}
\textbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\
\textbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \le B \\
& x_i \in \{0, 1\}, \forall i
\end{aligned}
$$

LP Relaxation (Fractional Knapsack)

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes)
- Find a max-value subset S from n items

Let x_i be a binary decision variable of whether to include the item i

$$\begin{aligned} \mathbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\ \mathbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \leq B \\ & x_i \in \{0, 1\}, \forall i \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \mathbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\ \mathbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \leq B \\ & 0 \leq x_i \leq 1, \forall i \end{aligned}$$

LP Relaxation: allowing for fractional allocation

LP Relaxation (Fractional Knapsack)

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes)
- Find a max-value subset S from n items

Let x_i be a fraction of item i to be included

$$\begin{array}{ll} \textbf{max} & \sum_{i=1}^{n} v_i x_i \\ \textbf{s.t.} & \sum_{i=1}^{n} s_i x_i \leq B \\ & 0 \leq x_i \leq 1, \forall i \end{array}$$

This is easy: greedy would suffice!

Re-index $\frac{v_1}{x_1} \geq \frac{v_2}{x_2} \geq \ldots \geq \frac{v_n}{x_n}$

The solution would be (1,1,...,1, α ,0,...,0,0) with $0 \leq \alpha < 1$

Back to (Integer Knapsack)

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes) Assume $B \geq s_i$ for each i
- Find a max-value subset S from n items

$$\begin{aligned} \textbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\ \textbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \leq B \\ & x_i \in \{0,1\}, \forall i \end{aligned}$$

Re-index $\frac{v_1}{x_1} \geq \frac{v_2}{x_2} \geq \ldots \geq \frac{v_n}{x_n}$

The solution would be (1,1,...,1, α ,0,...,0,0)

Trim

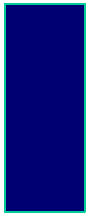
Greedy solution is (1,1,...,1,0,0,...,0,0)

However, this can be arbitrarily bad!

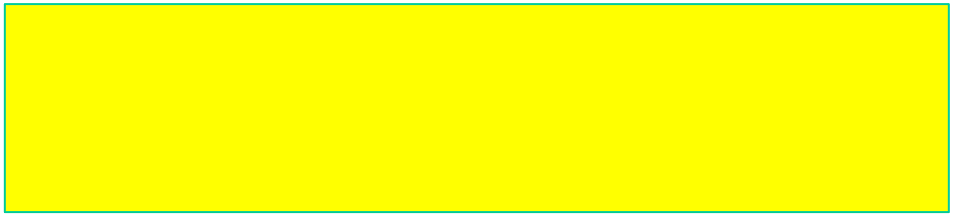
A Bad Example

- Greedy can be arbitrarily bad for 0-1 Knapsack

Capacity $B = 10000$



Item 1: $s_1 = 1, v_1 = 100$



Item 2: $s_2 = 10000, v_2 = 10000$

A 2-Approximation Algorithm

- n items $\{1, \ldots, n\}$
- Each item i has a value v_i and size s_i
- A capacity B (for sizes) Assume $B \geq s_i$ for each i
- Find a max-value subset S from n items

$$\begin{aligned} \textbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\ \textbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \leq B \\ & x_i \in \{0,1\}, \forall i \end{aligned}$$

Re-index $\frac{v_1}{x_1} \geq \frac{v_2}{x_2} \geq \ldots \geq \frac{v_n}{x_n}$

The solution would be (1,1,…,1, α ,0,…,0,0)

Choose the better of the two:

SOL1 = (1,1,…,1,0,0,…,0,0)

SOL2 = (0,0,…,0,1,0,…,0,0)

A 2-Approximation Algorithm

$$\begin{array}{ll} \mathbf{max} & \sum_{i=1}^{n} v_i x_i \\ \mathbf{s.t.} & \sum_{i=1}^{n} s_i x_i \leq B \\ & x_i \in \{0,1\}, \forall i \end{array} \leqslant \begin{array}{ll} \mathbf{max} & \sum_{i=1}^{n} v_i x_i \\ \mathbf{s.t.} & \sum_{i=1}^{n} s_i x_i \leq B \\ & 0 \leq x_i \leq 1, \forall i \end{array}$$

Re-index $\frac{v_1}{x_1} \geq \frac{v_2}{x_2} \geq \ldots \geq \frac{v_n}{x_n}$

The greedy solution would be (1,1,…,1, α ,0,…,0,0)

So $OPT(MILP) \leq OPT(LP) \leq v_1 + v_2 + \cdots + v_{k-1} + v_k$

Choose the better of the two:

SOL1 = (1,1,…,1,0,0,…,0,0)

SOL2 = (0,0,…,0,1,0,…,0,0)

So we are getting

$\max(v_1 + v_2 + \cdots + v_{k-1}, v_k) \geq 0.5OPT(LP) \geq 0.5OPT(MILP)$

Is there a better solution?

$$\begin{aligned} \textbf{max} \quad & \textstyle\sum_{i=1}^{n} v_i x_i \\ \textbf{s.t.} \quad & \textstyle\sum_{i=1}^{n} s_i x_i \le B \\ & x_i \in \{0,1\}, \forall i \end{aligned}$$

Our goal (FPTAS – fully polynomial time approximation scheme):

That is, for any $\epsilon > 0$, we can get a $(1-\epsilon)$ approximation in time polynomial in N and $\frac{1}{\epsilon}$ where $N = n \times \log\left(\max_{i\in[n]} \{v_i, s_i\}\right)$.

Look at a related problem (KS)

Consider a target value u, we find the min-size subset

$$\min_{S\subseteq[n]} \quad \sum_{i\in S} s_i$$
$$\text{s.t.} \quad \sum_{i\in S} v_i \geq u$$

Let $T[i,w]$ be the min-size of subset $S \subseteq \{1,\ldots,i\}$ such that $\sum_{k\in S} v_k = w$

Let $v_{\max} = \max_{i\in[n]} v_i$.

1. For $w = 0,\ldots,nv_{\max}$:

   Set $T[1,w] = \begin{cases} 0 & w=0 \\ s_1 & w=v_1 \\ \infty & \text{otherwise} \end{cases}$

2. For $i = 2,\ldots,n$:

   For $w = 0,\ldots,nv_{\max}$:

   $\to$ Set $T[i,w] = \min\{T[i-1,w], T[i-1,w-v_i]+s_i\}$

To find the optimal solution to 0-1 knapsack, it suffices to find

$$\arg\max_{w}\{T[n,w] : T[n,w] \leq B\}$$

Look at a related problem (KS)

Consider a target value u, we find the min-size subset

$$\textbf{min}_{S\subseteq[n]} \quad \textstyle\sum_{i\in S} s_i$$
$$\textbf{s.t.} \quad \textstyle\sum_{i\in S} v_i \geq u$$

Let $T[i,w]$ be the min-size of subset $S\subseteq\{1,\ldots,i\}$ such that $\sum_{k\in S} v_k = w$

Let $v_{\max} = \max_{i\in[n]} v_i$.

1. For $w = 0,\ldots,nv_{\max}$:

   Set $T[1,w] = \begin{cases} 0 & w = 0 \\ s_1 & w = v_1 \\ \infty & \text{otherwise} \end{cases}$

2. For $i = 2,\ldots,n$:

   For $w = 0,\ldots,nv_{\max}$:

   $\rightarrow$ Set $T[i,w] = \min\{T[i-1,w], T[i-1,w-v_i]+s_i\}$

Sizes:	$s_1 = 3, s_2 = 3, s_3 = 8, s_4 = 5$
Values:	$v_1 = 4, v_2 = 4, v_3 = 6, v_4 = 5$

Find $\arg\max_w\{T[n,w] : T[n,w] \leq B\}$

Polynomial time in n and $v_{\max}$ (but exponential in $\log v_{\max}$)!

FPTAS

We know how to solve it exactly:

$$\textbf{ExactKS}(s_1, \ldots, s_n, v_1, \ldots, v_n, B) \text{ requires } poly(n, \textstyle\sum v_i)$$

Goal is FPTA: Need an algorithm that runs in $poly(n, 1/\epsilon)$
but you can relax the solution to be within $(1-\epsilon)\mathbf{OPT}$

Algorithm: $(s_1, \ldots, s_n, v_1, \ldots, v_n, B, \epsilon)$

1 : $M \leftarrow \max_i v_i$

2 : $v_i' \leftarrow \left\lfloor \frac{v_i}{\epsilon M/n} \right\rfloor$

3 : $A \leftarrow \text{ExactKnapsack}\,(s_1, \ldots, s_n, v_1', \ldots, v_n', B)$

4 : return solution A

FPTAS

We know how to solve it exactly:

$$\textbf{ExactKS}(s_1, \ldots, s_n, v_1, \ldots, v_n, B) \text{ requires } poly(n, \sum v_i)$$

Goal is FPTA: Need an algorithm that runs in $poly(n, 1/\epsilon)$
but you can relax the solution to be within $(1-\epsilon)\mathbf{OPT}$

Algorithm: $(s_1, \ldots, s_n, v_1, \ldots, v_n, B, \epsilon)$

1 : $M \leftarrow \max_i v_i$

2 : $v_i' \leftarrow \left\lfloor \frac{v_i}{\epsilon M/n} \right\rfloor$

3 : $A \leftarrow \text{ExactKnapsack}\,(s_1, \ldots, s_n, v_1', \ldots, v_n', B)$

4 : return solution A

Running time analysis:

$$\sum_{i=1}^{n} v_i' = \sum_{i=1}^{n} \left\lfloor \frac{v_i}{\epsilon M/n} \right\rfloor \leq \sum_{i=1}^{n} \frac{n}{\epsilon} \leq \frac{n^2}{\epsilon}$$

So the running time of ExactKS is $\text{poly}\left(\frac{n^2}{\epsilon}\right) = \text{poly}\left(n, \frac{1}{\epsilon}\right)$

FPTAS

We know how to solve it exactly via DP:

$$\textbf{ExactKS}(s_1,\ldots,s_n,v_1,\ldots,v_n,B) \text{ requires } \mathit{poly}(n,\textstyle\sum v_i)$$

Goal is FPTA: Need an algorithm that runs in $\mathit{poly}(n,1/\epsilon)$
but you can relax the solution to be within $(1-\epsilon)\mathbf{OPT}$

Algorithm: $(s_1,\ldots,s_n,v_1,\ldots,v_n,B,\epsilon)$

1 : $M \leftarrow \max_i v_i$
2 : $v_i' \leftarrow \left\lfloor \frac{v_i}{\epsilon M/n} \right\rfloor$
3 : $A \leftarrow \mathrm{ExactKnapsack}\,(s_1,\ldots,s_n,v_1',\ldots,v_n',B)$
4 : return solution A

Optimality analysis: Shall show $\overline{OPT} \ge (1-\epsilon)\frac{n}{\epsilon M}\cdot OPT$. Then, we are done:

$$\sum_{i\in A} v_i \ge \frac{\epsilon M}{n}\sum_{i\in A} v_i' = \frac{\epsilon M}{n}\overline{OPT} \ge (1-\epsilon)OPT.$$

$$OPT = \sum_{i\in S} v_i = \sum_{i\in S} \frac{v_i}{\frac{\epsilon M}{n}}\frac{\epsilon M}{n} \le \sum_{i\in S}(v_i'+1)\frac{\epsilon M}{n} = \left(\sum_{i\in S} v_i'\right)\frac{\epsilon M}{n} + \epsilon M \le \frac{\epsilon M}{n}\overline{OPT} + \epsilon OPT$$

true original optimal set