# 组合优化理论

## 第7章 背包问题

主讲人：陈安龙

# 第7章 背包问题

§1 背包问题的描述

§2 背包问题的分支定界法

§3 背包问题的近似算法

§4 背包问题的一些相关问题

# 第7章 背包问题

背包问题(*Knapsack Problem*)是一个有着广泛应用的组合优化问题，它不仅在投资决策、装载、库存等方面有应用，而且常以子问题形式出现在大规模优化问题中，它的理论与算法具有一定的代表性.

## §1 背包问题的描述

背包问题的一般描述为：设有物品集 $U=\{u_1,u_2,\ldots,u_n\}$ 是一个准备放入容量为 $C\in Z^+$ 的背包中的 $n$ 项物品的集合.

如何选择 $U$ 中的一些物品装入背包，使这些物品的总重量不超过 $C$，且使总价值达到最大?

# §1 背包问题的描述

背包问题的数学模型：

$w_j \in Z$—— 重量

$p_j \in Z$—— 价值

$C \in Z^+$——容量

$$\max \quad z = \sum_{j=1}^{n} p_j x_j$$

$$(KP) \qquad s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$

$$x_j = 0 \ \ or \ \ 1 \qquad j \in N = \{1, 2, \ldots, n\}$$

因为决策变量 $x_j = 0 \ \ or \ \ 1$，所以也称 0–1 背包问题.

一般背包问题： $x_j \in Z^+ \cup \{0\} \qquad j \in N = \{1, 2, \ldots, n\}$

( *General Knapsack Problem* )

# 为讨论方便，总可假定（相当于标准化）：

1、 $w_j > 0, \quad p_j > 0 \qquad j = 1 \sim n;$

2、 $w_j \le C \qquad j = 1 \sim n;$

3、 $\sum_{j=1}^{n} w_j > C$

4、 $\frac{p_1}{w_1} \ge \frac{p_2}{w_2} \ge \cdots \ge \frac{p_n}{w_n}$ 即按价值密度从大到小排列.

# §1 背包问题的描述

对 4 只需 $O(n\ln n)$次运算即可;

对 3 若 $\sum_{j=1}^{n} w_j \le C$，最优解为 $x_1 = x_2 = \cdots = x_n = 1$;

对 2 若 $w_j > C$，则最优解中 $x_j = 0$，$\therefore$ 事先可去掉;

对 1 分三种情况讨论:

(1) 若 $p_j \le 0$ 且 $w_j \ge 0$ 令 $N^0 = \left\{ j \in N \middle| p_j \le 0,\ w_j \ge 0 \right\}$

此时，最优解中 $x_j = 0$ 所以，该物品事先可去掉;

(2) 若 $p_j \ge 0$ 且 $w_j < 0$ 令 $N^1 = \left\{ j \in N \middle| p_j \ge 0,\ w_j < 0 \right\}$

此时，最优解中 $x_j = 1$ 所以，该物品事先可去掉;

(3) 若 $p_j < 0$ 且 $w_j < 0$ 令 $N^- = \{ j \in N \mid p_j < 0, \ w_j < 0 \}$

只需在模型中，令 $x_j = 1 - y_j$，则系数即为大于零了.

综上，对不满足（1）、（2）、（3）的假定，可作如下处理，使之满足：

对 $\forall \ j \in N^-$ 令 $x_j = 1 - y_j \quad \overline{p}_j = -p_j \quad \overline{w}_j = -w_j$

对 $\forall \ j \in N^+ = N \setminus (N^0 \cup N^1 \cup N^-)$ 令

$x_j = y_j \quad \overline{p}_j = p_j \quad \overline{w}_j = w_j$

**则原问题化为：**

$$\max \ z = \sum_{j \in N^- \cup N^+} \overline{p}_j y_j + \sum_{j \in N^- \cup N^1} p_j$$

$$s.t. \quad \sum_{j \in N^- \cup N^+} \overline{w}_j y_j \le C - \sum_{j \in N^- \cup N^1} w_j$$

$$y_j = 0 \ \ or \ \ 1, \qquad j \in N^- \cup N^+$$

# §1 背包问题的描述

如果在（*KP*）中，令 $x_j = 1 - y_j$

$$\max \quad z = \sum_{j=1}^{n} p_j - \sum_{j=1}^{n} p_j y_j$$

$$s.t. \quad \sum_{j=1}^{n} w_j - \sum_{j=1}^{n} w_j y_j \le C$$

$$y_j = 0 \quad or \quad 1 \qquad j \in N = \{1, 2, \dots, n\}$$

令 $$q = \sum_{j=1}^{n} w_j - C$$

**该问题的实际意义是求不放在包中的物品的价值和最小**

$$(KP) \quad \max \quad z = \sum_{j=1}^{n} p_j x_j$$

$$s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$

$$x_j = 0 \quad or \quad 1$$

$$j \in N = \{1, 2, \dots, n\}$$

$$\min \quad u = \sum_{j=1}^{n} p_j y_j$$

$$s.t. \quad \sum_{j=1}^{n} w_j y_j \ge q$$

$$y_j = 0 \quad or \quad 1 \qquad j \in N = \{1, 2, \dots, n\}$$

# §2 背包问题的分支定界法

分支定界法 ( *Branch and Bound Method* ) 的基本思想在课程中已介绍，它的重要在于它提出了一类新的思路（隐枚举法），使得许多原来不好解决的问题有了解决的可能性。（具有普适性）

## △ 确定问题（子问题）的最优值的上（下）界

通常是通过求解松弛问题，用松弛问题的解作为界

***Note*：** 松弛问题选择的原则

1、松弛问题要与原问题的最优值尽量接近；

2、松弛问题要尽量容易解 .

**这两个原则不易统一，所以可选择不同的松弛问题**

# §2 背包问题的分支定界法

## △ 划分方法的选择

原则是希望分出来的子问题容易被查清，可加快计算.

## △ 选哪个活问题先检查

1、先检查最大上界（极大化问题）的活问题

优点：检查子问题较其他规则为少；

缺点：计算机储存量较大 .

2、先检查最新产生的最大上界的活问题

优点：计算机储存量较少 ；

缺点：需要更多的分支运算 .

**选择的不同，提供了发挥的余地**

考虑 *KP* 的松弛问题： $x_j \in \{0,1\} \Rightarrow x_j \in [0,1]$

$$\max \quad z = \sum_{j=1}^{n} p_j x_j$$

$$C(KP) \quad s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$

$$0 \le x_j \le 1 \quad j \in N = \{1, 2, \ldots, n\}$$

思路：将物品按价值密度从大到小的顺序放入包内，……

记 $s = \min\left\{ j \;\middle|\; \sum_{i=1}^{j} w_i > C \right\}$ —— 关键项

第一个放不下的物品的序号

# §2 背包问题的分支定界法

*Theorem* 7.1 $C(KP)$ 最优解为

$$\bar{x}_j = 1 \quad j = 1 \sim s-1 \qquad \bar{x}_j = 0 \quad j = s+1 \sim n$$

$$\bar{x}_s = \frac{\bar{C}}{w_s}$$

其中 $\bar{C} = C - \sum_{j=1}^{s-1} w_j$

最优解值为: $z_{opt}(C(KP)) = \sum_{j=1}^{s-1} p_j + \bar{C}\frac{p_s}{w_s}$

显然, $z_{opt}(KP) \le z_{opt}(C(KP))$ ， 由于 $p_j$ 的整数性,

得到 $z(KP)$ 的一个上界:

$$U_1 = \lfloor z_{opt}(C(KP)) \rfloor = \sum_{j=1}^{s-1} p_j + \left\lfloor \bar{C}\frac{p_s}{w_s} \right\rfloor$$

$\lfloor \alpha \rfloor$ 表示不超过 $\alpha$ 的最大整数.

**Theorem 7.1** **Proof：** 显然 $C(KP)$ 的最优解必满足 $\sum_{j=1}^{n} w_j x_j = C$

设 $x^*$ 是其最优解，要证 $x^* = \bar{x}$ 若存在 $k < s$ 使 $x_k^* < 1$

则至少存在 $q \geq s$ 使 $x_q^* > \bar{x}_q$．取充分小的 $\varepsilon > 0$

(满足：1、$x_k^* + \varepsilon \leq 1$ 2、$x_q^* - \varepsilon \frac{w_k}{w_q} \geq 0$ )

$x_k^* + \varepsilon$ 则重量增加 $\varepsilon w_k$，$x_q^* - \varepsilon \frac{w_k}{w_q}$ 则重量减少 $\varepsilon w_k$

将 $x_k^*$ 增加 $\varepsilon$，$x_q^*$ 减少 $\varepsilon \frac{w_k}{w_q}$，此时, 仍是一个可行解,

且目标函数值增加 $\varepsilon(p_k - p_q \frac{w_k}{w_q}) > 0$，矛盾．

$\therefore$ 对$\forall$ $k < s$，$x_k^* = \bar{x}_k = 1$ 同理可证 对$\forall$ $k > s$ $x_k^* = 0$

又由极大性知：$x_s^* = \frac{\bar{C}}{w_s} = \bar{x}_s$ 因此，$\bar{x}$ 是最优解.

易证 $z_{opt}(C(KP))$ 为最优解值

# §2 背包问题的分支定界法

**Theorem 7.2** 设 $U^0 = \sum_{j=1}^{s-1} p_j + \left\lfloor \bar{C}\frac{p_{s+1}}{w_{s+1}} \right\rfloor$

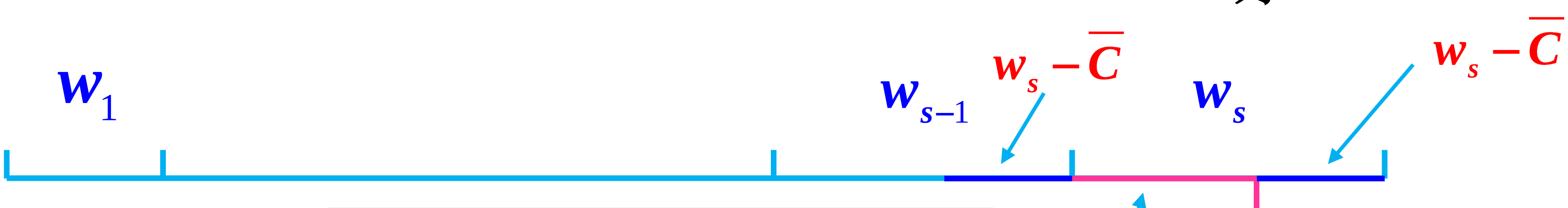
$$U^1 = \sum_{j=1}^{s-1} p_j + \left\lfloor p_s - (w_s - \bar{C})\frac{p_{s-1}}{w_{s-1}} \right\rfloor$$

其中 $s$ 与 $\bar{C}$ 定义同前. 则

1、 $z(KP)$ 的一个上界为 $U_2 = \max(U^0, U^1)$；

2、对背包问题任一实例, $U_2 \le U_1$

作为上界$U_2$比$U_1$更好




$$\sum_{j=1}^{s-1} p_j + p_s - (w_s - \bar{C})\frac{p_{s-1}}{w_{s-1}}$$

$$\bar{C} = C - \sum_{j=1}^{s-1} w_j$$

***Proof*：** 1、因为 ***KP*** 中 $x_s$ 不能取分数， 所以，***KP*** 的最优解一定是 $\overline{x}_s = 0$ ***or*** $\overline{x}_s = 1$情形之一．

当 $\overline{x}_s = 0$， 由 ***Th*** **7.1** 可知，$U^0$是此情形的上界；

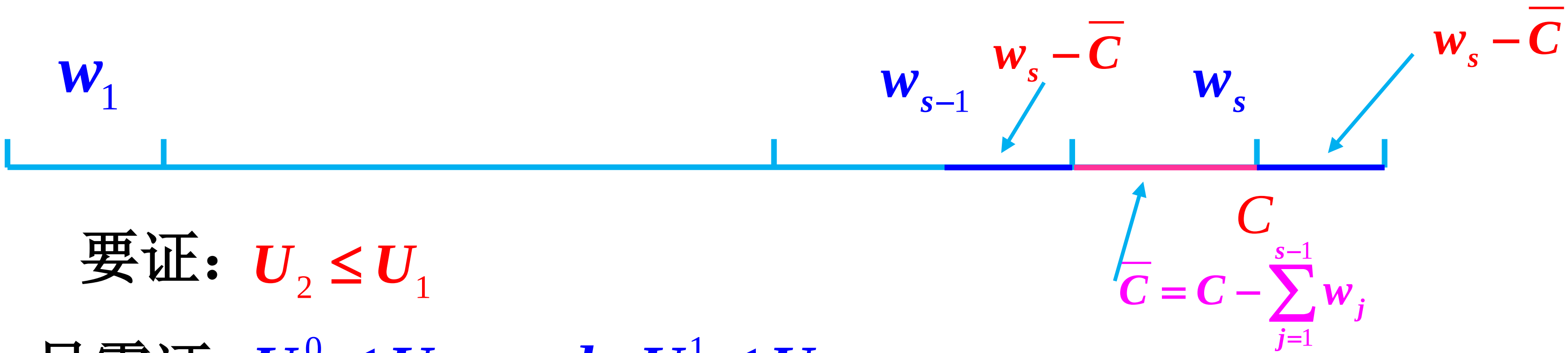
当 $\overline{x}_s = 1$， 这时，若 $\overline{x}_1 = \overline{x}_2 = \cdots = \overline{x}_{s-1} = 1$

重量超出 $w_s - \overline{C}$ ，而此时价值密度值最小的是 $\dfrac{p_{s-1}}{w_{s-1}}$ ．

$\therefore$ $U^1$是此情形的上界 ．

从而 $U_2 = \max\left(U^0, U^1\right)$是 $z(KP)$ 的上界．

# §2 背包问题的分支定界法




要证：$U_2 \le U_1$

只需证：$U^0 \le U_1 \quad and \quad U^1 \le U_1$

$$U_1 = \sum_{j=1}^{s-1} p_j + \left\lfloor \overline{C} \frac{p_s}{w_s} \right\rfloor$$

$$U^0 = \sum_{j=1}^{s-1} p_j + \left\lfloor \overline{C} \frac{p_{s+1}}{w_{s+1}} \right\rfloor$$

$$\because \quad \overline{C} \frac{p_{s+1}}{w_{s+1}} \le \overline{C} \frac{p_s}{w_s}$$

# § 2 背包问题的分支定界法

2、 $U^0 \le U_1$ 是显然的；

$C(KP)$ 的最优解值 $z_{opt}(C(KP)) = \sum_{j=1}^{s-1} p_j + \overline{C}\frac{p_s}{w_s}$

$$= \sum_{j=1}^{s} p_j - (w_s - \overline{C})\frac{p_s}{w_s} \qquad (*)$$

$C(KP)$ 当 $\overline{x}_s = 1$ 的最优解值

$$\sum_{j=1}^{s} p_j - (w_s - \overline{C})\frac{p_{s-1}}{w_{s-1}} \qquad (**)$$

$\because\ w_s > \overline{C},\ \frac{p_{s-1}}{w_{s-1}} \ge \frac{p_s}{w_s} \quad \therefore \sum_{j=1}^{s} p_j - (w_s - \overline{C})\frac{p_s}{w_s} \ge \sum_{j=1}^{s} p_j - (w_s - \overline{C})\frac{p_{s-1}}{w_{s-1}}$

$\therefore\ U^1 \le U_1$ 从而 $U_2 \le U_1$.

一般给出的上界越小，计算量越大，但越容易被剪枝 .

## Example 1　用分支定界法求如下 *KP*：

$$n=7 \quad (p_j)=(70,20,39,37,7,5,10)$$

$$(w_j)=(31,10,20,19,4,3,6) \quad C=50$$

$$u=70+20+\left\lfloor\frac{39\times 9}{20}\right\rfloor=107$$

**Solution：**　可验证

$$\frac{70}{31}\ge\frac{20}{10}\ge\frac{39}{20}\ge\cdots\ge\frac{10}{6}$$

$\widetilde{K}_0$　$\bar{x}_0=\left(1,1,{}^{9}\!/_{20},0,0,0,0\right)$

$u=107$

$x_3=0$　　$x_3=1$

$\widetilde{K}_1$　$\bar{x}_1=\left(1,1,0,{}^{9}\!/_{19},0,0,0\right)$

$u=107$

$\widetilde{K}_2$　$\bar{x}_2=\left({}^{30}\!/_{31},0,1,0,0,0,0\right)$

$u=106$　*

$x_4=0$　　$x_4=1$

$\widetilde{K}_3$　$\bar{x}_3=\left(1,1,0,0,1,1,{}^{1}\!/_{3}\right)$

$u=105$　*

$\widetilde{K}_4$　$\bar{x}_4=(1,0,0,1,0,0,0)$

$u=107$　*

$$\therefore \quad x_{opt}=(1,0,0,1,0,0,0) \qquad z_{\max}=107$$

# §3 背包问题的近似算法

有改进的吗？

通过前面介绍的 $C(KP)$，自然想到如下贪婪算法 (*Greedy Algorithm*): $x_1 = \cdots = x_{s-1} = 1,\quad x_s = \cdots = x_n = 0$

其目标函数值为 $\sum_{j=1}^{s-1} p_j$ .

$GA_0$是近似算法吗？

$GA_0$   *step* 1   $x_0 = w_0 = 0 \qquad k = 1$

*step* 2   若 $\sum_{j=0}^{k-1} w_j x_j + w_k \le C$，则 $x_k = 1$，

否则 $x_k = 0$；

*step* 3   若 $k = n$，则结束;

否则 $k := k+1$，转 *step* 2 .

构造例子 I：$n=2,\ \ p_1=1+\varepsilon,\ \ w_1=1,\ \ p_2=k,\ \ w_2=k,\ \ C=k.$

$\varepsilon$ 为充分小的正数 .

按上述算法，得： $x_1=1,\ \ x_2=0,\ \ z_{GA_0}(I)=1+\varepsilon;$

而显然最优解为： $x_1=0,\ \ x_2=1,\ \ z_{opt}(I)=k.$

$$\frac{z_{GA_0}(I)}{z_{opt}(I)}=\frac{1+\varepsilon}{k}\to 0 \qquad (k\to+\infty)$$

说明 $GA_0$ 的绝对性能比不会大于任意给定的正数，所以，它不能作为近似解 . 但稍加改进，就可得到一个绝对性能比为常数的较好的近似算法 .

# §3 背包问题的近似算法

$GA_1$ $step\ 1$ 求解 $C(KP)$，得关键项记为 $s$；

$step\ 2$ 取 $\max\left\{\sum_{j=1}^{s-1} p_j, p_s\right\}$ 作为近似解值.

即若 $\sum_{j=1}^{s-1} p_j > p_s$，则 $x_1 = \cdots = x_{s-1} = 1, x_s = \cdots = x_n = 0$

否则 $x_1 = \cdots = x_{s-1} = x_{s+1} = \cdots = x_n = 0, \quad x_s = 1$

***Example 2*** 用 $GA_1$ 求如下 $KP$：

$$n = 4 \quad (p_j) = (3, 7, 17, 20), (w_j) = (1, 3, 8, 10) \quad C = 11$$

***Solution***： 易验证 $\frac{3}{1} \geq \frac{7}{3} \geq \frac{17}{8} \geq \frac{20}{10}$

有改进的吗？

显然，物品3 为关键项（即 $s = 3$） $\because$ $p_1 + p_2 = 3 + 7 = 10$,

而 $p_3 = 17$ 近似解为 $x_1 = x_2 = x_4 = 0, x_3 = 1$; $z_{GA_1} = 17$

$GA_2$ step 1 求解 $C(KP)$，得关键项记为 $s$；

step 2 令 $p_i = \max_j \{p_j\}$

若 $\sum_{j=1}^{s-1} p_j > p_i$ 则 $x_1 = \cdots = x_{s-1} = 1, x_s = \cdots = x_n = 0$

否则 $x_1 = \cdots = x_{i-1} = x_{i+1} = \cdots = x_n = 0, \quad x_i = 1$

*Example* 3 用 $GA_2$ 求如下 $KP$：

$$n = 4 \quad (p_j) = (3,7,17,20), (w_j) = (1,3,8,10) \quad C = 11$$

*Solution*： $\because \quad p_1 + p_2 = 3 + 7 = 10,$

而 $\max_j \{p_j\} = p_4 = 20$ 近似解为

$$x_1 = x_2 = x_3 = 0, x_4 = 1; \quad z_{GA_2} = 20$$

# § 3 背包问题的近似算法

*Theorem* 7.3 $R_{GA_1}=\frac{1}{2}$ · $R_{GA_1}=\sup_I\left\{r\le 1 \middle| \frac{z_{GA_1}(I)}{z_{opt}(I)}\ge r\right\}$

*Proof*：由 *s* 的定义知：对于任意实例 I

$z_{opt}(I)\le\sum_{j=1}^{s-1}p_j+p_s$ 又 $z_{GA_1}(I)=\max\left\{\sum_{j=1}^{s-1}p_j, p_s\right\}\ge\frac{1}{2}(\sum_{j=1}^{s-1}p_j+p_s)$

因此，$\frac{z_{GA_1}(I)}{z_{opt}(I)}\ge\frac{1}{2}$ 从而 $R_{GA_1}\ge\frac{1}{2}$

取实例 I：$n=3, p_1=1+\varepsilon, w_1=1,$ $\varepsilon$为充分小的正数．

$p_2=p_3=w_2=w_3=k,\ C=2k$

$z_{GA_1}(I)=1+\varepsilon+k,\quad x_1=x_2=1,\quad x_3=0$

$z_{opt}(I)=2k,\quad x_1=0,\quad x_2=x_3=1$ $\frac{z_{GA_1}(I)}{z_{opt}(I)}=\frac{1+\varepsilon+k}{2k}\to\frac{1}{2}\quad(k\to\infty)$

则 $R_{GA_1}\le\frac{1}{2}$ 从而 $R_{GA_1}=\frac{1}{2}$

# §3 背包问题的近似算法

**Theorem 7.4** $R_{GA_2} = \frac{1}{2}$ . 证明与 $Th$ 7.3 的类似 .

对 $Ex.3$

$$n = 4 \quad (p_j) = (3,7,17,20), (w_j) = (1,3,8,10) \quad C = 11$$

考虑对 $GA_2$ 进行修改，进一步可取

$x_1 = x_4 = 1, \quad x_2 = x_3 = 0$ 则 $z = 23$

这在实际计算中是会有好处的 . 但绝对性能比不会改进 .

# §3　背包问题的近似算法

已知 $GA_0$ 对解 0-1 背包问题效果很差，但在一般背包问题中却是可以的．

设
$$\max \quad z=\sum_{j=1}^{n} p_j x_j$$
$$s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$
$$x_j \in z^+ \cup \{0\},\ j=1\sim n$$

显然，$x_1=\left\lfloor \frac{C}{w_1} \right\rfloor$，$x_2=\cdots=x_n=0$

是一个可行解．$\therefore \quad z_{GA_0} \ge p_1 \left\lfloor \frac{C}{w_1} \right\rfloor$

松弛问题的最优解值

通过求解 $C(KP)$ 得 $z_{opt}(I) \le p_1 \frac{C}{w_1}$

$I: p_1 = k+3\varepsilon,\ p_2 = k$

$w_1=\varepsilon+k/2,\ w_2=k/2$

$\varepsilon>0, \quad C=k$

$$\because \quad \frac{z_{GA_0}(I)}{z_{opt}(I)} \ge \frac{\left\lfloor C/w_1 \right\rfloor}{C/w_1} \ge \frac{\left\lfloor C/w_1 \right\rfloor}{1+\left\lfloor C/w_1 \right\rfloor} \ge \frac{1}{2}$$

$$\therefore \quad R_{GA_0} \ge \frac{1}{2}$$

进一步可证 $R_{GA_0}=\frac{1}{2}$

1975年 *Sahni* 给出一个多项式时间近似算法．

算法$S_k$：

*step* 1 对任意满足 $M \subseteq N = \{1,2,\cdots,n\}, |M| \le k$

且 $\sum_{i \in M} w_i \le C$ 的子集 $M$，先将 $M$ 中的物品放入包内,

然后用算法 $GA_1$ 或 $GA_2$ 求解一个如下定义的 $KP$：

物品集为 $N \backslash M$，包容量为 $C - \sum_{i \in M} w_i$，将得到的解与

$M$ 的并作为原问题的近似解．

*step* 2 取上述所有不同解中最好一个作为输出．

*Theorem* 7.5 $R_{S_k} = 1 + \dfrac{1}{k}$

# §4 0-1背包问题的一些相关问题

## 一、有界背包问题 (*Bounded Knapsack Problem*)

0-1 背包问题： $x_j = 0 \ \ or \ \ 1$ ;

一般背包问题： $x_j \in z^+ \cup \{0\}$ ; （无界背包问题）

有界背包问题： $0 \le x_j \le b_j$ $b_j$ 为给定的正整数 .

显然，*GKP* 可化为 *BKP*，只需令 $b_j = \left\lfloor \frac{C}{w_j} \right\rfloor$

*BKP* 可化为等价的 0-1 *KP*

思想：任一整数可用 0 , 1 变量来表示，如 $x < 16$ 非负整数

$x = x_1 + 2x_2 + 2^2 x_3 + 2^3 x_4$ 如 13 $x_1 = 1, x_2 = 0, x_3 = 1, x_4 = 1$

# §4 0-1背包问题的一些相关问题

与前面讨论一样，总可假定 $p_j, w_j, b_j, C$ 都是正整数;

$$\sum_{j=1}^{n} b_j w_j > C,\quad b_j w_j \le C,\quad \frac{p_1}{w_1} \ge \frac{p_2}{w_2} \ge \cdots \ge \frac{p_n}{w_n}.$$

***Theorem* 7.7** 记 $s = \min\left\{ j \,\middle|\, \sum_{i=1}^{j} b_i w_i > C \right\}$, $\overline{C} = C - \sum_{j}^{s-1} b_j w_j$,

则 （1）$U_1 = \sum_{j=1}^{s-1} b_j p_j + \left\lfloor \overline{C} \frac{p_s}{w_s} \right\rfloor$ 是 *BKP* 的一个上界；

（2）取 $\overline{x}_j = b_j (j = 1, 2, \cdots, s-1)$ $\overline{x}_j = 0 (j = s, \cdots, n)$ 得到一个可行解，将此解的值与 $b_s p_s$ 的值比较，取大者为输出．

该算法的绝对性能比 $R = \frac{1}{2}$，计算复杂性为 $O(n \log n)$.

# 二、子集和问题

在组合优化问题中，很多问题是相通的，参数稍作修正，可能就是另一个问题．

在 0-1 背包问题中，令 $p_j = w_j$（这时 $\frac{p_j}{w_j} = 1, j = 1 \sim n$）

即

$$\max \quad z = \sum_{j=1}^{n} w_j x_j$$

称为子集和问题

$$(SSP) \quad s.t. \quad \sum_{j=1}^{n} w_j x_j \le C$$

***S**ubset **S**um **P**roblem*

$$x_j = 0 \quad or \quad 1 \qquad j \in N = \{1, 2, \ldots, n\}$$

***SSP*** 是 **0-1*KP*** 的特殊情形，所以原方法皆可用. 但因其特殊，它应有更简单的方法．

# §4 0-1背包问题的一些相关问题

1、贪婪算法(*GA*):

按任意顺序将物品逐个放入包内，直到放不下为止，然后将其结果与最大重量的物品比较，取优者为输出．

可证 $R_{GA}=\dfrac{1}{2}$ 计算复杂性为 $O(n)$．

2、近似算法 *MTGS*：

将上述的贪婪算法分别用于物品集 $\{1,2,\cdots,n\}$，$\{2,3,\cdots,n\}$，$\{3,\cdots,n\}$，$\cdots$，$\{n-1,n\}$，$\{n\}$，取最好者为输出．

这里假定 $w_1\ge w_2\ge\cdots\ge w_n$．

可证 $R_{MTGS}=\dfrac{3}{4}$ 计算复杂性为 $O(nlogn+n^2)=O(n^2)$．

# 本章结束