背包之01背包、完全背包、多重背包详解

—— Tanky Woo(2010.07.31)

首先说下动态规划，动态规划这东西就和递归一样，只能找局部关系，若想全部列出来，是很难的，比如汉诺塔。你可以说先把除最后一层的其他所有层都移动到2，再把最后一层移动到3，最后再把其余的从2移动到3，这是一个直观的关系，但是想列举出来是很难的，也许当层数n=3时还可以模拟下，再大一些就不可能了，所以，诸如递归，动态规划之类的，不能细想，只能找局部关系。

图1.汉诺塔图片

（引至杭电课件:DP最关键的就是状态，在DP时用到的数组时，也就是存储的每个状态的最优值，也就是记忆化搜索）

要了解背包，首先得清楚动态规划：

动态规划算法可分解成从先到后的4个步骤：

> 1. 描述一个最优解的结构；
> 2. 递归地定义最优解的值；
> 3. 以“自底向上”的方式计算最优解的值；
> 4. 从已计算的信息中构建出最优解的路径。

其中步骤1~3是动态规划求解问题的基础。如果题目只要求最优解的值，则步骤4可以省略。

背包的基本模型就是给你一个容量为V的背包 在一定的限制条件下放进最多(最少?)价值的东西

> 当前状态→ 以前状态

看了dd大牛的《背包九讲》，迷糊中带着一丝清醒，这里我也总结下01背包，完全背包，多重背包这三者的使用和区别，部分会引用dd大牛的《背包九讲》，如果有错，欢迎指出。 (www.wutianqi.com留言即可)

首先我们把三种情况放在一起来看：

> 01背包（ZeroOnePack）: 有N件物品和一个容量为V的背包， 每种物品均**只有一件。**第i件物品的费用是c[i]，价值是w[i]。求解将哪些物品装入背包可使价值总和最大。
> 完全背包(CompletePack): 有N种物品和一个容量为V的背包，每种物品都有**无限件**可用。第i种物品的费用是c[i]，价值是w[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量，且价值总和最大。
> 多重背包(MultiplePack): 有N种物品和一个容量为V的背包，第i种物品**最多有n[i]件**可用。每件费用是c[i]，价值是w[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量，且价值总和最大。

比较三个题目，会发现不同点在于每种背包的数量，01背包是每种只有一件，完全背包是每种无限件，而多重背包是每种有限件。

先来分析**01背包**：

01背包（ZeroOnePack）: 有N件物品和一个容量为V的背包，每种物品均只有一件。第i件物品的费用是c[i]，价值

是w[i]。求解将哪些物品装入背包可使价值总和最大。
这是最基础的背包问题，特点是：每种物品仅有一件，可以选择放或不放。
用子问题定义状态：即f[i][v]表示前i件物品恰放入一个容量为v的背包可以获得的最大价值。则其状态转移方程便是：

$$f[i][v]=\max\{f[i-1][v],f[i-1][v-c[i]]+w[i]\}$$

把这个过程理解下：
在前i件物品放进容量v的背包时，它有两种情况

> 情况一: 第i件不放进去，这时所得价值为:f[i-1][v]
> 情况二: 第i件放进去，这时所得价值为：f[i-1][v-c[i]]+w[i]

（第二种是什么意思？就是如果第i件放进去，那么在容量v-c[i]里就要放进前i-1件物品）
最后比较第一种与第二种所得价值的大小，哪种相对大，f[i][v]的值就是哪种。（这里是重点，理解！）

这里是用二维数组存储的，可以把空间优化，用一维数组存储。
用f[0..v]表示，f[v]表示把前i件物品放入容量为v的背包里得到的价值。把i从1~n(n件)循环后，最后f[v]表示所求最大值。

这里f[v]就相当于二维数组的f[i][v]。那么，如何得到f[i-1][v]和f[i-1][v-c[i]]+w[i]？（重点！思考）
首先要知道，我们是通过i从1到n的循环来依次表示前i件物品存入的状态。
即：for i=1..N
现在思考如何能在是f[v]表示当前状态是容量为v的背包所得价值，而又使f[v]和f[v-c[i]]+w[i]标签前一状态的价值？
逆序
这就是关键！

```
for i=1..N
   for v=V..0
        f[v]=max{f[v],f[v-c[i]]+w[i]};
```

分析上面的代码：当内循环是逆序时，就可以保证后一个f[v]和f[v-c[i]]+w[i]是前一状态的！这里给大家一组测试数据：　测试数据：10,3 3,4 4,5 5,6

最大容量M	物品个数N			j=0-m										
10	3		C	0	1	2	3	4	5	6	7	8	9	10
物品大小w	物品价值p	编号	0	0	0	0	0	0	0	0	0	0	0	0
3	4	i= 1	1	0	0	0	4	4	4	4	4	4	4	4
4	5	1-n 2	2	0	0	0	4	5	5	5	9	9	9	9
5	6	3	3	0	0	0	4	5	6	6	9	10	11	11

图2: 01背包图(1)

这个图表画得很好，借此来分析：
C[v]从物品i=1开始，循环到物品3，期间，每次逆序得到容量v在前i件物品时可以得到的最大值。
（请在草稿纸上自己画一画）

这里以一道题目来具体看看：
题目：http://acm.hdu.edu.cn/showproblem.php?pid=2602
代码：http://www.wutianqi.com/?p=533
分析：

物品	0	1	2	3	4	5	6	7	8	9	10
0	0	0	0	0	0	0	0	0	0	0	0
(1,2)	0	2	2	2	2	2	2	2	2	2	2
(3,4)	0	2	2	4	6	6	6	6	6	6	6
(8,10)	0	2	2	4	6	6	6	6	10	12	12
(2,5)	0	2	5	7	7	9	11	11	11	11	15
(4,7)	0	2	5	7	7	9	12	14	14	14	18

图2: 01背包图(2)

具体根据上面的解释以及我给出的代码分析。这题很基础，看懂上面的知识应该就会做了。

完全背包：

完全背包(CompletePack): 有N种物品和一个容量为V的背包，每种物品都有无限件可用。第i种物品的费用是c[i]，价值是w[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量，且价值总和最大。 完全背包按其思路仍然可以用一个二维数组来写出：

$$f[i][v]=\max\{f[i-1][v-k*c[i]]+k*w[i] \mid 0<=k*c[i]<=v\}$$

同样可以转换成一维数组来表示:

伪代码如下:

```
for i=1..N
    for v=0..V
        f[v]=max{f[v],f[v-c[i]]+w[i]}
```

顺序

想必大家看出了和01背包的区别，这里的内循环是顺序的，而01背包是逆序的。

现在关键的是考虑：为何完全背包可以这么写？

> 在次我们先来回忆下，01背包逆序的原因？是为了是max中的两项是前一状态值，这就对了。 那么这里，我们顺序写，这里的max中的两项当然就是当前状态的值了，为何？ 因为每种背包都是无限的。当我们把i从1到N循环时，f[v]表示容量为v在前i种背包时所得的价值，这里我们要添加的不是前一个背包，而是当前背包。所以我们要考虑的当然是当前状态。

这里同样给大家一道题目：
题目：http://acm.hdu.edu.cn/showproblem.php?pid=1114
代码：http://www.wutianqi.com/?p=535

多重背包

多重背包(MultiplePack): 有N种物品和一个容量为V的背包。第i种物品最多有n[i]件可用，每件费用是c[i]，价值是w[i]。求解将哪些物品装入背包可使这些物品的费用总和不超过背包容量，且价值总和最大。

这题目和完全背包问题很类似。基本的方程只需将完全背包问题的方程略微一改即可，因为对于第i种物品有n[i]+1种策略：取0件，取1件……取n[i]件。令f[i][v]表示前i种物品恰放入一个容量为v的背包的最大权值，则有状态转移方程：

$$f[i][v]=\max\{f[i-1][v-k*c[i]]+k*w[i] \mid 0<=k<=n[i]\}$$

这里同样转换为01背包：
普通的转换对于数量较多时，则可能会超时，可以转换成二进制（暂时不了解，所以先不讲） 对于普通的。就是多了一个中间的循环，把j=0~bag[i]，表示把第i中背包从取0件枚举到取bag[i]件。

给出一个例题：
题目：http://acm.hdu.edu.cn/showproblem.php?pid=2191
代码：http://www.wutianqi.com/?p=537

因为限于个人的能力，我只能讲出个大概，请大家具体还是好好看看dd大牛的《背包九讲》。
如果大家有问题或者资料里的内容有错误，可以留言给出，博客:http://www.wutianqi.com/

老版下载地址：
Word: http://download.csdn.net/source/2587577

最后更新：2012.02.22
这次更新变动较大，显示结构更为工整清晰，如果大家转载过我以前的版本，可以重新换成这个版本。