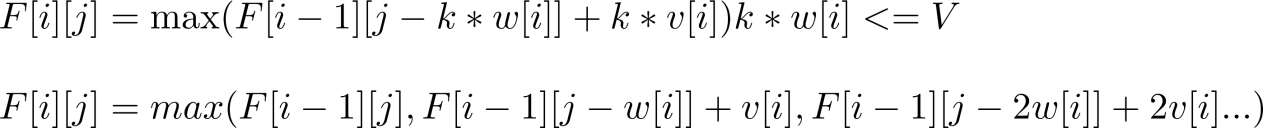
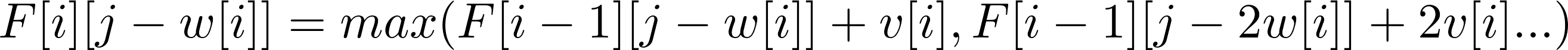
在01背包问题中，j是按照从V...0的顺序来逆序循环的，之所以这么做是为了保证在第i次循环中的是由状态递推而来。这也是为了保证每件物品只选一次：保证在考虑“选入第i件物品”这个策略时，依据的是一个从来没有选入第i件物品的子结果。

而完全背包问题是每件物品可以选无限件，所以在考虑“选入第i件物品”这个策略时，需要一个可能选入第i件物品的子结果，所以j的顺序必须采用从0...V的正序。

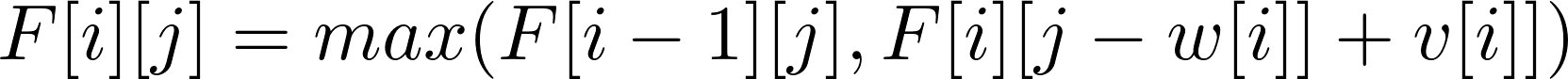
在《基本思路中》，状态转移方程为：



对于F[i][j-w[i]]：



则由上述两式可得：



在对主件k的“附件集合”进行了一次01背包后，可以得到费用依次为0...V-Wk(不选第k个主件，只考虑附件集合)，他们对应的最大价值为Fk[0...V-Wk]。

那么这个第k个主件和他的附件集合就相当于一个拥有V-Wk+1个物品的物品组，其中费用为w的物品对应的价值为Fk[w-Wk]+Vk，w的取值范围为[Wk，V]

在背包容量为V的背包问题中，泛化物品是一个定义域为[0, V]中的整数的函数h，当分配给它的费用为w时，能得到的价值时h(w)。

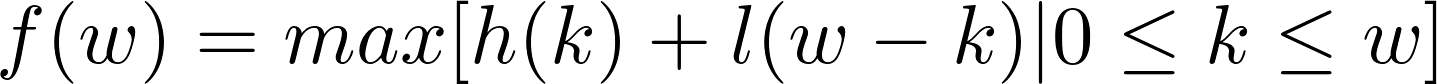
对于一个费用为w，价值为v的物品，

如果是01背包中的物品，当把它看成泛化物品时，它就是除了h(w)=v外，其他都是0的函数

如果是完全背包中的物品，h就是仅当x被w整除时有h(x)=，其他都是0的函数

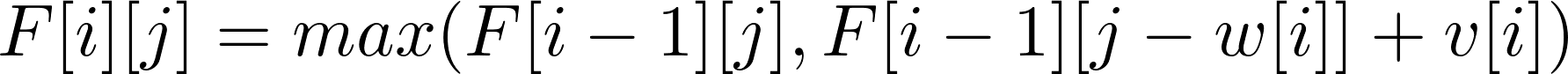
如果是多重背包中的物品，其重复次数最多为m，那么h(x)=，且，其他都为0

如果给定了两个泛化物品h和l，他们能获得的最大值价值为



假设h，l都是泛化物品，若函数f满足上述关系式，则称f是h与l的和。由上可知，如果将两个泛化物品用他们的和替代，则不影响结果

在01背包问题中，遍历到j=6时，此时能取到的最大价值为



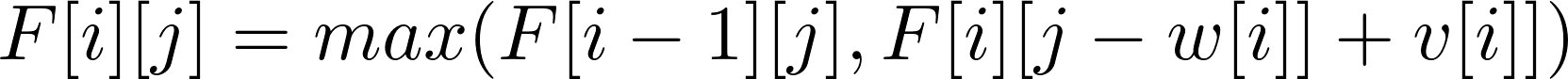
又因为是倒序遍历，所以dp[6]=4。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  | 4 |

在遍历到j=3时，因为dp[3]依赖的是dp[0]，所以此时dp[3]还是4，故不会有一件物品选择两次的情况。

在完全背包问题中，首先遍历到j=3，此时能取到的最大价值为



所以dp[3]=4。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 |

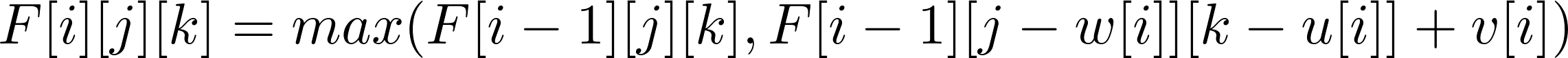
在遍历到j=6时，dp[6]依赖的是dp[3]，而此时dp[3]已经有值了，所以dp[6]=8，从而达到了一件物品选多次的目的

将第i种物品使用二进制的思想拆分成若干个物品，例如p[i]=13，那么第i物品可以拆分为（w[i],v[i]），（2w[i],2v[i]），（4w[i],4v[i]），（6w[i],6v[i]）四件物品，此时问题被转化成了一个01背包问题

二维费用的背包问题是指：对于每件物品，具有两种不同的费用，选择这件物品必须同时付出这两种费用。对于每种费用都有一个可付出的最大值（背包容量）。问怎样选择物品可以得到最大的价值。

设第 i 件物品所需的两种费用分别为 Wi 和 Ui。两种费用可付出的最大值（也即两种背包的容量）分别为 W 和 U。物品的价值为 Vi。

此背包问题的维度增加了一维，则状态页增加一维：令表示第i件物品付出的代价为j和k时，能获取的最大价值。那么状态转移方程为：



其他遍历顺序等不变