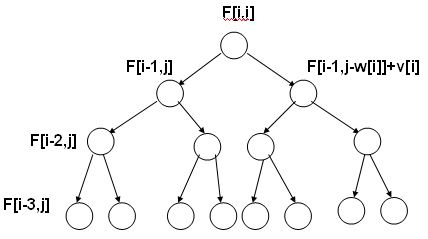
**[01背包问题](http://www.cnblogs.com/gzydn/archive/2009/06/11/1501439.html)**

Posted on 2009-06-11 15:55 [YDN](http://www.cnblogs.com/gzydn/) 阅读(673) [评论(4)](http://www.cnblogs.com/gzydn/archive/2009/06/11/1501439.html#commentform) [编辑](http://www.cnblogs.com/gzydn/archive/2009/06/11/1501439.html) [收藏](http://www.cnblogs.com/gzydn/archive/2009/06/11/1501439.html) 所属分类: [动态规划](http://www.cnblogs.com/gzydn/category/191175.html) http://www.cnblogs.com/gzydn/aggbug/1501439.html?type=1&webview=1

问题描述：  
      有 n 件物品x1, x2, …, xn , 每件物品有一个价值和一个重量，分别记为：   
      v1,v2, …vn   
      w1,w2, …wn   
其中所有的 wi 均为整数。 现有一个背包，其最大载重量为m，要求从这n件物品中任取若干件（这些物品每样只有一件，要么被装入要么被留下）。问背包中装入哪些物品可使得所装物品的价值和最大？ （我们只需要求出最大价值，不需要求出具体拿的是哪些物品）  
例如，m=23, n = 5,   
        vi : 19 24 33 45 50   
        wi : 5   6  8   11 12   
最大价值为：95

分析：  
       如果想用贪心，先求出平均价值，然后从高到低的方法来取，如果有一个背包的容量为10，共有3个物品，体积分别是3、3、5，价值分别是6、6、9，那么你的方法取到的是前两个物品，总价值是12，但明显最大值是后两个物品组成的15。因此贪心的方法不能得到正确结果。  
       换一个更简单的方式来思考：每个物品只有2种选择，要么放入，要么不放入。  
      （1）放入：问题转换为在背包载重为m-wi的情况下，在其它n-1件物品中挑选，求得价值和最大。等把这个子问题求出后，再加上vi的价值就是整个问题的最优解了。   
      （2） 没放入：那么就当xi 根本不存在，直接解物品数量为n-1，背包载重为m的子问题。子问题的最优解就是问题的最优解。  
      定义函数f(i,j)为在1~i 件物品中选若干件装入限重为 j 的背包中的最大价值和， 那么根据上面关于第 i 件物品是否装入了背包的情况分析，我们得出关系式：   
      (1)当第I件物品要装入背包时，f(i,j) := (i-1件物品，限重为j-w[i]的最优解)+ v[i], 即： f(i,j) := f(i-1, j-w[i]) + v[i]   
     当然，第i件物品要装入是有条件限制的：第i 件物品重量小于等于背包限重，即 w[i] <= j   
    （2）当第i件物品不装入背包时，f(i,j) :=i-1件物品，限重为m的最优解，即： f(i,j) := f(i-1, j)   
     求得装入或者不装入第 i 件物品的限重为J的背包的最大价值，只需要比较这两种情况下谁的价值更大，更大者为当前问题的最优解。  
      f(i,j)=max{ f(i-1, j-w[i]) + v[i] , f(i-1, j) }    该方程递归结束的边界条件是：当j=0时，f(i,0)=0。  
      在按自底向上的动态规划方式求解问题时，其实主要就是做一件事：   
      按问题规模从小到大地求解问题，把每阶段求得的问题的最优解保存在表格（数组）中，以便在下一阶段求解更大的问题时，可以直接查表引用子问题的最优解。 （类似于递推）  
   阶段的分析：   
        将n件物品放入背包，故可以把阶段划分为n个。把在前I件物品中选取物品放入背包作为第I个阶段。  
   状态的分析：  
        在第i 个阶段有多少个状态呢?因为包的容量为m，故在每个i 阶段都有m个状态：f(i,1)、f(i,2)、f(i,3)……f(i,m-1)、f(i,m)。  
        而这m个问题的求解基础是第I-1个阶段的m个子问题。这些子问题的最优解已经先于此时求得，保存在f[I-1,1], f[I-1,2], …, f[I-1,m]中，现在只需要直接引用它们的值就可以了。这就是动态规划自底向上的体现。  
         之所以能求得之前的状态，是因为有一个边界：f(i,0)=0。  
   决策的分析：  
         只有两个，在前I件物品中，限重为J的条件下，装入第I件物品还是不装入第I件物品。比较这两种情况的价值和，谁大谁就是最优解。  


http://www.cnblogs.com/Images/OutliningIndicators/ContractedBlock.gifhttp://www.cnblogs.com/Images/OutliningIndicators/ExpandedBlockStart.gifCode  
 1http://www.cnblogs.com/Images/OutliningIndicators/None.gif//01背包问题  
 2http://www.cnblogs.com/Images/OutliningIndicators/None.gifprogram beibao01;  
 3http://www.cnblogs.com/Images/OutliningIndicators/None.gifvar  
 4http://www.cnblogs.com/Images/OutliningIndicators/None.gif    v,w:array[1..100] of integer;  //物品的价值和重量  
 5http://www.cnblogs.com/Images/OutliningIndicators/None.gif    f:array[0..1000,0..1000] of integer;  //取得的价值  
 6http://www.cnblogs.com/Images/OutliningIndicators/None.gif    m,n,i,j:integer;   //m为包的容量，n为物品数  
 7http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
 8http://www.cnblogs.com/Images/OutliningIndicators/None.gifprocedure init;  
 9http://www.cnblogs.com/Images/OutliningIndicators/None.gifbegin  
10http://www.cnblogs.com/Images/OutliningIndicators/None.gif  readln(m,n);  
11http://www.cnblogs.com/Images/OutliningIndicators/None.gif  for i:=1 to n do  
12http://www.cnblogs.com/Images/OutliningIndicators/None.gif    read(w[i],v[i]);  
13http://www.cnblogs.com/Images/OutliningIndicators/None.gifend;  
14http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
15http://www.cnblogs.com/Images/OutliningIndicators/None.gifprocedure work;  
16http://www.cnblogs.com/Images/OutliningIndicators/None.gifbegin  
17http://www.cnblogs.com/Images/OutliningIndicators/None.gif  for i:=1 to n do    //阶段  
18http://www.cnblogs.com/Images/OutliningIndicators/None.gif    for j:=1 to m do   //状态  
19http://www.cnblogs.com/Images/OutliningIndicators/None.gif       if (j>=w[i]) then  //如果包的容量还能够放下一个包  
20http://www.cnblogs.com/Images/OutliningIndicators/None.gif          if (f[i-1,j-w[i]]+v[i])>f[i-1,j] then  //比较不放包和放包两中策略中价值大的一种  
21http://www.cnblogs.com/Images/OutliningIndicators/None.gif            f[i,j]:=f[i-1,j-w[i]]+v[i]  //能放时候的价值  
22http://www.cnblogs.com/Images/OutliningIndicators/None.gif          else  
23http://www.cnblogs.com/Images/OutliningIndicators/None.gif            f[i,j]:=f[i-1,j];    //不能放时候的价值  
24http://www.cnblogs.com/Images/OutliningIndicators/None.gifend;  
25http://www.cnblogs.com/Images/OutliningIndicators/None.gif  
26http://www.cnblogs.com/Images/OutliningIndicators/None.gifbegin  
27http://www.cnblogs.com/Images/OutliningIndicators/None.gif  init;  
28http://www.cnblogs.com/Images/OutliningIndicators/None.gif  work;  
29http://www.cnblogs.com/Images/OutliningIndicators/None.gif  writeln(f[n,m]);  
30http://www.cnblogs.com/Images/OutliningIndicators/None.gif  readln;  
31http://www.cnblogs.com/Images/OutliningIndicators/None.gifend.