多重背包单调队列优化

2014-05-28

问题

`N种物品和容量为V的背包，若第i种物品大小(重量)为we[i]，价值为va[i],共有n[i]件,怎样才能使背包内物品总重量最大`

多重背包一般解法

朴素的三重枚举，
将物品的数量拆成0、1、2、4 等二的指数，分别计算其价值，当做单独的物品。因为用这些数可以组合出各种<=m(m为此种物品的数量)的数，也就可以用普通的01背包

分析:

```
dp[i][j]表示前i种物品，背包大小为j的最大总价值

m[i] = min(n[i], j / we[i])

dp[i][j] = max{f[i - 1][j - k * we[i]] + k * va[i]} {0 <= k <= m[i]}
```

> 对于第一种方法，基本只能用来对拍，几乎所有的多重背包的题目，用第一种方法都会超时。对于第二种方法，时间复杂度为 $O(n*w*log(m))$ (n为物品种数，w为背包大小，m为此物品数量)，是一种比较高效的方法。

单调队列优化 O(n*w)

另外一种就是用单调队列。时间复杂度为$O(n*w)$O(n∗w)，只是常数可能会稍大。

根据上面的式子，每个j值对应的k有m[i] + 1个里面去找最大的那个，就相当于在一个区间里面找一个最大值，可以考虑用单调队列来做这个事情，每次维护队列是单调递减的，每次取出来队列头作为转移，每次加入的时候，把前面的比它小的就出队，然后超过m[i]+1的也出队。
根据上面的式子，发现对于j这个维数来说，如果两个体积值对于we[i]取余的值不一样，是不可能转移的。这样的话直接按照mod的这个值来做转移，因为mod的值是在[0,we[i]-1]，后面再枚举k的值。

假设mod = j % we[i]，a = j / we[i]，那么j = a * we[i] + mod，可得:

dp[i][j] = max{dp[i - 1][mod + (a - k) *we[i]] + k* va[i]} 其中{0 <= k <= m[i]}

化简一下，把a - k 用k来替换就可以得到:

dp[i][j] = max{dp[i - 1][mod + k *we[i]] - k* va[i]} + a * va[i] 其中{a - m[i] <= k <= a}

这样扫描的时候就是，第一重循环是枚举i为N(物品种数)，第二重循环是枚举的mod从0到we[i]-1,然后第三重循环是枚举的余数为mod的情况下的k值，k值是从0到mod + k * we[i] <= V的对于每一个k对应一个这么大小的背包，然后找以这个为单调队列末尾的那个队列头的值，完成转移方程。

多重背包的需要单调队列优化的一些例题

POJ 2392

POJ 1742