二进制思想和多重背包问题

二进制思想

问题描述：

假设有1000个苹果，现在要取n个苹果，如何取？正常的做法应该是将苹果一个一个拿出来，直到n个苹果被取出来。

又假设有1000个苹果和10只箱子，如何快速的取出n个苹果呢？可以在每个箱子中放 2^i (i<=0<=n)个苹果，也就是 1、2、4、8、16、32、64、128、256、489（最后的余数），相当于把十进制的数用二进制来表示，取任意n个苹果时，只要推出几只箱子就可以了。

多重背包问题

问题描述：

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。

问题分析：

很容易想到可以把该问题转化成01背包问题来考虑，把每n[i] 中的每个物品都当成一个独立的物品，而这 n[i] 个物品能够表示的重量其实用 log n[i]个组合后的物品就能表示。

内层循环代码如下：

```
int k, t;
k = 1;
t = n[i];
while(t > k)
{
    for(j=W; j>=c[i]*k; --j)
    {
        f[j] = max(f[j], f[j-c[i]*k] + w[i]*k);
    }
    t -= k;
    k *= 2;
}
for(j=W; j>=c[i]*t; --j)
{
    f[j] = max(f[j], f[j-c[i]*t] + w[i]*t);
}
```

poj上的1014是一道简单的多重背包问题。http://poj.org/problem?id=1014

源码如下：