原创 【luogu 1450】 HAOI 2008 容斥原理

2019-09-24 17:14:33 我是一只计算鸡 阅读数 8

不定方程非负整数解：$\sum_{i=1}^{n} x_i = m$ ，$x_i \le b_i$

$$\left|\bigcap_{i=1}^{n} S_i\right| = |U| - \left|\bigcup_{i=1}^{n} S_i'\right|$$

对于$S_{ai}{}'$表示$a_i >= b_i + 1$的解的数目。

有的下界大于0，我们可以同时减掉这个下界，使下界都为0。

那么$\left|\bigcap_{a_i<a_i+1}^{1\le i\le k} S_{a_i}\right|$的不定方程形式为$\sum_{i=1}^{n} x_i = m - \sum_{i=1}^{k}(b_{a_i}+1)$，这个长度为 k 的 a 数组相当于在枚举子集。

HAOI 2008：有4种不同的硬币c[i]，每种硬币都有一定的数量上限d[i]，买价值为S的物品，一共有多少种付款方式。实际上就是求解有限制的[illegible]种$\sum_{i=1}^{n} c_i x_i = S, x_i \le d_i$ 非负整数解的个数。

等价于求解：$\sum_{i=1}^{n} c_i x_i = m - \sum_{i=1}^{k}(d_i+1) * c_i$

二进制枚举子集

```
#include<bits/stdc++.h>
using namespace std;
const int maxn = 1e5 + 7;
typedef long long ll;
ll c[5], d[5], tot, S, f[maxn];
int main()
{
        scanf("%lld %lld %lld %lld %lld", &c[1], &c[2], &c[3], &c[4], &tot);
        f[0] = 1;  // 递推 预处理 每种都可以买无限制次的方案数
        for(int i = 1; i <= 4; i++) {
                for(int j = c[i]; j < maxn; j++) f[j] += f[j - c[i]];
        }
        while(tot--) {
                scanf("%lld %lld %lld %lld %lld", &d[1], &d[2], &d[3], &d[4], &S);
                ll ans = 0;
                for(int i = 1; i < 16; i++) {  // 二进制枚举子集 买还是不买
                        ll m = S, bit = 0;
                        for(int j = 1; j <= 4; j++) {  // 超出限制d[j] + 1，剩下的随便买
                                if(i >> (j - 1) & 1) m -= (d[j] + 1) * c[j], bit++;
                        }
                        if(m >= 0) ans += (bit % 2 * 2 - 1) * f[m];  // 容斥
                }
                printf("%lld\n", f[S] - ans);
        }
        return 0;
}
```

dfs 枚举

```
#include<bits/stdc++.h>
using namespace std;
const int maxn = 1e5 + 7;
typedef long long ll;
ll d[5], c[5], tot, S, f[maxn], ans;
```

```
void init()
{
        f[0] = 1;
        for(ll i = 1; i <= 4; i++) {
                for(ll j = c[i]; j < maxn; j++) f[j] += f[j - c[i]];
        }
}
void dfs(ll num, ll k, ll sum)
{
        if(sum < 0) return ;
        if(num == 5) {
                if(k & 1) ans -= f[sum];
                else ans += f[sum];
                return ;
        }
        dfs(num + 1, k + 1, sum - (d[num] + 1) * c[num]);
        dfs(num + 1, k, sum);
        return ;
}
int main()
{
        scanf("%lld %lld %lld %lld %lld",  &c[1], &c[2], &c[3], &c[4], &tot);
        init();
        while(tot --) {
                scanf("%lld %lld %lld %lld %lld", &d[1], &d[2], &d[3], &d[4], &S);
                        ans = 0;
                        dfs(1, 0, S);
                        printf("%lld\n", ans);
        }
        return 0;
}
```


文章最后发布于: 201