

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

版权声明：本文为博主原创文章，遵循 CC 4.0 BY-SA 版权协议，转载请附上原文出处链接和本声明。

本文链接：<https://blog.csdn.net/giftedpanda/article/details/101295617>

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

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

对于 S_{a_i}' 表示 $a_i \geq b_i + 1$ 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 $\left| \bigcap_{a_i < a_i + 1}^{1 \leq i \leq k} S_{a_i} \right|$ 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

$\sum_{i=1}^n c_i x_i = S, x_i \leq d_i$
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

```
1 #include<bits/stdc++.h>
2 using namespace std;
3 const int maxn = 1e5 + 7;
4 typedef long long ll;
5 ll c[5], d[5], tot, S, f[maxn];
6 int main()
7 {
8     scanf("%lld %lld %lld %lld %lld", &c[1], &c[2], &c[3], &c[4], &tot);
9     f[0] = 1; // 递推 预处理 每种都可以买无限次数的方案数
10    for(int i = 1; i <= 4; i++) {
11        for(int j = c[i]; j < maxn; j++) f[j] += f[j - c[i]];
12    }
13    while(tot--) {
14        scanf("%lld %lld %lld %lld %lld", &d[1], &d[2], &d[3], &d[4], &S);
15        ll ans = 0;
16        for(int i = 1; i < 16; i++) { // 二进制枚举子集 买还是不买
17            ll m = S, bit = 0;
18            for(int j = 1; j <= 4; j++) { // 超出限制  $d[j] + 1$ ，剩下的随便买
19                if(i >> (j - 1) & 1) m -= (d[j] + 1) * c[j], bit++;
20            }
21            if(m >= 0) ans += (bit % 2 * 2 - 1) * f[m]; // 容斥
22        }
23        printf("%lld\n", f[S] - ans);
24    }
25    return 0;
26 }
```

dfs 枚举

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

1024

程序员节，为程序员加油！

关闭

```
6 void init() {
7     {
8         f[0] = 1;
9         for(ll i = 1; i <= 4; i++) {
10             for(ll j = c[i]; j < maxn; j++) f[j] += f[j - c[i]];
11         }
12     }
13 void dfs(ll num, ll k, ll sum)
14 {
15     if(sum < 0) return ;
16     if(num == 5) {
17         if(k & 1) ans -= f[sum];
18         else ans += f[sum];
19         return ;
20     }
21     dfs(num + 1, k + 1, sum - (d[num] + 1) * c[num]);
22     dfs(num + 1, k, sum);
23     return ;
24 }
25 int main()
26 {
27     scanf("%lld %lld %lld %lld %lld", &c[1], &c[2], &c[3], &c[4], &tot);
28     init();
29     while(tot --) {
30         scanf("%lld %lld %lld %lld %lld", &d[1], &d[2], &d[3], &d[4], &S);
31         ans = 0;
32         dfs(1, 0, S);
33         printf("%lld\n", ans);
34     }
35     return 0;
36 }
```

1024

程序员节，
为程序员加油！

关闭

有 0 个人打赏

文章最后发布于: 201

©2019 CSDN 皮肤主题: 终极编程指南 设计师: CSDN官方博客