万能欧几里得算法

算法简介

一种 $O(\log n)$ 计算形如 $f(n)=\sum_{i=1}^n \lfloor \frac {ai+b}c \rfloor$ 的函数的算法。

算法原理

不妨考虑待计算函数就是 $f(n)=\sum_{i=1}^n \lfloor \frac {ai+b}c\rfloor$，同时假设 $b\lt c$，考虑按如下过程计算 $f(n)$。

考虑在 $(0,n]$ 范围内逐渐增加 x，当 $y=\frac {ax+b}c$ 恰好等于某个整数时 $y\gets y+1$，称为 U 操作。

当 x 恰好等于某个整数时有 $\text{ans}\gets \text{ans}+y$，称为 R 操作。如果遇到整点，先进行 U 操作再进行 R 操作。

于是计算 $f(n)$ 等价于处理由 $y=\frac {ax+b}c(x\in (0,n])$ 产生的操作序列 S。

先考虑如何得到这个操作序列，设 $g(a,b,c,n,U,R)$ 表示由 $y=\frac {ax+b}c(x\in (0,n])$ 产生的操作序列 S。

定义字符串幂操作 $S^k=S^{k-1}S$。

如果 $a\ge c$，显然 R 操作前面至少有 $\lfloor \frac ac\rfloor$ 个 U 操作。将这 $\lfloor \frac ac\rfloor$ 个 U 操作和一个 R 操作视为一个整体，于是有

$$ g(a,b,c,n,U,R)=g(a\bmod c,b,c,n,U,U^{\lfloor \frac ac\rfloor}R) $$

如果 $b\ge c$，显然有

$$ g(a,b,c,n,U,R)=U^{\lfloor \frac bc\rfloor}g(a,b\bmod c,c,n,U,R) $$

最后，如果 $c\gt \max (a,b)$，则考虑构造 $y=\frac {ax+b}c$ 按 $y=x$ 做对称变换得到的直线 $y=\frac {cx-b}a$。

不难发现，$y=\frac {cx-b}a$ 产生的序列和 $y=\frac{ax+b}c$ 产生的序列大致是互补的，即 U/R 互补。

但要处理遇到整点的情况，此时两条直线都是先 U 再 R 不满足互补关系，考虑将 $y=\frac {cx-b}a$ 整体向下偏移得到 $y=\frac {cx-b-1}a$。

这样原先在 $y=\frac {cx-b}a$ 先 U 再 R 的整点等价于 $y=\frac {cx-b-1}a$ 先 R 在 U，于是整点也满足了互补性质。

但偏移也导致 $y=\frac {cx-b-1}a$ 的边界发生了变化，考虑暴力处理边界，设 $m=\frac {an+b}c$，于是有

$$ g(a,b,c,n,U,R)=R^{\lfloor \cfrac{c-b-1}{a}\rfloor}Ug(c,(c-b-1)\bmod a,a,m-1,R,U)R^{n-\lfloor

\cfrac{cm-b-1}{a}\rfloor} $$

边界条件为 $m=0$，此时说明只有 R 操作，于是 $g(a,b,c,n,U,R)=R^n$。

假如不考虑字符串拼接等操作的复杂度，上述式子可以 $O(\log n)$ 计算。

接下来考虑如何通过得到的字符串即操作序列来计算答案。事实上可以在拼接字符串时维护答案。

设 $h(S)$ 表示字符串 S 对应的答案，两个串分别为 $S1,S2$，$dx=\text{cntR}(S1),dy=\text{cntU}(S1),n=\text{cntR}(S2)$。

$$ h(S1S2)=h(S1)+\sum_{i=1}^n (\lfloor \frac {ai+b}c\rfloor+dy)=h(S1)+\sum_{i=1}^n (\lfloor \frac {ai+b}c\rfloor)+n\times dy=h(S1)+h(S2)+n\times dy $$

至于 S^k 可以利用快速幂计算。整体操作复杂度为 $T(a,c)=T(c\bmod a,a)+O(\log \frac ca)$。

$T(a,c)$ 产生的 $O(\log c)-O(\log a)$ 可以和随后 $T(c\bmod a,a)$ 产生的 $O(\log a)-O(\log (c\bmod a))$ 部分抵消。

最终 $T(a,c)=O(\log c)-O(\log\gcd (a,c))\sim O(\log c)$。

算法例题

例题一

洛谷p5170

题意

求

$$ f(n)=\sum_{i=0}^n \lfloor \frac {ai+b}c\rfloor\\ g(n)=\sum_{i=0}^n (\lfloor \frac {ai+b}c\rfloor)^2\\ h(n)=\sum_{i=0}^n i\lfloor \frac {ai+b}c\rfloor $$

题解

设 F,G,H 分别表示序列 S 对应 f,g,h 的答案，于是有

$$ \begin{equation}\begin{split} F(S1S2)&=F(S1)+\sum_{i=1}^n (\lfloor \frac {ai+b}c\rfloor+dy)=F(S1)+F(S2)+n\times dy\\ G(S1S2)&=G(S1)+\sum_{i=1}^n (\lfloor \frac {ai+b}c\rfloor+dy)^2 \\ &=G(S1)+\sum_{i=1}^n (\lfloor \frac {ai+b}c\rfloor)^2+ 2dy\sum_{i=1}^n \lfloor \frac {ai+b}c\rfloor+dy+\sum_{i=1}^n (dy)^2\\ &=G(S1)+G(S2)+2dyF(S2)+n\times (dy)^2\\ H(S1S2)&=H(S1)+\sum_{i=1}^n (i+dx)(\lfloor \frac {ai+b}c\rfloor+dy) \\ &=H(S1)+\sum_{i=1}^n \left(i(\lfloor \frac {ai+b}c\rfloor)+i\times dy+dx(\lfloor \frac {ai+b}c\rfloor)+dxdy\right)\\ &=H(S1)+H(S2)+\frac {n(n+1)}2dy+dxF(S1)+n\times dxdy \end{split}\end{equation} $$ 最后这题的 i 是从 0 开始的，万能欧几里得算法的 i 是从 1 开始的，注意补上 $i=0$ 的贡献。

```
const int mod=998244353;
struct Node{
    LL cntr,cntu,f,g,h;
    Node operator * (const Node &b)const{
        Node c;
        LL n=b.cntr,dx=cntr,dy=cntu;
        c.cntr=(cntr+b.cntr)%mod;
        c.cntu=(cntu+b.cntu)%mod;
        c.f=(f+b.f+dy*n)%mod;
        c.g=(g+b.g+dy*b.f*2+dy*dy%mod*n)%mod;
        c.h=(h+b.h+n*(n+1)/2%mod*dy+dx*b.f+dx*dy%mod*n)%mod;
        return c;
    }
};
Node quick_pow(Node n,int k){
    Node ans=Node{0,0,0,0,0};
    while(k){
        if(k&1)ans=ans*n;
        n=n*n;
        k>>=1;
    }
    return ans;
}
Node asgcd(int a,int b,int c,int n,Node su,Node sr){
    if(a>=c)
    return asgcd(a%c,b,c,n,su,quick_pow(su,a/c)*sr);
    int m=(1LL*a*n+b)/c;
    if(!m)
    return quick_pow(sr,n);
    else
    return quick_pow(sr,(c-b-1)/a)*su*asgcd(c,(c-
b-1)%a,a,m-1,sr,su)*quick_pow(sr,n-(1LL*c*m-b-1)/a);
}
Node cal(int a,int b,int c,int n){
    Node su=Node{0,1,0,0,0},sr=Node{1,0,0,0,0};
    return quick_pow(su,b/c)*asgcd(a,b%c,c,n,su,sr);
}
int main()
{
    int T=read_int();
    while(T--){
        int n=read_int(),a=read_int(),b=read_int(),c=read_int();
        Node ans=cal(a,b,c,n);
        ans.f=(ans.f+b/c)%mod;
        ans.g=(ans.g+1LL*(b/c)*(b/c))%mod;
        space(ans.f);space(ans.g);enter(ans.h);
    }
    return 0;
}
```

例题二

LOJ#138

题意

求

$$ f(n)=\sum_{i=0}^n i^{k_1}(\lfloor \frac {ai+b}c\rfloor)^{k_2}\\ $$

题解

设 $F(S,k_1,k_2)$ 表示 $f(n)=\sum_{i=1}^n i^{k_1}(\lfloor \frac {ai+b}c\rfloor)^{k_2}$ 关于 S 操作序列的答案。

\begin{equation}\begin{split} F(S1S2,k_1,k_2)&=F(S1,k_1,k_2)+\sum_{i=1}^n (i^{k_1}+dx)^{k_1}(\lfloor \frac {ai+b}c\rfloor+dy)^{k_2}\\ &=F(S1,k_1,k_2)+\sum_{i=1}^n\sum_{t_1=0}^{k_1}\sum_{t_2=0}^{k_2}{k_1\choose t_1}{k_2\choose t_2}dx^{k_1-t_1}dy^{k_2-t_2}i^{t_1}(\lfloor \frac {ai+b}c\rfloor+dy)^{t_2}\\ &=F(S1,k_1,k_2)+\sum_{t_1=0}^{k_1}\sum_{t_2=0}^{k_2}{k_1\choose t_1}{k_2\choose t_2}dx^{k_1-t_1}dy^{k_2-t_2}F(S2,t_1,t_2) \end{split}\end{equation}

对每个 S□维护矩阵 $F(S,i,j)(0\le i\le k_1,0\le j\le k_2)$□于是可以实现 $O\left(k_1^2k_2^2\right)$ 信息合并。

边界 $F(L,i,j)=0,F(R,i,j)=[j==0](0\le i\le k_1,0\le j\le k_2)$□另外注意对答案补上 $i=0$ 的贡献。时间复杂度 $O\left(k_1^2k_2^2\log c\right)$□

```
const int mod=1e9+7,MAXK=11;
int C[MAXK][MAXK],k1,k2;
struct Node{
    int cntr,cntu,f[MAXK][MAXK];
    Node(int cntr=0,int cntu=0){
        this->cntr=cntr;
        this->cntu=cntu;
        mem(f,0);
    }
    Node operator * (const Node &b)const{
        static int px[MAXK],py[MAXK];
        Node c;
        int dx=cntr,dy=cntu;
        px[0]=py[0]=1;
        _rep(i,1,k1)
        px[i]=1LL*px[i-1]*dx%mod;
        _rep(i,1,k2)
        py[i]=1LL*py[i-1]*dy%mod;
        c.cntr=(cntr+b.cntr)%mod;
        c.cntu=(cntu+b.cntu)%mod;
```

```
        _rep(i,0,k1)_rep(j,0,k2){
            c.f[i][j]=f[i][j];
            _rep(i2,0,i)_rep(j2,0,j)
c.f[i][j]=(c.f[i][j]+1LL*b.f[i2][j2]*C[i][i2]%mod*C[j][j2]%mod*px[i-
i2]%mod*py[j-j2])%mod;
        }
        return c;
    }
};
Node quick_pow(Node n,int k){
    Node ans=Node(0,0);
    while(k){
        if(k&1)ans=ans*n;
        n=n*n;
        k>>=1;
    }
    return ans;
}
Node asgcd(int a,int b,int c,int n,Node su,Node sr){
    if(a>=c)
    return asgcd(a%c,b,c,n,su,quick_pow(su,a/c)*sr);
    int m=(1LL*a*n+b)/c;
    if(!m)
    return quick_pow(sr,n);
    else
    return quick_pow(sr,(c-b-1)/a)*su*asgcd(c,(c-
b-1)%a,a,m-1,sr,su)*quick_pow(sr,n-(1LL*c*m-b-1)/a);
}
Node cal(int a,int b,int c,int n){
    Node su=Node(0,1),sr=Node(1,0);
    _rep(i,0,k1)
    sr.f[i][0]=1;
    return quick_pow(su,b/c)*asgcd(a,b%c,c,n,su,sr);
}
int main()
{
    C[0][0]=1;
    _for(i,1,MAXK){
        C[i][0]=1;
        _rep(j,1,i)
        C[i][j]=(C[i-1][j-1]+C[i-1][j])%mod;
    }
    int T=read_int();
    while(T--){
        int n=read_int(),a=read_int(),b=read_int(),c=read_int();
        k1=read_int(),k2=read_int();
        Node ans=cal(a,b,c,n);
        int base=1;
        _rep(i,0,k2){
            ans.f[0][i]=(ans.f[0][i]+base)%mod;
            base=1LL*base*(b/c)%mod;
```

```
            }
            enter(ans.f[k1][k2]);
        }
        return 0;
}
```

参考资料

这两篇还有配图，方便理解。

C3H5ClO的博客

ix35_ 的博客