生成函数理论 5

基本应用

例 1

P2000 拯救世界

依照题意可列出 10 个条件的生成函数 $f_i(x)$将它们相乘再化简即可。

- $f_1(x)=(1+x^6+x^{12}+\cdots)=\frac{1}{1-x^6}$
- $f_2(x)=(1+x+x^2+\cdots+x^9)=\frac{1-x^{10}}{1-x}$
- $f_3(x)=(1+x+x^2+\cdots+x^5)=\frac{1-x^6}{1-x}$
- $f_4(x)=(1+x^4+x^8+\cdots)=\frac{1}{1-x^4}$
- $f_5(x)=(1+x+x^2+\cdots+x^7)=\frac{1-x^8}{1-x}$
- $f_6(x)=(1+x^2+x^4+\cdots)=\frac{1}{1-x^2}$
- $f_7(x)=(1+x)=\frac{1-x^2}{1-x}$
- $f_8(x)=(1+x^8+x^{16}+\cdots)=\frac{1}{1-x^8}$
- $f_9(x)=(1+x^{10}+x^{20}+\cdots)=\frac{1}{1-x^{10}}$
- $f_{10}(x)=(1+x+x^2+x^3)=\frac{1-x^4}{1-x}$

$\displaystyle f(x)=\prod_{i=1}^{10}f_i(x)=\frac{1}{(1-x)^5}=\sum_{n=0}^{\infty}\binom{n+4}{n}x^n$

$\displaystyle ans_n=\binom{n+4}{4}=\frac{(n+4)(n+3)(n+2)(n+1)}{24}$

由于 $10^{100000}\le n<10^{100001}$需要用高精度，正解是用 NTT用 Python 会 TLE但是竟然可以用 Ruby AC

Ruby 代码：

```
n=Integer(gets)
puts (n+4)*(n+3)*(n+2)*(n+1)/24
```

C++ 代码：

```
#include<bits/stdc++.h>
using namespace std;
const int N=5e6+5,P=998244353,G=3,Gi=332748118;
typedef long long ll;
int rev[N];
char s[N];
ll n1[N],n2[N],n3[N],n4[N];
ll fastpow(ll x,ll y){
    ll ret=1;
    for(;y;y>>=1,x=x*x%P)
```

```
        if(y&1) ret=ret*x%P;
    return ret;
}
void NTT(ll *y,int len,int on){
    for(int i=0;i<len;i++)
        if(i<rev[i])
            swap(y[i],y[rev[i]]);
    for(int mid=1;mid<len;mid<<=1){
        ll wn=fastpow(on==1?G:Gi,(P-1)/(mid<<1));
        for(int j=0;j<len;j+=(mid<<1)){
            ll w=1;
            for(int k=0;k<mid;k++,w=w*wn%P){
                ll u=y[j+k],t=w*y[j+k+mid]%P;
                y[j+k]=(u+t)%P;
                y[j+k+mid]=(u-t+P)%P;
            }
        }
    }
}
ll c[N];
void mul(ll *a,ll *b,int &n,int &m){
    for(int i=0;i<=n;i++) a[i]%=P;
    for(int i=0;i<=m;i++) b[i]%=P;
    int len=1,l=0;
    while(len<=n+m) len<<=1,l++;
    for(int i=0;i<len;i++) rev[i]=(rev[i>>1]>>1)|((i&1)<<(l-1));
    NTT(a,len,1);
    NTT(b,len,1);
    for(int i=0;i<len;i++) a[i]=a[i]*b[i]%P;
    NTT(a,len,-1);
    ll inv=fastpow(len,P-2);
    memset(c,0,sizeof(c));
    for(int i=0;i<=n+m;i++){
        a[i]=a[i]*inv%P;
        c[i]+=a[i];
        c[i+1]+=c[i]/10;
        c[i]%=10;
    }
    if(c[n+m+1]) n++;
    for(int i=0;i<=n+m;i++) a[i]=c[i];
    n=n+m;
}
ll ans[N];
int main(){
    scanf("%s",s);
    int n=strlen(s);
    n--;
    int len1=n,len2=n,len3=n,len4=n;
    for(int i=0;i<=n;i++) n1[i]=n2[i]=n3[i]=n4[i]=s[n-i]-'0';
    n1[0]+=1;
    n2[0]+=2;
```

```
    n3[0]+=3;
    n4[0]+=4;
    for(int i=0;i<=n;i++){
        n1[i+1]+=n1[i]/10;
        n1[i]%=10;
        n2[i+1]+=n2[i]/10;
        n2[i]%=10;
        n3[i+1]+=n3[i]/10;
        n3[i]%=10;
        n4[i+1]+=n4[i]/10;
        n4[i]%=10;
    }
    if(n1[n+1]) len1++;
    if(n2[n+1]) len2++;
    if(n3[n+1]) len3++;
    if(n4[n+1]) len4++;
    mul(n1,n2,len1,len2);
    mul(n1,n3,len1,len3);
    mul(n1,n4,len1,len4);

    ll p=0;
    int lenans=0;
    for(int i=len1;i>=0;i--){
        p=p*10+n1[i];
        ans[i]=p/24;
        p%=24;
        if(ans[i]&&!lenans) lenans=i;
    }
    for(int i=lenans;i>=0;i--)
        printf("%lld",ans[i]);
    return 0;
}
```

例 2 （Bell 数）

令 B_n 表示 $[n]$ （n 元集合）所有划分的个数，试找到计算 B_n 的公式。

讨论 $[n]$ 的划分中包含元素 n 的那个子集，设其含有 k （$1\le k\le n$）个元素，则剩余的 $k-1$ 个元素是从 $[n-1]$ 中选取的。而剩余的那些子集为 $n-k$ 个元素的一个划分，所以 $$ B_n=\sum_{k=1}^{n}\binom{n-1}{k-1}B_{n-k},~~~~n\ge 1. $$ 初始值为 $B_0=1,B_1=1$。令 $\displaystyle B(x)=\sum_{n\ge 0}\frac{B_n}{n!}x^n$，两边求导数，得到 $$ \begin{align} \frac{dB(x)}{dx}&=\sum_{n=1}^{\infty}\frac{B_n}{(n-1)!}x^{n-1}\\ &=\sum_{n=1}^{\infty}\frac{1}{(n-1)!}\left(\sum_{k=1}^{n}\binom{n-1}{k-1}B_{n-k}\right)x^{n-1}\\ &=\sum_{n=1}^{\infty}\sum_{k=1}^{n}\frac{x^{k-1}}{(k-1)!}\frac{B_{n-k}x^{n-k}}{(n-k)!}\\ &=\sum_{k=1}^{\infty}\frac{x^{k-1}}{(k-1)!}\sum_{n\ge k}\frac{B_{n-k}x^{n-k}}{(n-k)!}\\ &=\sum_{k=1}^{\infty}\frac{x^{k-1}}{(k-1)!}\sum_{i\ge 0}\frac{B_ix^i}{i!}~~(i=n-k)\\ &=e^xB(x). \end{align} $$ 解微分方程 $\displaystyle \frac{dB(x)}{dx}=e^xB(x)$，可得 $$ B(x)=Ce^{e^x}, $$ 其中 C 为待定常数。由初始条件 $B(0)=1$ 得到 $C=e^{-1}$，即 $$ B(x)=e^{e^x-1}, $$ 从而 $$ \begin{align} B(x)&=\frac{1}{e}\sum_{k=0}^{\infty}\frac{(e^x)^k}{k!}=\frac{1}{e}\sum_{k=0}^{\infty}\frac{

1}{k!}\sum_{n=0}^{\infty}\frac{k^nx^n}{n!}\\ &=\frac{1}{e}\sum_{n=0}^{\infty}\left(\sum_{k=0}^{\infty}\frac{k^n}{k!}\right)\frac{x^n}{n!}. \end{align} $$ 因此 $$ B_n=\frac{1}{e}\sum_{k=0}^{\infty}\frac{k^n}{k!}. $$ $\Box$

例 3

题意

P4451 [国家集训队]整数的lqp拆分

题解

Fibonacci 数的生成函数是这个（基础知识□□ $$ F(x)=\frac{x}{1-x-x^2} $$ 当拆分为恰好 k 个数时，答案的生成函数是 $F(x)^k$□那么答案的生成函数即为 $$ \begin{align} G(x)&=\sum_{i=0}^{\infty}F(x)^i\\ &=\frac{1}{1-F(x)}\\ &=\frac{1-x-x^2}{1-2x-x^2}\\ &=1+\frac{x}{1-2x-x^2}\\ &=1+\frac{\sqrt 2}{4}(\frac{1}{1-(1+\sqrt 2)x}-\frac{1}{1-(1-\sqrt 2)x})\\ &=1+\frac{\sqrt 2}{4}\sum_{n=0}^{\infty}[(1+\sqrt 2)^n-(1-\sqrt 2)^n]x^n \end{align} $$ 所以通项公式为 $$ a_n\equiv \frac{\sqrt 2}{4}[(1+\sqrt 2)^n-(1-\sqrt 2)^n]\pmod{1e9+7} $$ 需要用二次剩余解出 $x^2\equiv 2\pmod{1e9+7}$

评价

对于理解生成函数和公式推导有一定的帮助

代码

```
#include<bits/stdc++.h>
using namespace std;
typedef long long ll;
const ll mod=1e9+7;
const ll two=59713600;
ll ksm(ll x,ll y){
    ll ret=1;
    for(;y;y>>=1,x=x*x%mod)
        if(y&1) ret=ret*x%mod;
    return ret;
}
int main(){
    string s;
    cin>>s;
    ll n=0;
    int len=s.length();
    for(int i=0;i<len;i++){
        n=(n*10+s[i]-'0')%(mod-1);
    }
```

```
    n=(n+(mod-1))%mod;
    n++;
    ll ans=(ksm(1+two,n)-ksm((1-two+mod)%mod,n)+mod)%mod*two/4%mod;
    printf("%lld",ans);
    return 0;
}
// sqrt(2):59713600
```

拓展

令 a_n 表示在一个 n-集合上完成某个任务的方法数□$a_0=0$□对于 $n\ge 1$□令 b_n 表示将集合 $[n]$ 划分成任意的连续、非空区间段，然后在这些非空区间段上完成前面任务的方法数□$b_0=1$□设 $\{a_n\}_{n=0}^{\infty}$ 和 $\{b_n\}_{n=0}^{\infty}$ 的普通生成函数分别为 $f(x)$ 和 $g(x)$□则对任意固定的正整数 i□将 $[n]$ 划分成 i 个非空子区间时，所对应的方法数的普通生成函数为 $f^i(x)$□且有 $$ g(x)=\frac{1}{1-f(x)}. $$