集训六问题2题解

Uva的题网上搜得到题解的我就不多写了。

第五题hdu 1812网上好像都是用java实现，这题我之前已经把我写的c++代码发上去了。

Problem 2 BT的Boss

首先，虽然每次是选取一个格子组成的矩形区域，但事实上，每一个格子在每个单位时间被选取是互不关联的，也就是我们可以逐个计算格子的期望值，然后相加。具体实现如下：

1. 枚举当前格子(i,j)
2. 求出格子(i,j)在被选中的概率，即（所有包含(i,j)的矩形/(n\*m\*n\*m)），令为概率p

此时，p[i][j]=(2.0\*i\*(n-i+1)-1)/n/n\*((2.0\*j\*(m-j+1)-1)/m/m)

1. 令f[i][1]：时间i后，当前格子没被攻击的概率；

f[i][2]：时间i后，当前格子为火焰状态的概率；

f[i][3]：时间i后，当前格子为寒冰状态的概率。

f[i][1]=f[i-1][1]\*(1-p)+f[i-1][2]\*p/2+f[i-1][3]\*p/2; //由i-1转移过来

f[i][2]=f[i-1][1]\*p/2+f[i-1][2]\*(1-p/2);

f[i][3]=f[i-1][1]\*p/2+f[i-1][3]\*(1-p/2);

当然这时候还不算玩，我们考虑下时间复杂度，惊讶地hua现：

O(T\*n\*m\*t)=O(10^9)，写了半天Time limit excceed有没有，回忆我们学过的线性递推优化技巧，想到了什么？矩阵快速幂!（这个优化技巧大家应该熟练掌握。）

这样成功优化到:O（T\*n\*m\*logt）=O(10^7)，这样就完美解决了这道题。

具体实现看代码，之前写的代码，很丑，还是希望大家自建结构体解决矩阵乘法问题。

#include<cstdio>

#include<cstdlib>

#include<iostream>

#include<cstring>

#include<algorithm>

#include<cmath>

#define nn 110

using namespace std;

typedef double dbl;

dbl p[nn][nn];

dbl a[4][4],b[4][4],c[4][4];

int i,j,T,k,m,t,n;

void calcp()

{

memset(p,0,sizeof(p));

for (i=1;i<=n;i++)

for (j=1;j<=m;j++)

p[i][j]=(2.0\*i\*(n-i+1)-1)/n/n\*((2.0\*j\*(m-j+1)-1)/m/m);

}

void equal(dbl a[][4],dbl b[][4])

{

for (int i=1;i<=3;i++)

for (int j=1;j<=3;j++)

a[i][j]=b[i][j];

}

void plu(dbl a[][4],dbl b[][4],dbl c[][4])

{

int i,j,k;

for (i=1;i<=3;i++)

for (j=1;j<=3;j++)

{

a[i][j]=0;

for (k=1;k<=3;k++)

a[i][j]+=b[i][k]\*c[k][j];

}

}

dbl equ(dbl p)

{

memset(b,0,sizeof(b));

b[1][1]=1-p;

b[2][2]=b[3][3]=1-p/2;

b[1][2]=b[1][3]=b[2][1]=b[3][1]=p/2;

memset(a,0,sizeof(a));

a[1][1]=a[2][2]=a[3][3]=1;

k=t;

while (k)

{

if (k&1)

{

equal(c,a);

plu (a,b,c);

}

equal(c,b);

plu (b,c,c);

k>>=1;

}

return a[1][1];

}

int main()

{

cin>>T;

while (T--)

{

dbl ans=0;

cin>>n>>m>>t;

calcp();

for (i=1;i<=n;i++)

for (j=1;j<=m;j++)

ans+=equ(p[i][j]);

printf("%.4lf\n",ans);

}

return 0;

}

另外这里讲一个小技巧，我们知道可以重载 [] 这个操作符，但是只限于一维，[][]这样的操作符是无法重载的，遇到下面的问题，怎么解决？

Struct matrix{

Int a[100][100];

};

Matrix mat;

我们访问a[i][j]一般要用mat.a[i][j]这种格式，但希望能直接用mat[i][j]来替代它不是方便多了？可是又不能重载[][]双操作符，有这样的方法：

Struct matrix{ Int a[100][100];

Int\* operator [](int i){return a[i];} //此处我们改为返回一个指针即可。

};