

圆的面积并问题和一些拓展

问题

给 N 个圆，求其面积并 $N \leq 1000$

解法 1

扫描线 + 辛普森积分。这里被积函数 $f(t)$ 为 $x=t$ 这条直线被圆覆盖的长度。

在实现时考虑辛普森积分的特点，需要预处理出圆在 x 轴的投影。

code

```
#include <bits/stdc++.h>
using namespace std;
const double eps = 1e-8;
const int maxn = 1e3 + 23;

struct Circle{
    double x, y, r;
}a[maxn];

int n;

double f(double x){
    vector<pair<double, int>> v;
    for(int i = 1; i <= n; i++){
        if (x >= a[i].x + a[i].r || x <= a[i].x - a[i].r) continue;
        double d = a[i].r * a[i].r - (x - a[i].x) * (x - a[i].x);
        d = sqrt(d);
        v.push_back({a[i].y - d, 1});
        v.push_back({a[i].y + d, -1});
    }
    sort(v.begin(), v.end());
    double ret = 0;
    for(int i = 0, cnt = 0; i < (int)v.size() - 1; i++){
        cnt += v[i].second;
        if(cnt) ret += v[i + 1].first - v[i].first;
    }
    return ret;
}

double calc(double l, double r){
    return (r - l) * (f(l) + f(r) + 4 * f((l + r) / 2)) / 6;
}
```



```

#include <bits/stdc++.h>
using namespace std;
typedef pair<double,int> pdd;
const double eps=1e-10;
const double pi=acos(-1.0);
const double pi2=2*pi;
const int maxn=1e3+23;
struct Point{
    double x,y;
    Point(double x=0,double y=0):x(x),y(y) {}
    bool operator < (const Point &b) const
    {
        return x<b.x||(x==b.x&&y<b.y);
    }
};
typedef Point Vector;
Point operator + (Point A,Point B){return Point(A.x+B.x,A.y+B.y);}
Point operator - (Point A,Point B) {return Point(A.x-B.x,A.y-B.y);}
Point operator * (Point A,double B) {return Point(A.x*B,A.y*B);}
Point operator / (Point A,double B) {return Point(A.x/B,A.y/B);}
int dcmp(double x)
{
    if(fabs(x)<eps) return 0;
    return x<0 ? -1 : 1;
}
bool operator == (const Point &a,const Point &b)
{
    return dcmp(a.x-b.x)==0&& dcmp(a.y-b.y)==0;
}

double dot(Vector a,Vector b)
{
    return a.x*b.x+a.y*b.y;
}
double length(Vector a)
{
    return sqrt(dot(a,a));
}
double angle(Vector v)
{
    return atan2(v.y,v.x);
}
double cross(Vector a,Vector b)
{
    return a.x*b.y-a.y*b.x;
}
struct Circle
{
    Point c;double r;
    Point point(double a)
    {

```

```
        return Point(c.x+cos(a)*r,c.y+sin(a)*r);
    }
    void read()
    {
        scanf("%lf%lf",&c.x,&c.y);
    }
    double oint(double t1,double t2)
    {
        return r*(r*(t2-t1)+c.x*(sin(t2)-sin(t1))-c.y*(cos(t2)-cos(t1)));
    }
    bool operator < (const Circle rhs) const
    {
        return c<rhs.c||(c==rhs.c&&r<rhs.r);
    }
    bool operator == (const Circle rhs) const
    {
        return c==rhs.c&&(dcmp(r-rhs.r)==0);
    }
}a[maxn];
pdd p[maxn<=2];int cnt,pb;
void getCircleIntersection(const Circle &C1,const Circle &C2)
{
    double d=length(C1.c-C2.c);
    if((C1.r+C2.r-d)<=0||C1.r-C2.r-d>=0){
        return ;
    }
    if(((C2.r-C1.r)-d)>=0) {pb++;return ;}
    double a=angle(C2.c-C1.c);
    double da=acos((C1.r*C1.r+d*d-C2.r*C2.r)/(2*C1.r*d));
    double l=a-da,r=a+da;
    if(l<-pi) l+=pi2;
    if(r>=pi) r-=pi2;
    if(l>r) pb++;
    p[cnt++]={l,1};
    p[cnt++]={r,-1};
}
int n;

double cal(int x)
{
    double ans=0;cnt=0;
    pb=0;
    for(int i=1;i<=n;i++){
        if(i==x) continue;
        getCircleIntersection(a[x],a[i]);
    }
    p[cnt++]={-pi,0};p[cnt++]={pi,0};
    sort(p,p+cnt);
    for(int i=1;i<cnt;i++){
        if(pb==0) ans+=a[x].oint(p[i-1].first,p[i].first);
    }
}
```

```

        pb+=p[i].second;
    }
    return ans;
}

int main(){
    cin>>n;
    for(int i=1;i<=n;i++){
        scanf("%lf%lf%lf",&a[i].c.x,&a[i].c.y,&a[i].r);
    }
    double ans=0;
    for(int i=1;i<=n;i++) ans+=cal(i);
    printf("%.4f",ans/2);
}

```

一些拓展问题

问题 1

给 N 个圆，每个圆出现的概率为 p_i 求面积并的期望 $N \leq 1000$

解法

将扫描线的区间加转换为区间乘概率或轮廓上的区间加转为区间乘概率即可。

[code](#)

问题 2

给 N 个圆，求被至少 k 个圆覆盖部分的面积。

解法 1

扫描线求被覆盖至少 k 次的长度之后辛普森积分。

解法 2

被覆盖至少 k 次部分的轮廓线由恰好被覆盖 $k-1$ 次的圆弧组成。

[code](#)

PS：上面两题笔者的辛普森积分均不能通过，可能是辛普森积分的姿势太低了。

From:
<https://wiki.cvbbacm.com/> - **CVBB ACM Team**

Permanent link:
https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:i_dont_know_png:qxforever:circleunion 

Last update: **2020/09/17 17:17**