平面图的处理方法

清华大学　钱桥

一、平面几何中的基本元素

- 点、向量
- 角度
- 直线、射线、线段
- 简单多边形
- 求向量夹角
- 向量旋转
- 点与线的位置关系
- 点到线的投影
- 线段求交
- 点与简单多边形的位置关系

- 给定一个凸多边形A，查询一个点是否在它内部。

  ——O(n)预处理，O(logn)在线回答

- 给定一个顶点满足极角序的多边形B，查询一个点是否在它内部。

  ——O(n)预处理，O(logn)在线回答

- 给定一个多边形C，查询一个点是否在它内部。
- 给定一个平面图D，查询一个点在哪个域。

  O(nlogn)预处理，O(logn)回答！这就是我们今天的目标！

二、平面图的基本性质

定义:

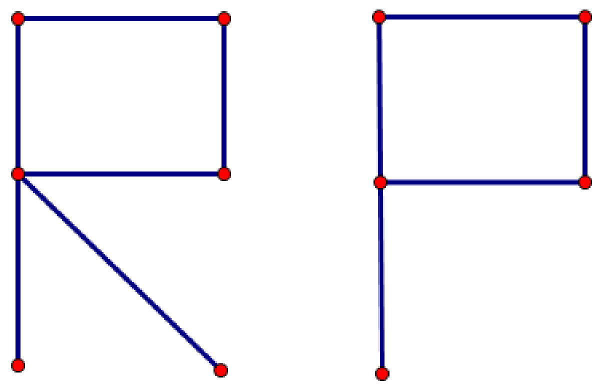
若图G可画在平面上，使得任意两条边都不会在非端点处相交，则称G是平面图。

n：平面图中的点数

m：平面图中的边数

k：平面图中连通块的数量

r：平面图中域的数量



性质1：$r = m - n + k + 1$

证明：

对于n个点，生成为k个连通块的森林，需要n-k条边

在此基础上，保持连通块数量不变，每增加一条边，会导致增加一个域

由于初始有1个域，增加的边数为$m - (n-k)$，故域数为$m - n + k + 1$。

性质2：r <= 2*n – 4

性质3：m <= 3*n - 6

证明：

每个域最少也要有3条边包围，每条边可用2次，故有2*m >= 3*r。

带入等式r = m – n + k + 1，再结合k>=1即可。

结论：平面图中边、域的数量均是O(n)的。

三、平面图的存储

一个优秀的存储结构应满足:

- 占用空间小
- 构建时间复杂度低
- 支持快速的查询

	邻接矩阵	邻接表
占用空间:	O(n^2)	O(n + m)
构建时间:	O(n^2 + m)	O(n + m)
查询相邻边:	O(n)	O(deg(u))
遍历全图:	O(n^2)	O(n + m)

平面图需要支持哪些查询操作?

- 对于一个点，查询其相邻的边——邻接表
- 对于一个域，查询其周围的点和边

  为每个域创建一个列表，存储其边界上的点和边（1）

- 对于一个域，查询其相邻的域

  为平面图创建对偶图（2）

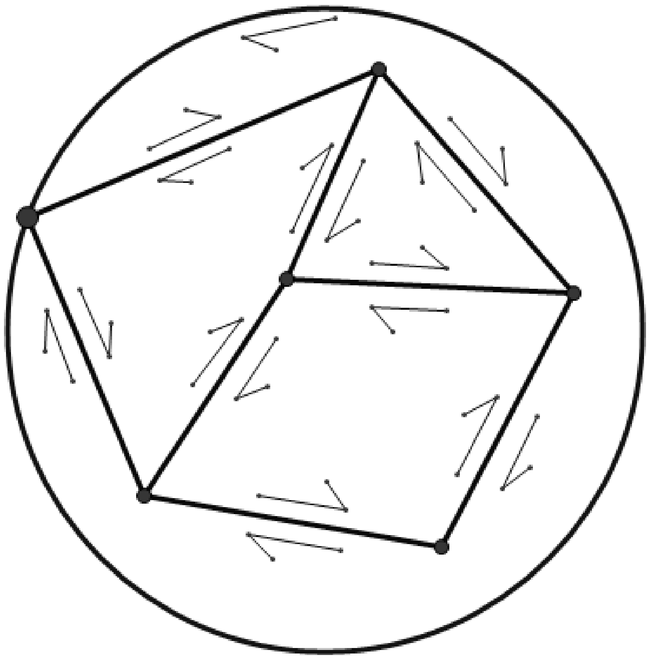
构建结构（1）：

步骤0：化无界为有界

步骤1：将每条边拆成两条有向边，需要有指针指向对方

步骤2：将每个点发出的有向边按照顺时针排序

步骤3：任取一条未访问的边出发，在每个节点处选择顺时针第一条边，直到返回出发的边为止。

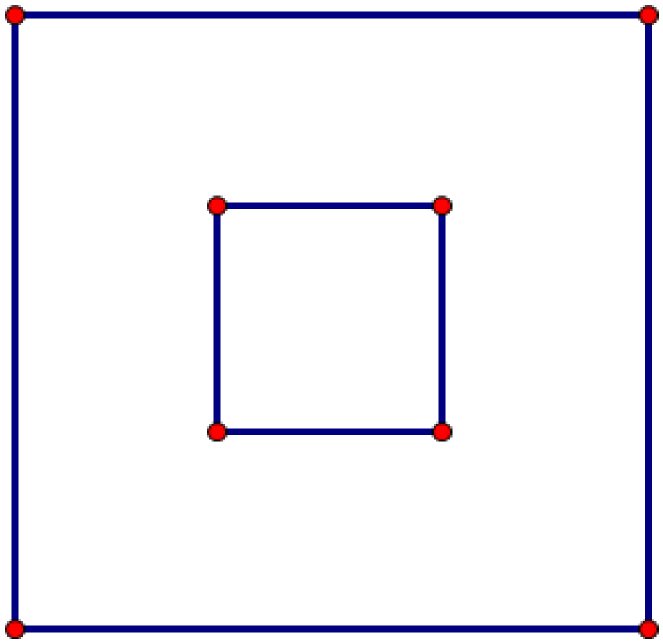


构建结构（2）：

- 记录每条边从属于哪个域
- 为每个域在对偶图中创造一个点
- 为每条边在对偶图中创造一条边

对于不连通的图，应如何处理?

　　通过这个算法，无法得到图中的圆环域，也无法得到这个域与中心方块域相邻的关系。

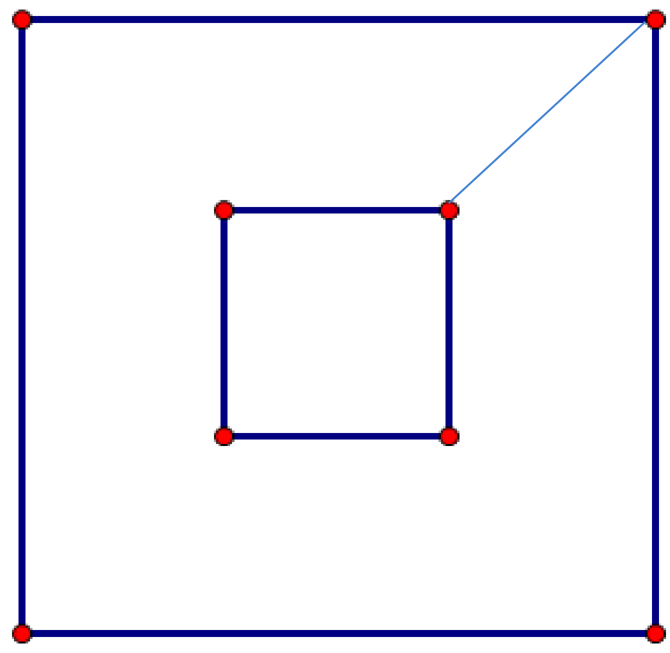


对于不连通的图，应如何处理?

添加这条边之后:

平面图中域的数量不变，即对偶图中点的数量不变。

平面图中边的数量增加了，但对应的对偶图中只增加了一个自环。



对偶图的应用：

平面图最大流

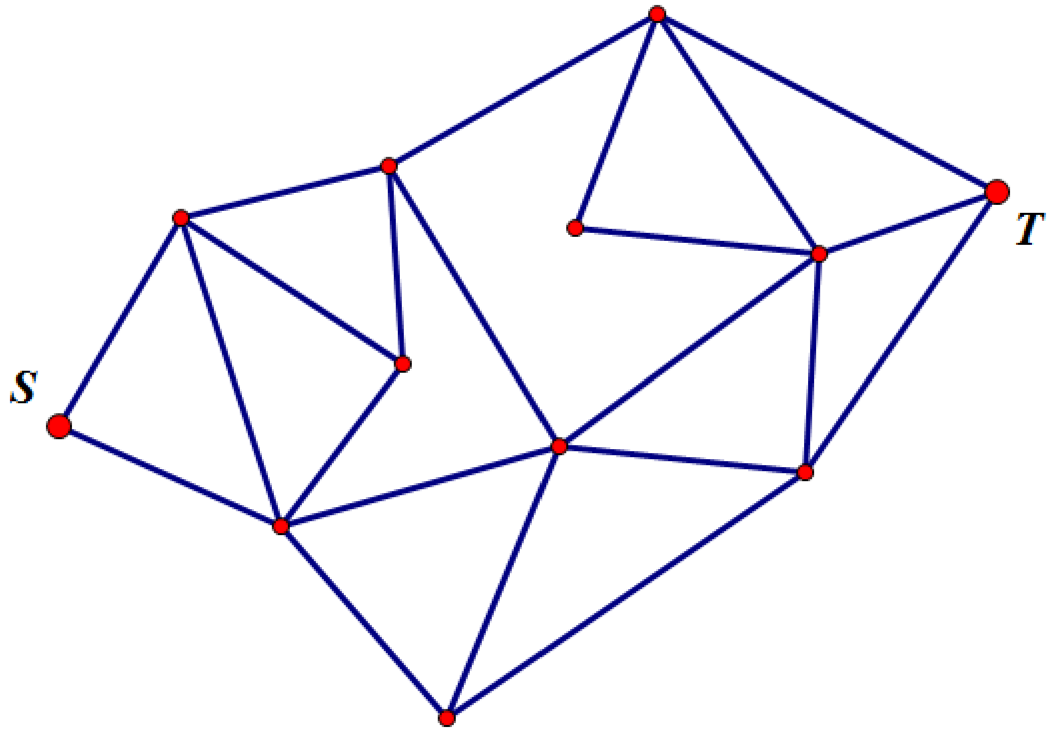
↓

平面图最小割

↓

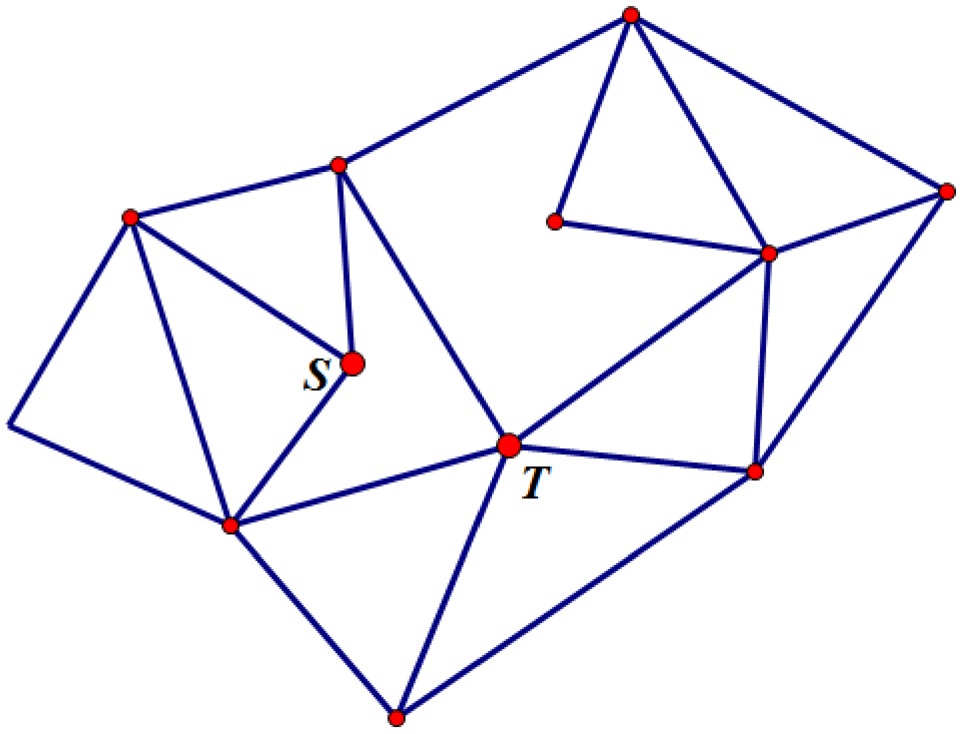
对偶图最短路

Case 1:

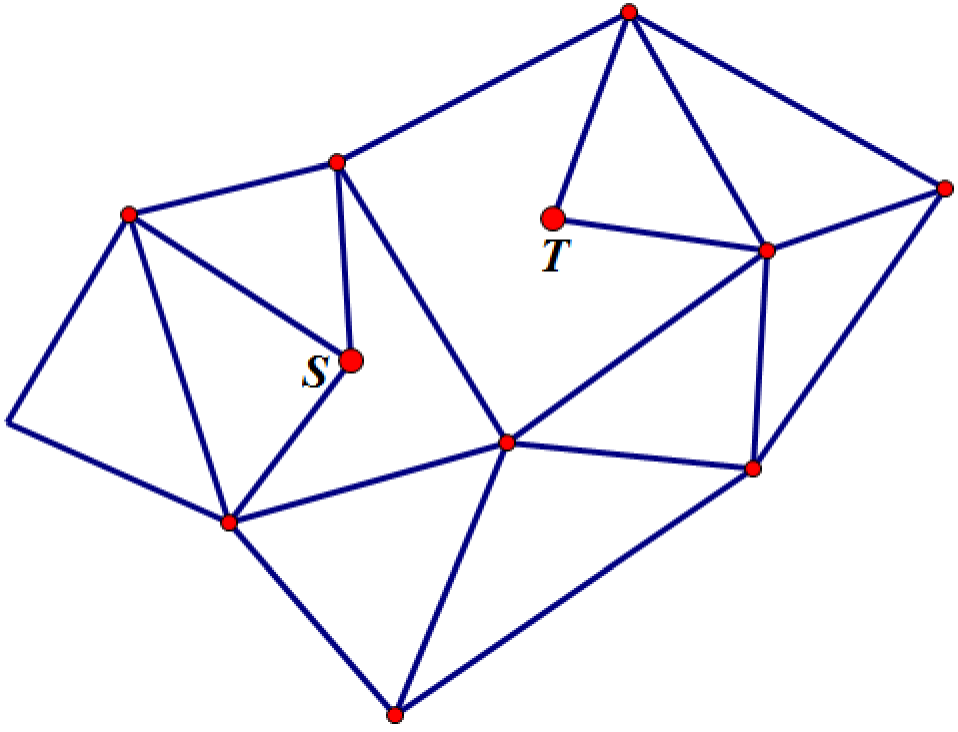


- 无向图
- 有向图

Case 2 :



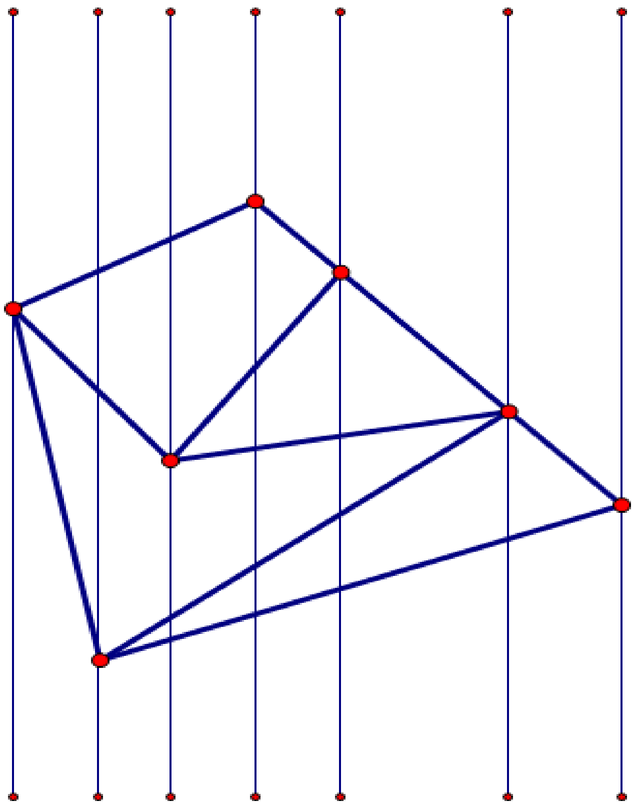
Case 3 :



四、梯形图及其查找结构

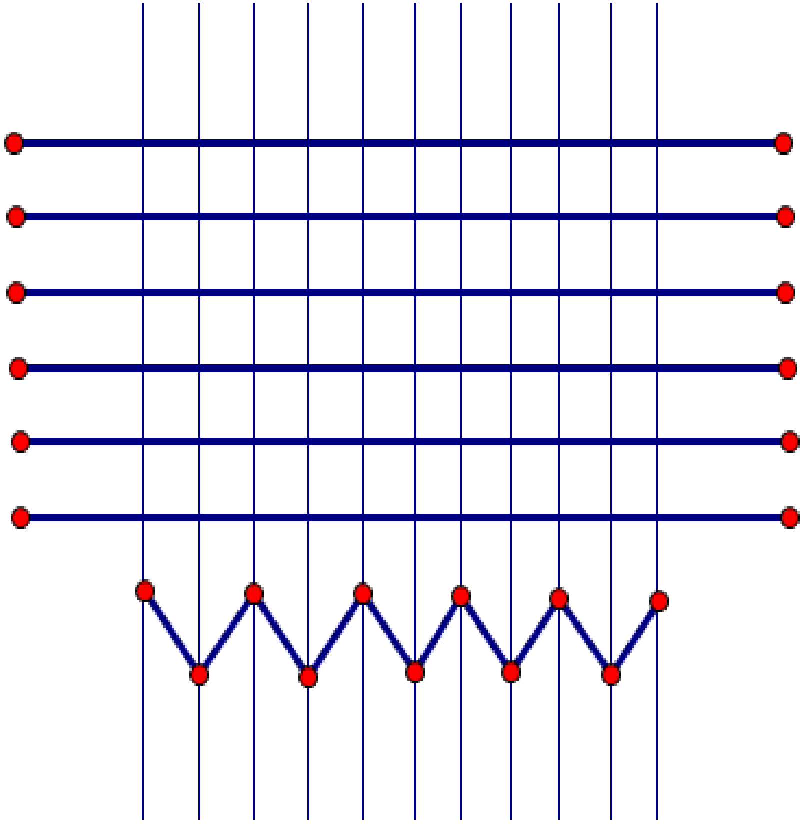
查询一个点在哪个域，在上述的结构（1）、结构（2）中仍然需要O(n)的时间。为了使查询操作更快，我们需要更强大的结构来维护平面图。

一种思路是，引入若干条垂直竖线将平面图切割：



查询可做到O(logn)

构建需要O(n^2*logn)



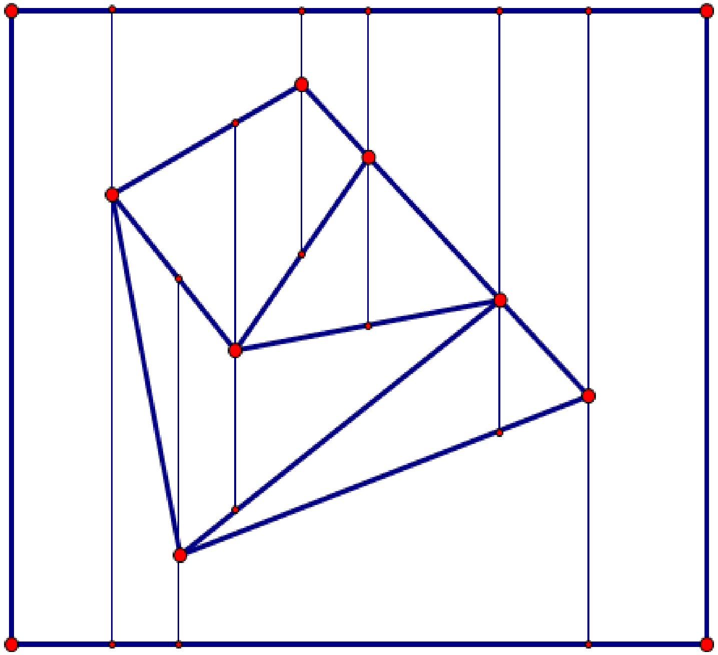
域的数量会增加至O(n^2)

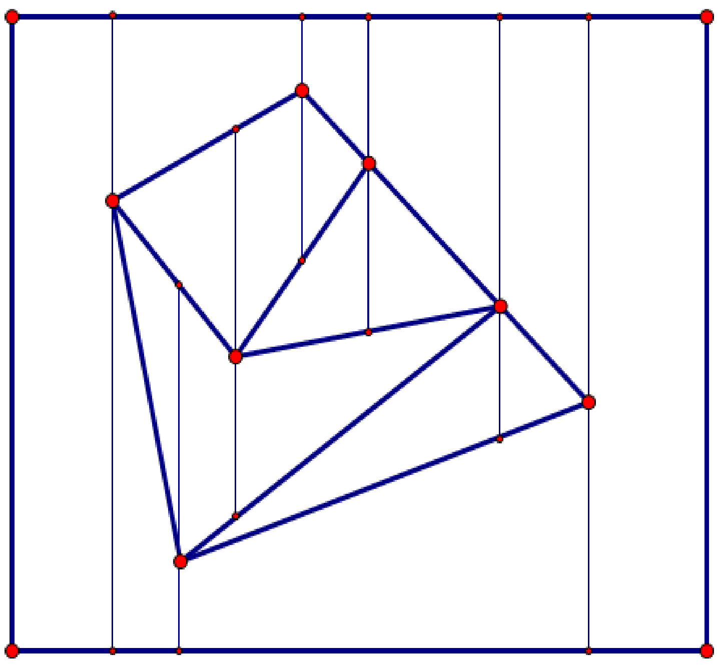
尝试改进划分方式

另一种思路是，每个点沿竖直方向发出射线，碰到其他点或线段即停止

在这个结构中，每个域均是梯形（三角形可看做是退化的梯形）。域的数量是O(n)的。如何证明?

基本假设：所有点横坐标互异





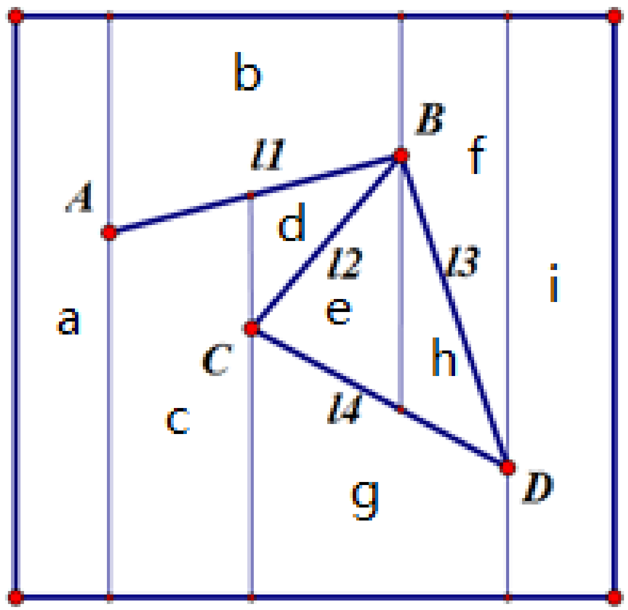
证明：

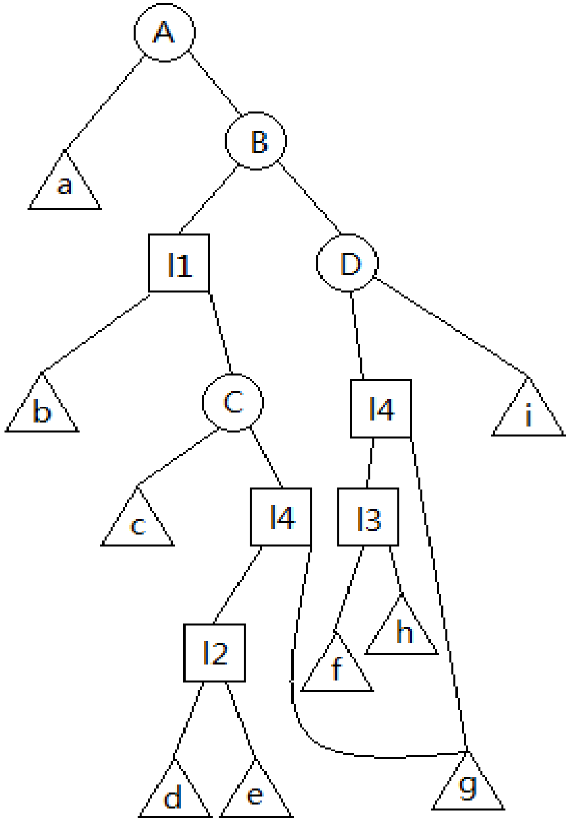
每个点引出两条射线，至多增加两个节点。故节点总数不超过3*n。

除节点处无交点，仍然是平面图。

故域的数量仍是O(n)。

除了这张梯形图外，我们还需要一个查找结构：





五、构造梯形图查找结构
——随机增量法

思考题:

任给平面上n个节点，请在O(n)时间内找到一个尽可能小的圆，使得所有的点都在圆内或圆上

- 将i个点随机排列，则第i个点在前i-1个点的轮廓圆外的概率不大于3/i。

任务：在O(n)的时间内，为n个点构造轮廓圆

- 将n个点的顺序随机打乱，时间复杂度O(n)

- 已得到前i-1个点的轮廓圆，判断第i个点是否在该圆内：

  (i-3)/i的概率：在圆内或圆上，可直接忽略该点

  3/i的概率：在圆外，需要重新构造前i个点的轮廓圆

- 若可在O(i)的时间内为前i个点构造轮廓圆，则可在O(1)时间内，将前i-1个点的轮廓圆扩展为前i个点的轮廓圆

子任务：在O(i)时间内，为前i个点构造轮廓圆。

已知其中i号点一定在圆上！称其为点a。

- 将前i-1个点的顺序随机打乱，时间复杂度O(i)
- 已得到点a与前j-1个点的轮廓圆，判断第j个点是否在该圆内：

  (j-2)/j的概率：在圆内或圆上，可直接忽略该点

  2/j的概率：在圆外，需要重新构造前j个点的轮廓圆

- 若可在O(j)的时间内为前j个点与点a构造轮廓圆，则

  可在O(1)时间内，将点a与前j-1个点的轮廓圆扩展为点

  a与前j个点的轮廓圆

子任务：在O(j)时间内，为前j个点构造轮廓圆。

已知其中点a与点b（j号点）一定在圆上！

一次循环即可搞定！

构造梯形图及其查找结构：

最初，梯形图中没有任何点和边，只有一个梯形域，对应的查找结构中只有一个三角节点。

以一个随机的顺序向梯形图中插入平面图的边，同时维护梯形图及其查找结构。

整个过程中，先不要过于纠结时间复杂度的问题。

插入一条新的线段时：

1、查询待插入线段的两个端点在原梯形图的哪个域

这就是我们的查询操作，可直接套用！

- 若该点在某条竖直的射线上，算哪个域？

线段左端点往右算，右端点往左算

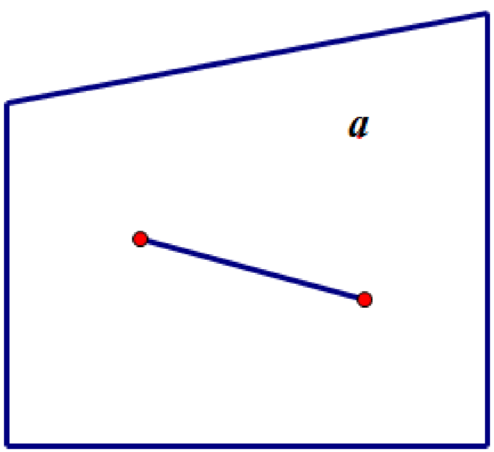
- 若该点在某条边上，算哪个域？

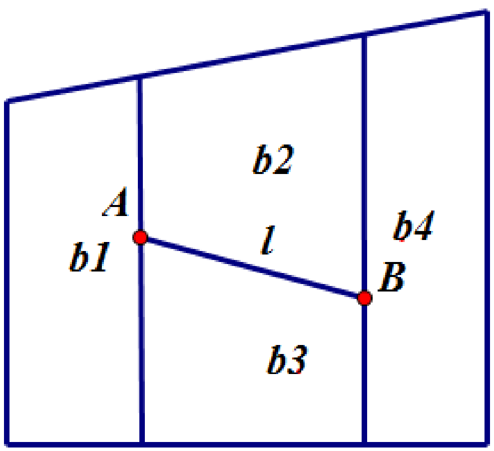
比较斜率

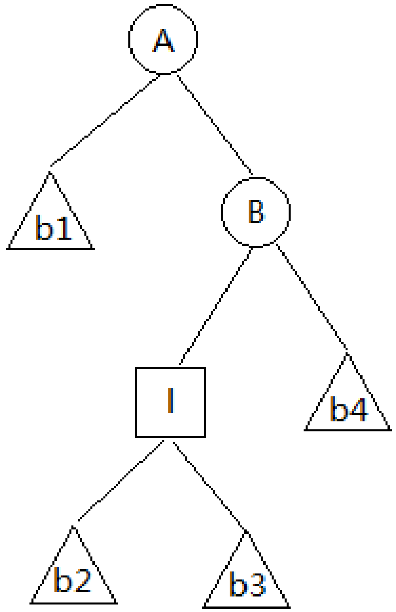
算在需要切割的域中！

2、插入线段，维护梯形图及查找结构

情况1：待插入线段的两个端点在同一个梯形域中：

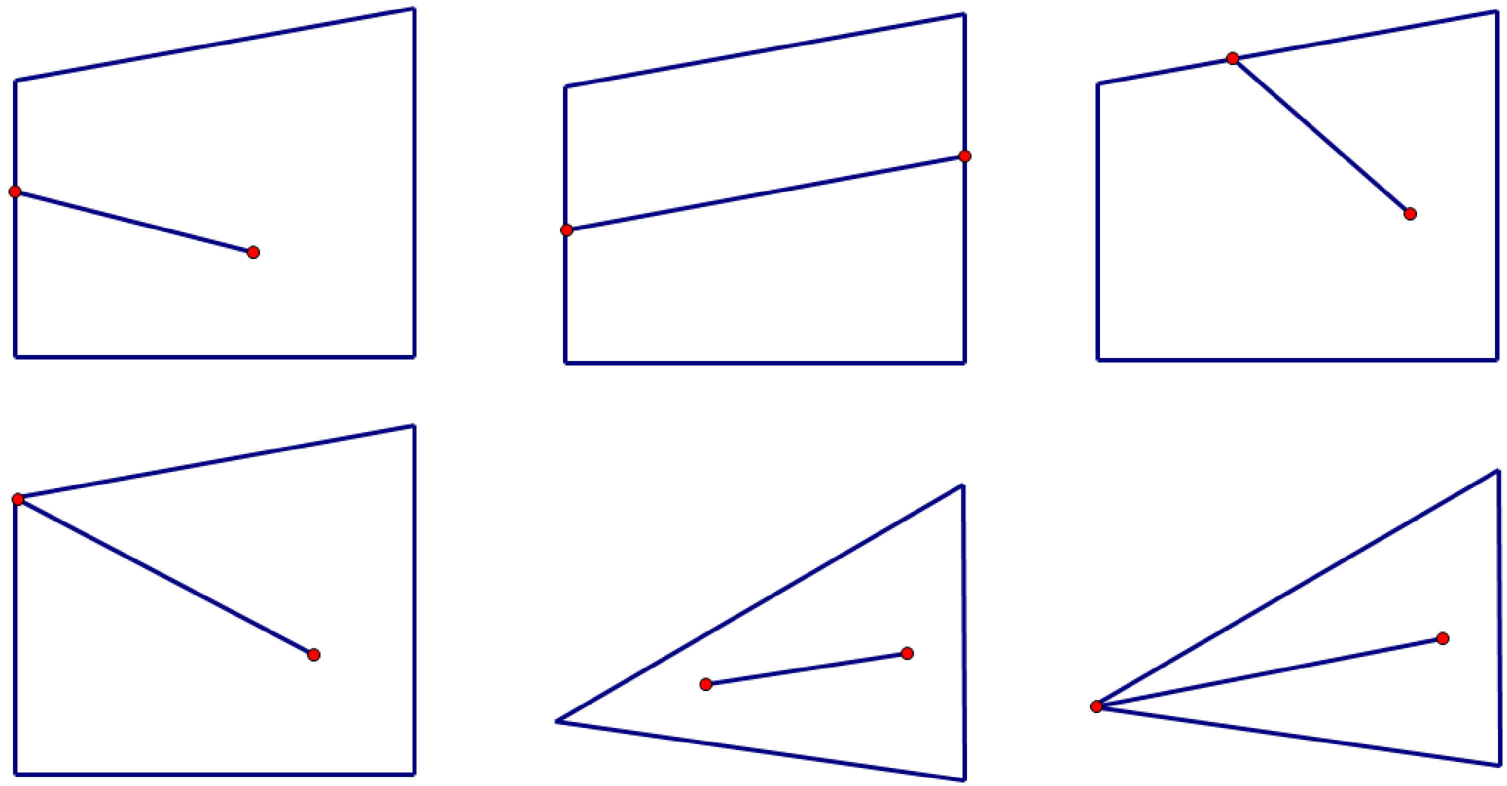






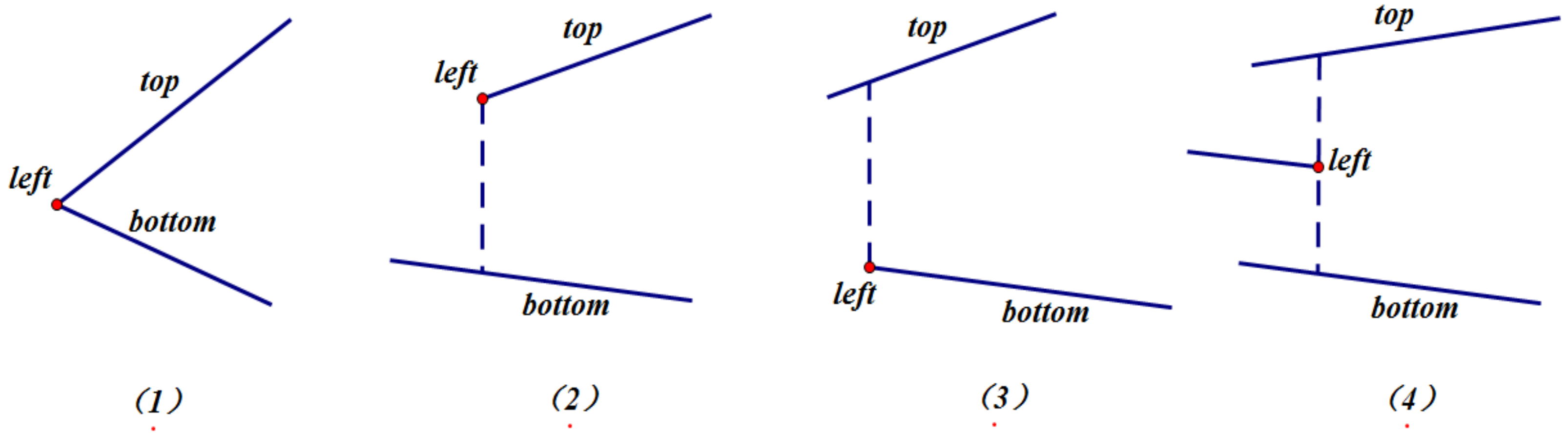
情况1中：域增加O(1)个，引入新节点O(1)个，时间复杂度O(1)。

对应的一些边界情况如何处理?



情况2：待插入线段横穿了若干个梯形域

为了找到这些梯形，我们引入邻居的概念：



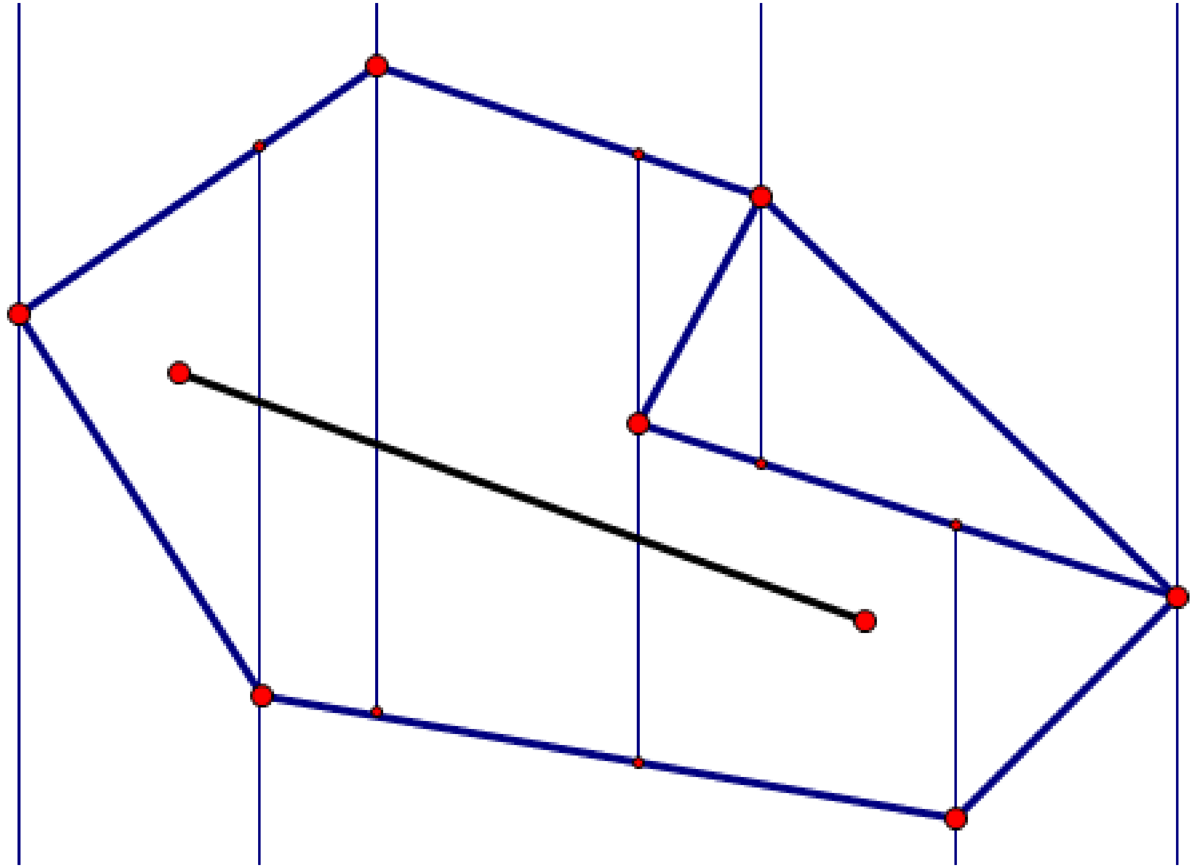
（1）没有左邻居

（2）有左下邻居

（3）有左上邻居

（4）有左上、左下邻居

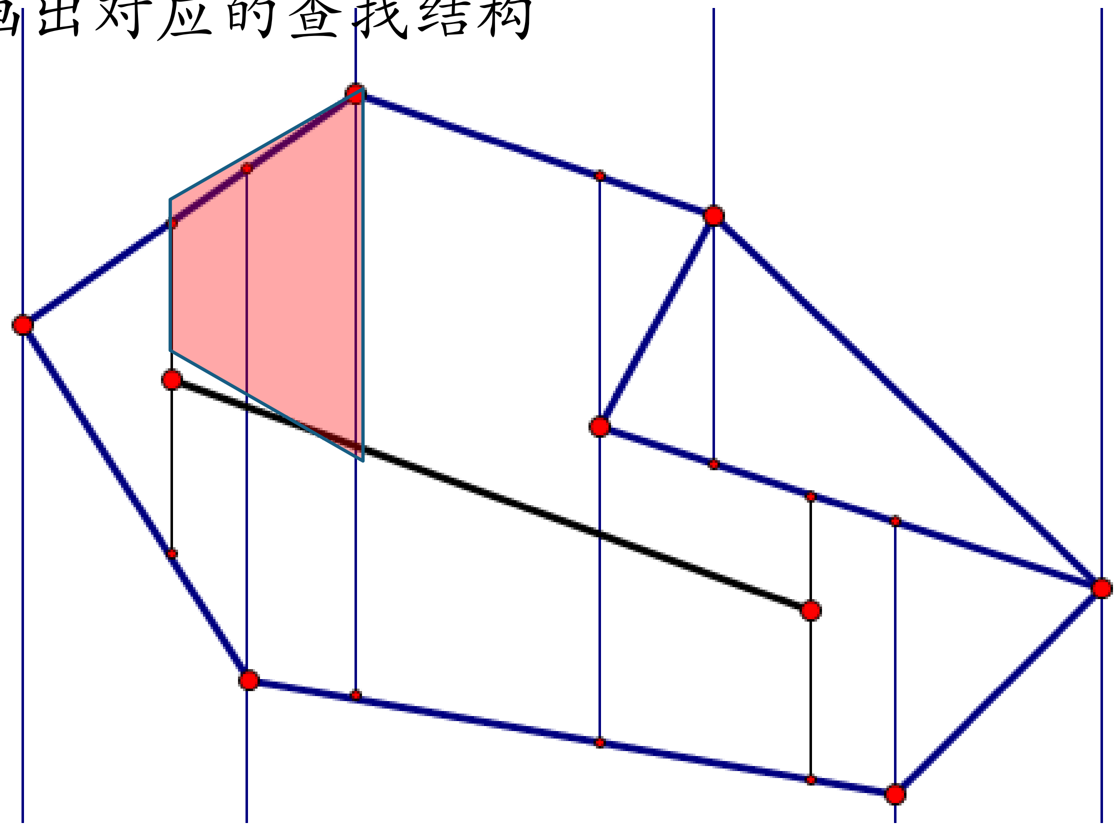
通过邻居的信息，可以顺藤摸瓜找到所有涉及到的梯形：



对于两侧的梯形，引入新的射线，一分为三

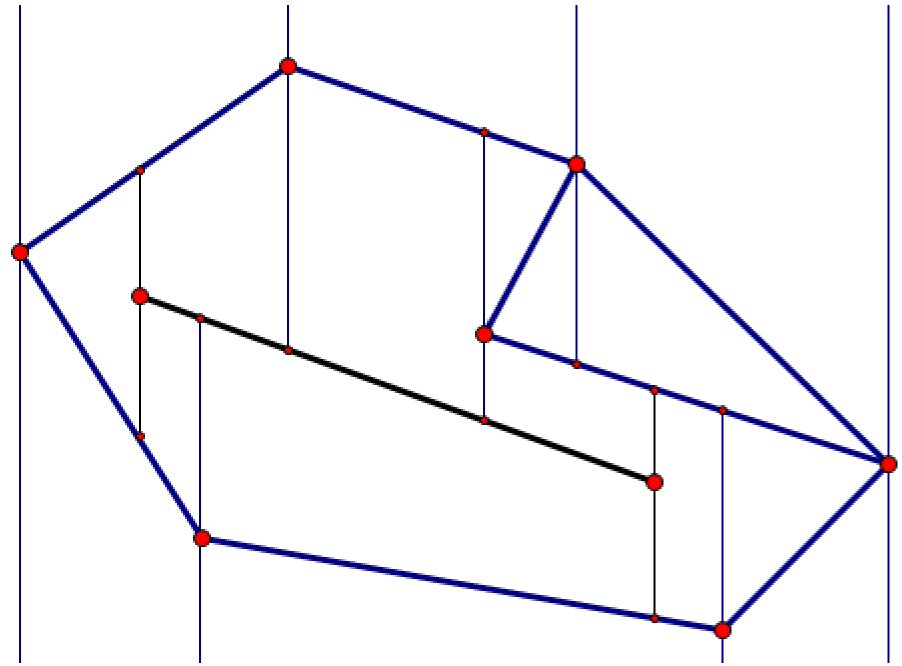
对于中间的梯形，一分为二

请画出对应的查找结构



问题：这是一个梯形，应该合并！如何合并？

需要合并的矩形公用相同的上边/下边。



合并后，梯形域的数量增加了常数个。如何证明?

证明:

对于相邻的两个梯形域，他们共用一条边。

插入前，有k个梯形域，则有k+1条边。

合并后，除了两端分别多出一个域外，每条边最多对应一个梯形域。

故合并后最多有k+3个梯形域。

情况2中：域增加O(1)个，引入新节点O(k)个，时间复杂度O(k)。

算法回顾：

1、寻找两个端点位置，套用查询操作。

时间复杂度O(L)。L为查找路径长度。

2、修改平面图及其查找结构：

顺藤摸瓜找到所有梯形域→剖分梯形构造查找结构→

合并梯形→维护边界以及邻居信息

情况1：域增加O(1)个，引入新节点O(1)个，时间复杂度O(1)。

情况2：域增加O(1)个，引入新节点O(k)个，时间复杂度O(k)。

*复杂度的分析

*处理退化情况:

去掉基本假设“所有点横坐标互异”，该如何处理?

将图中所有点绕原点旋转一个角度，与原图等价!

同时，待查询点也要随之旋转。

只要旋转的角度足够小，一定保证没有两个点横坐标相同!

但是，这一方法会引入精度误差。如何处理?

剪切变换法：

(x, y) → (x+ay, y)　其中a足够小

事实上，我们并不需要确定a的值。在存储点的信息时，我们存储(x, y)，但我们必须清楚这个点的坐标实际上是(x+ay, y)。在处理下面的问题时，也要随之做一些改动：

- 过点P做一条竖直垂线，判断点Q在其左侧、右侧或线上
- 对于线段P_1P_2，判断点Q在其上方、下方或线上

- 过点P做一条竖直垂线，判断点Q在其左侧、右侧或线上

对于点P(X_p+aY_p, Y_p)和点Q(X_q+aY_q, Y_q):

若X_p != X_q，（由于a足够小）已经可以比较出他们横坐标的大小关系。

若X_p = X_q，则Y_p与Y_q的大小关系，就是两个点横坐标的大小关系。

所以，我们把x当做第一关键字，y当做第二关键字进行两次比较，就可判断出点Q的位置。

- 对于线段P1P2，判断点Q在其上方、下方或线上

  剪切变换前后，点与线的位置关系不变!

  只需对未经过剪切变换的Q与P1P2作比较即可。

行胜于言

谢谢大家！

钱桥 qianqiaodecember29@126.com