A sweep line algorithm for computing Voronoi diagram of circle sites

穆黎森、靳力、黄畅

2004.1

内容

- 引言
- 背景和前人的工作
- 算法和原理
- 系统设计和数据结构
- 结果和讨论，复杂度分析

引言

问题的定义：

- arc：给定平面上一组circle（相交或不相交），根据它们的交点划分为一组互不相交的arc（端点除外），每一个arc定义为一个site；
- cell：一个site的cell，定义为平面上到该site的距离小于到其他site的距离的点的集合，其中点到site的距离定义为该点到site（arc）上点的距离的最小值；
- 平面上去掉所有cell之后剩下的部分定义为circle sites的Voronoi diagram，此时的Voronoi diagram由一些双曲线（段）和椭圆（弧）组成。

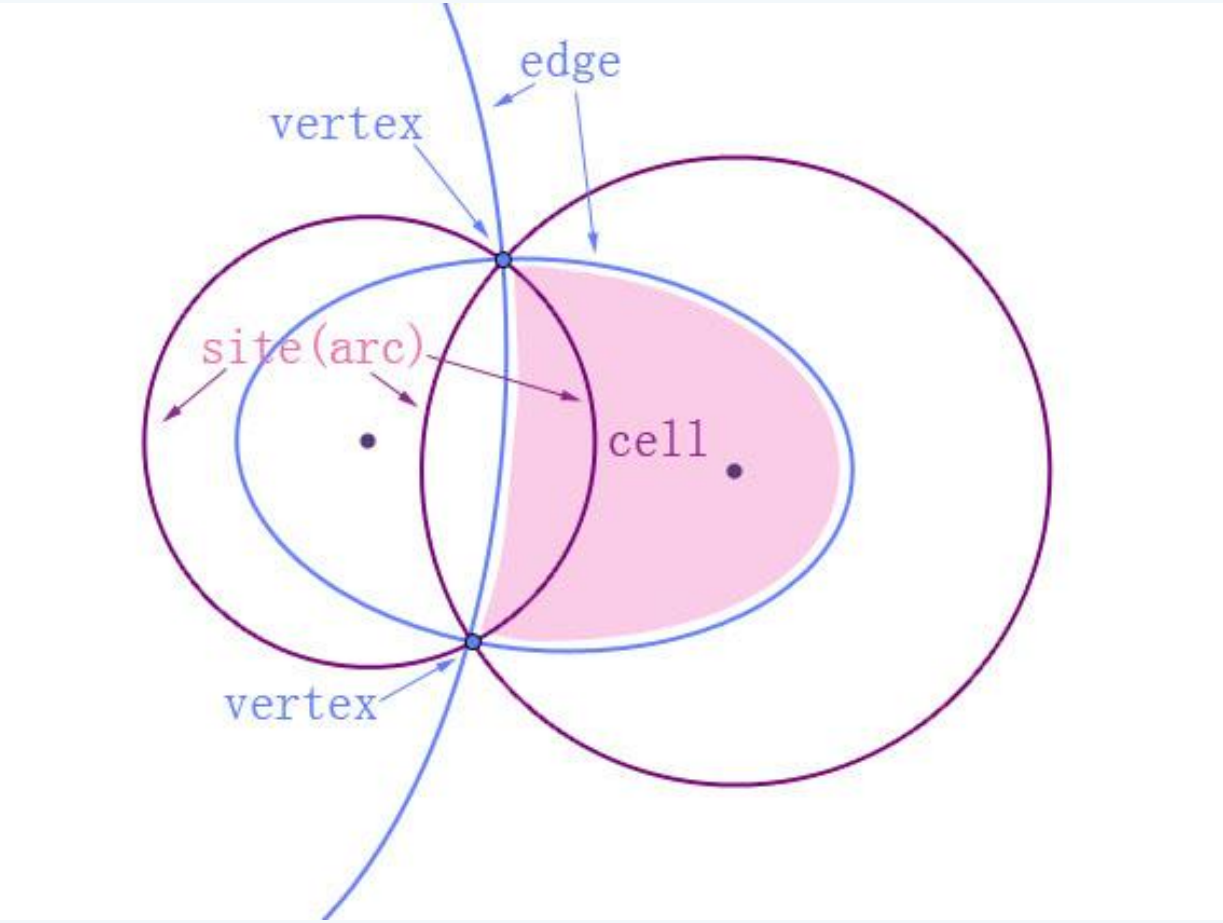
引言

Voronoi diagram for circle sites的边（edge）和顶点（vertex）：

- edge，定义为Voronoi diagram上同一半支双曲线（段）或同一椭圆（弧）上连续的最长的一段。
- vertex，定义为edge的交点。

- 我们的算法就是要从给出的一组circle，得出circle的Voronoi diagram的
- 一种类DCEL（Double Connected Edge List）的表示。

引言



内容

- 引言
- 背景和前人的工作
- 算法和原理
- 系统设计和数据结构
- 结果和讨论，复杂度分析

背景和前人的工作

- Voronoi diagram定义中的要件：
  - distance定义：欧氏距离、加权、乘系数等等。
  - site定义：点、线段、圆等等。

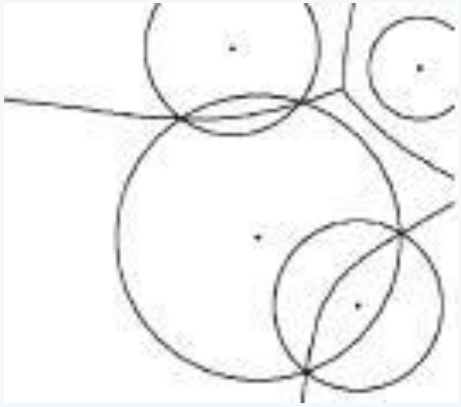
- 前人研究的Voronoi diagram基本上可以按照上述两个要件进行分类。

背景和前人的工作

- Fortune的扫描线算法[1]：
  - point sites：site为点，distance为欧氏距离。
  - line segment sites：site为直线段，distance为欧氏距离。
  - weighted point sites：site为点，distance为欧氏距离加上一个非负数。
  - Fortune扫描线算法的时间复杂度O(nlogn)，空间复杂度O(n)。

背景和前人的工作

- Kim.的Voronoi diagram of circle set算法[2,3]：
  - site为圆上的点，distance为欧氏距离
  - 首先计算以圆心为point set的点的Voronoi diagram，其后通过改变Voronoi diagram拓扑等手段生成circle set的Voronoi diagram。
  - 采用莫比乌斯变换的方法求三圆的公切圆。
  - 算法复杂度为 $O(n^2)$。
  - 对于圆相交的情况的处理并不完备。



内容

- 引言
- 背景和前人的工作
- 算法和原理
- 系统设计和数据结构
- 结果和讨论，复杂度分析

算法和原理

- 本节介绍我们的扫描线算法的关键问题，分为4部分：
  - 扫描线算法的基本原理和海岸线的形状。
  - Event的定义。
  - circle event的生成和判定。
  - 边的生成和表示。

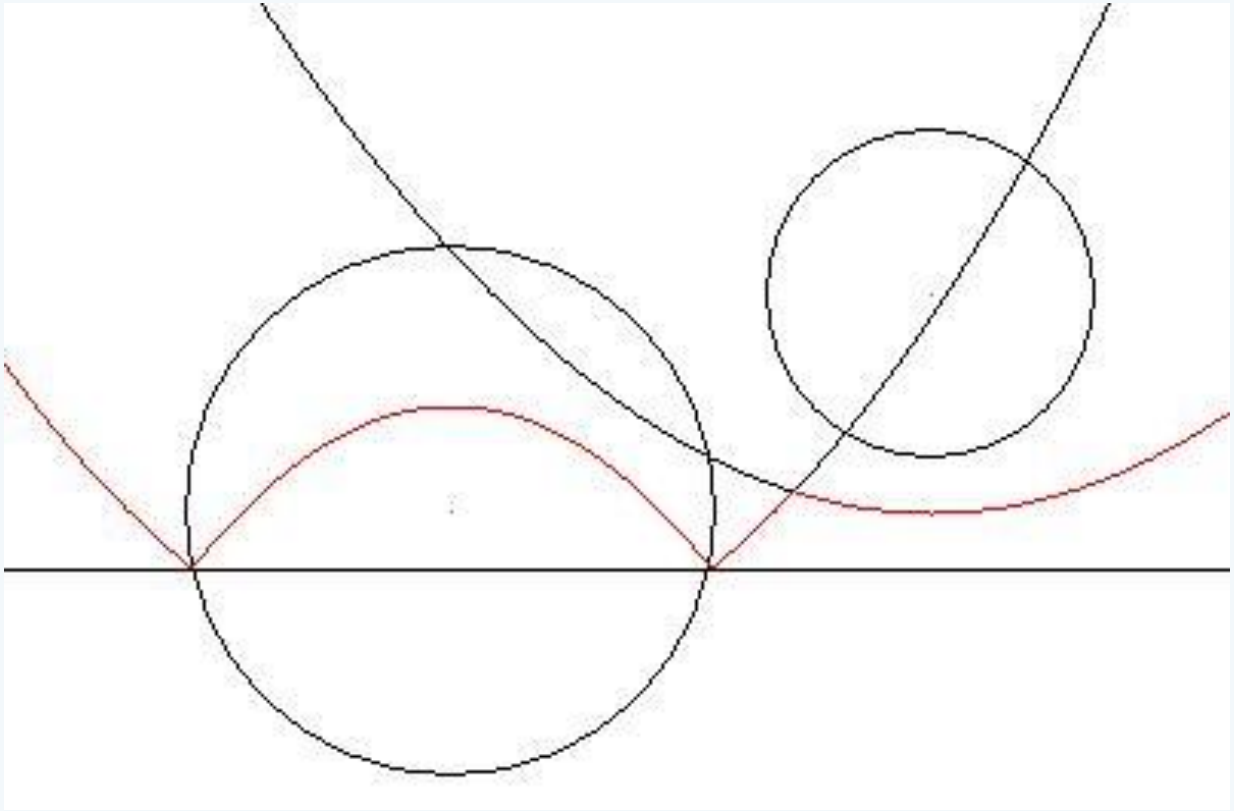
算法的基本原理和海岸线的形状

- 将平面划分为“安全”区域和“待处理”区域两个部分
- 不断将“待处理”的区域变为“安全”的区域
- 两区域的界面：海岸线(beach line)

算法的基本原理和海岸线的形状

- beach line上的点A
  - 到扫描线扫过区域中的site的距离：d1
  - 到扫描线的距离：d2
  - d1=d2
- 两种形状的beach line
  - 圆外：下凸的抛物线
  - 圆内：上凸的抛物线

算法的基本原理和海岸线的形状



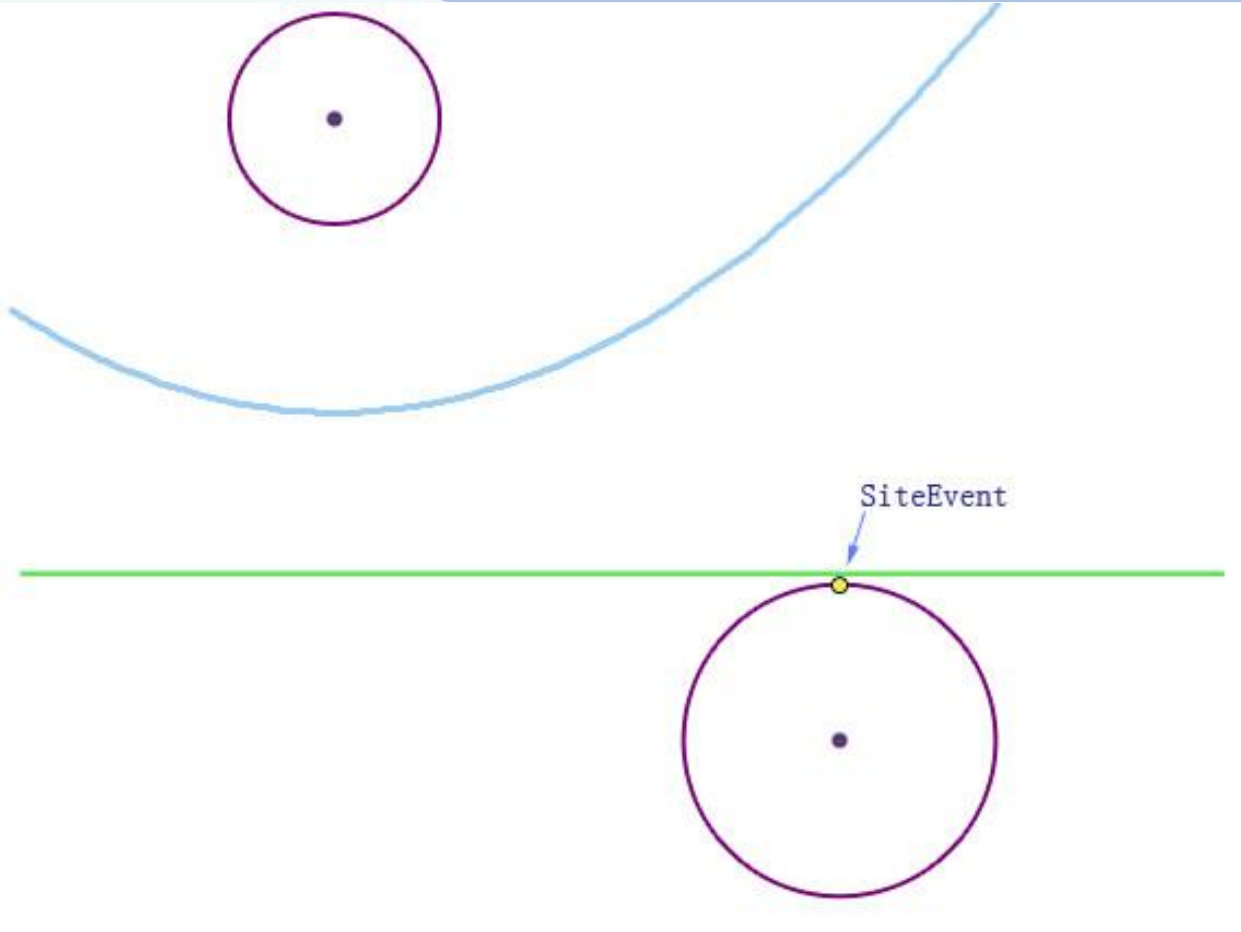
Event的定义

- 共有四种event：
  - site event，指扫描线碰到一个圆时，位置为圆周的最高点。
  - cross event，是指sweep line扫过圆的交点时，位置为交点的坐标。
  - merge event，是指sweep line离开一个圆的时候，位置为圆周的最低点。
  - circle event，指Voronoi diagram中有新的顶点出现时，位置为与扫描线上相邻的三段抛物线对应的三个圆的外切圆的最低点。

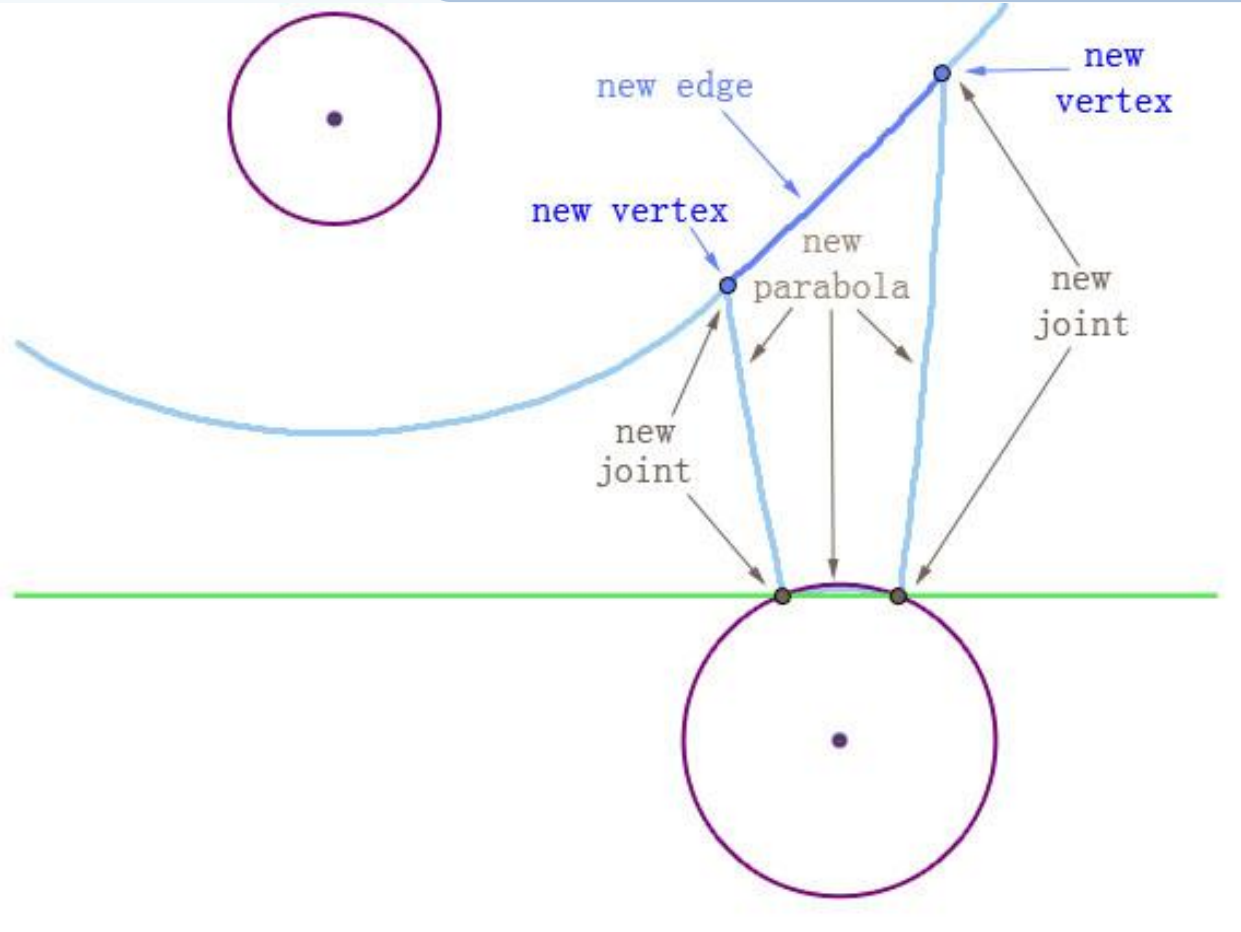
Event的定义

- 整个扫描线算法就是由这些event的处理构成，所以对event的处理是本算法的核心。
- 以下分别说明各event发生时所作的操作。

Event的定义：site event



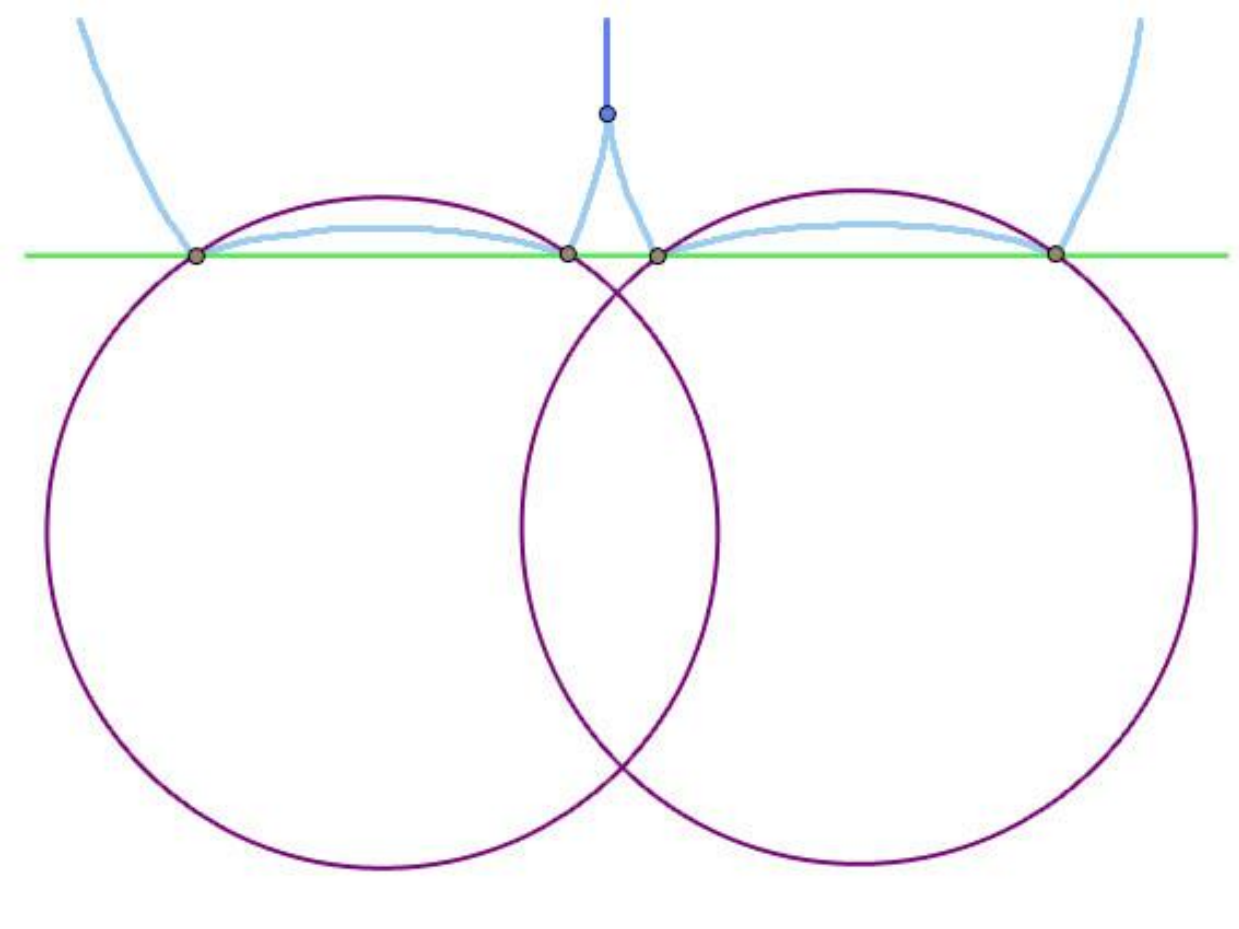
Event的定义：site event



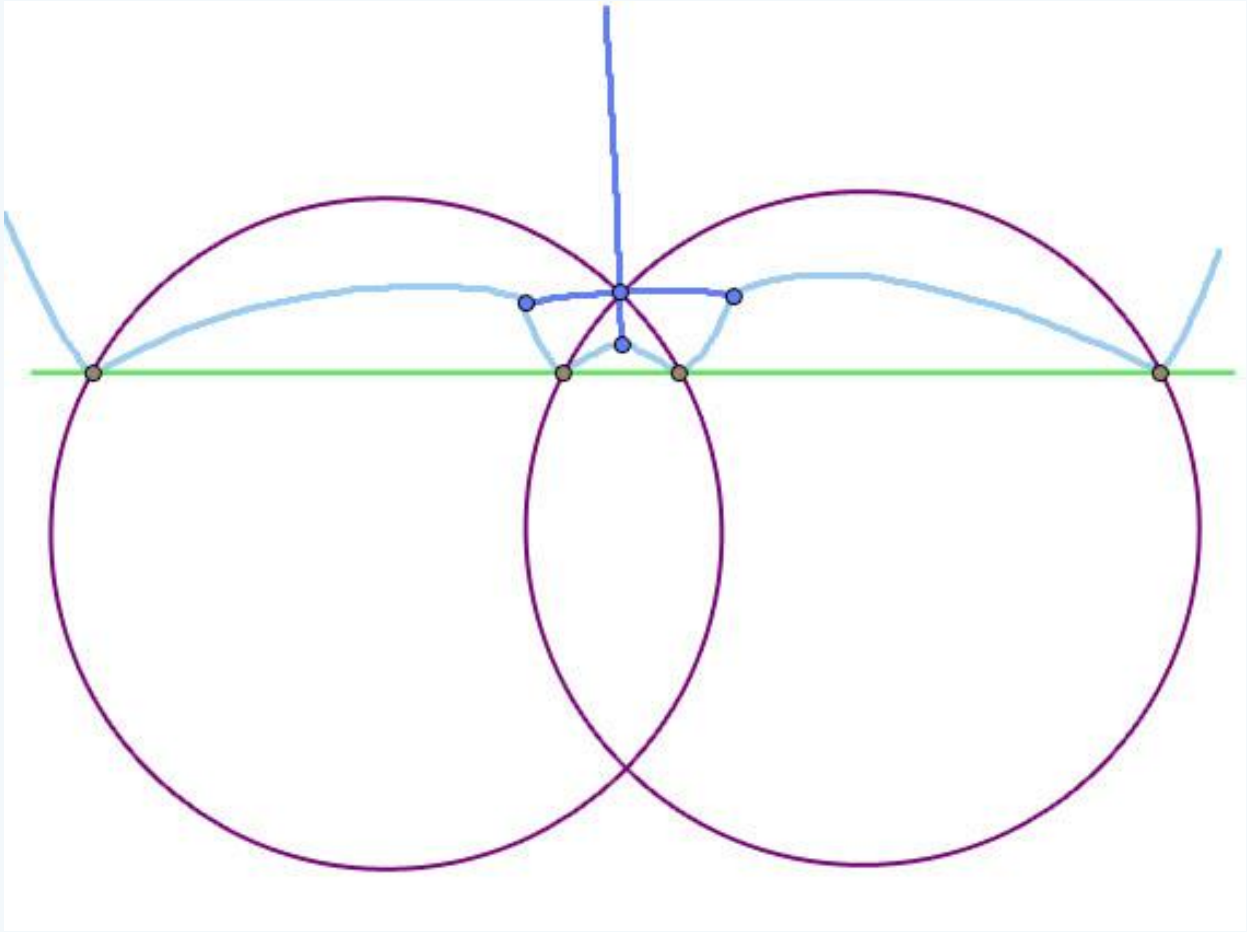
Event的定义：site event

- 计算新的beach line：原来的1段抛物线分为2段；中间插入3段。
- 计算对circle event的影响：删除不复存在的circle event，加入新的circle event。
- 修改Voronoi diagram：生成新的Voronoi diagram的边。

Event的定义： cross event



Event的定义：cross event



Event的定义：cross event

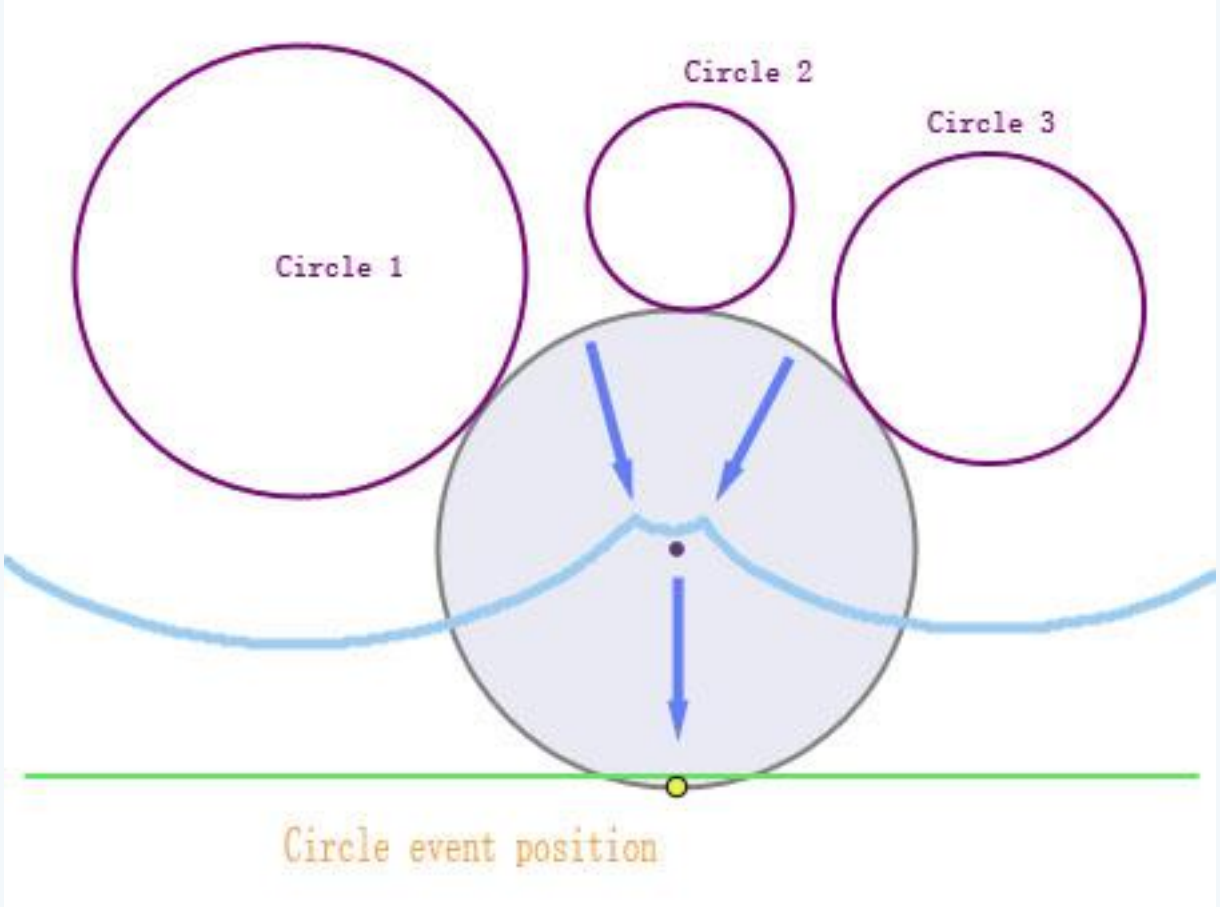
- 计算新的beach line：2段抛物线消失，新的海岸线生成。
- 计算对circle event的影响：删除不复存在的circle event，加入新的circle event。
- 修改Voronoi diagram：终结1条边，新生成3条边。

Event的定义：merge event

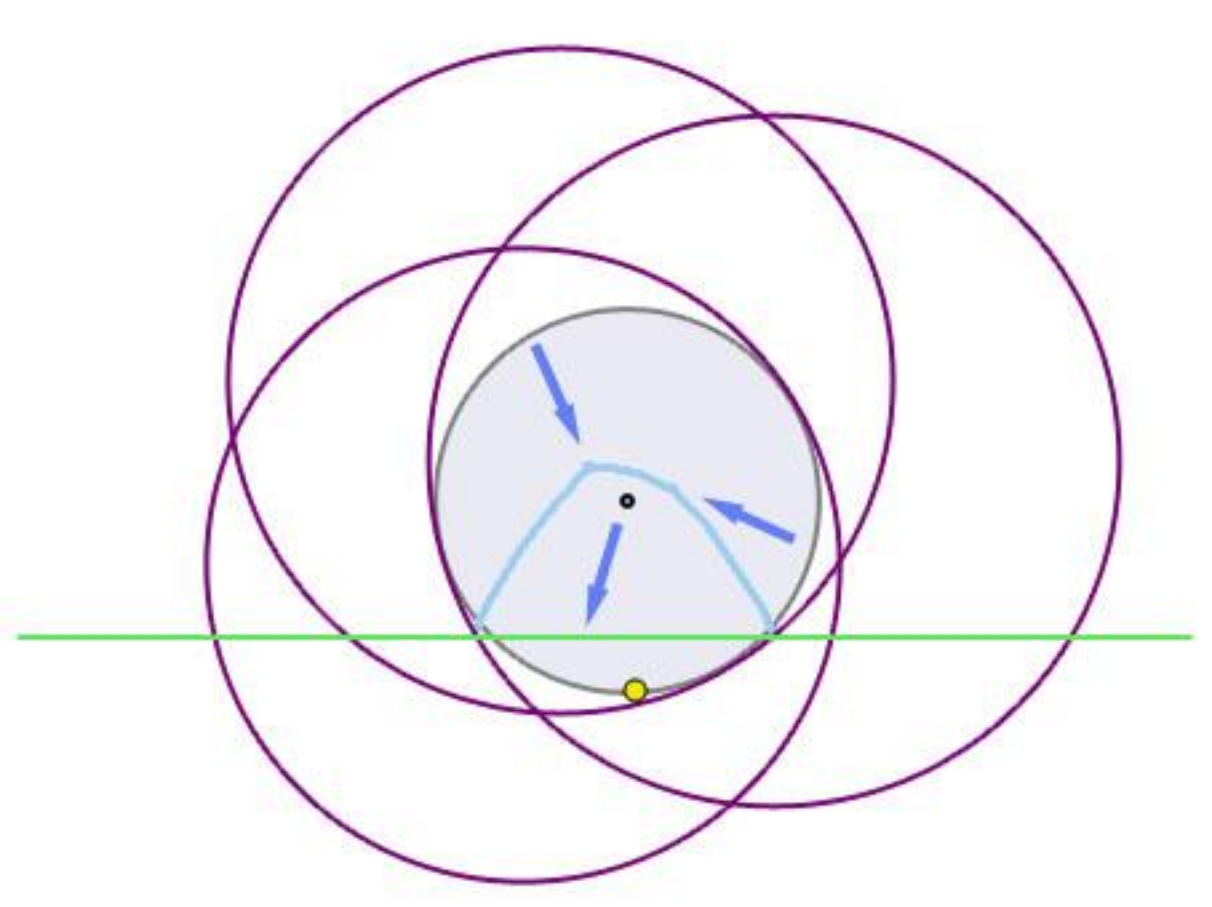
Event的定义：merge event

- 计算新的beach line：圆外的2段抛物线合成1段，圆内的抛物线消失。
- 计算对circle event的影响：删除不复存在的circle event，加入新的circle event。
- 修改Voronoi diagram：2条边合成1条边。

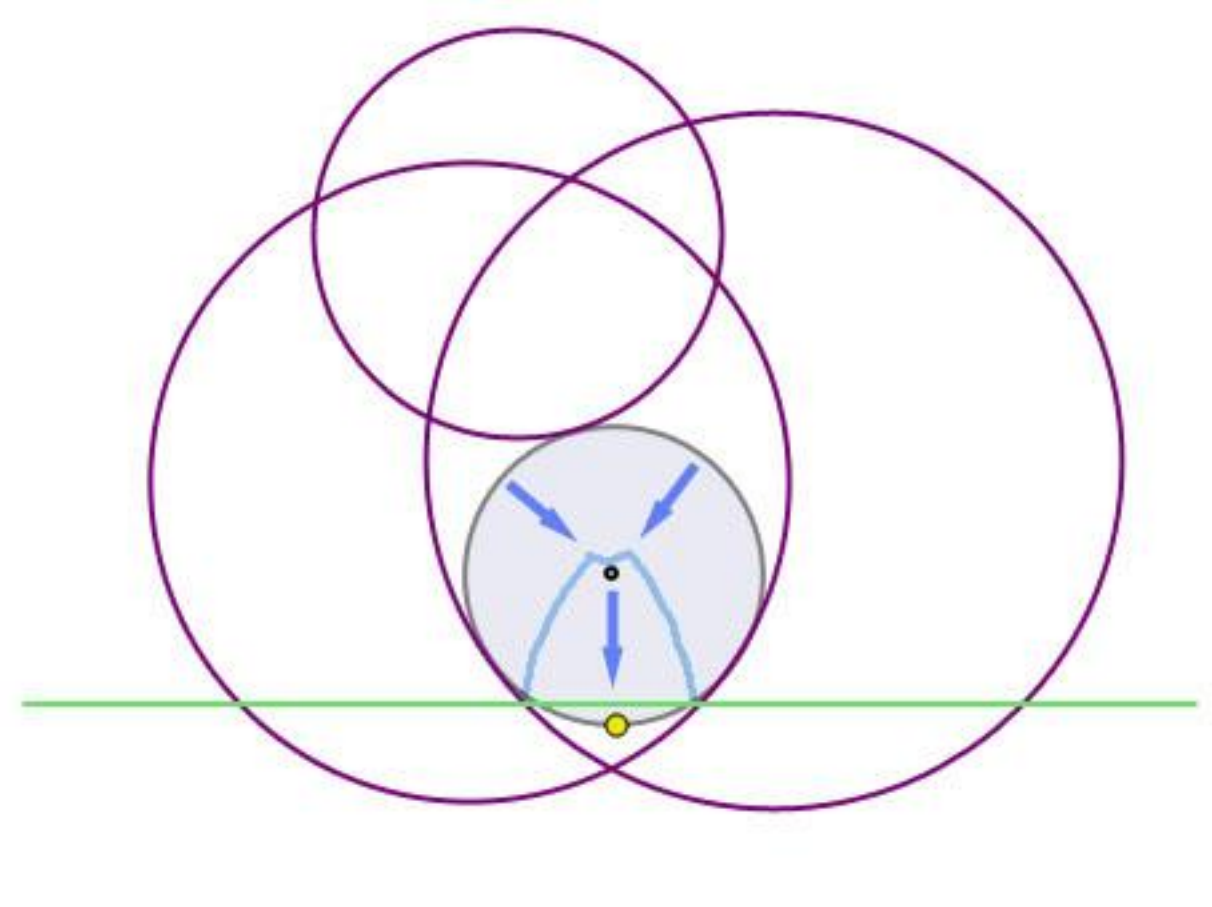
Event的定义： circle event



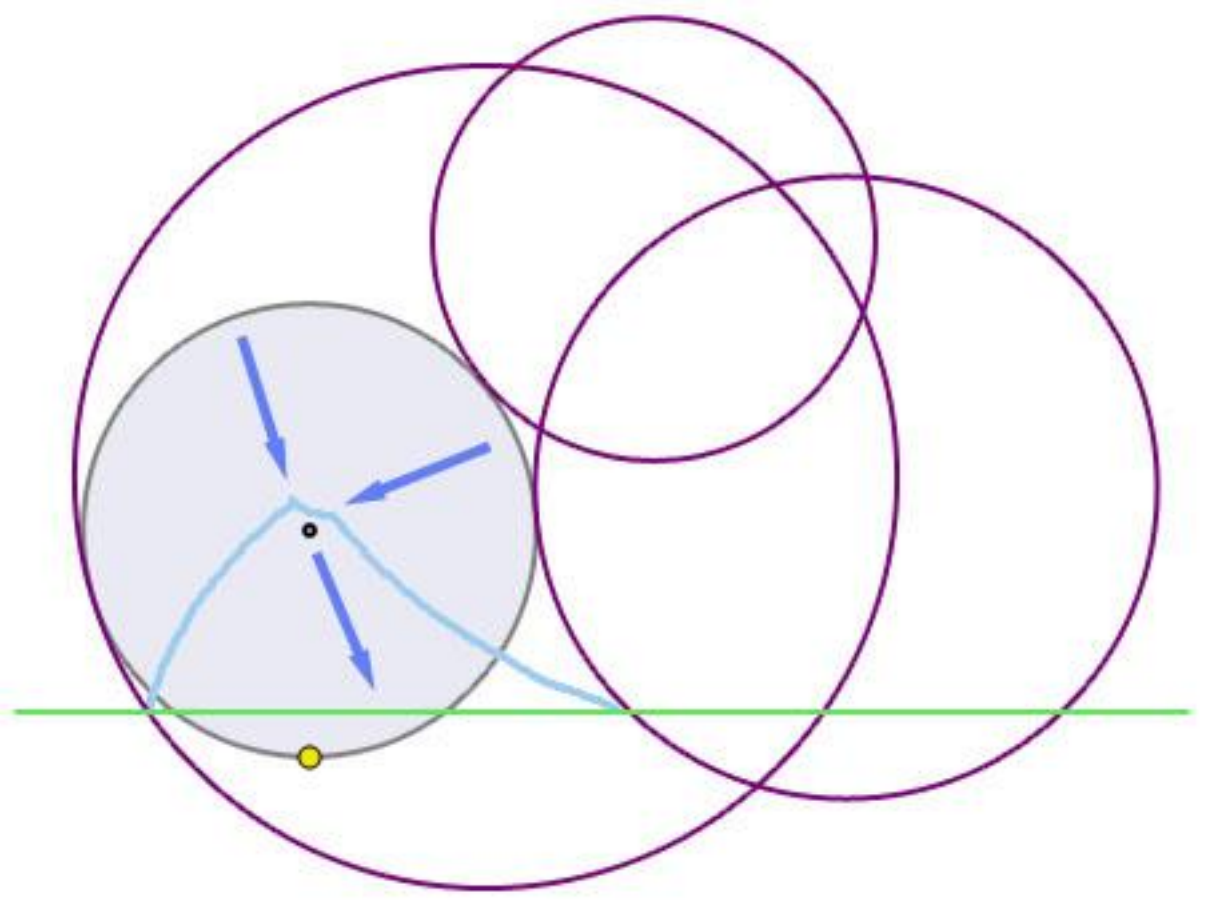
Event的定义： circle event



Event的定义： circle event



Event的定义： circle event



Event的定义：circle event

- 计算新的beach line：3条beach line交于1点，中间的1条消失，两边的保留。
- 计算对circle event的影响：删除不复存在的circle event，加入新的circle event。
- 修改Voronoi diagram：2条边结束，新生成1条边。

circle event的生成和判定

- 求解circle event是扫描线算法的一个关键问题。
- 根本问题是求解三个圆的公切圆。
- 三个圆的公切圆问题 => 两个圆过特定点的公切圆问题。
- 适应各种相切情况，推广Kim[2,3]的方法，使用莫比乌斯变换（Möbius transformation）。

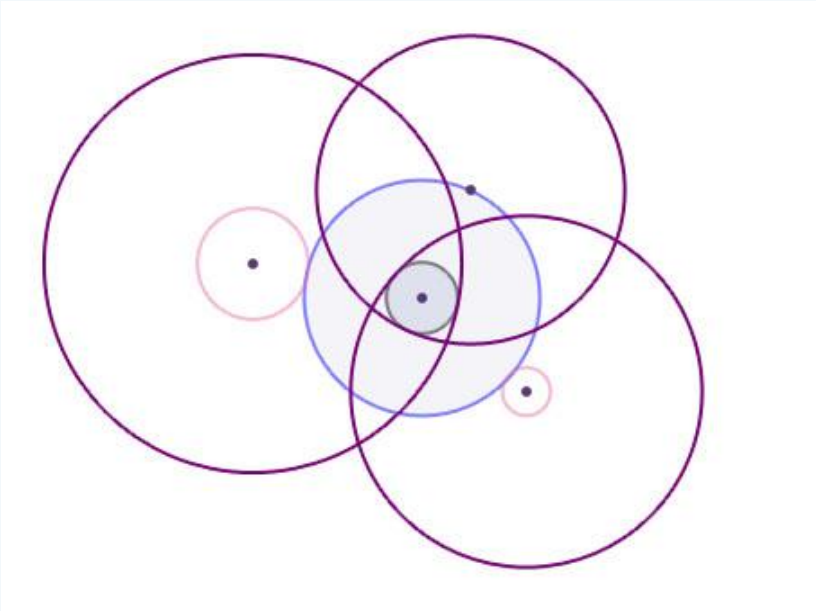
circle event的生成和判定

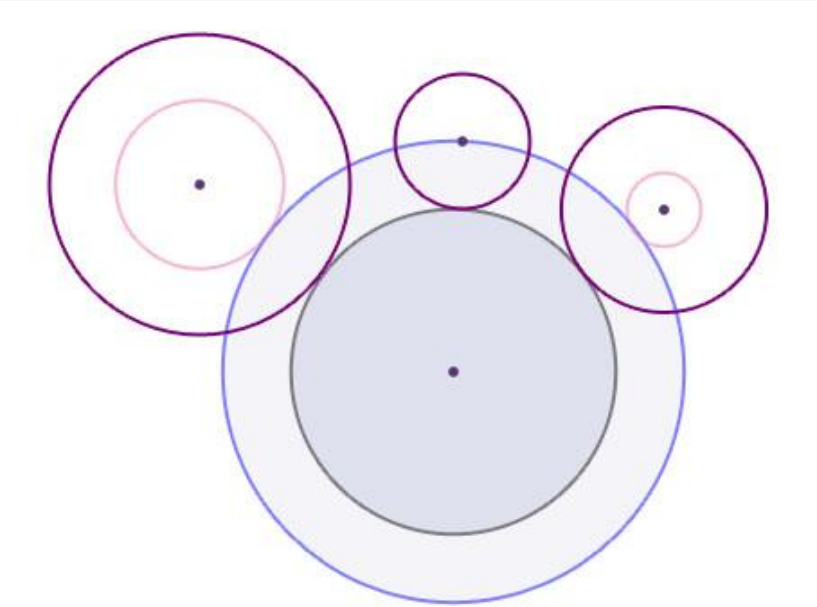
- 问题转换：
  - 三圆公切圆 => 两圆过一定点的公切圆

- 四种相切关系：
  - K=0：全内切
  - K=1：两内切、一外切
  - K=2：一内切、两外切
  - K=3：全外切

circle event的生成和判定

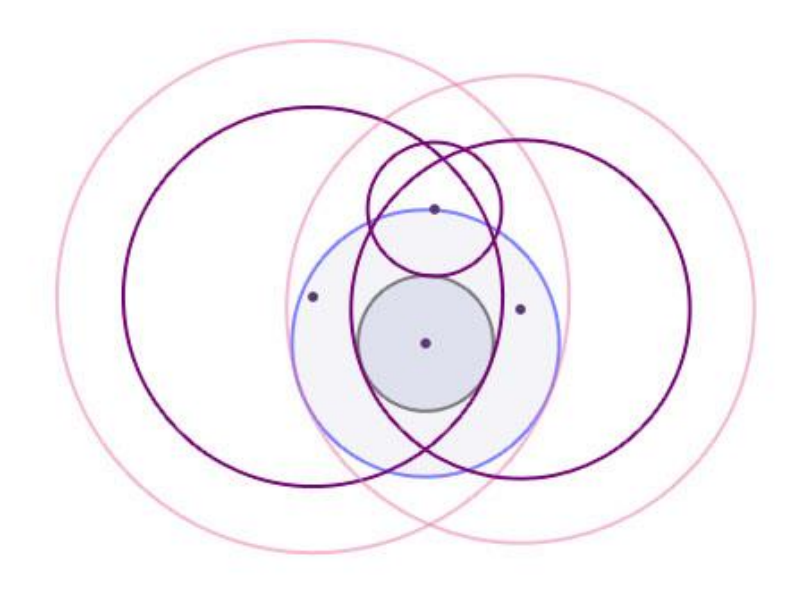
- 全内切和全外切（k=0, 3）：
  - r1=>r1-r3，r2=>r2-r3，（r3为最小圆的半径）

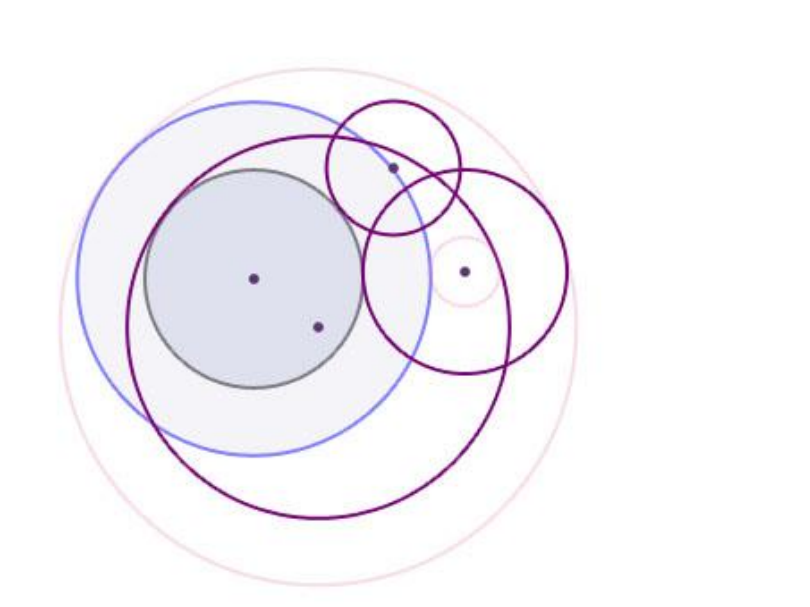




circle event的生成和判定

- 内外切同时存在（k=1,2）：
  - 内切+r3，外切-r3，（r3为外切的圆中小者的半径）





circle event的生成和判定

- 莫比乌斯变换：

$$w = \frac{1}{z - z_0}$$

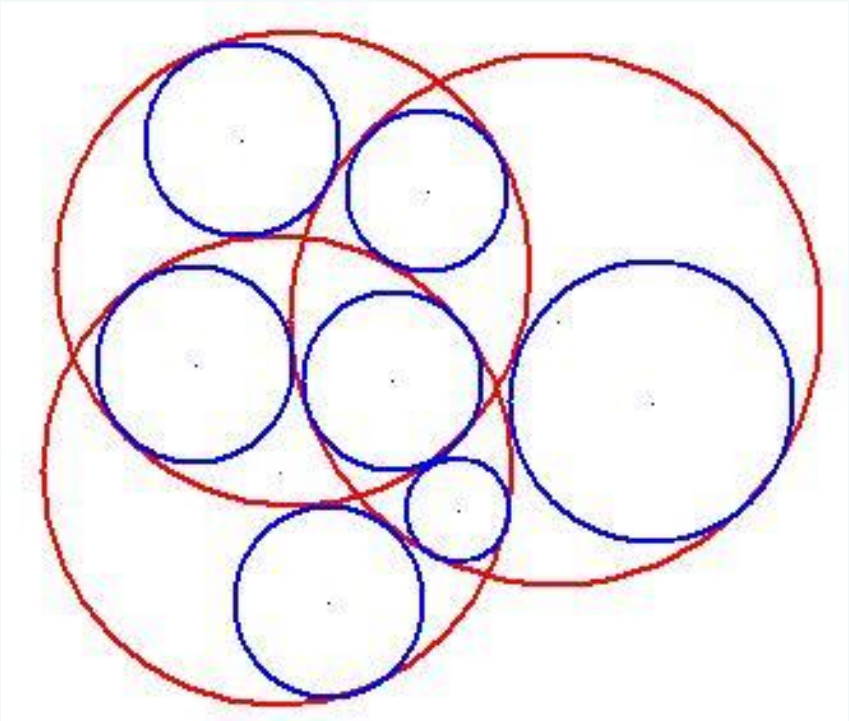
- Z-平面 => W-平面
  - 性质1：Z-平面上经过z0的直线和圆，变换为W-平面上的直线；
  - 性质2：Z-平面上不经过z0的直线和圆，变换为W-平面上的圆；
  - 性质3：Z-平面上的无穷远点，变换为W-平面上的原点；
  - 性质4：变换是保角的，即在Z-平面上相切的几何元素在W-平面上仍然相切。

circle event的生成和判定

- Z-平面上两个圆过一定点的公切圆

  =>

- W-平面上两个圆的公切线。



边的生成

- 边的生成（边的端点可以无限远）：
  - 一个端点到另一个端点：从一个circle event或cross event开始、到另一个circle event或cross event结束。
  - 从中间开始向两端生长：从一个 site event 开始、经过两个circle event 或 cross event 后结束。
  - 从两端到中间生长：两端分别从circle event或cross event开始、直到一个merge event 时连接完整。
  - 没有端点的情况：从一个site event 开始、到一个 merge event 结束，生成一个完整的椭圆。（一个圆包含另一个圆，且较大的圆不与其他圆相交的情况。）

边的表示

- DCEL
- 边为二次曲线
- 参数表示曲线的形状和类型
  - 曲线类型
  - 焦点
  - 到两焦点距离和或差

内容

- 引言
- 背景和前人的工作
- 算法和原理
- 系统设计和数据结构
- 结果和讨论，复杂度分析

系统设计和数据结构

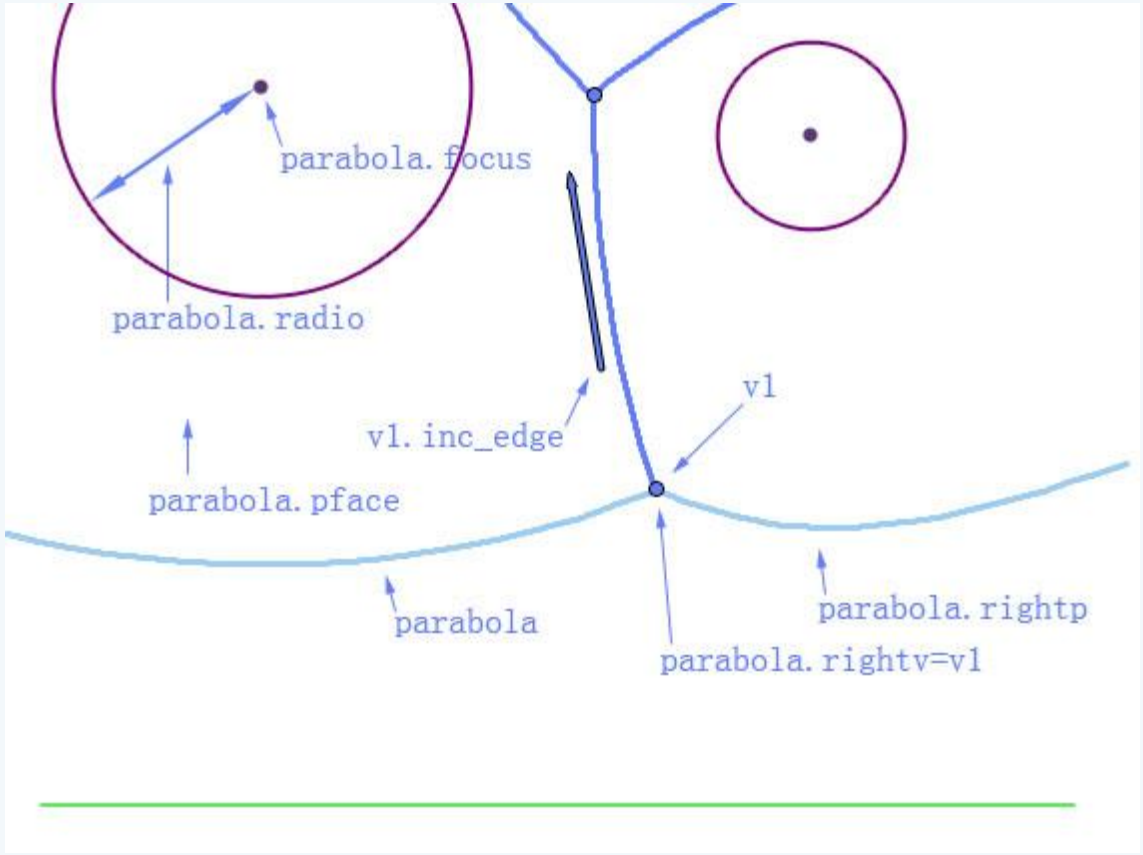
- 系统设计：
  - 预处理阶段：计算site event、cross event和merge event的事件点，并排序。
  - 扫描线推进阶段：按顺序访问事件点，执行对应事件点上的操作，直到所有事件都已处理完毕。

系统设计和数据结构

- 数据结构：
  - 事件队列：需要进行插入、删除、排序等操作。
  - beach line：需要进行插入、删除、查找操作。
  - Voronoi diagram：需要在其上逐渐的生成边，使用的类DCEL的表示方法。

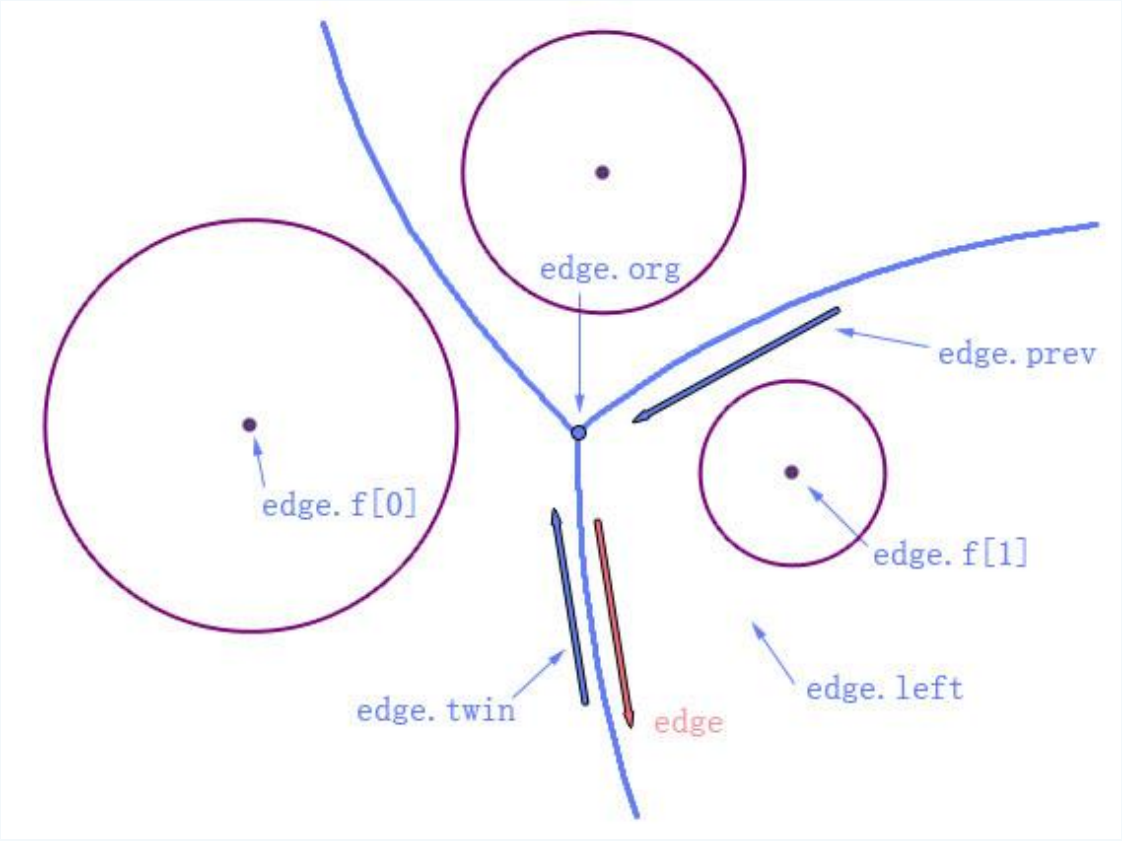
系统设计和数据结构

- 海岸线：



系统设计和数据结构

- Voronoi diagram的数据结构：



内容

- 引言
- 背景和前人的工作
- 算法和原理
- 系统设计和数据结构
- 结果和讨论，复杂度分析

复杂度分析

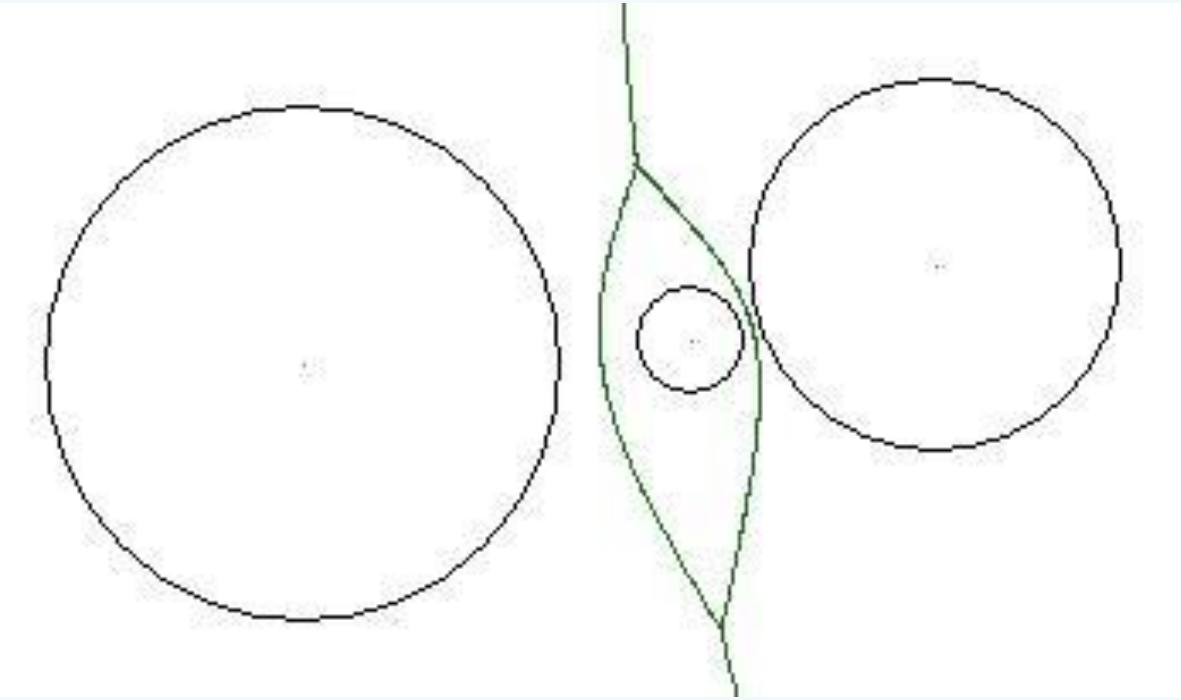
- 以下讨论中n表示输入的圆的数量。
- 预处理阶段：
  - 计算site event和site event的数量：$O(n)$
  - 计算cross event和cross event的数量：$O(n^2)$
  - 计算merge event和merge event的数量：$O(n)$
  - 排序：$O(n^2 \log n)$
- 预处理阶段时间复杂度：$O(n^2 \log n)$

复杂度分析

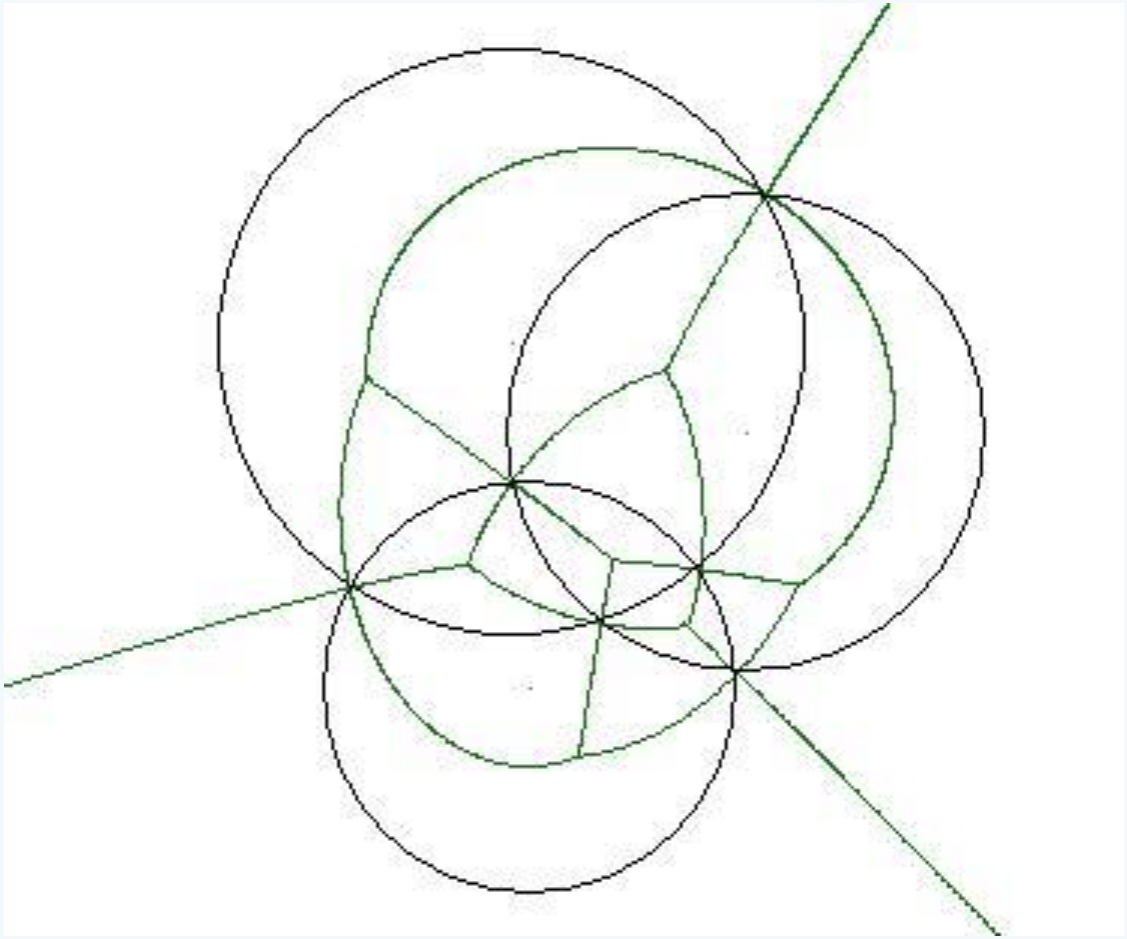
- 扫描线推进阶段：
  - Voronoi diagram顶点数量：每段圆弧对应一个cell，圆弧的数量为 $O(n^2)$ 顶点数量是 $O(2l-4)$ 其中l是cell数。所以顶点数量是 $O(n^2)$
  - 处理到的circle event数量：Voronoi diagram的vertex数 – cross event数。$O(n^2)$
  - 处理每个event的复杂度：$O(\log n^2) = O(\log n)$
- 扫描线推进阶段的时间复杂度：$O(n^2 \log n)$
- 整个算法的时间复杂度：$O(n^2 \log n)$

结果

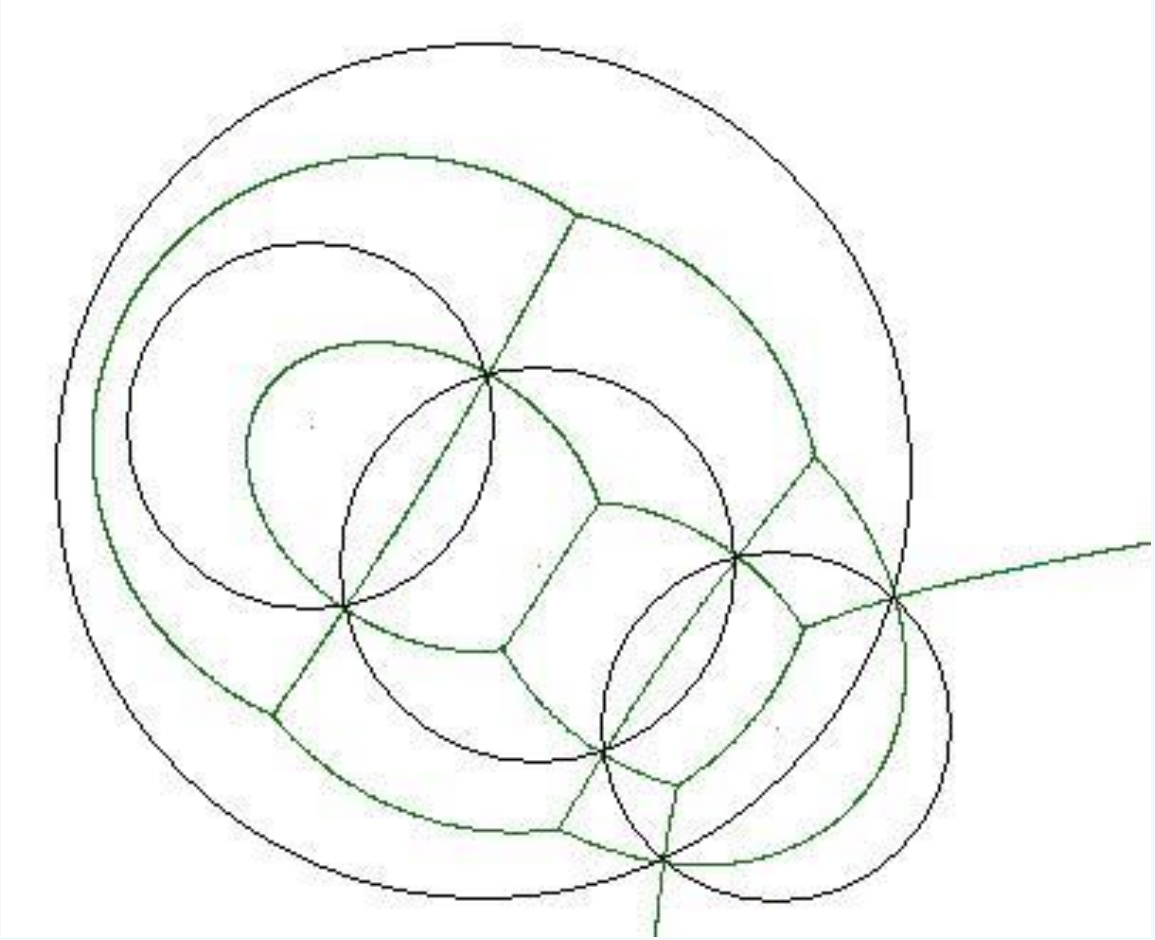
- 我们的结果达到了预期：



结果



结果



参考文献

- [1] Fortune, S., 1987. A sweepline algorithm for Voronoi diagrams. Algorithmica 2, 153–174.
- [2] Deok-Soo Kim, Donguk Kim, Kokichi Sugihara: Voronoi diagram of a circle set from Voronoi diagram of a point set: I. Topology. Computer Aided Geometric Design 18(6): 541-562 (2001)
- [3] Deok-Soo Kim, Donguk Kim, Kokichi Sugihara: Voronoi diagram of a circle set from Voronoi diagram of a point set: II. Geometry. Computer Aided Geometric Design 18(6): 563-585 (2001)

谢谢！