

計算幾何学講義 第11回  
ポロノイ図

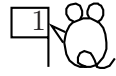


金子 晃

kanenko@atom.is.ocha.ac.jp

[http://atom.is.ocha.ac.jp/~kanenko/  
KOUGI/CompGeo/compgeo.html](http://atom.is.ocha.ac.jp/~kanenko/KOUGI/CompGeo/compgeo.html)

## Voronoi 図の背景



- スーパーマーケットチェーンが新しい支店を開くとき、どこにすれば既存の支店と競合せず最も有効に顧客を集められるか？
- ある国の各都市の商業的支配領域をどうやって推定するか？
- 住人はどの郵便局を利用するか？ 距離に著しい不平等は無いか？

抽象化すると、  
物資やサービスを供給するサイトが分布しているとき、  
住人はどこのサイトから自分の必要なものを得ようとするか？

モデル化のための仮定 (ボロノイ割り当てモデル) :

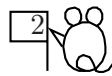
- サービスのコストはサイトによらない。
- サービスを得るコストは、サイトまでのユークリッド距離に比例する。
- 人々はこのコストを最小化するように行動する。

解くべき問題 :

平面に頂点の分布が与えられたとき、平面を多角形に分割し、各面がそれを含む唯一頂点への最も近い点のなす領域と一致するようにせよ。

答えはボロノイ図 (Voronoi diagram) で与えられる。

## Voronoi 図の定義と性質

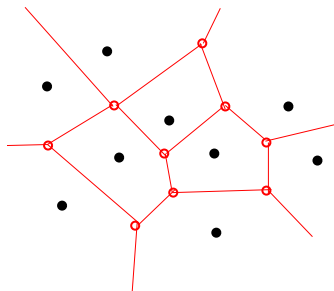


**定義** 平面の点集合  $P = \{p_1, \dots, p_n\}$  の Voronoi 図  $\text{Vor}(P)$  とは、  
平面の多角形への分割  $\mathcal{V}(p_1), \dots, \mathcal{V}(p_n)$  であって、  
 $q \in \mathcal{V}(p_i) \iff \text{dis}(q, p_i) < \text{dis}(q, p_j)$  for  $\forall j \neq i$  なるもの。  
ここに  $\text{dis}$  は平面の Euclid 距離

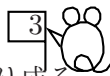
$\text{Vor}(P)$  で Voronoi 図の線分と頂点のみを表すこともある。

● 2点  $p, q$  に対する Voronoi 図は、線分  $\overline{pq}$  の垂直二等分線  
(perpendicular bisector)  
以下では単に“2点  $p, q$  の垂直二等分線”という。  
 $p$  を含む半平面を  $h(p, q)$  で、 $q$  を含む半平面を  $h(q, p)$  で表す。

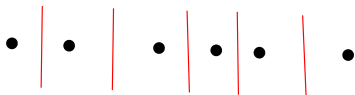
●  $\mathcal{V}(p_i) = \bigcap_{1 \leq j \leq n, j \neq i} h(p_i, p_j)$   
従って高々  $n - 1$  凸多角形となる。  
(非有界かもしれない)



## Voronoi 図の構造



定理 全ての点  $p_i$  が共線 (colinear) なら Voronoi 図は平行直線より成る.  
それ以外するとき Voronoi 図の辺集合は連結で半直線か線分より成る.



証明 共線でない点集合  $P$  の Voronoi 図が全直線  $l$  を含んだとせよ.

それは  $\exists p_i, p_j \in P$  の垂直 2 等分線である.

(辺の定義によりある 2 点から等距離にある点の集合だから.)

仮定により  $p_i, p_j$  と共線にない第 3 の点  $p_k \in P$  が有る.

$p_j, p_k$  の垂直 2 等分線は  $l$  と平行ではないので, 交わる.

$l$  の  $h(p_k, p_j)$  内の点は  $p_i, p_j$  よりも  $p_k$  に近いので,

Voronoi 図の边上の点ではなくなり, 不合理.

次に, Voronoi 図の辺が連結でなかったとすると,

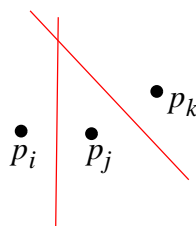
それを分ける多角形領域が存在する

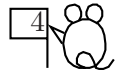
これは, Voronoi 図のセルの一つなので, 凸.

凸図形で, その補集合が非連結となるものは,

平行 2 直線に挟まれた帯しか有り得ない.

それは, Voronoi 図が全直線を含まないという, 証明済みの事実と反する.  $\square$





定理  $n \geq 3$  のとき,  $n$  点の Voronoi 図は  
高々  $2n - 5$  個の頂点と  $3n - 6$  個の辺を持つ.

証明 すべての点が共線のときは,  $v = 0, e = n - 1$  で明らかに成り立つ.  
それ以外の場合, Voronoi 図に無限遠点  $p_\infty$  を付け加えると,  
球面の多角形分割が得られるので,

Euler の多面体定理により  $(v + 1) - e + f = 2$ .

ここで,  $f = n$  (各面は一つずつもとの点を含む)

各辺はちょうど 2 個の端点を持つので,

$2e =$  各頂点での次数 (辺の数) の和.

各頂点は無限遠点も込めて次数 3 以上なので

$$2e \geq 3(v + 1)$$

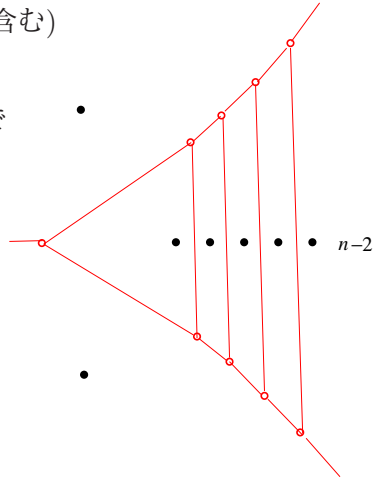
$$\text{これより, } v + 1 = e - n + 2 \leq \frac{2}{3}e.$$

$$\text{よって } e \leq 3n - 6.$$

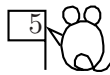
$$\text{従って更に, } v + 1 \leq \frac{2}{3}e \leq 2n - 4,$$

$$\text{i.e. } v \leq 2n - 5 \quad \square$$

左図は定理の限界を達成している例 :



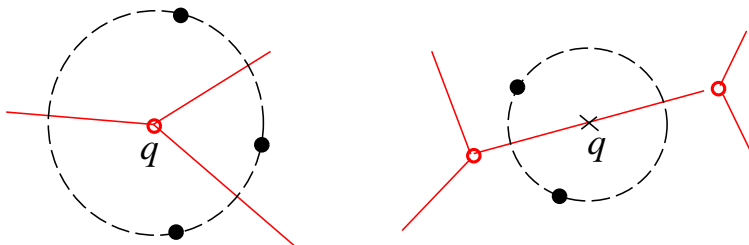
## Voronoi 図の頂点と辺の特徴付け



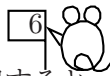
$n$  点の Voronoi 図は  $O(n)$  個の辺や頂点しか含まないので、  
 $O(n^2)$  個の垂直 2 等分線やそれらの交点の大部分は Voronoi 図に参加しない。  
どうするのが Voronoi 図に参加するか？

**定義** 点  $q$  の最大空円 (largest empty circle)  $C_P(q)$  とは、  
 $q$  を中心とする円で  $P$  の点を内部に含まないような最大の円。

- 定理** (i)  $q$  が  $\text{Vor}(P)$  の頂点  
 $\iff C_P(q)$  の境界上に  $P$  の少なくとも 3 点が存在。  
(ii)  $p_i, p_j$  の垂直 2 等分線が  $\text{Vor}(P)$  の辺を成す  
 $\iff$  垂直 2 等分線上のある点  $q$  で  
 $C_P(q)$  の境界上にある  $P$  の点が  $p_i, p_j$  だけ



## Voronoi 図の計算



$\mathcal{V}(p_i) = \bigcap_{1 \leq j \leq n, j \neq i} h(p_i, p_j)$  に半平面の交差計算アルゴリズムを適用すると、

各  $\mathcal{V}(p_i)$  は  $O(n \log n)$  で求まるが、Voronoi 図全体では  $O(n^2 \log n)$  になる。  
Voronoi 図を  $O(n \log n)$  で計算するアルゴリズムがいろいろ有る。

以下その一つ Fortune のアルゴリズムを解説する。

$O(n \log n)$  はベストである：

$n$  個のデータのソートの問題は  $n$  点 Voronoi 図の製作問題に帰着できる。

前者が  $\Omega(n \log n)$  であることが周知なので、Voronoi 図も  $\Omega(n \log n)$ 。

基本的に水平 sweep line 法を用いる。

sweep line  $l$  を上から降ろして来たとき、最初の頂点に出会う前に、  
既に Voronoi 図の一部が  $l$  の上にできている！

よって、 $l$  より上にある半平面を  $l^+$  で表すとき、ステータスの記述を、

“ $l^+$  に有る Voronoi 図の部分”

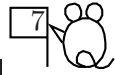
ではなく、

“ $l$  より下の頂点では変化しないような  $l^+$  に有る Voronoi 図の部分”

とする。

これは、 $q \in l^+$  で、そこから最短のサイトが  $l^+$  にあることが既知である  
ようなものの集合と一致

## sweep line 法



$q \in \ell^+$  で、そこから最短のサイト  $p_i$  が  $\ell^+$  にあることが確定  
 $\iff \text{dis}(q, p_i) \leq \text{dis}(q, \ell)$

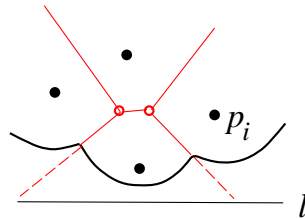
一般に、各  $p_i$  に対し、 $\{q \mid \text{dis}(q, p_i) \leq \text{dis}(q, \ell)\}$  は放物線の上部を定める。  
 $p_i$  が焦点で  $\ell$  が準線。

故に、ステータス構造は、

すべてのサイト  $p_i \in \ell^+$  に対するこれらの放物線内部の和集合  
内に含まれる Voronoi 図の部分

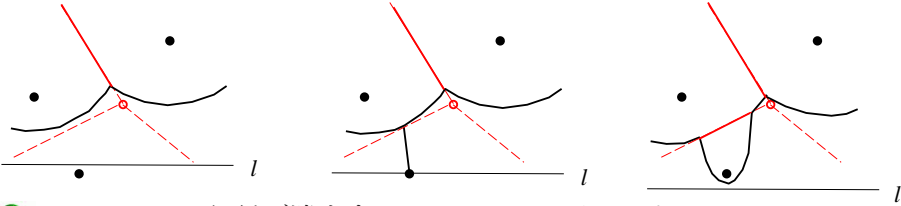
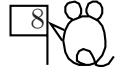
となる。

この領域の境界は全放物線の方程式  
 $y = a_i x^2 + b_i x + c_i$  ( $a > 0$ ) の  $\min$  で  
定義された区分的に滑らかな曲線となる  
(beach line 海岸線)。

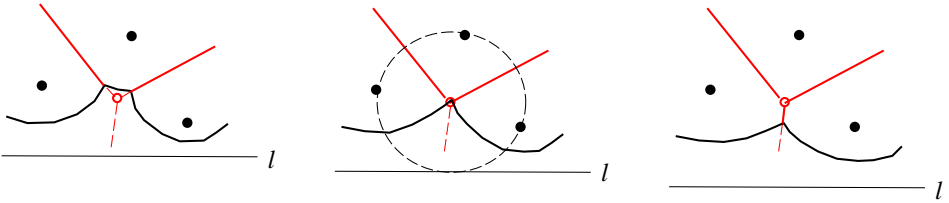


beach line は sweep line を下げると連続的に変化するので、  
そのまま記録することはできないが、  
組み合わせ構造が変化するところだけに注目すればよい。

● beach line に新しい弧が追加される : 点 (site) イベント  
 sweep line がサイトを通過するとき起こる :



● beach line から弧が消失する : 円 (circle) イベント  
 sweep line が三つ (以上) のサイトを通る円の最低点を通る時起こる :



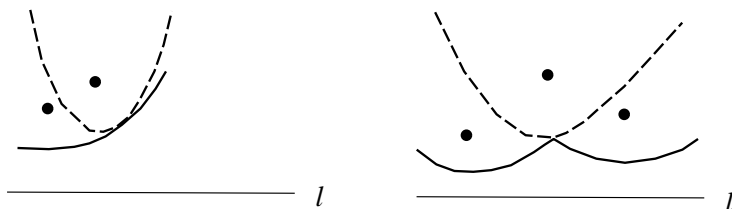
この円の中心  $q$  は Voronoi 図の頂点となる.

∵ beach line の角は Voronoi 図の辺上に有り,  $q$  でそれらの辺が交わるから.

この円の内部にはサイトは存在しない.

∵  $q$  は beach line 上の点なので,  $l$  への距離より近くにはサイトは無い.

補題 beach line に新しい弧が生ずるのは上の場合しか無い.



証明 上の図が想定される他の可能性

(点線で示した放物線弧が  $l$  とともに降りて来てどこかで接し、突き抜ける) いずれも,  $l$  を共通の準線とする場合には有り得ないことが, 方程式を立てれば容易に分かる.

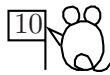
直感的にも分かる :

左の図では, 点線の放物線の方が曲率が小さいので, 逆に外側に来るはず. 右の図では,  $l$  が下がる時, 点線の放物線の方が下がり方が遅いので, 突っ込まずに離れて行ってしまうはず.

補題 beach line の弧が消失するのは上の場合しか無い.

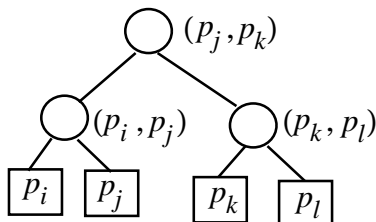
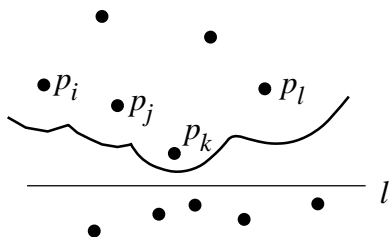
証明 弧が消失するのは, 二つの beach line の二つの角がぶつかる時で, それらの角が描いて来た Voronoi 図の辺もそこでぶつかり, 頂点を成す.

## Voronoi 図作成のためのデータ構造

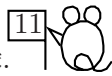


- イベントキュー  $Q$ : サイト, 三つのサイトの外接円の最低点
- ステータス構造  $T$ : beach line の組み合わせ構造を表す平衡二分探索木
- 作成途中の Voronoi 図  $D$ : 2重連結辺リストとして記録するが, 半無限辺を扱えないので, 全サイトを含む限界長方形の中で作る.

$T$  の構造: 葉は beach line の放物線弧,  
内部ノードは弧が交わる角 (かど) を表すサイトの順序対  $(p_i, p_j)$   
( $p_i$  の弧が左に,  $p_j$  の弧が右にあるように順序を定める.)  
ノード間もこの順序に関して辺を書く.



この構造を用いて新サイトの真上に有る beach line の弧を  $O(\log n)$  で計算可.  
 $\therefore$  内部ノード  $(p_i, p_j)$  と  $l$  から  $O(1)$  で対応する beach line 角の座標が  
計算できるから, 通常の平衡二分木の探索時間で済む.



- $T$  の内部ノードには、対応する beach line の角が載る Voronoi 図の辺 (2 重連結辺のいずれか) へのポインターを記録.
- $T$  の葉には、対応する beach line の弧が消失するときの円イベントへのポインタを記録 ..... ☆.

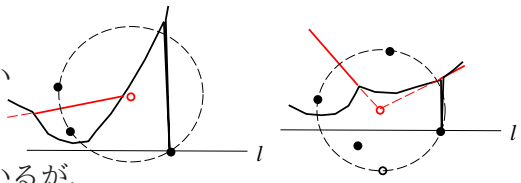
$Q$  の構造 : イベントの  $y$ -座標に関する先着権キュー (priority queue) イベント点が登場する毎に追加する.

- 点イベントのときは、そのサイトを記録
- 円イベントのときは、円の最低点と、このイベントで消失する弧を表す  $T$  の葉へのポインタ (上の☆の逆向きのポインタ)

点イベントは始めから分かるが、円イベントは sweep と同時に探索が必要  
その方法 : ステータス木  $T$  にある隣接した三つの弧に対応する  
三つのサイトの外接円の最低点を可能な円イベントの候補とし、  
 $T$  が変化する毎にそれを更新する.

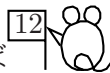
しかし、円イベントは必ずしも  
イベントとして実現すると限らない  
(false alarm 図の右の方がその例)

※ 教科書では、図左は求めた円がイベントの候補にならないとしているが、  
これも円イベントに含めないとこの図の頂点が Voronoi 図に追加されない !



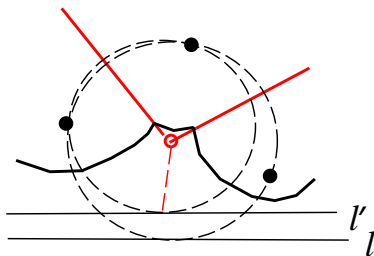
実際の手順：

弧が追加されたとき、外接円の中心を計算し、それが下方に有れば (サイトが右折れに並んでいれば) その最低点を  $Q$  に追加 (※ 下注)  
弧が消失したとき、対応する  $T$  の葉が円イベントへのポイントを持ってばそれを  $Q$  から削除

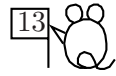


補題 Voronoi 図の全頂点は円イベントで見付かる。

証明  $\text{Vor}(P)$  の頂点は最低 3 個のサイト  $p_i, p_j, p_k \in P$  の外接円  $C$  の中心。  
簡単のため、 $C$  上にはこの 3 サイトしか無く、どれも最低点でないとする。  
sweep line  $l$  をほんの少し上に戻した  $l'$  を考えると、  
 $p_i, p_j$  を通り、 $l'$  に接する円  $C'$  が定まる。  
 $C'$  上にはこの 2 サイトしか無いので、  
これらに対応する beach line の弧が存在する。  
同様に、 $p_j, p_k$  に対応する弧も存在する。  
 $p_j$  に対応する弧は明らかに同一で、  
これが  $l'$  を  $l$  に下げたとき消失する。□



※ 実は前ページで注意したような場合も (左折れではあるが) 円イベントに含めないと、この証明は完全ではない！



## VoronoiDiagram( $P$ ) :

入力 平面の  $n$  点 (サイト) の集合  $P = \{p_1, \dots, p_n\}$

出力  $P$  の Voronoi 図  $\text{Vor}(P)$

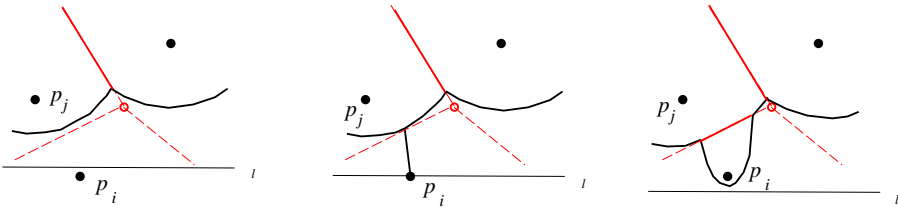
限界長方形内に 2 重連結辺リスト  $\mathcal{D}$  として与えられる

### アルゴリズム

- ①  $Q$  を全サイトの点イベントで初期化.  $\mathcal{T}, \mathcal{D}$  を空で初期化
- ② while  $Q$  が空でない
- ③      $Q$  から  $y$ -座標最大のイベントを除去
- ④     if それが点イベント  $p_i$  then  
       HandleSiteEvent( $p_i$ )
- ⑤     else  
       円イベントが指す  $\mathcal{T}$  の葉を  $\gamma$  として  
       HandleCircleEvent( $\gamma$ )  
       end if
- end while
- ⑥ 限界長方形を作り,  
   残っている beach line に対応する Voronoi 図の半無限辺を  $\mathcal{D}$  に追加
- ⑦  $\mathcal{D}$  の半辺を辿り, Voronoi 図の面データを作成

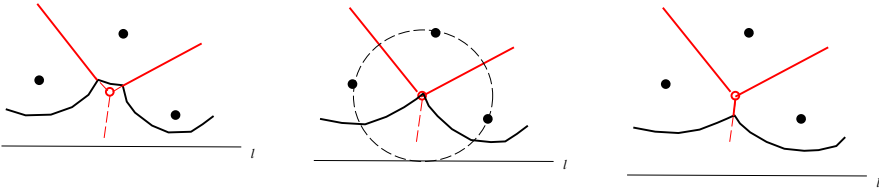
**HandleSiteEvent( $p_i$ ) :**

- ① if  $\mathcal{T}$  が空 then  
 $p_i$  をただ一つの葉として追加し return
- ②  $p_i$  の真上にある弧  $\alpha$  を  $\mathcal{T}$  の葉から探す.
- ③ if この葉が  $Q$  の円イベントへのポインタを持つ then  
このイベントを  $Q$  から削除
- ④  $\alpha$  に対応する  $\mathcal{T}$  の葉を  
内部ノード  $(p_j, p_i)$ ,  $(p_i, p_j)$  および葉  $p_i$  より成る部分木と取り替える.  
(ここに,  $p_j$  は  $\alpha$  に対応するサイト)  
必要なら木の平衡化も行う.
- ⑤  $\mathcal{V}(p_j)$  と  $\mathcal{V}(p_i)$  をへだてる Voronoi 図の新しい辺を  $\mathcal{D}$  に追加  
(2重辺リストのノードを用意するだけで, 端点等は NULL ポインタ)
- ⑥  $p_i$  に対応する新しい弧を最右端, および最左端にもつ3連続弧の各々  
について, それらに対応する円イベントを調べ, 必要なら  $Q$  に追加.

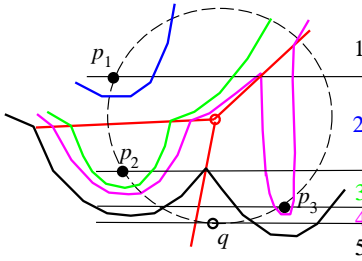






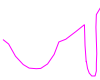


**HandleCircleEvent( $\gamma$ ):**

- ① 葉  $\gamma$  を  $\mathcal{T}$  から消去し,  
この葉に対応する弧  $\alpha$  の消失で変わる内部ノードの組み合わせを修正。  
必要なら木の平衡化も行う。  
消失した弧がからんでいた円イベントを  $Q$  から消去。  
(この葉の左隣と右隣の葉に円イベントへのポインタが有るかどうか見る.)
- ② 円の中心を新たな頂点として  $\mathcal{D}$  に追加。  
既存の辺との隣接関係を更新。  
新しい角の動きに対応してそこから発する辺を  $\mathcal{D}$  に追加。
- ③ 消失弧の左隣, および右隣を中央弧として新たに生じた二つの3隣辺  
の各々に対し, 円イベントを調べ, 必要なら  $Q$  に追加。



処理の実例



領域	$Q$	$\mathcal{T}$	$\mathcal{D}$	beach line
1	$\{p_1, p_2, p_3\}$	$\emptyset$	$\emptyset$	$\emptyset$
2	$\{p_2, p_3\}$	$\boxed{p_1}$	$\emptyset$	
3	$\{p_3\}$	<pre>           (p1, p2)          /   \         [p1]  (p2, p1)               / \              [p2] [p1]             </pre>		
4	$\{q\}$	<pre>           (p2, p1)          /   \         (p1, p2) (p1, p3)        / \   / \       [p1] [p2] [p1] (p3, p1)                        / \                       [p3] [p1]             </pre>		
5	$\emptyset$	<pre>           (p1, p2)          /   \         [p1]  (p2, p3)               / \              [p2] [p3]             </pre>		

## アルゴリズムの正当化と計算量



定理  $n$  点の Voronoi 図は  $O(n)$  の記憶域と  $O(n \log n)$  の計算量で作れる。

証明 上の説明が通用する非退化な場合をまず論ずる、

点イベントの総数は明らかに  $n$ 。

円イベントは Voronoi 図の頂点に対応するので、その数は高々  $2n - 5$

(false alarm の分も有るが、実際には処理されず、一時的に  $\mathcal{T}$  と  $\mathcal{Q}$  の記憶域を占めるだけで、いずれも点イベントで消失するので、 $O(n)$ ) ※1

以上により、 $\mathcal{Q}$  の記憶サイズは  $O(n)$

$\mathcal{T}$  のサイズは、内部ノードが beach line の角の数、葉が高々その倍で、やはり  $O(n)$  ※2

よって、 $\mathcal{T}$ 、 $\mathcal{Q}$  の基本操作 (ノードの追加や削除) は  $O(\log n)$  の計算量。

イベント当たりの  $\mathcal{D}$  の作り替えの計算量は  $O(1)$  なので、

イベント当たりの計算量は  $O(\log n)$

よって全計算量は  $O(n) \times O(\log n) = O(n \log n)$ 。

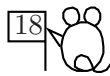
必要な記憶域は、途中の  $\mathcal{T}$ 、 $\mathcal{Q}$  と、答の  $\mathcal{D}$  のサイズの和で  $O(n)$  □

※1 実は消失イベントの総数はイベント処理が生じないので問題にならず、問題となるのは、一時記憶で  $\mathcal{Q}$  などを占めるサイズだけだが、

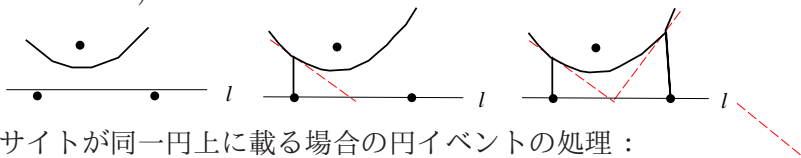
これも明らかに beach line の弧の数以下なので、常時  $O(n)$  である。

※2 同一のサイトが弧として複数回現れることが有り得ることが前ページの例から分かるが、それでもこの評価は成り立つ。

次に、退化した場合を考える.

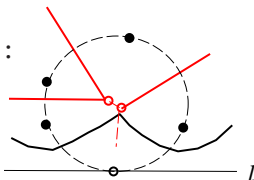


- 二つのサイトが同一の  $y$ -座標を持つ場合の点イベントの処理：  
 $x$ -座標の小さい物から順に処理する。  
(実は任意の順序でよい.)



- 四つ以上のサイトが同一円上に載る場合の円イベントの処理：  
 $y$ -座標が異なれば、退化ではない。

- 最後の二つが同じ高さのときの円イベントの処理：  
長さ 0 の辺を持つ二つの頂点を作っておき、  
後で一つにまとめれば、例外処理は不要。



- 円イベント点がサイトでもある場合、  
まず点イベントを処理して新たな 2 辺を作り、  
次いで円イベントを処理してそれらを頂点で結ぶと、  
例外処理が不要となる。  
(先に円イベントを処理すると右図の点線のように  
false edge が現れてしまう.)

