Università
della
Svizzera
italiana

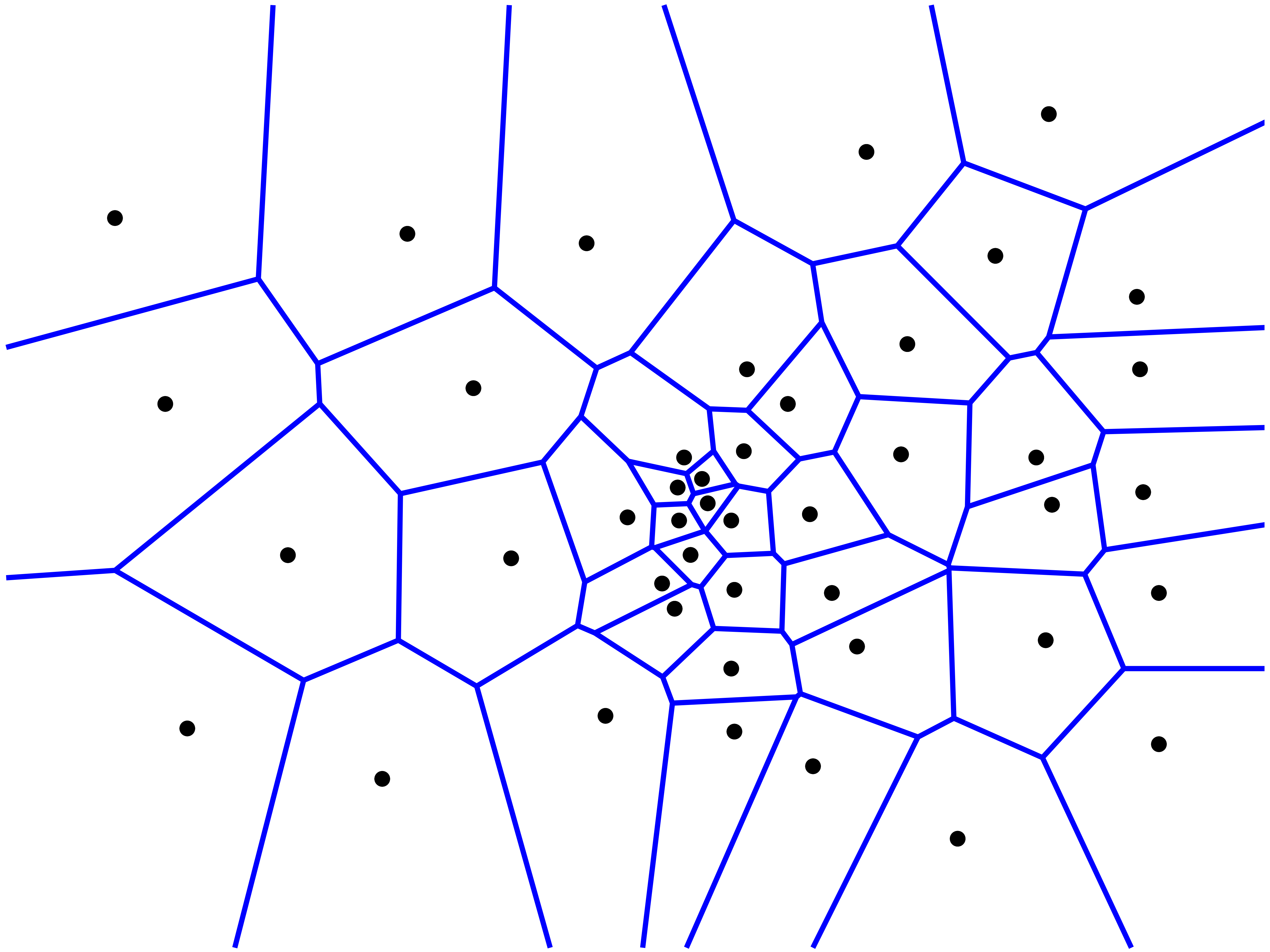
Generalized Voronoi Diagrams and Applications in VLSI Design for Manufacturing

Evanthia Papadopoulou

Università della Svizzera italiana (USI Lugano)

Voronoi diagrams

- A versatile geometric partitioning structure.



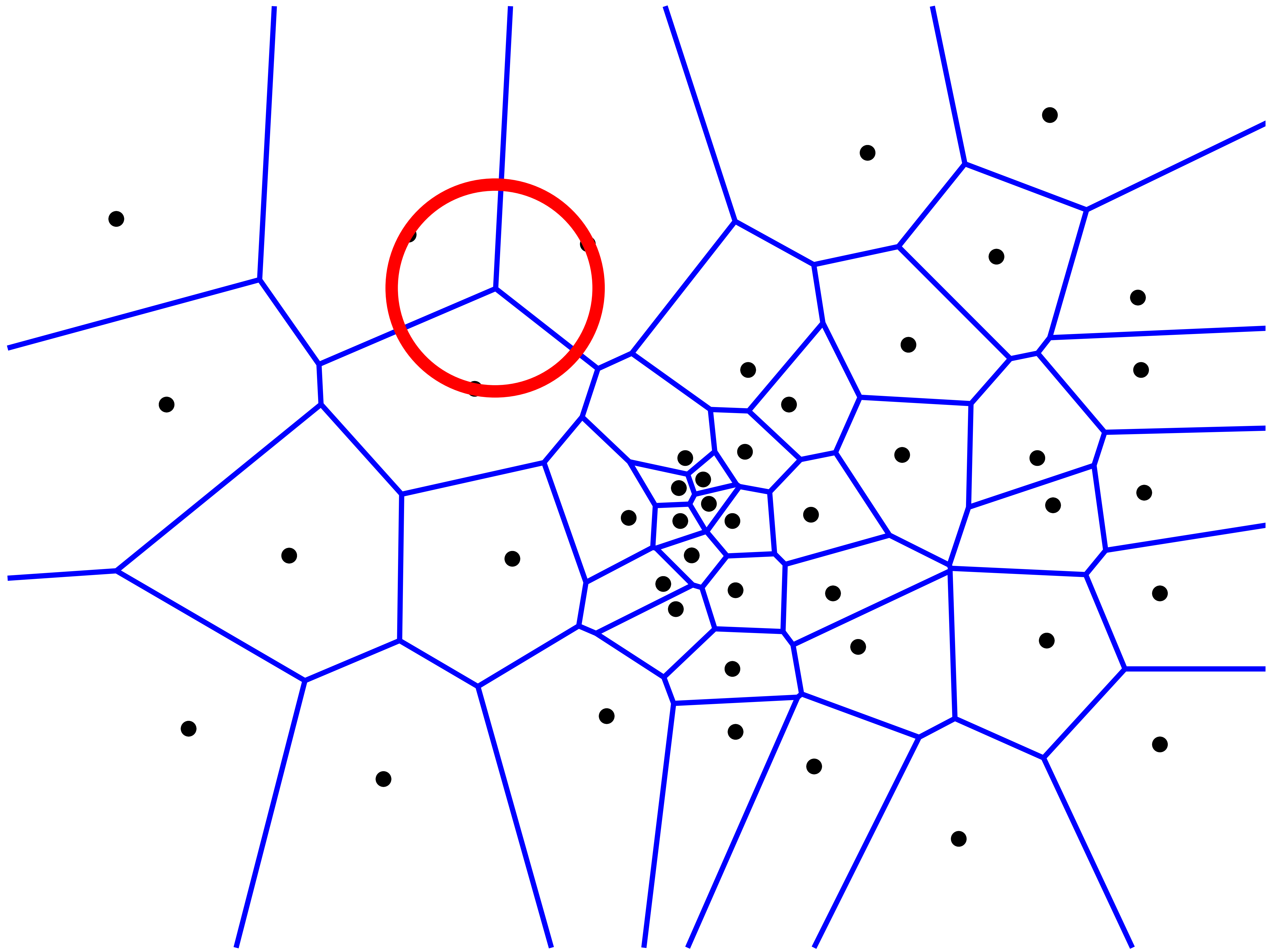
S: set of **n** simple geometric objects, called **sites**.

The **Voronoi region** of a site p is the locus of points closer to p than to any other site in S.

The **Voronoi diagram** of S is the resulting space subdivision

Voronoi diagram of **points** in **Euclidean plane**

- A plane graph of linear (O(n)) size.



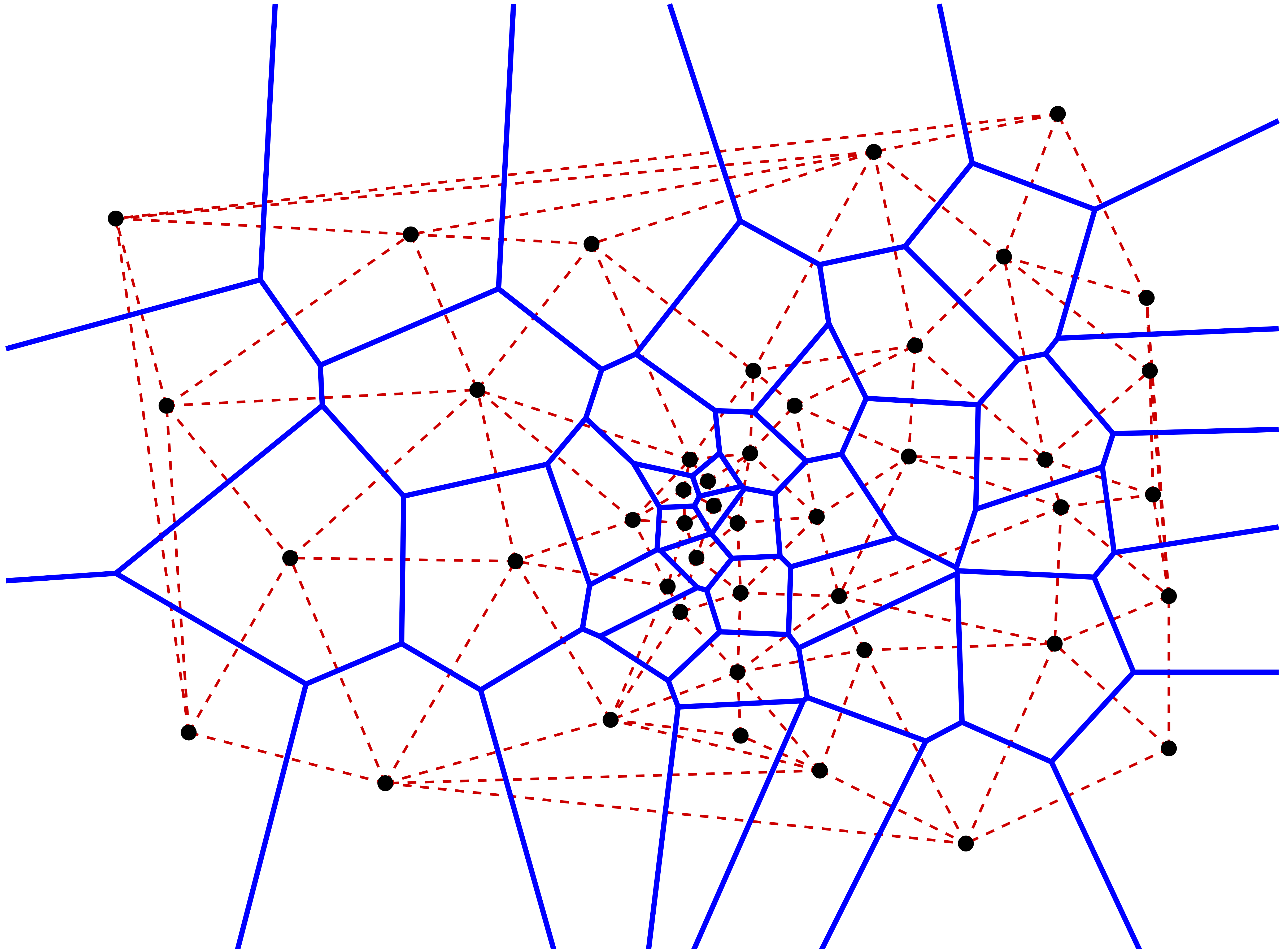
Voronoi regions are convex

Voronoi edges ⊆ line bisectors between two points

Voronoi vertices are points equidistant from 3 sites

Voronoi vertex: the center of a circle defined by 3 sites, which is empty of other sites.

Dual: Delaunay Graph / Triangulation



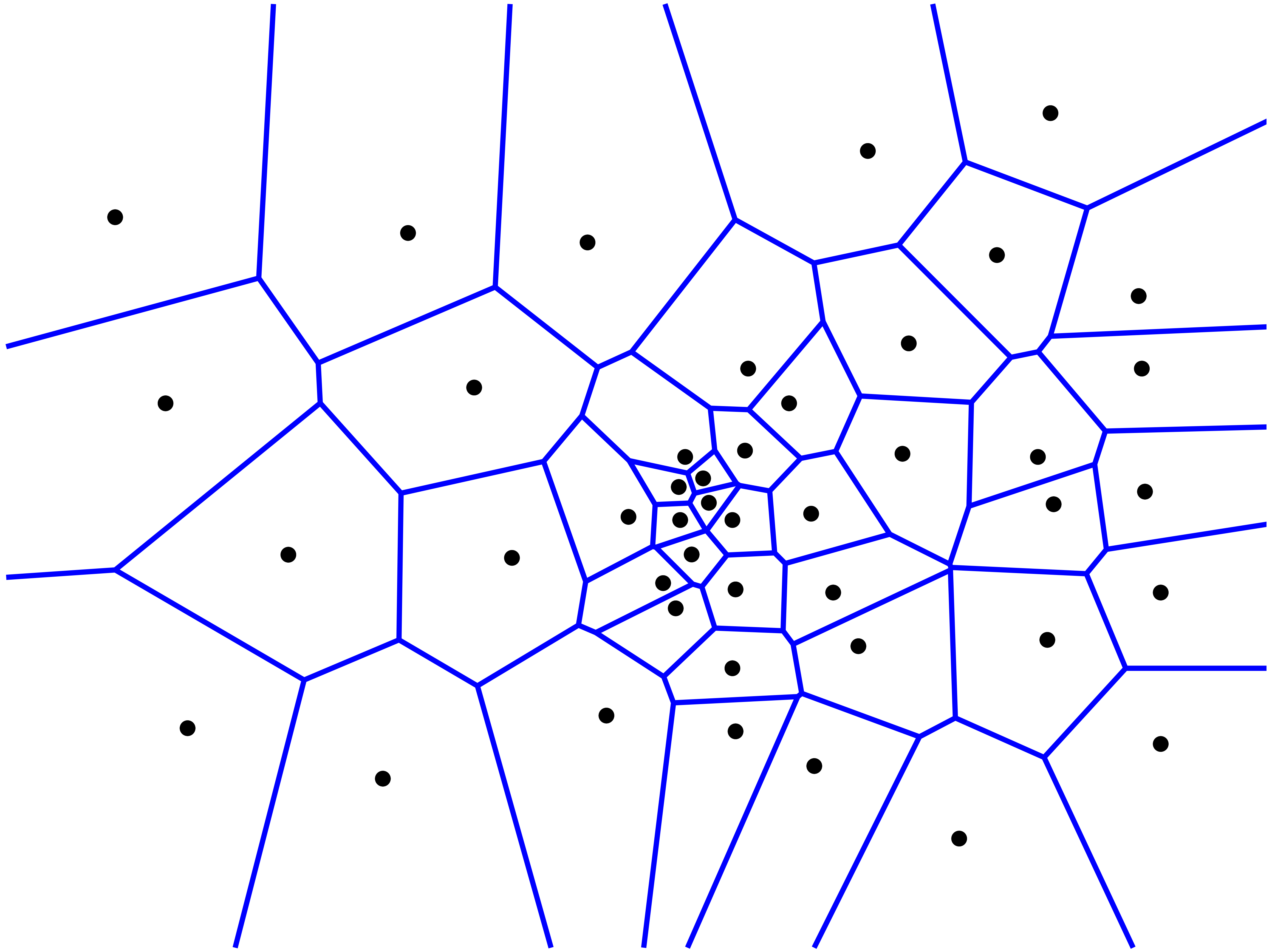
The graph nodes are sites

Two nodes are joined by an edge if their Voronoi regions are neighboring.

Equiv.: if there exists a circle passing through the two sites, which is empty of other sites

Voronoi diagrams

- A versatile geometric partitioning structure.



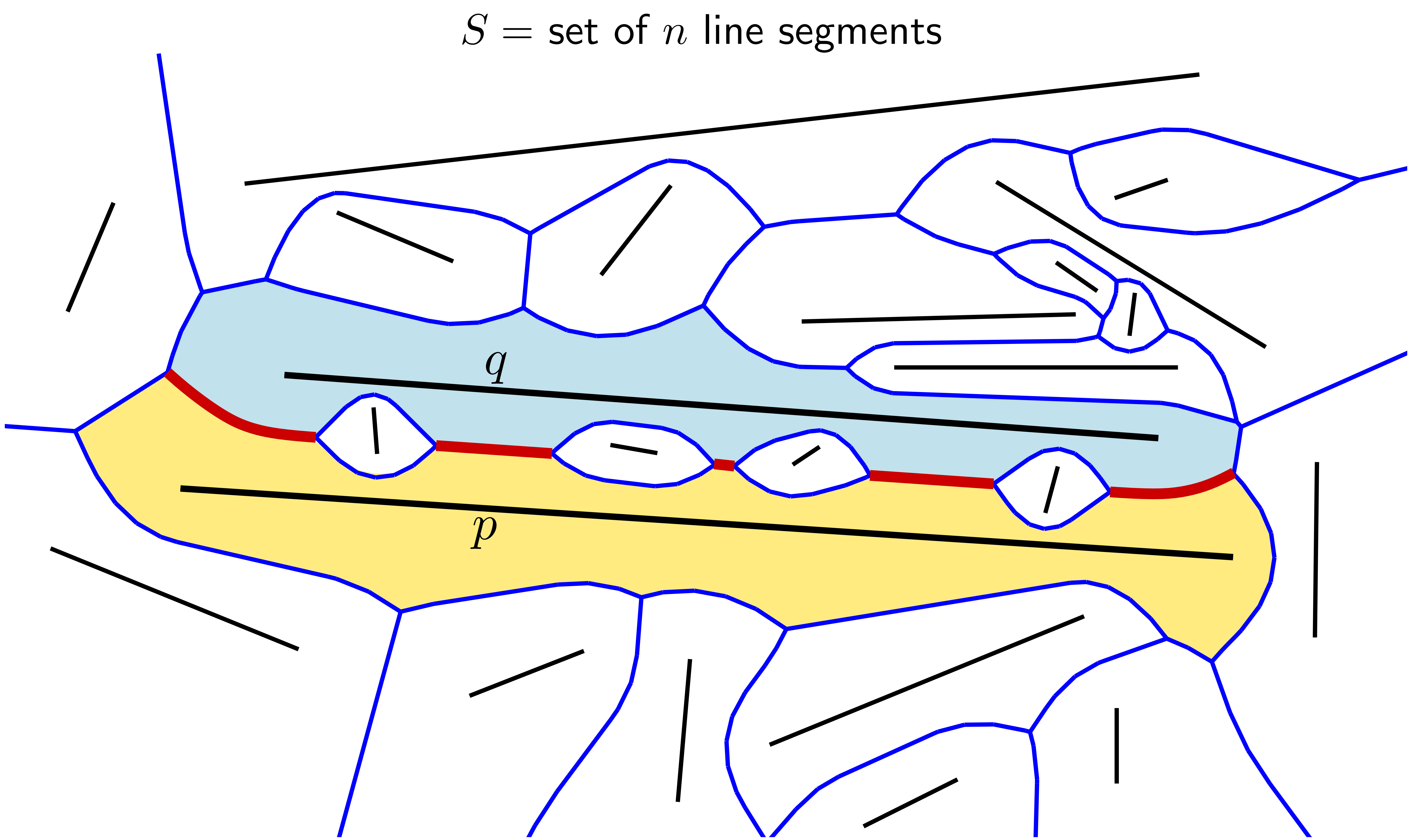
Voronoi diagrams of different **sites,**

generalized **metrics,**

higher **dimensions**

Voronoi diagram of segments

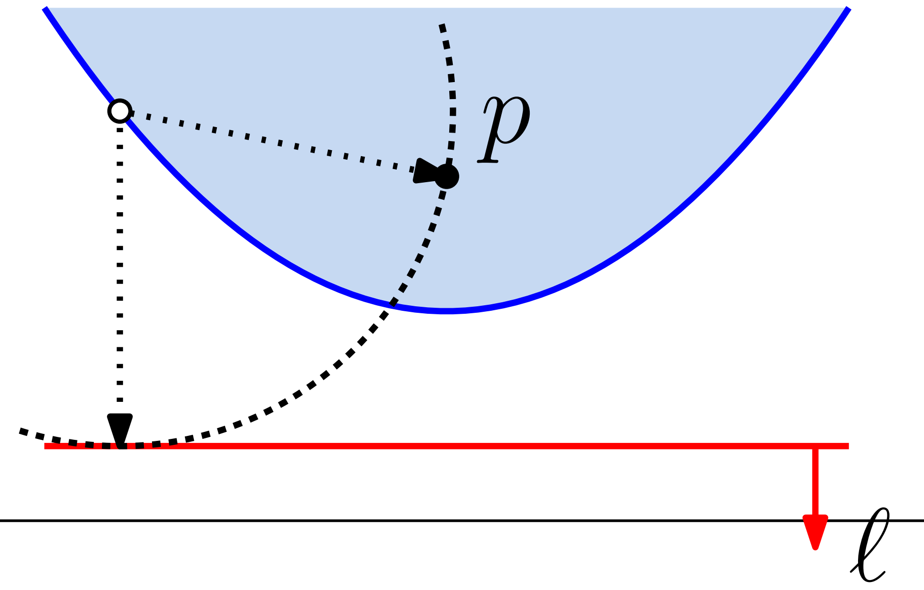
- Well known differences



Bisectors (**Voronoi edges**) are not lines

Voronoi regions are not convex

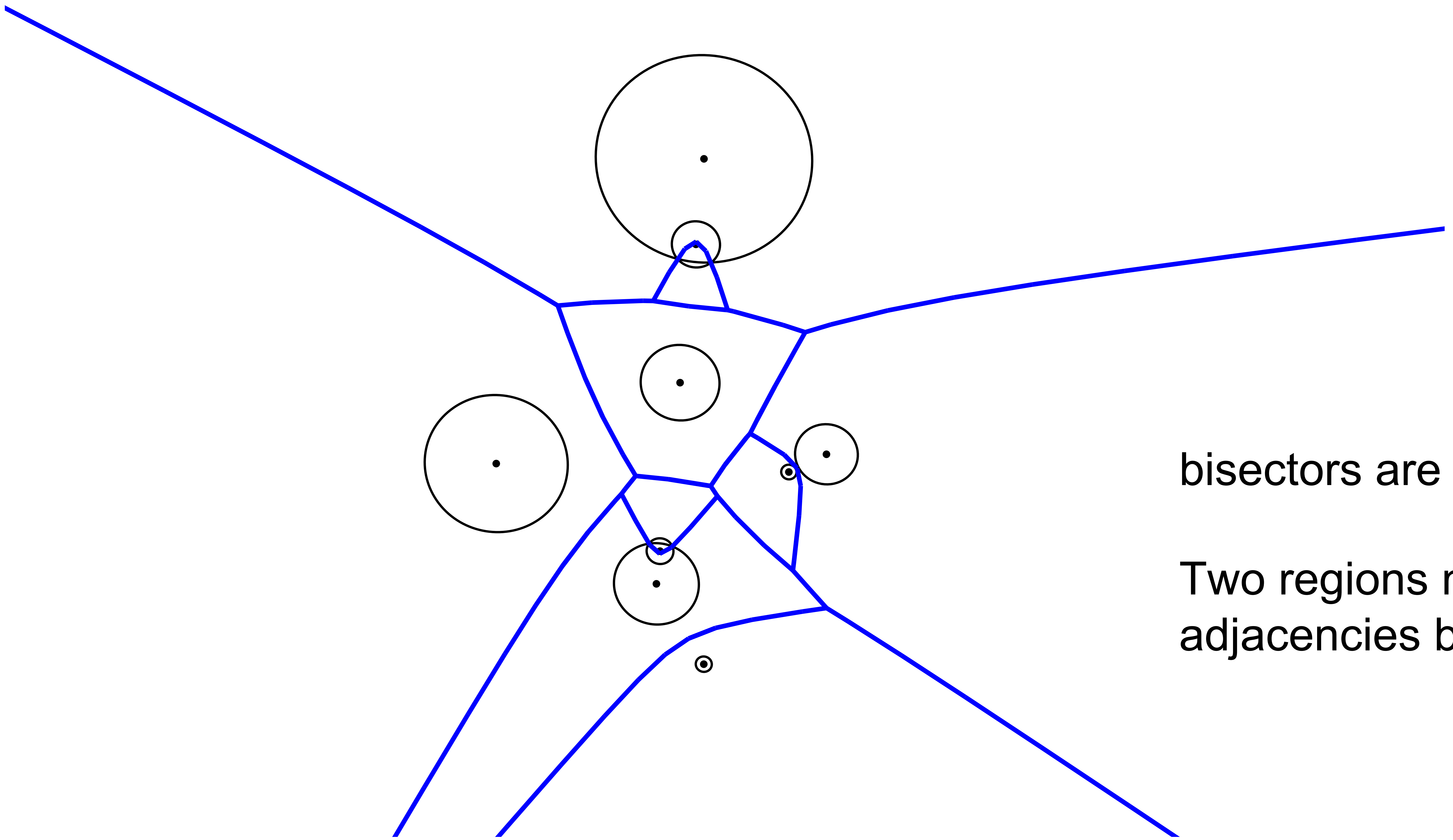
Multiple adjacencies between the regions of two sites



Voronoi diagram of circles

- Similar issues

$S =$ set of n circles (weighted points)

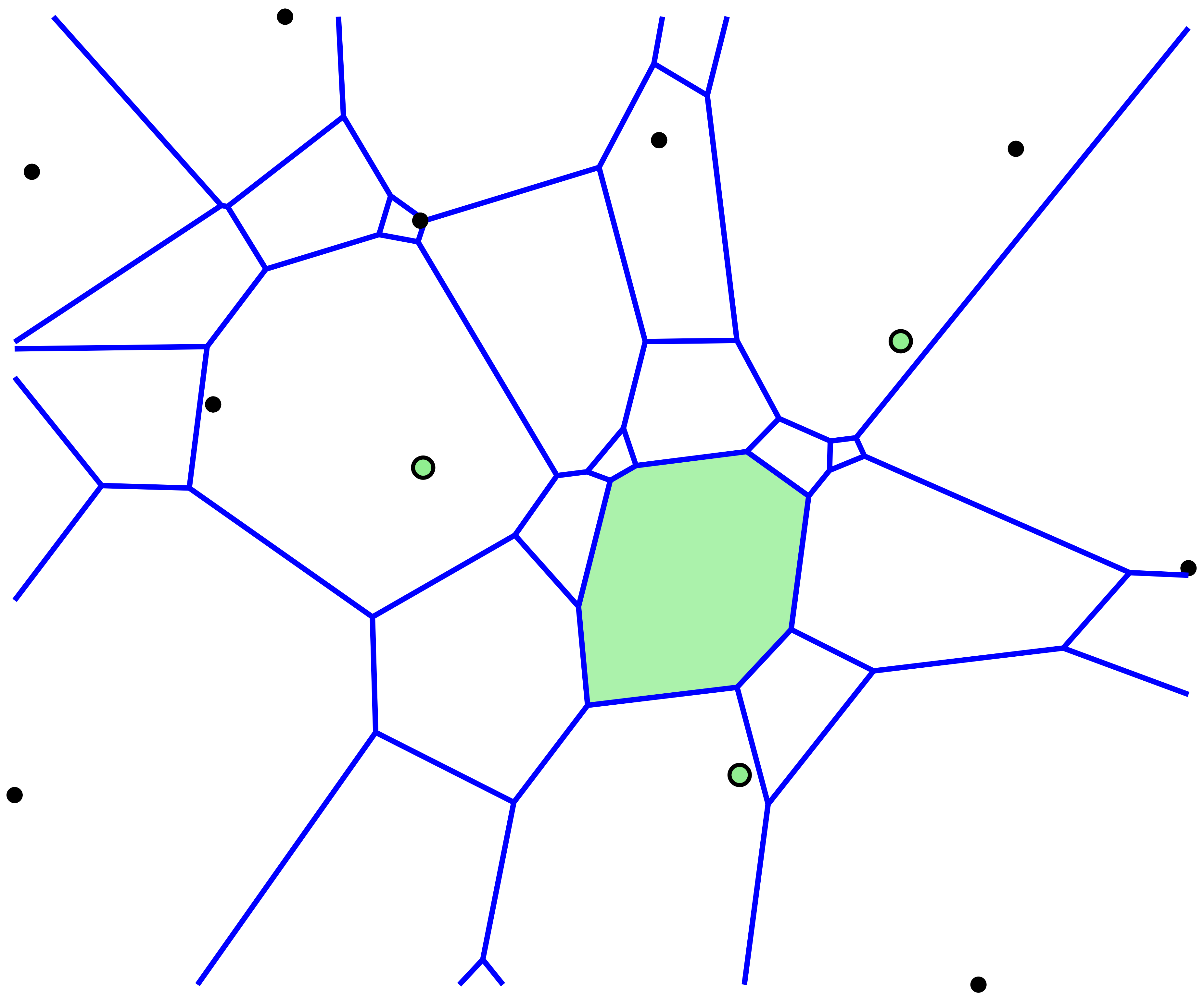


bisectors are hyperbolic arcs

Two regions may have multiple adjacencies between them

Voronoi diagrams of higher order

- k-nearest neighbor information, $1 \le k \le n-1$

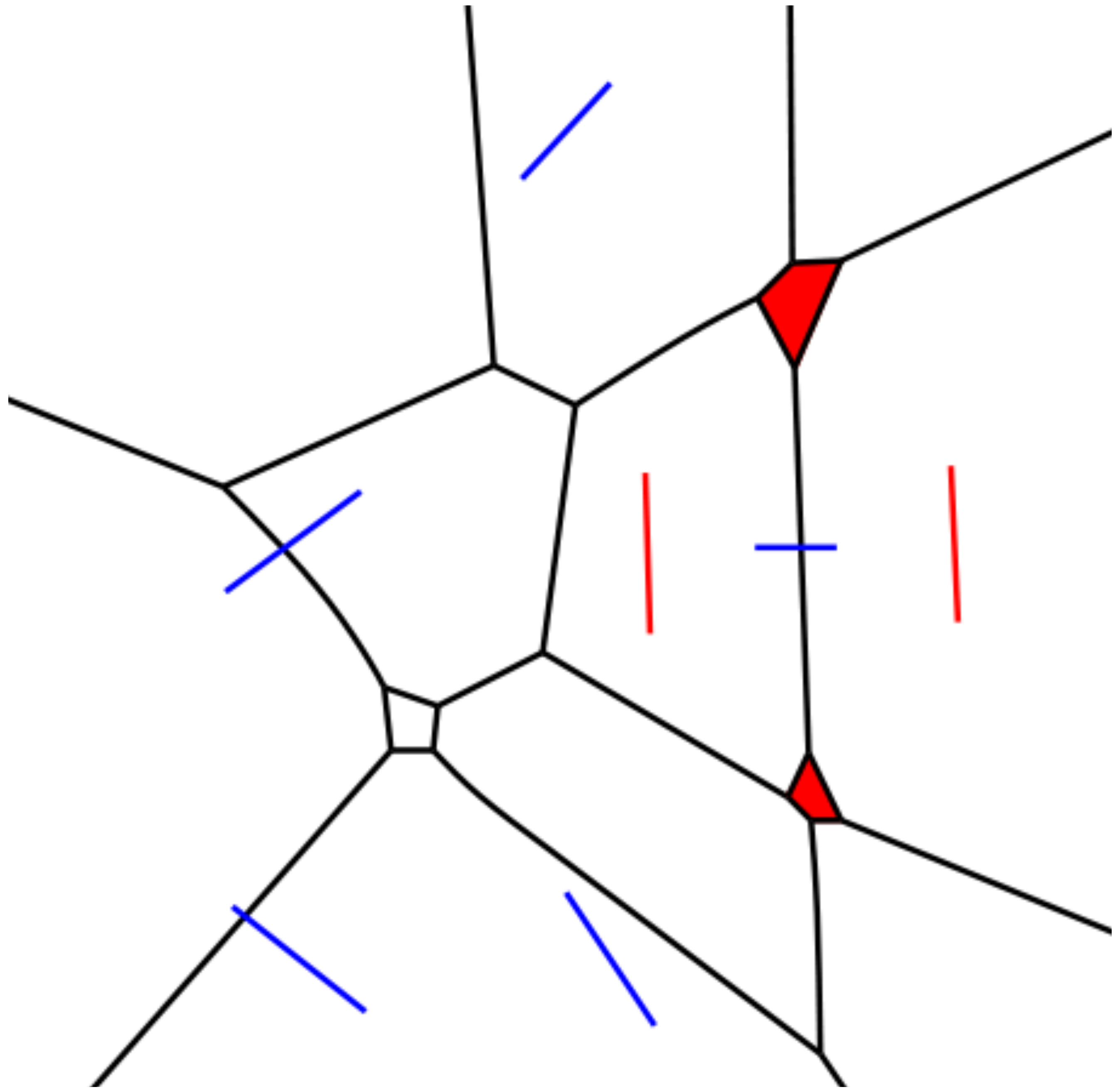


here, k=3

- **Order-k Voronoi region:** locus of points that have the same k closest sites
- **Order-k Voronoi diagram**: subdivision into maximal order-k Voronoi regions

Order-2 Voronoi diagram of segments

- The (order-2) Voronoi region of two segments may be disconnected



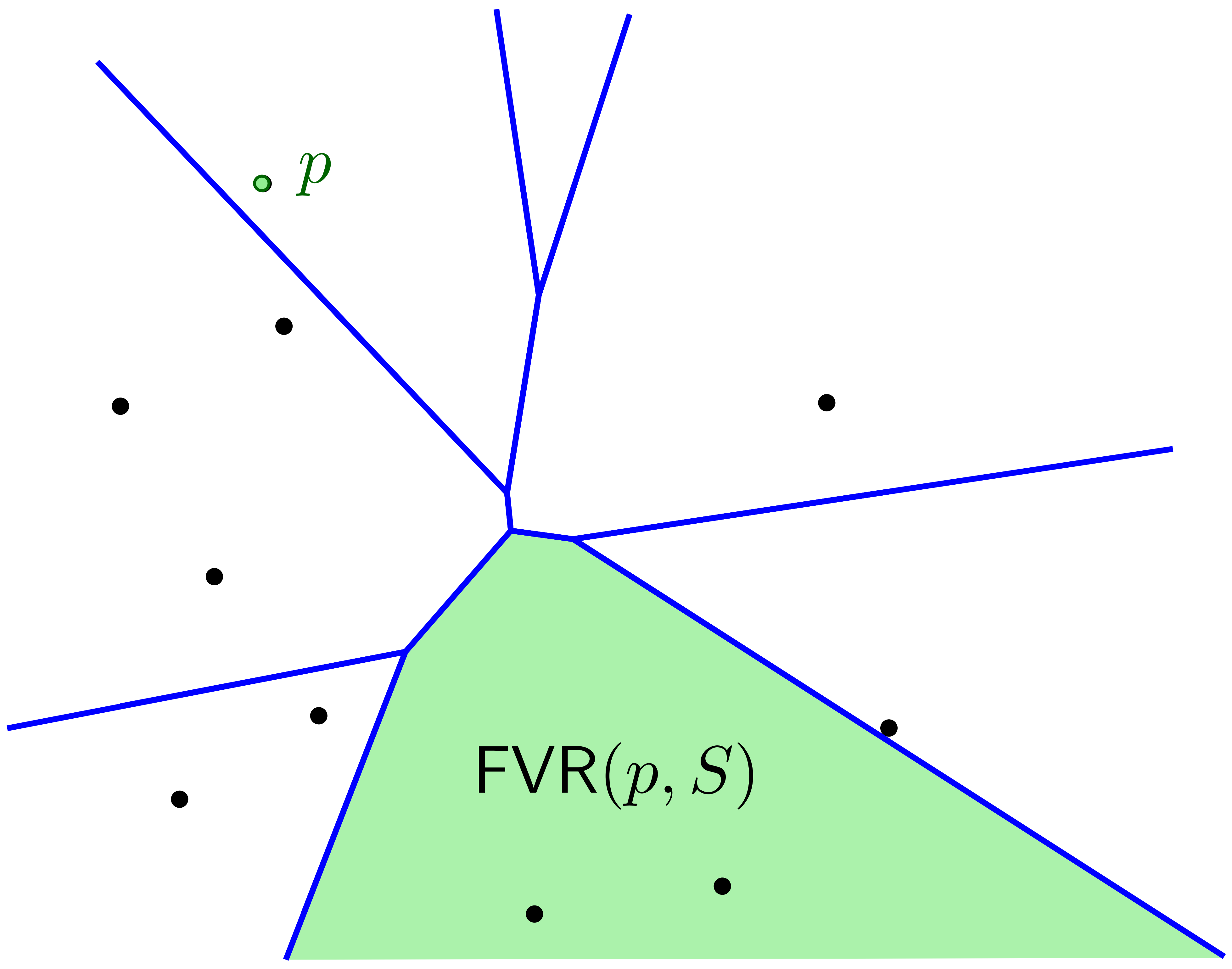
Disconnected regions become a theme for non-point VDs

For points, order-k regions are connected

- **Order-k Voronoi region:** locus of points that have the same k closest sites

Farthest-site Voronoi diagram

- **Farthest Voronoi region** of a site p**:** locus of points further away from p than from any other site.



Point-sites:

only points on the **convex hull** have a non-empty farthest Voronoi region.

FVD: a tree structure

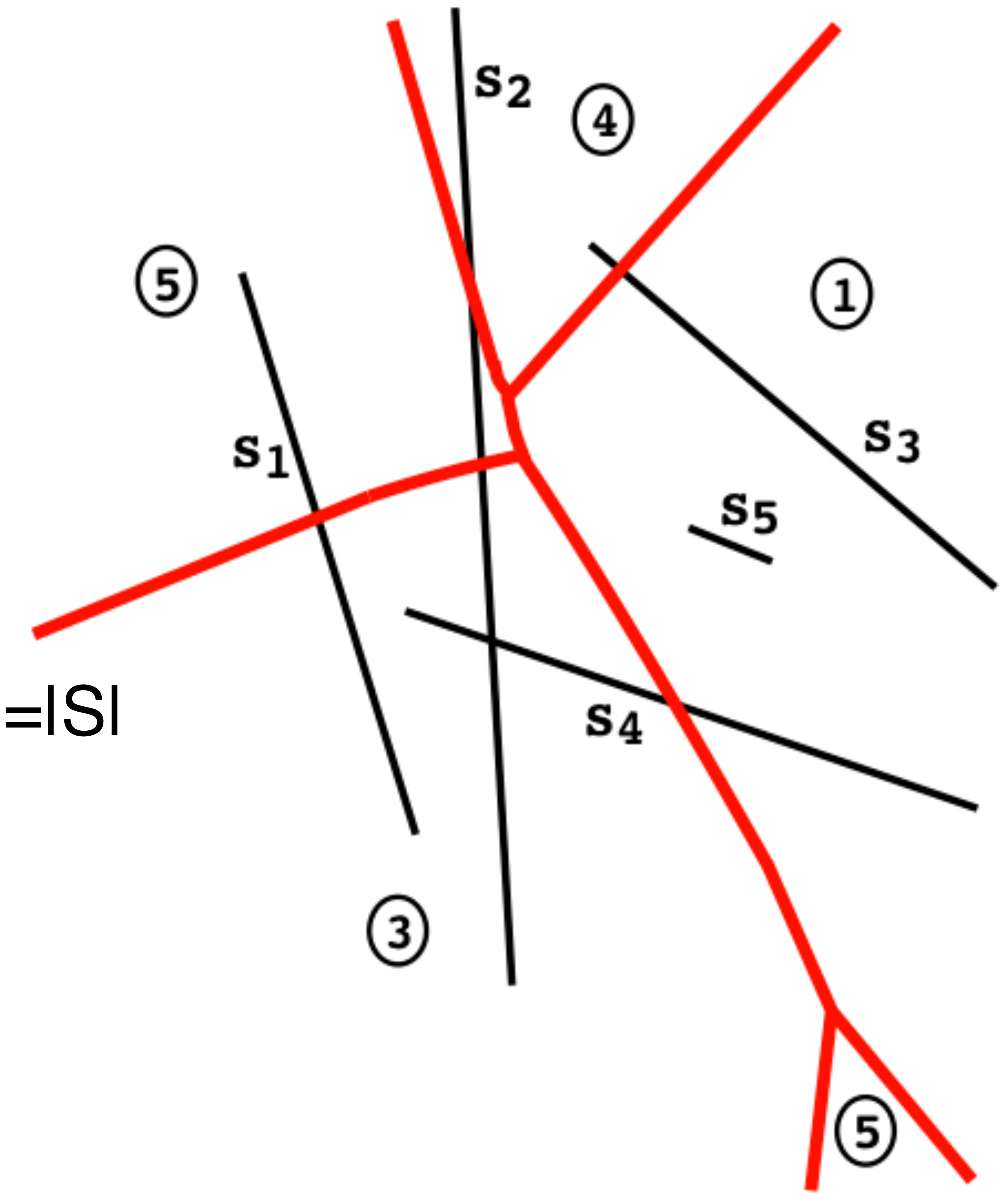
can be computed in **linear time**, after the convex hull is known

Farthest-segment Voronoi diagram

- Properties surprisingly different from points.
  - Not related to convex hull.
  - Disconnected Voronoi regions.
  - A single segment may have **Ω(n)** disconnected faces!

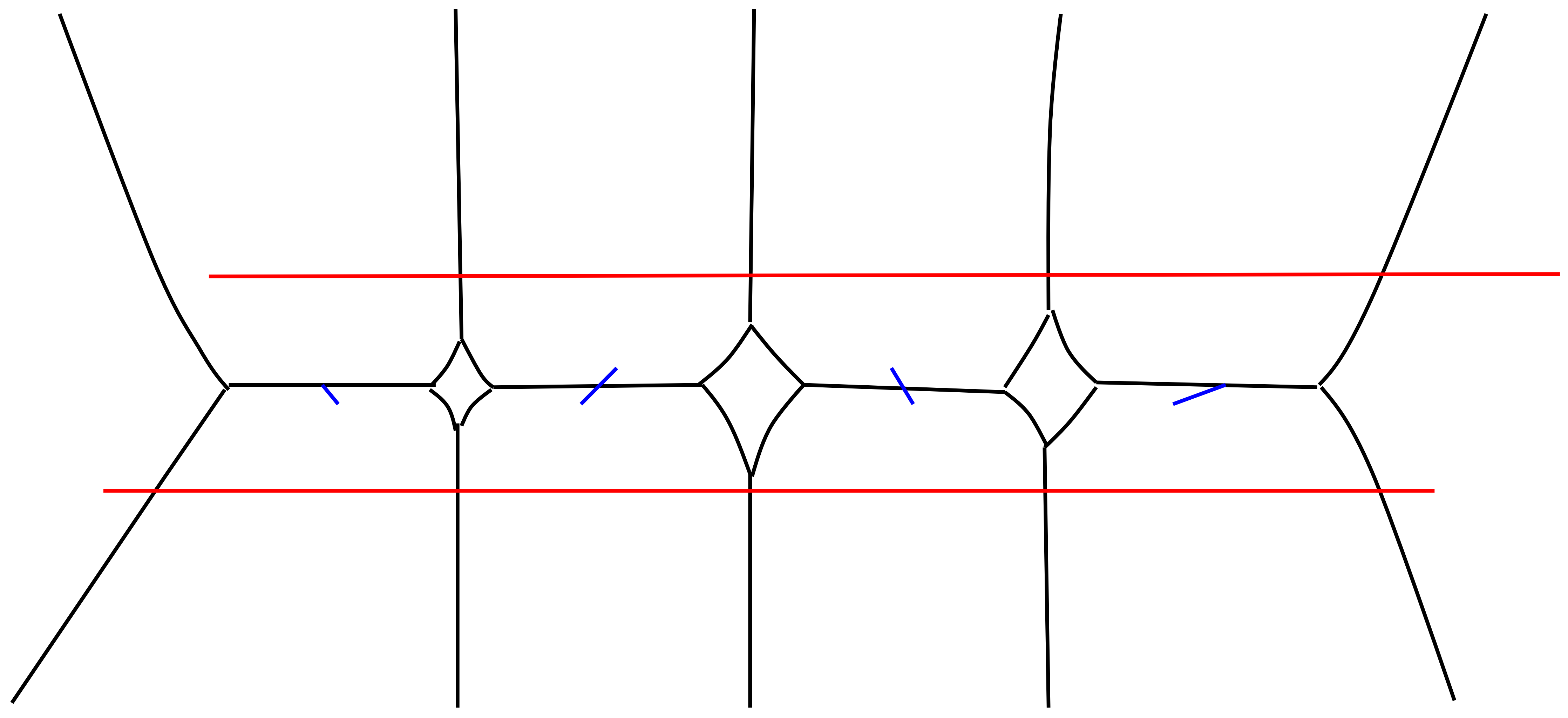
- Tree structure (disconnected regions), size: O(n), n=ISI
- Can be constructed in O(nlog n) time

[Aurenhammer, Drysdale, Krasser, IPL 06]



Order-k segment Voronoi diagram

- A single order-k Voronoi region may disconnect into **Ω(n)** faces
  - **Ω(n-k)** bounded faces; for 1<k<n/2, Ω(n-k) = Ω(n)
  - **Ω(k)** unbounded faces; for k > n/2, Ω(k) = Ω(n)



For points, order-k regions are connected

Order-2 Voronoi diagram of 6 segments

Region of red segments disconnects into 5 faces

[Pap., Zavershynskyi, ’14]

Classic Voronoi diagrams in the plane

- Differences between VDs of points, vs segments/polygons/etc, sometimes forgotten

- Classic variants of VDs for line segments/ polygons/ circles had been surprisingly ignored in CG, until relatively recently
  - farthest segment VD: **[Aurenhammer, Drysdale, Kraser, ’06]**
  - order-k segment VD: **[Pap., Zavershynskyi, ’14]**
  - order-k AVD, defined**: [Bohler, Cheilaris, Klein, Liu, Pap., Zavershynskyi, ’15]**

- Higher-order Voronoi diagrams of polygons are still ignored (current research)
  - only the farthest-polygon Voronoi Diagram has been considered

    [Cheong, Everett, Glisse, Gudmundsson, Hornus, Lazard, Lee, and Na., 2011]

Higher dimensions

- Voronoi diagrams / Delaunay triangulations in higher dimensions have an exponential dependency on the dimension, in the worst case

- For n points in Euclidean d-space the complexity can be $\mathbf{\Theta}\left(\mathbf{n}^{\left\lceil \frac{\mathbf{d}}{\mathbf{2}} \right\rceil}\right)$
  - It is **expected Θ(n)**, if d is a constant **[Dwyer DCG'99]**

- For n lines (or segments) the complexity is a **major open problem**, even in 3D:
  - lower bound $\Omega(n^2)$ **[Aronov 02]**
  - upper bound $O(n^{3+\epsilon})$; **[Sharir DCG'94]**
  - upper bound believed to be near quadratic (open problem)

- Voronoi diagram of line segments / polyhedra in 3D – a major open problem

Powerful unifying framework

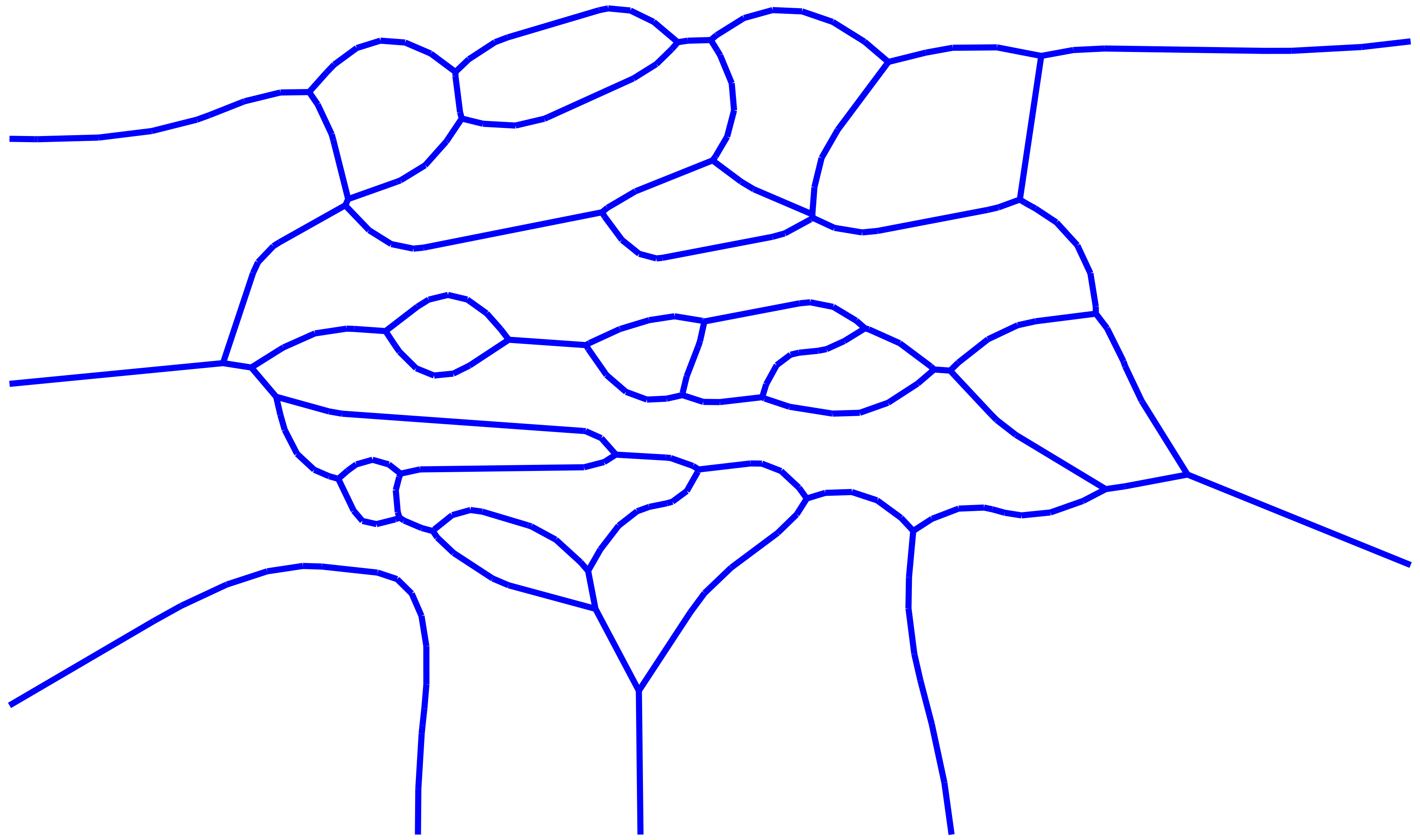
- General framework connecting **Voronoi diagrams** and **arrangements of hypersurfaces**, in a space one dimension higher
  [Edelsbrunner, Seidel, DCG 1986]

  - The set of sites S is a set of indices in a domain X;
  - For each site p, there is a real valued function $f_p\colon \mathrm{X} \rightarrow \mathrm{R}$.
  - The graph of f_p is a hypersurface in $X \times R$: the **Voronoi surface** of site p
  - The Voronoi diagram $\mathcal{V}(S)$ is the **lower envelope** of the arrangement of Voronoi surfaces
  - The order-k Voronoi diagram $\mathcal{V}_{\mathrm{k}}(S)$ is the **level-k** in this arrangement

- Results on envelopes of hypersurfaces directly apply to Voronoi diagrams,
  e.g., [Sharir, DCG 94], [Sharir and Agarwal 95]
- Still, important differences between arrangements of general surfaces vs arrangements of planes

Abstract Voronoi Diagrams (AVDs)

- Defined on bisecting curves satisfying axioms, rather than sites and distances
- Offer a unifying view to various concrete Voronoi diagrams in the plane

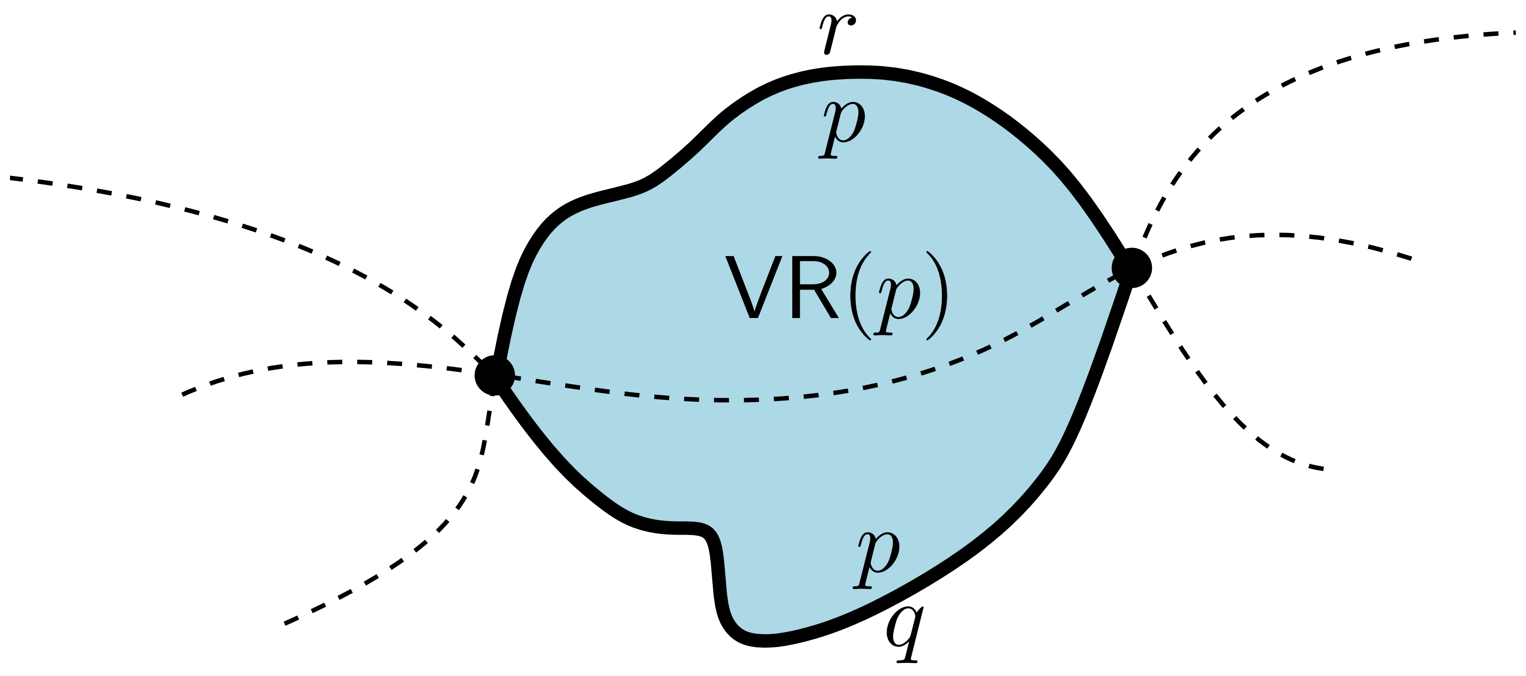


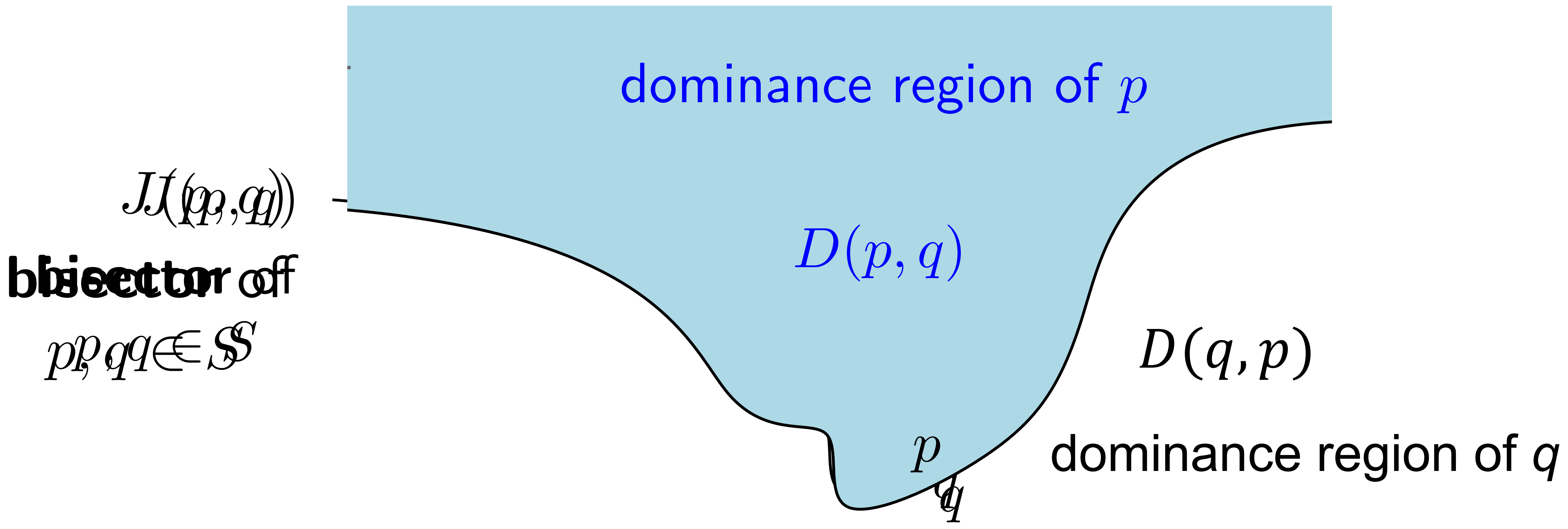
[R. Klein, Concrete and Abstract Voronoi Diagrams, 1989]

Abstract Voronoi diagrams

A bisector system in 2D (abstract sites (indices), no metrics)

- Bisectors are unbounded simple curves.
- Bisectors intersect transversally (a finite # times).
- For every subset of sites S′ ⊆ S:
  - Voronoi regions are non-empty and connected
  - Voronoi regions cover the plane





Voronoi region:

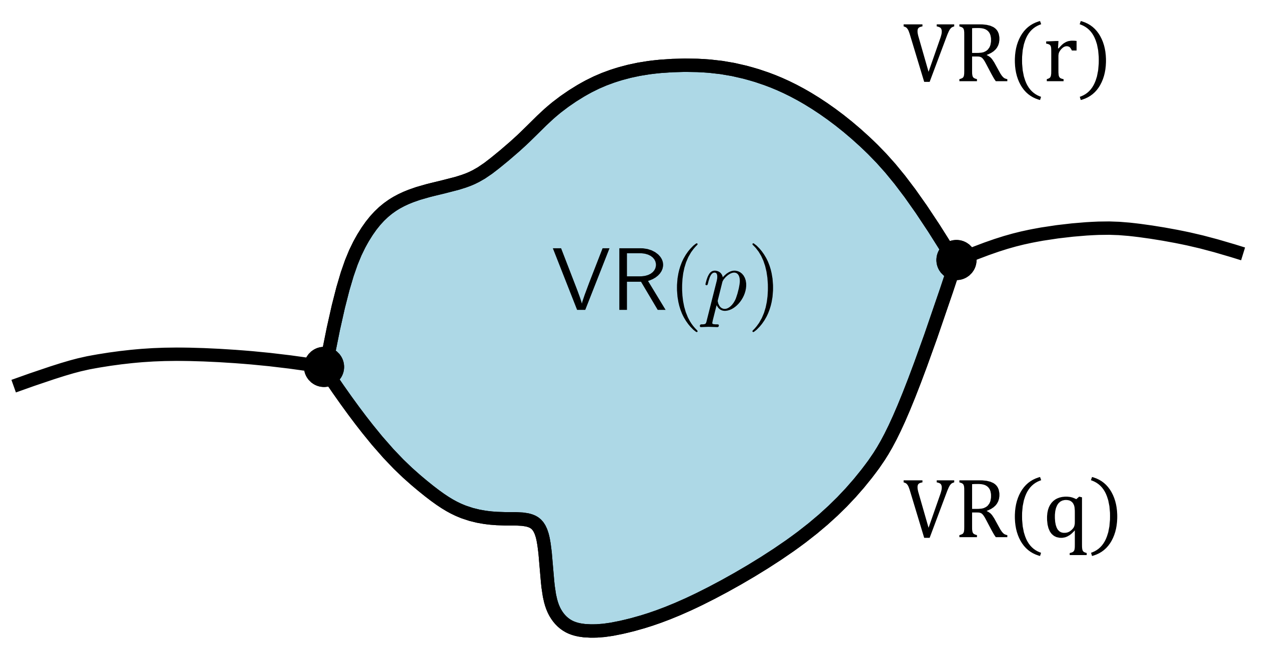
$$\mathrm{VR}(p,S) = \bigcap_{q \in S \setminus \{p\}} D(p,q)$$

[R. Klein, Concrete and Abstract Voronoi Diagrams, 1989]

Abstract Voronoi diagrams

A bisector system in 2D (abstract sites (indices), no metrics)

- Bisectors are unbounded simple curves.
- Bisectors intersect transversally (a finite # times).
- For every subset of sites S′ ⊆ S:
  - Voronoi regions are non-empty and connected
  - Voronoi regions cover the plane



Voronoi diagram:

$$\mathcal{V}(S) = \mathbb{R}^2 \setminus \bigcup_{p \in S} \mathsf{VR}(p, S)$$

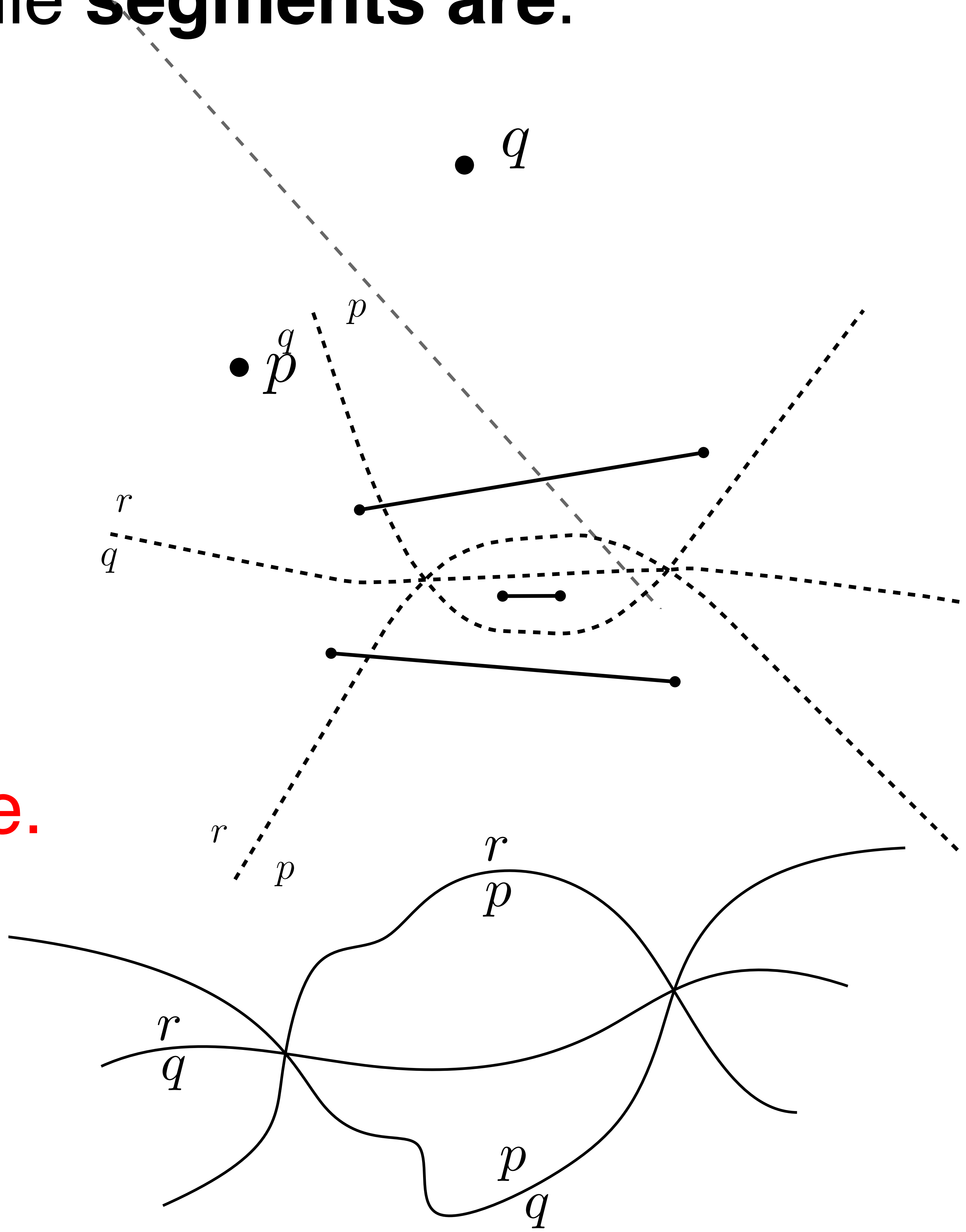
Voronoi region:

$$\mathsf{VR}(p, S) = \bigcap_{q \in S \setminus \{p\}} D(p, q)$$

[R. Klein, Concrete and Abstract Voronoi Diagrams, 1989]

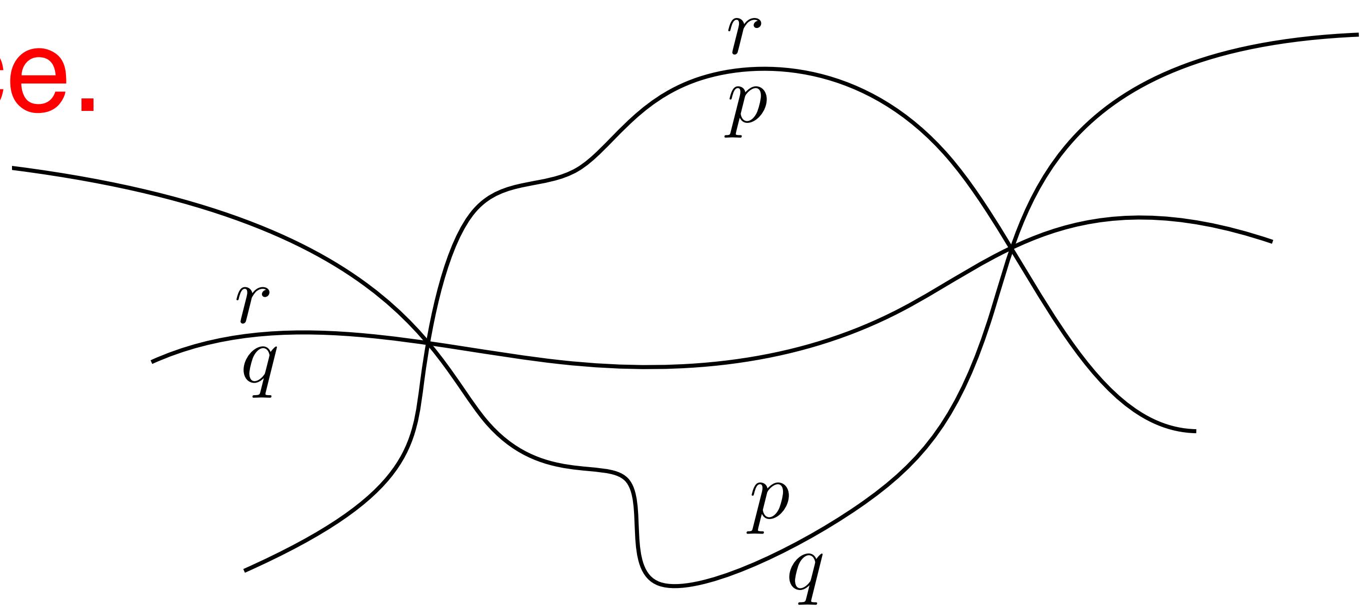
Points vs segments and AVDs

- Point-sites are **not representative** of the AVD model while **segments are**. Why?
- Point bisectors are lines. Intersect once (unless parallel)
- Segment (or circle) bisectors are not even pseudo-lines.
  - Simple curves of constant complexity, not pseudo-lines.
- **Related** segment (circle) bisectors intersect at most twice.
- **Related** abstract bisectors intersect at most twice.



Points vs segments and AVDs

- **Related** segment (circle) bisectors intersect at most twice.
- **Related** abstract bisectors intersect at most twice.



- 2 vs 1 intersections make a significant difference: properties, proof techniques
  - A bound may turn out the same but reasons why can be different
  - Reasons of AVDs/segments apply to points but not vice versa
- **> 2** intersections result in disconnected Voronoi regions – different model

Research Goal

- Generalize algorithmic techniques, combinatorial results, which are available for points, to Voronoi diagrams of generalized sites and metrics
  - These diagrams are often driven by applications, but good tools are still missing, to date

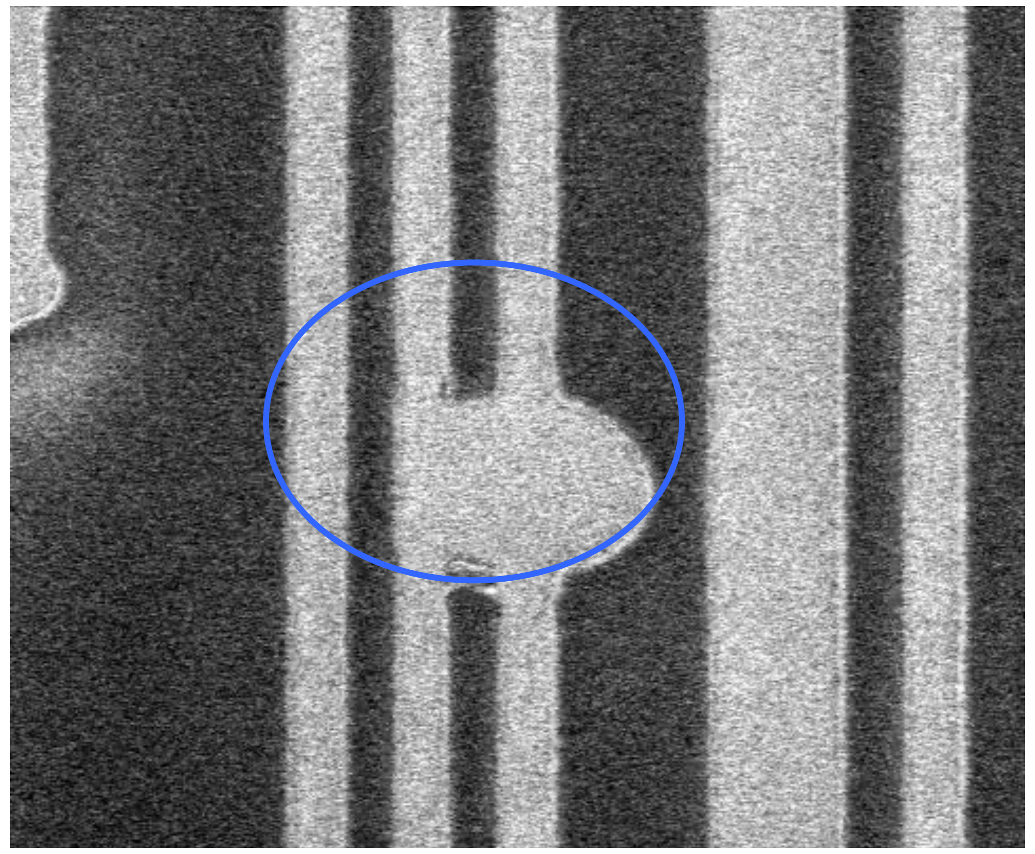
Generalized Voronoi diagrams

- Generalized (non-point) Voronoi diagrams often driven by applications
- Example from Microelectronics: **VLSI Yield Prediction/ Critical Area Analysis**
  - resulted in identifying some surprising holes in Computational Geometry literature, (filled out later)
  - resulted in a VLSI CAD tool (**Voronoi CAA**) used widely in semiconductor industry through Cadence

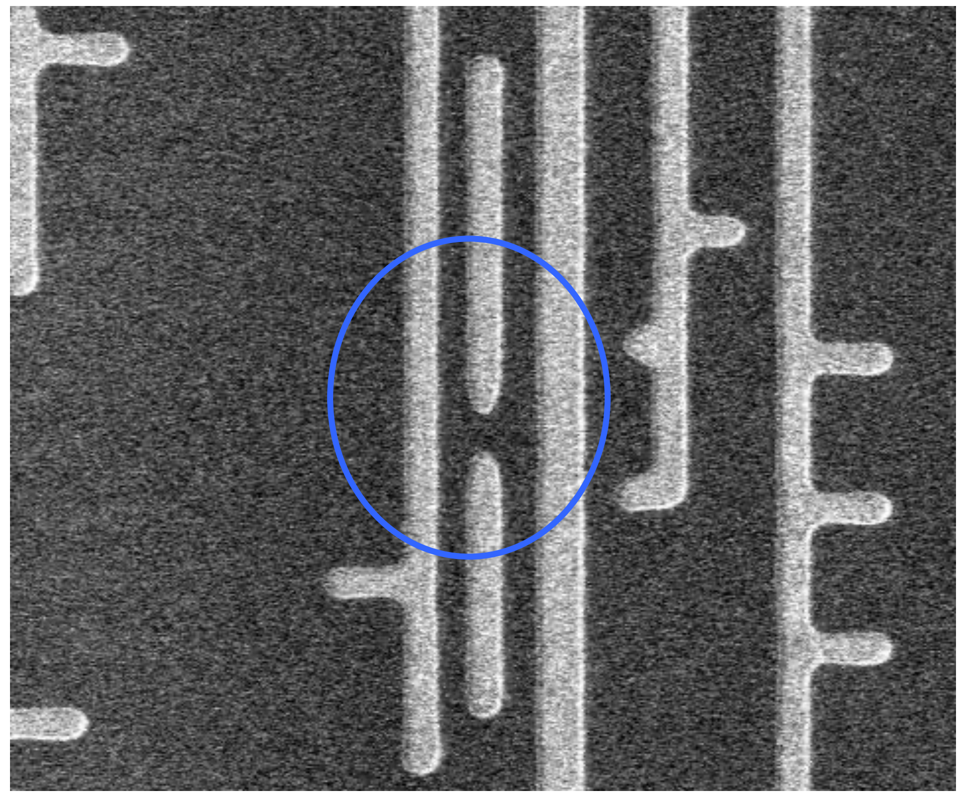
VLSI Critical Area Analysis

- **VLSI Yield**: Percentage of working chips over the chips manufactured
  - Factors of Yield loss: Random defects and Systematic defects
- **Random defects**: dust/contaminants on materials and equipment
- Prediction of yield loss due to random defects: **Critical Area Analysis**
- **Critical Area**: Measure reflecting the sensitivity of a VLSI design to random defects during manufacturing
  - Now a solved problem – but still essential to IC manufacturing
- **VLSI Layout**: layers of different materials; each layer a collection of shapes; manufacturing: optical processing layer by layer

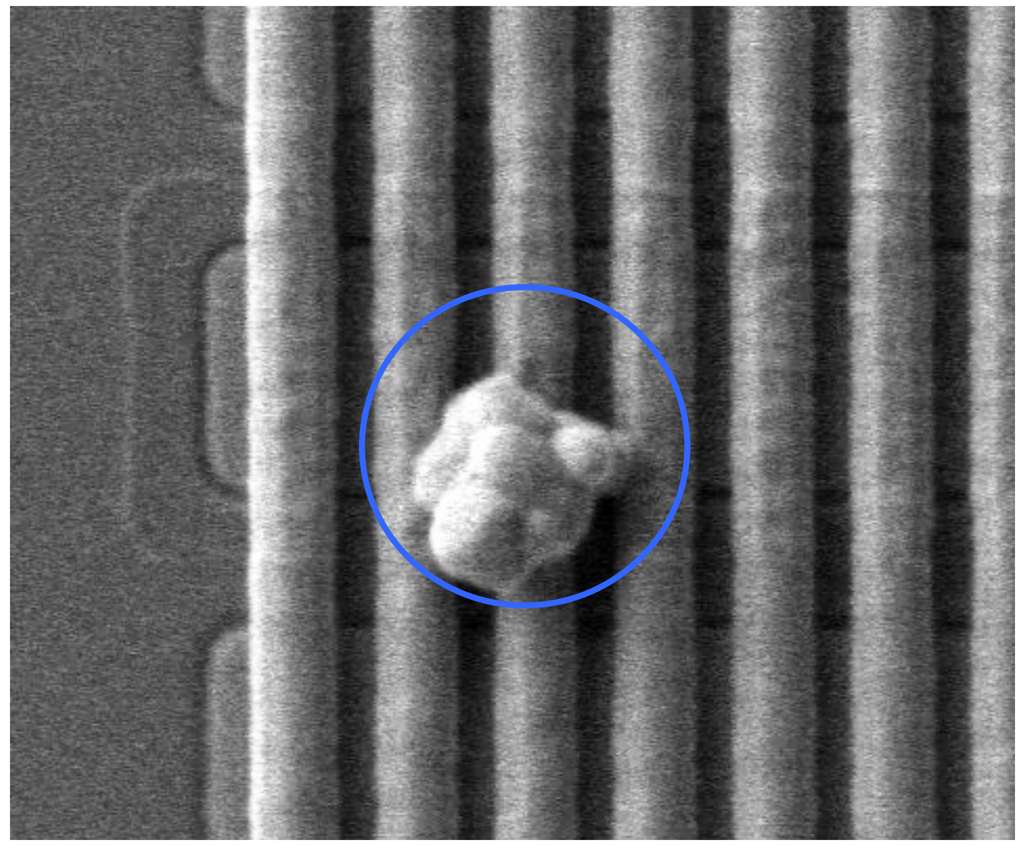
Examples of faults due to random defects



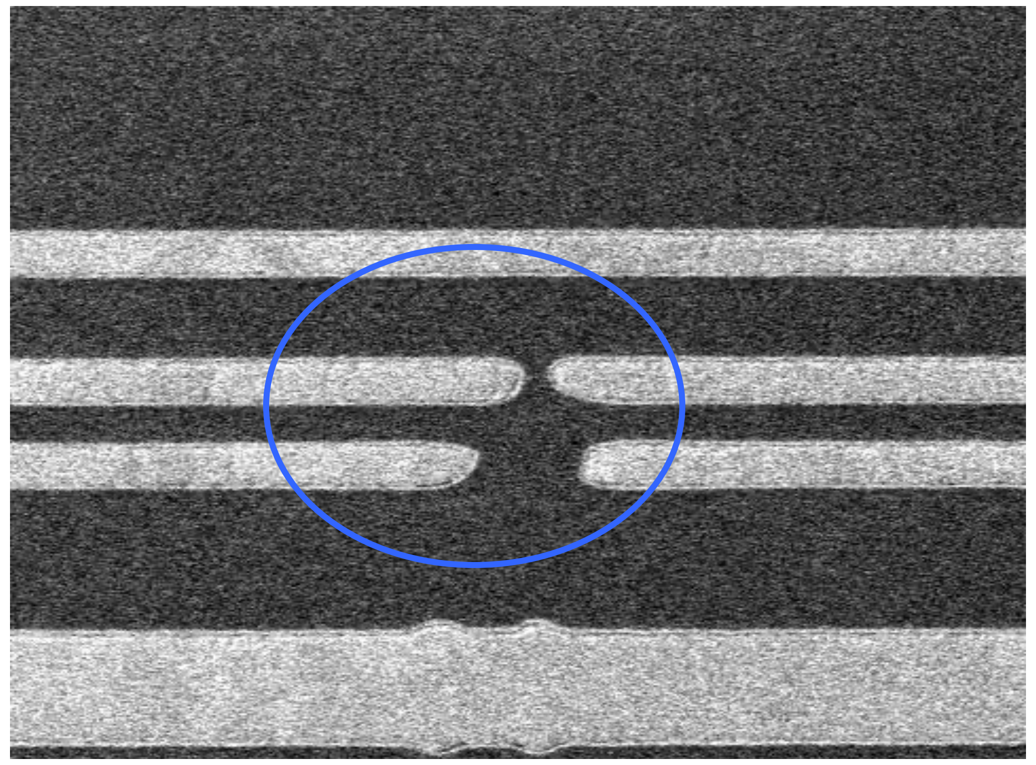
Shorted Metal



Open Metal



Foreign Material Short



Open Metal

Critical Area

- **Critical Area**:

$$A_c = \int_0^\infty A(r) D(r) dr$$

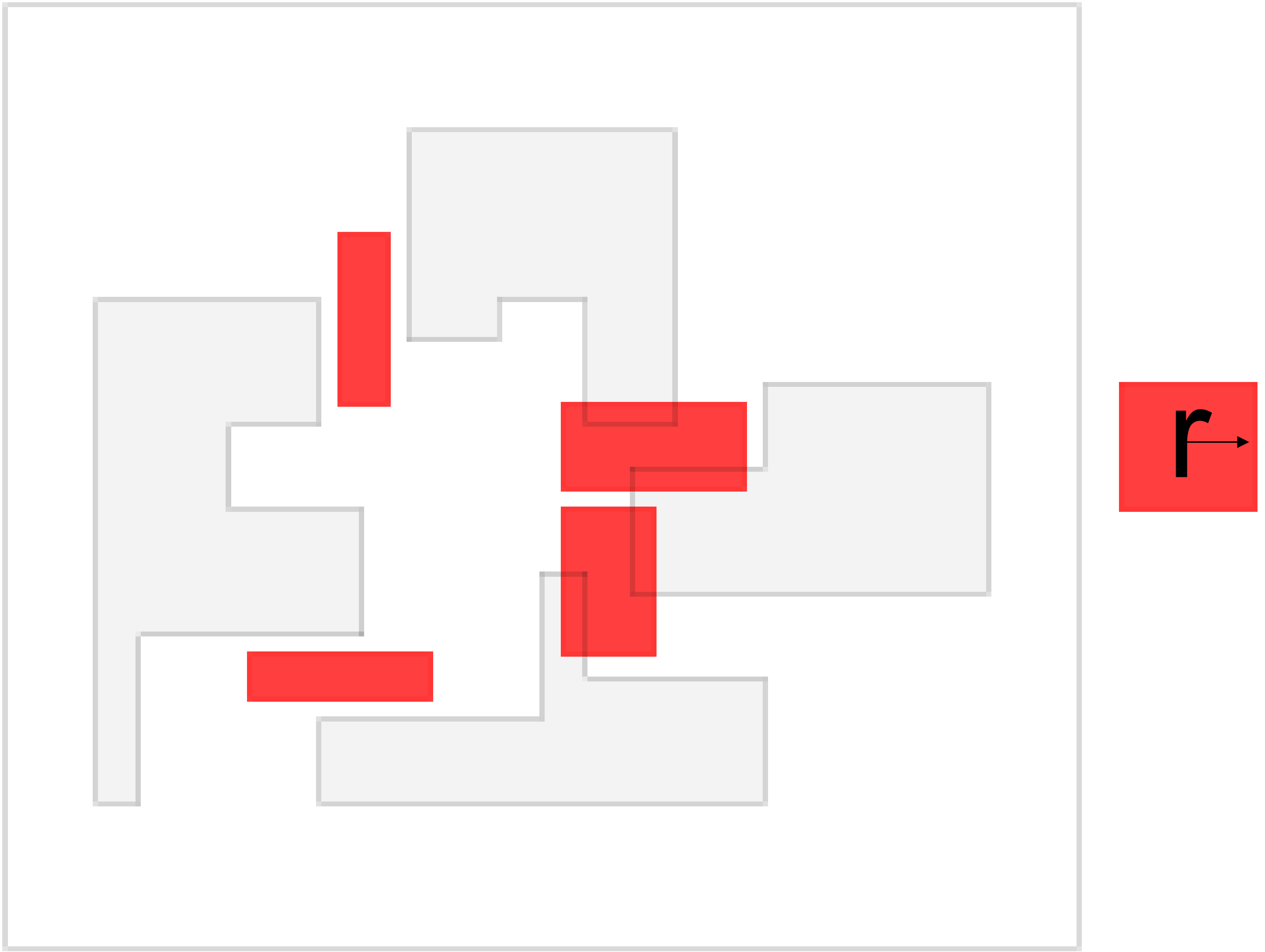
$A(r)$: area where if a defect of radius r is centered causes a circuit failure

$D(r)$: density function of the defect size

$$D(r) = \frac{r_0^2}{r^3}$$

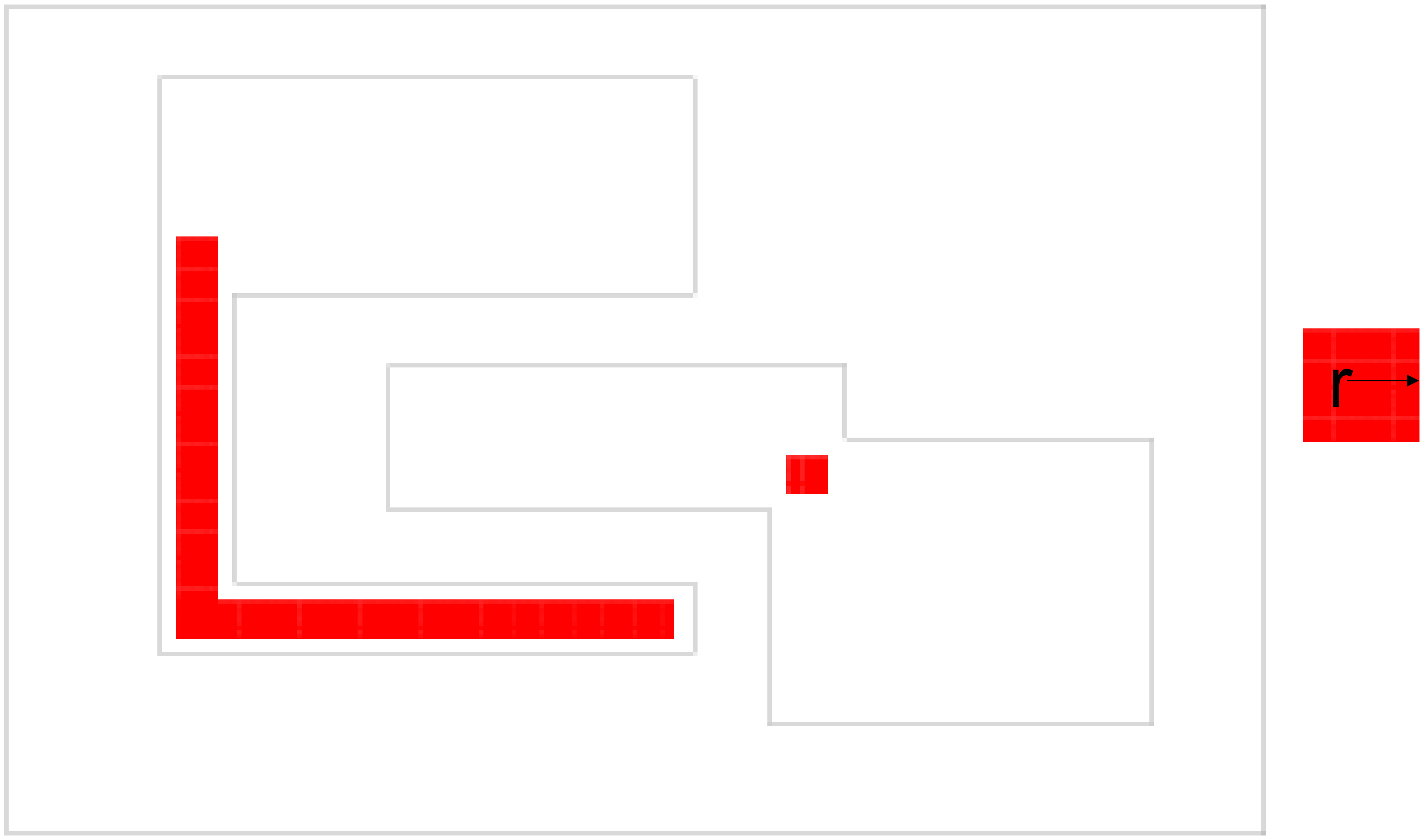
Defect of size r = disk of radius r

A(r) -- **shorts** for **one** defect size r



Critical Area $A_c = \int_0^\infty A(r)D(r)dr \quad \text{where} \quad D(r) = r_0^2/r^3$

A(r) – open faults for **one** defect size r



Critical Area $A_c = \int_0^\infty A(r)D(r)dr \quad \text{where} \quad D(r) = r_0^2/r^3$

Methods to compute Critical Area

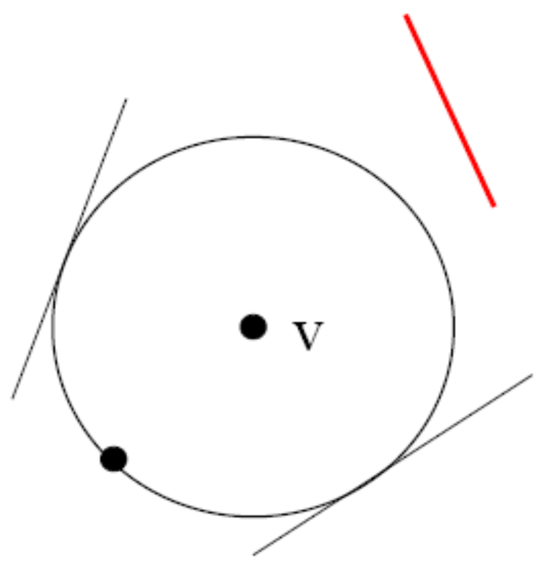
- **Monte Carlo** simulation [Initial work at IBM [e.g. Stapper & Rosner Trans. Semic. Manuf. 95)]
  - Randomly draw large number of defects following *D*(*r*); check for faults
  - Oldest, widely implemented technique. Computationally, very intensive

- **Shape shifting** methods [AFFCA '95 , Allan& Walton TCAD99, Zachariah & Chacravarty TVLSI 00]
  - Based on **shape expansion / shrinking - m**any variants
  - Very expensive to compute *A*(*r*) for medium/large *r,* needed in integration.

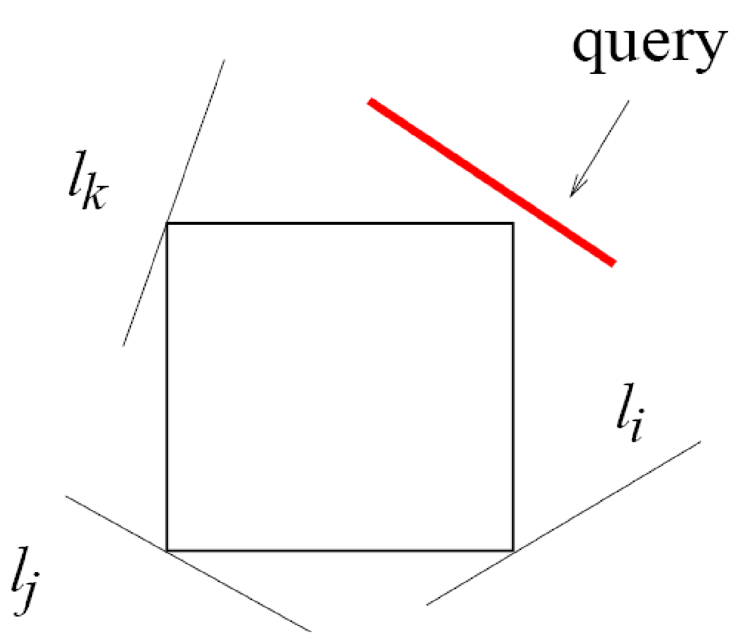
- **The Voronoi method** [P. & Lee TCAD99, P. TCAD01, P. TCAD11, various patents]
  - **Idea**: partition layout into regions where critical area integral can be computed analytically
  - Critical area computation is easy (trivial) once appropriate Voronoi diagram derived
  - Combined with layout sampling techniques for fast critical area estimate at chip level

L_∞ metric

- **Algorithmic degree**
  - Degree *d*: tests - evaluation of multivariate polynomials of arithmetic degree ≤ *d*.



In-circle test (segments): degree ≤ 40
[Burnikel 96]



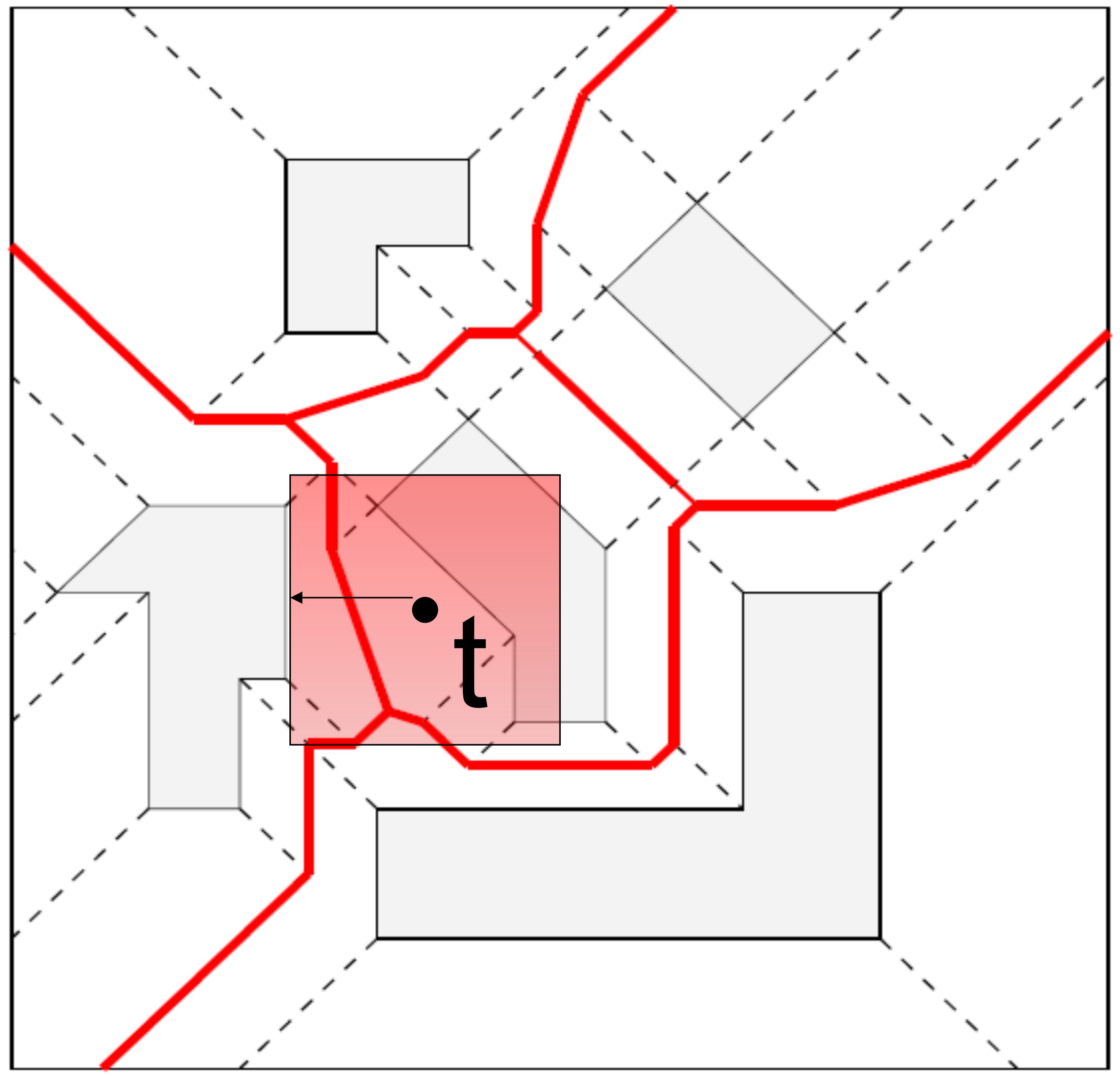
L_∞ in-circle test (segments): degree ≤ 5
[Papadopoulou & Lee IJCGA 01]

VLSI shapes: typically, ortho-45: degree 1

- L_∞ Voronoi diagram construction: significantly lower algorithmic degree
  - Robust, faster, easier to derive implementation

Shorts

- A defect on layer A forms a **short** if it overlaps two different shapes in different nets
- **Critical radius** of any point t: size of smallest defect centered at t causing a fault.



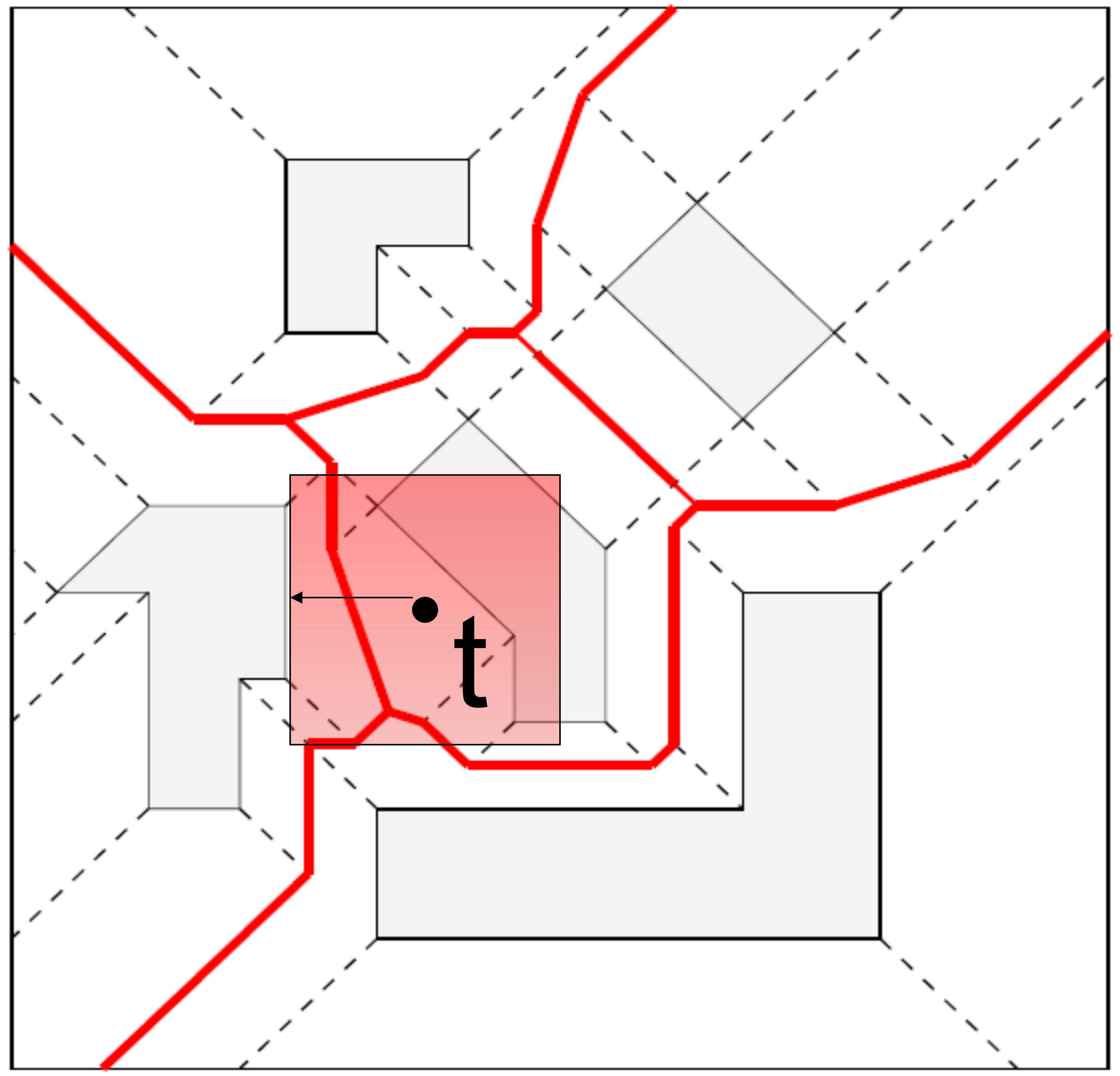
Model defects as squares ⇒ L_∞ metric

Simplicity in computation

Much lower algorithmic degree

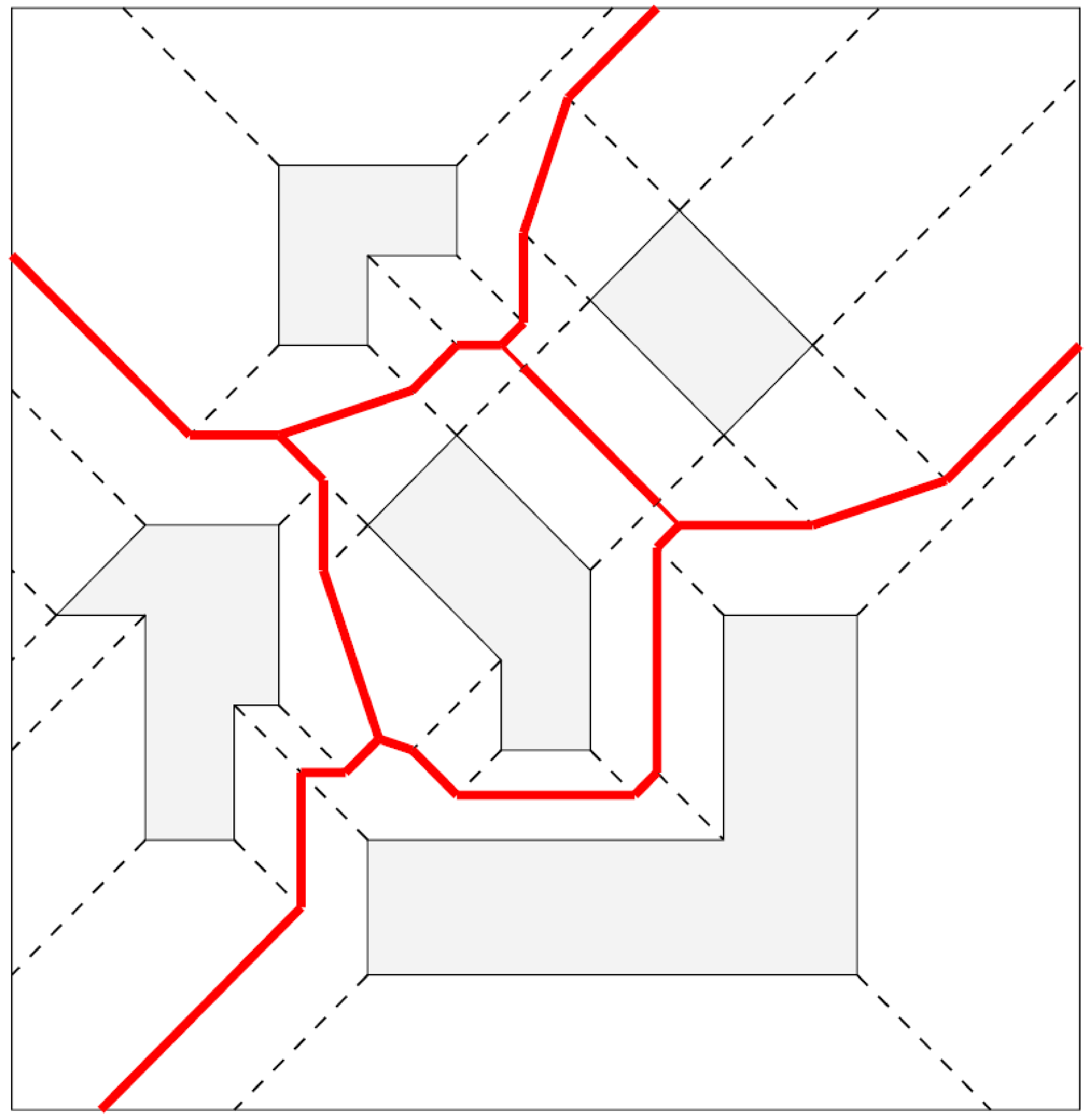
Shorts

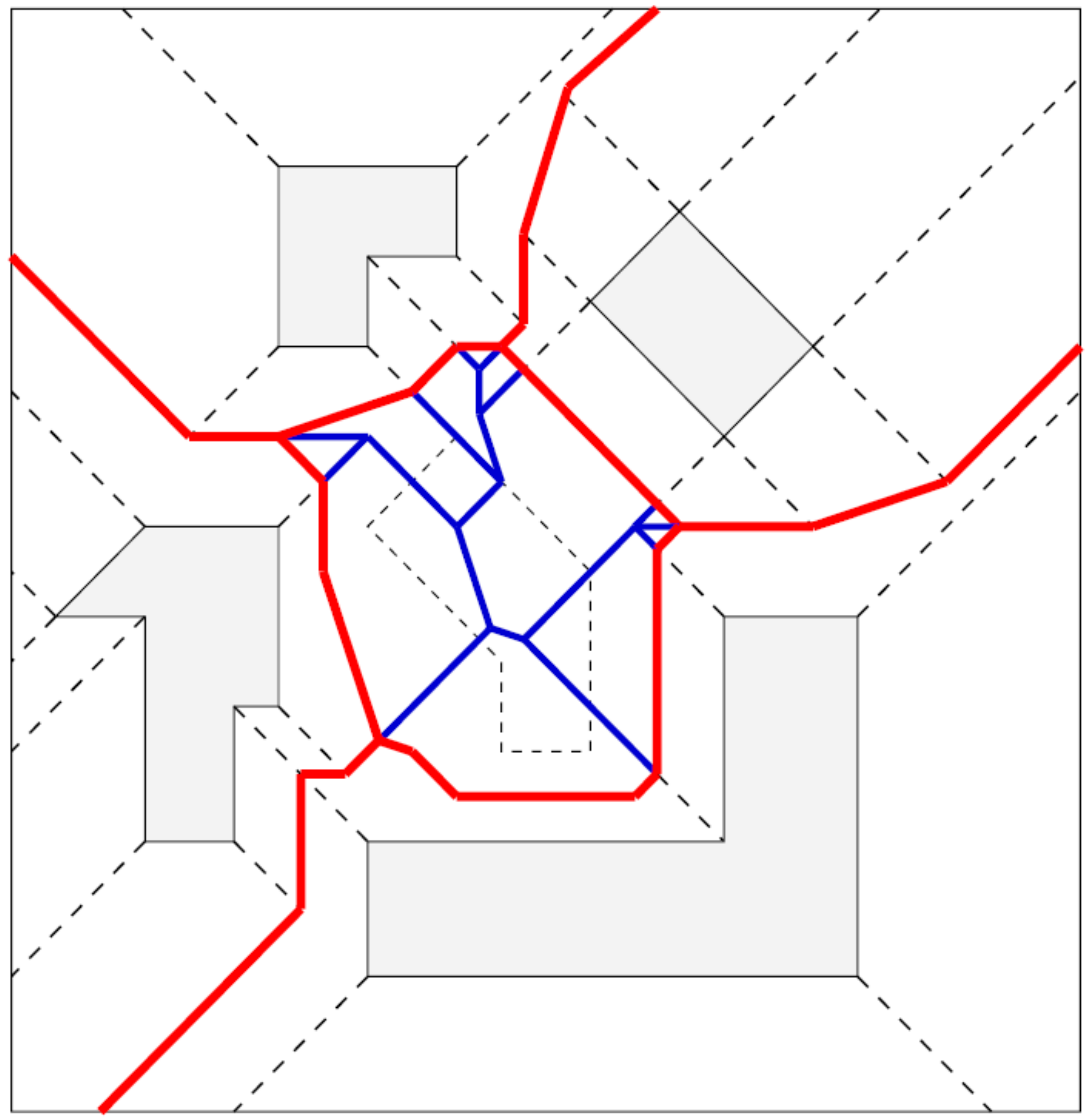
- **Critical radius**: distance from 2nd nearest polygon (in different net)
- **Need:** 2nd nearest neighbor information



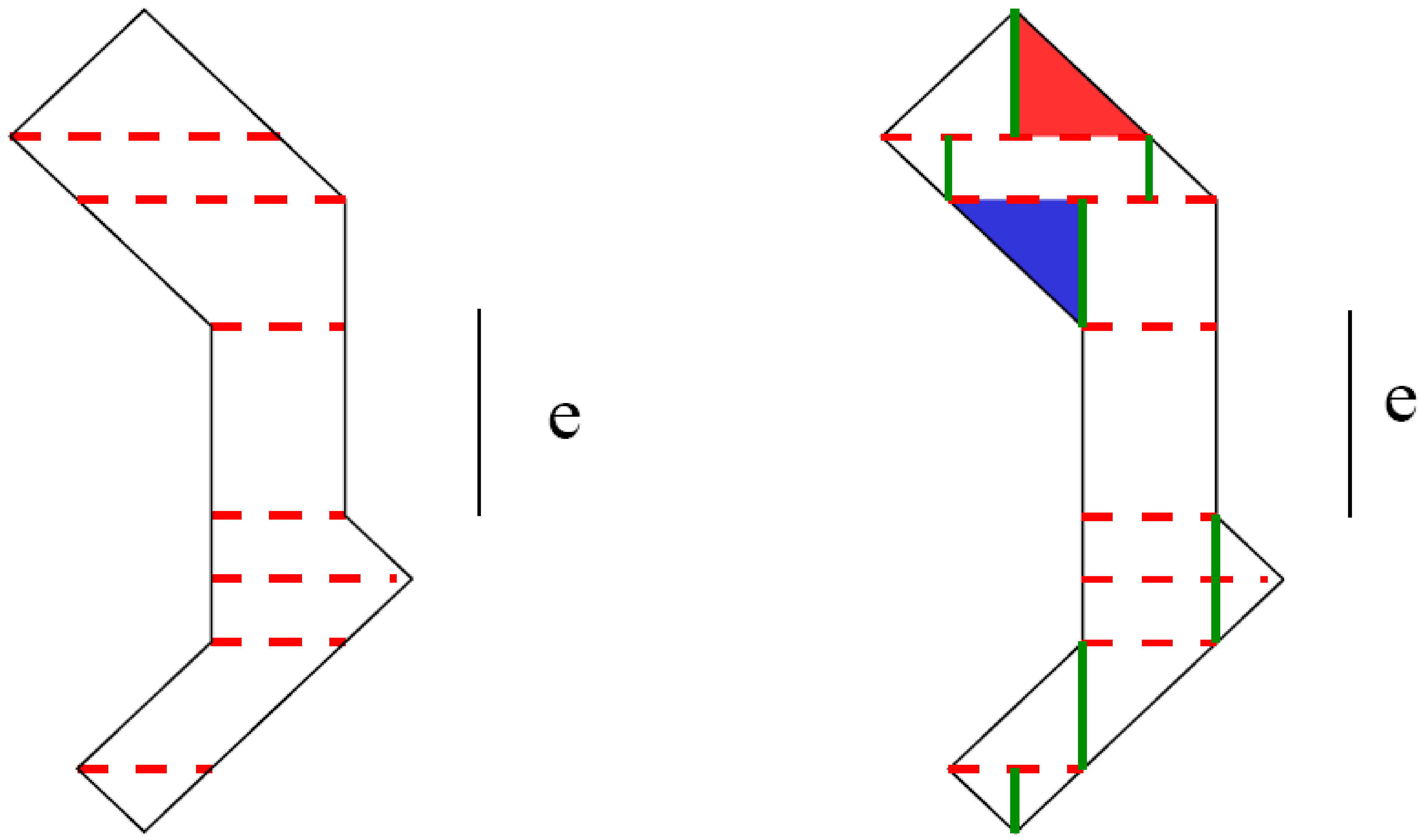
Voronoi diagram for shorts

- **2nd order Voronoi diagram of polygons**
  - Every region has a unique owner responsible for shorts within region
  - Critical radius at any point t: distance to owner of region



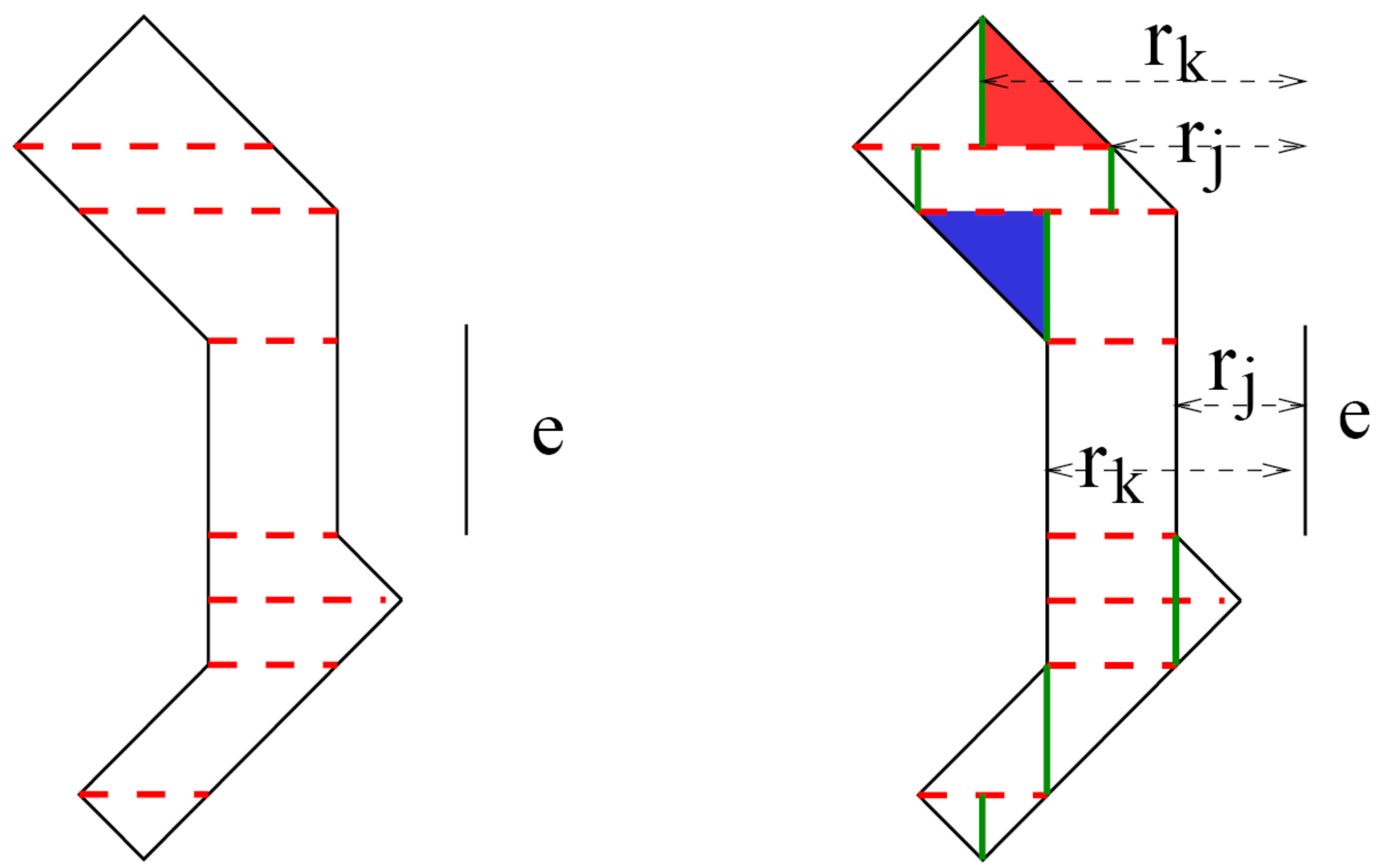


Critical Area Integration within a Voronoi region



- Subdivide Voronoi region into simple rectangles/ triangles
- Compute critical area within each analytically
- Add up formulas to derive critical area for entire region

Critical Area Integration within a Voronoi region



$$A_c(\mathcal{R}) = \frac{r_0^2}{2}\left(\frac{l}{r_j} - \frac{l}{r_k}\right)$$

$$A_c(T_{red}) = \frac{r_0^2}{2}\left(\ln\left(\frac{r_k}{r_j}\right) - \frac{l}{r_k}\right)$$

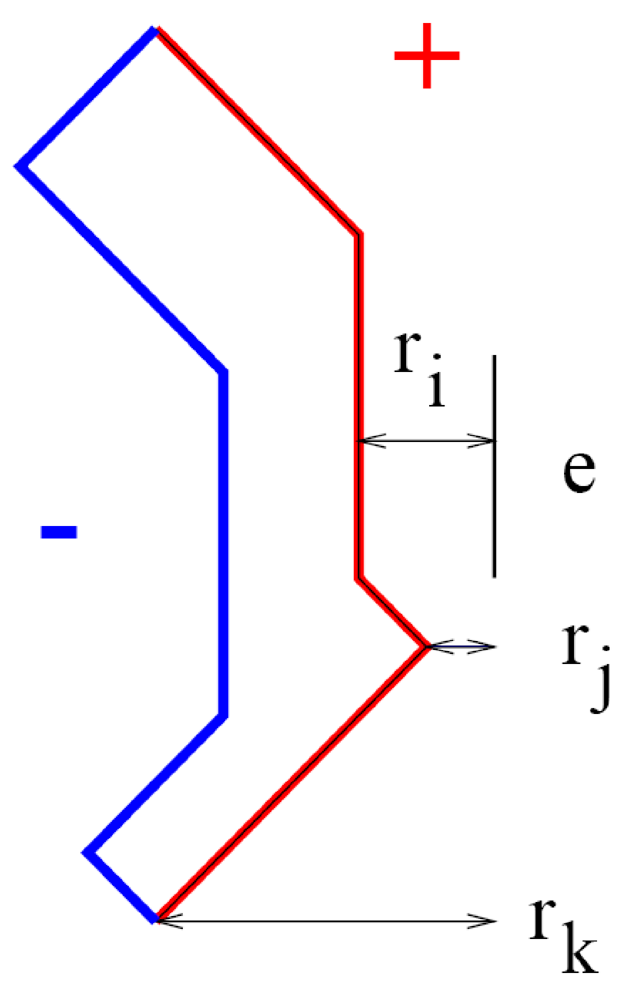
$$A_c(T_{blue}) = \frac{r_0^2}{2}\left(\frac{l}{r_j} - \ln\left(\frac{r_k}{r_j}\right)\right)$$

l = length of vertical side, r_k = max critical radius, r_j = min critical radius

Add up formulas $\Rightarrow$ internal terms $\frac{l_i}{r_i}$, $\ln\frac{r_k}{r_j}$ cancel out

Critical Area = Summation of Voronoi edges

35



Critical area within V:

$$A_c(V) = \frac{r_0^2}{2}\left(\sum_{red\ e_i}\frac{l_i}{r_i} - \sum_{blue\ e_m}\frac{l_m}{r_m} + \sum_{red\ e_{45}}\ln\frac{r_k}{r_j} - \sum_{blue\ e_{45}}\ln\frac{r_k}{r_j}\right)$$

Critical area computation: trivial once the Voronoi diagram computed

Critical Area via Voronoi diagrams

- **Shorts**: $A_c \leq$ 2nd order Voronoi diagram of polygons
- **Simple Open Faults**: $A_c \leq$ Voronoi diagram of (additively weighted) segments
- **Via Blocks**: $A_c \leq$ Hausdorff Voronoi diagram (a Voronoi diagram of point clusters)
- **General Open Faults**: $A_c \leq$ Higher order Voronoi diagram of (weighted) segments
  - Analytical Critical Area integration – no error
  - O(n log n) – type of algorithms in most cases
- All are variants of **generalized Voronoi diagrams of polygons**
  - Higher order Voronoi diagrams of segments/shapes had not been available in CG literature
- IBM **Voronoi CAA** CAD tool – (licensed to Cadence, used extensively in industry)

Research Goal

- Generalize algorithmic techniques or combinatorial results, which are available for points, to generalized Voronoi diagrams

- Example: **linear-time** algorithms to compute **tree Voronoi diagrams**

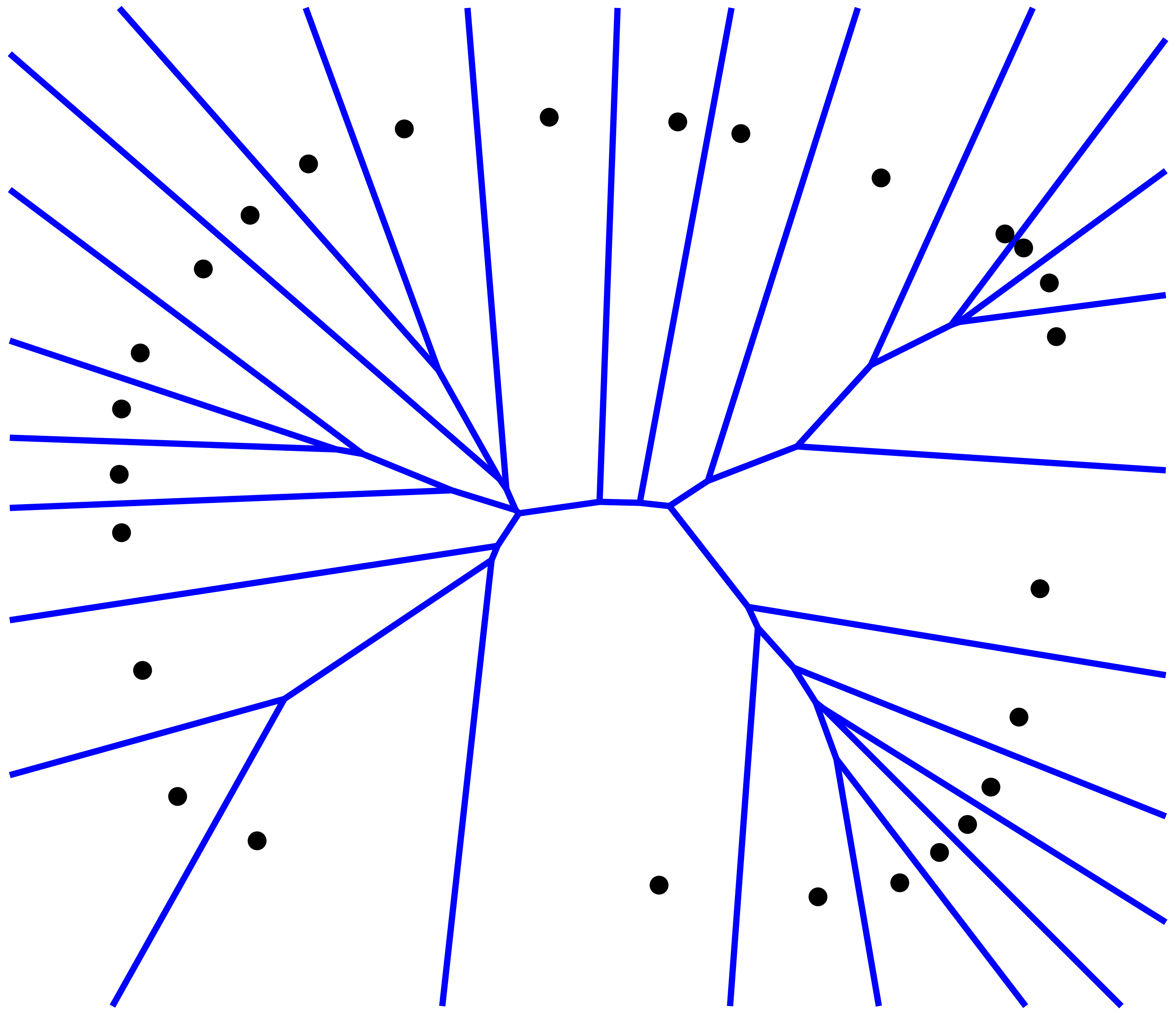
E.g., **Delaunay triangulation of a convex polygon** – very simple randomized incremental algorithm by **[Chew 1990]**

Linear-time Voronoi algorithms

- Voronoi diagram of points in convex position – a **tree diagram**

  [Aggarwal, Guibas, Saxe and Shor, DCG'89]

  [Chew 1990] randomized



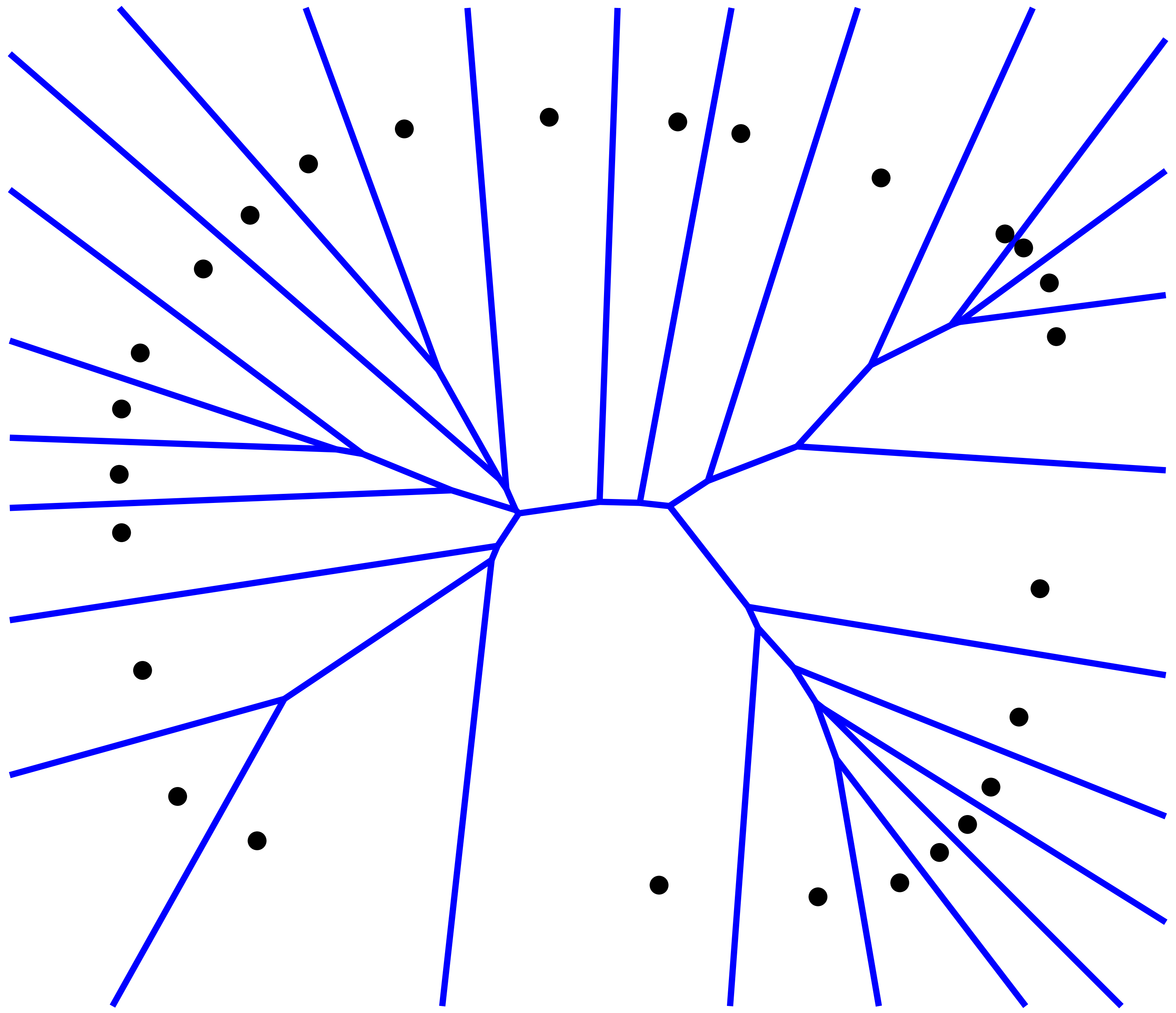
Related problems:
- Delaunay triangulation of a convex polygon
- Site deletion in a point VD
- Farthest-point VD, given the convex hull
- Iterative order-k Voronoi construction

Non-point-sites ? Segments? AVDs?

Linear-time Voronoi algorithms

The randomized incremental algorithm of Chew is extremely simple:

- Consider a random permutation of the input points
- **Phase 1**: delete points 1-by-1, recording their neighbors at the time of deletion
- **Phase 2**: Insert points 1-by-1 in reverse order, updating their VD after each insertion

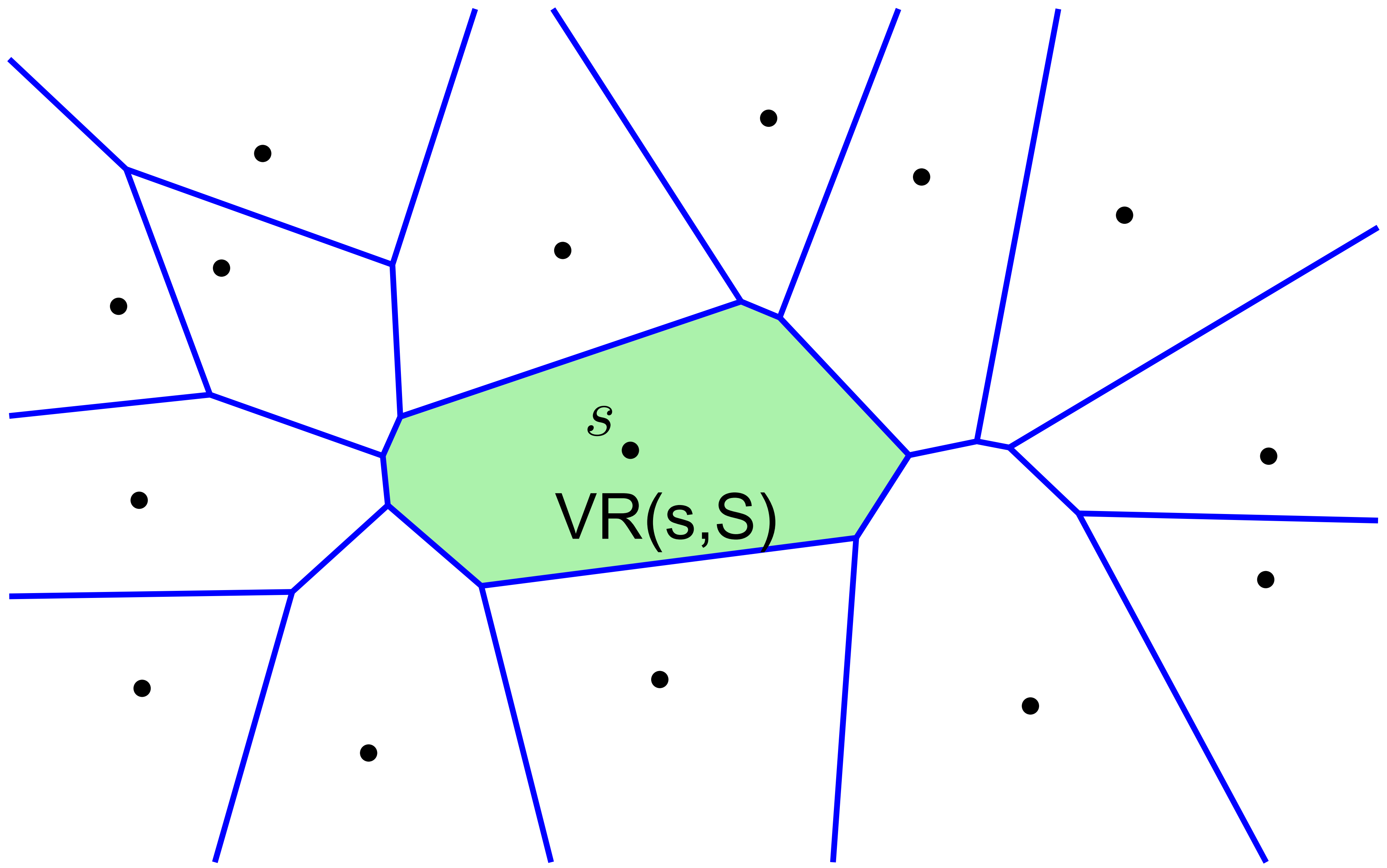


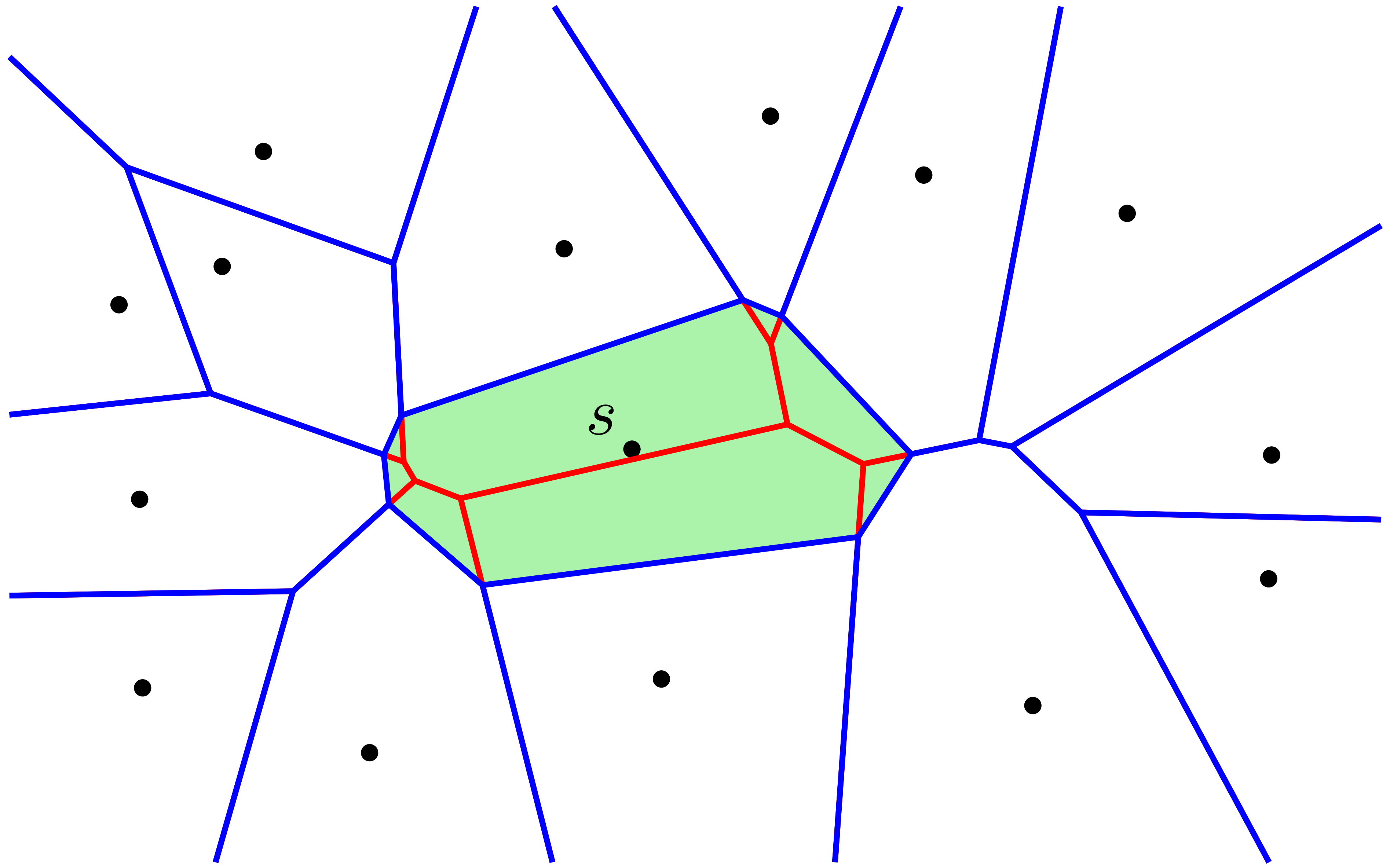
insertion point given by the stored neighbors – no point-location

Each insertion performed in expected O(1) time

Site deletion

- Given the Voronoi diagram VD(S) of a set of sites S, **delete** the region of a site **s** and update the diagram
- Compute the red diagram in VR(s,S), which is VD(S\{s}) ∩ VR(s,S) (a **tree** for point sites)



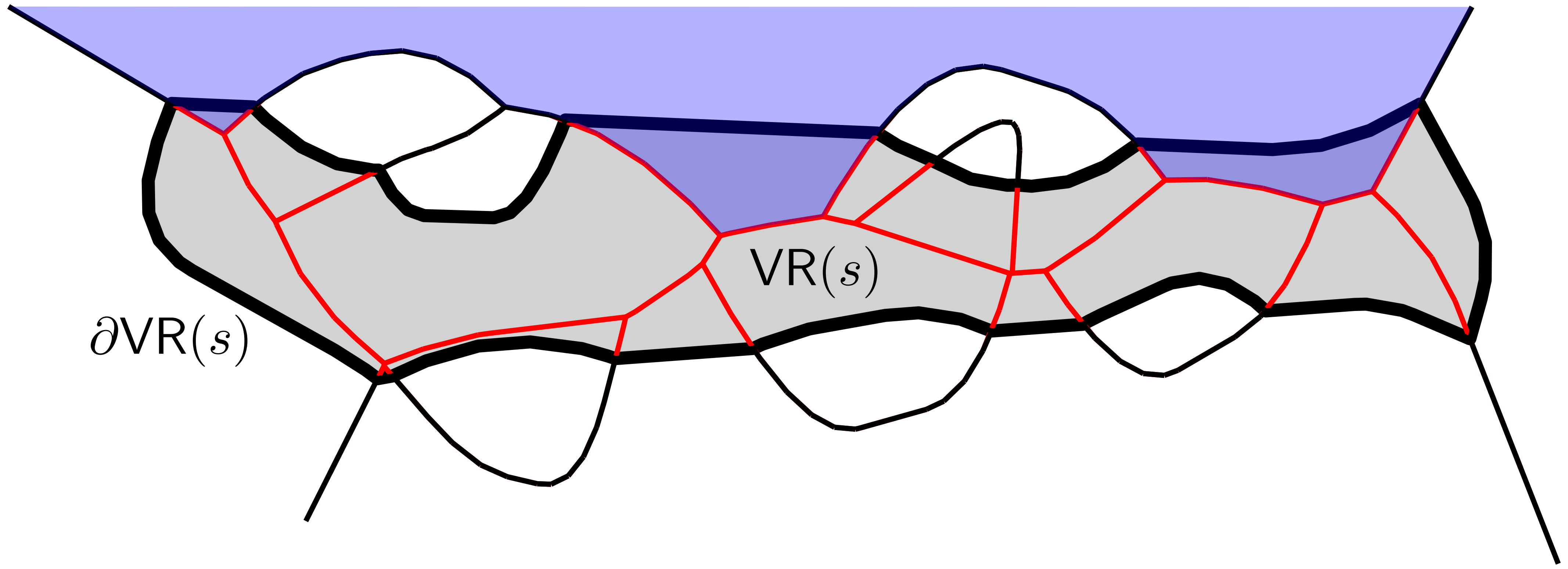


Site deletion – non-points

For non-point sites (e.g., line segments, circles, AVDs), considerably more difficult problem

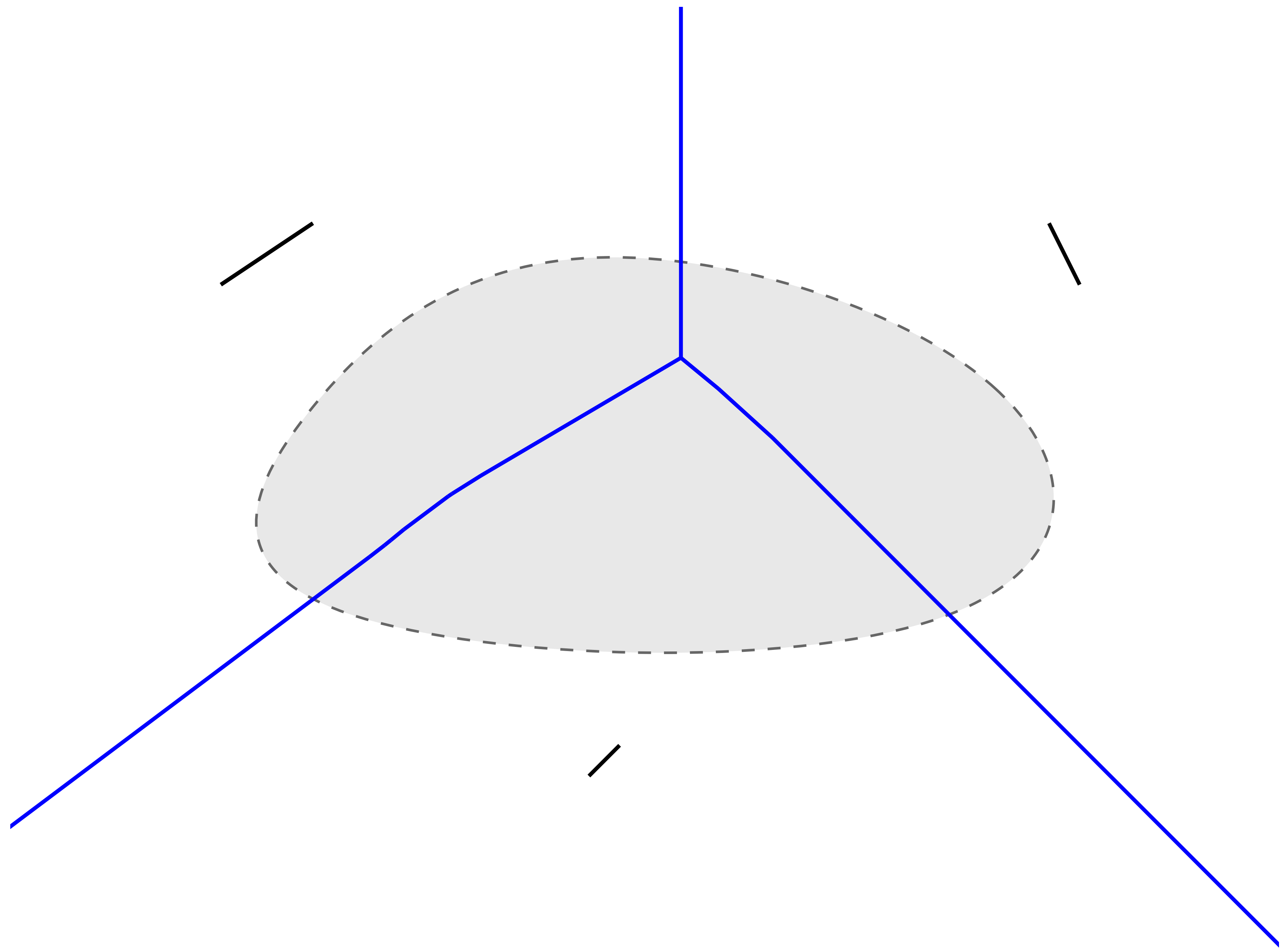
- open problem since the late 80's
- randomized linear time algorithm for AVDs **[Junginger, Pap., SoCG 2018]**
- deterministic linear-time algorithm still an open problem

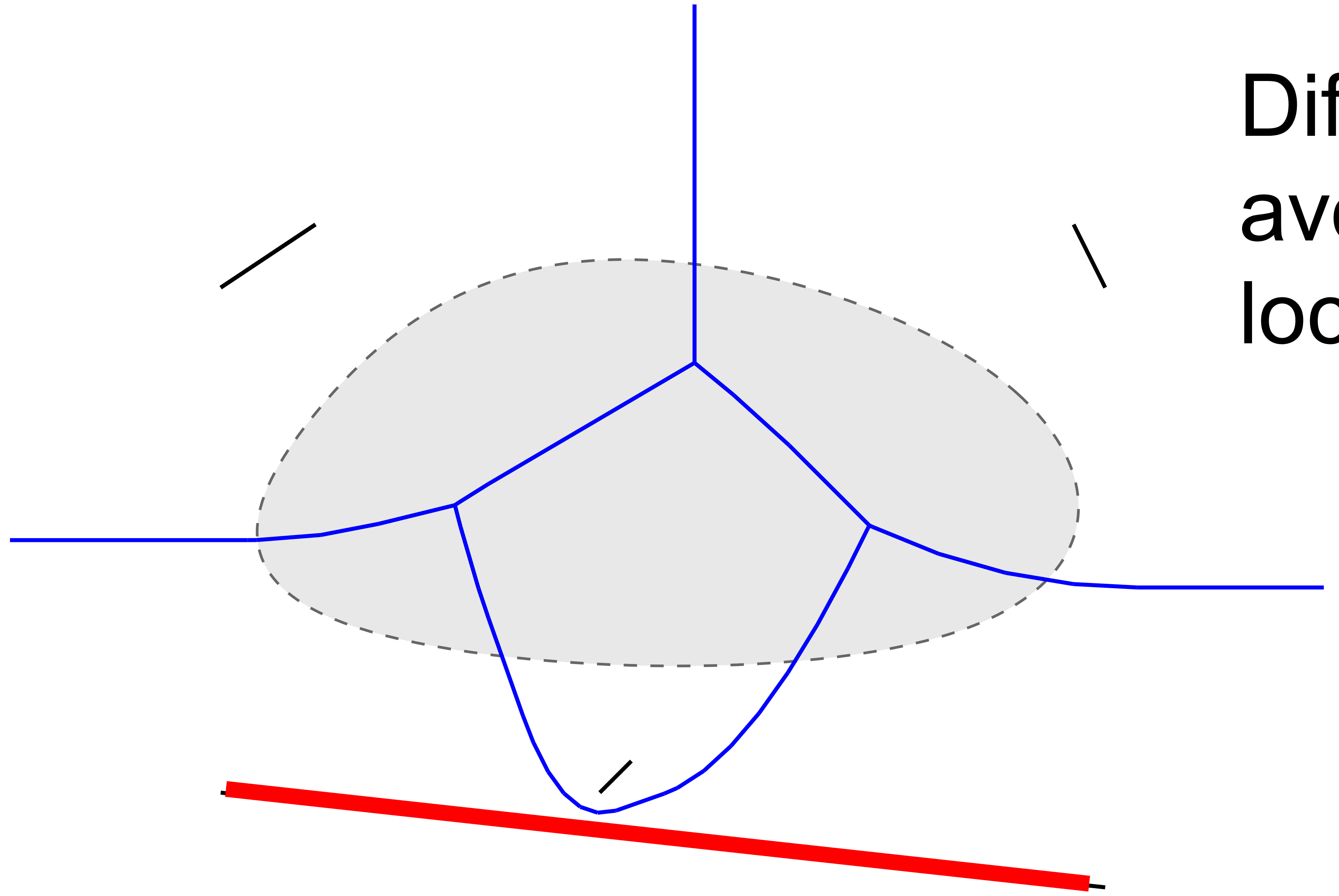
- Why difficult?
- **Disconnected Voronoi regions**



Incremental construction

- Compute the Voronoi diagram of segments in the shaded domain incrementally (a tree)
- When we consider a new segment, **many faces** may need to be inserted
- Step i may trigger the insertion of **Θ(i) new** faces in the diagram, whose location we do not know in advance

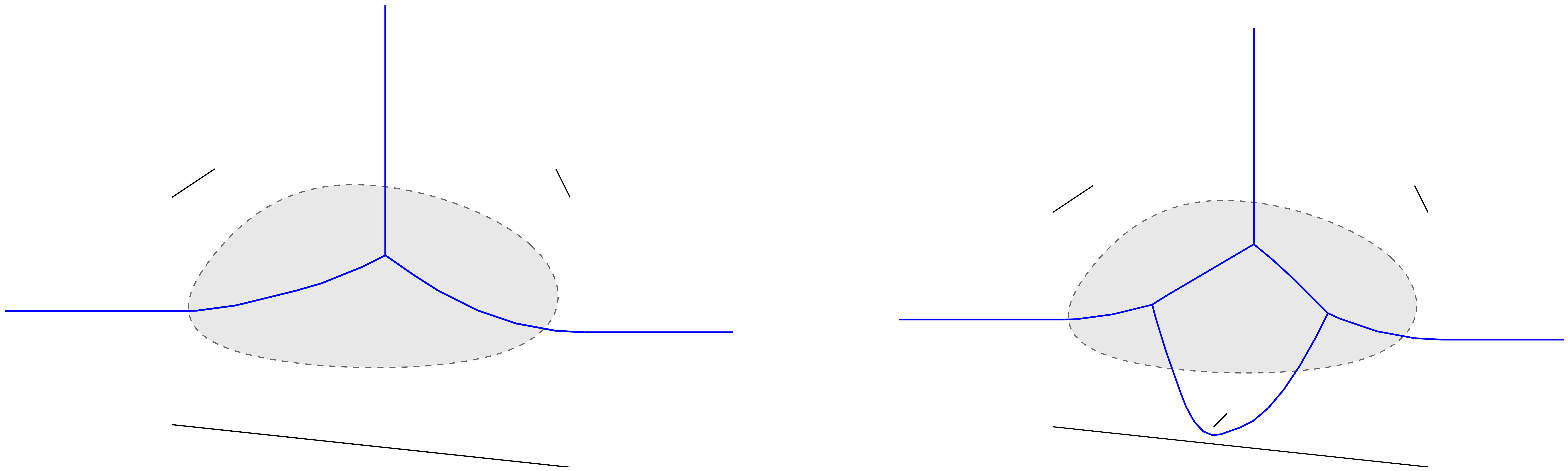




Difficult to avoid point location

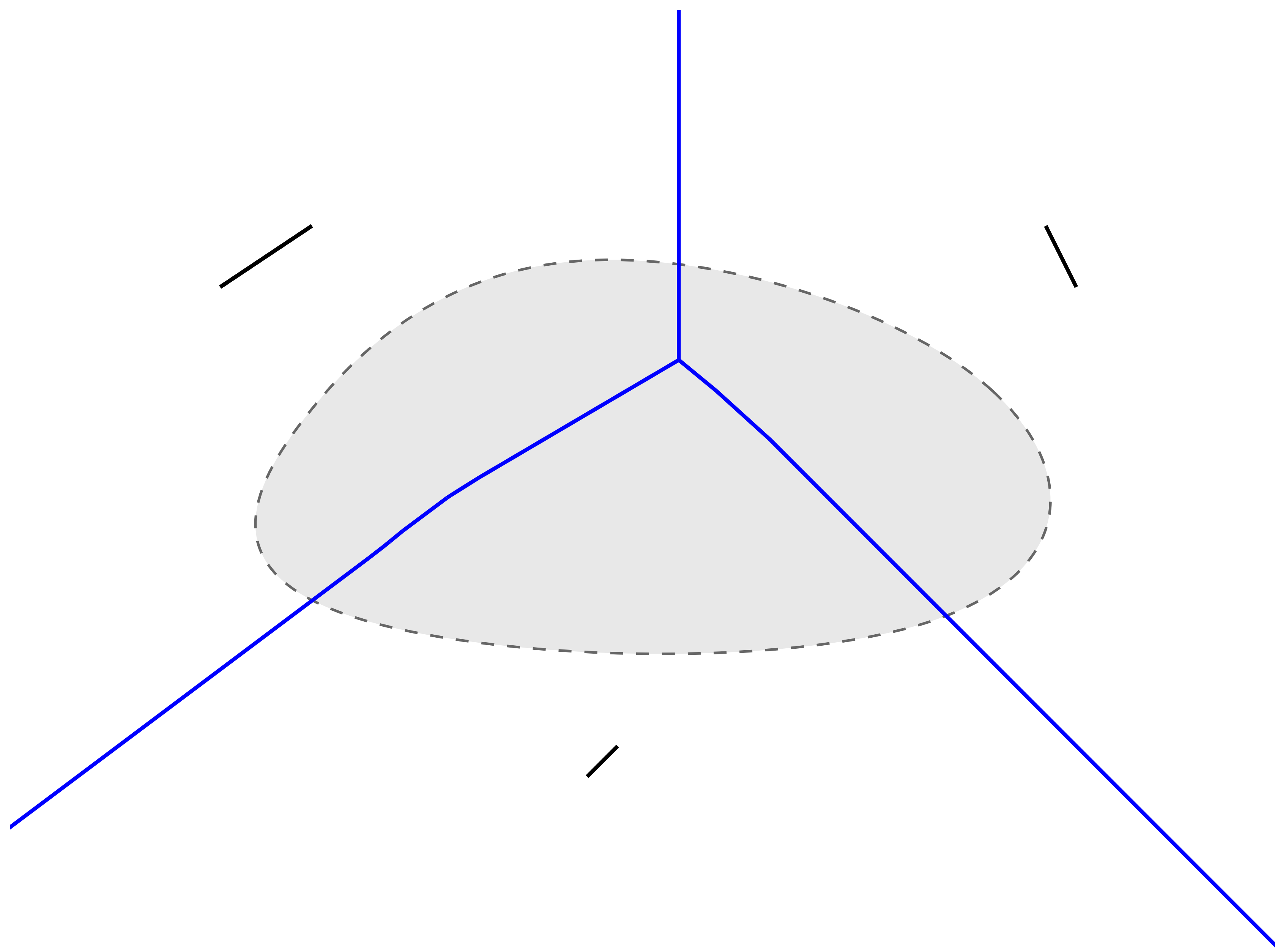
Incremental construction

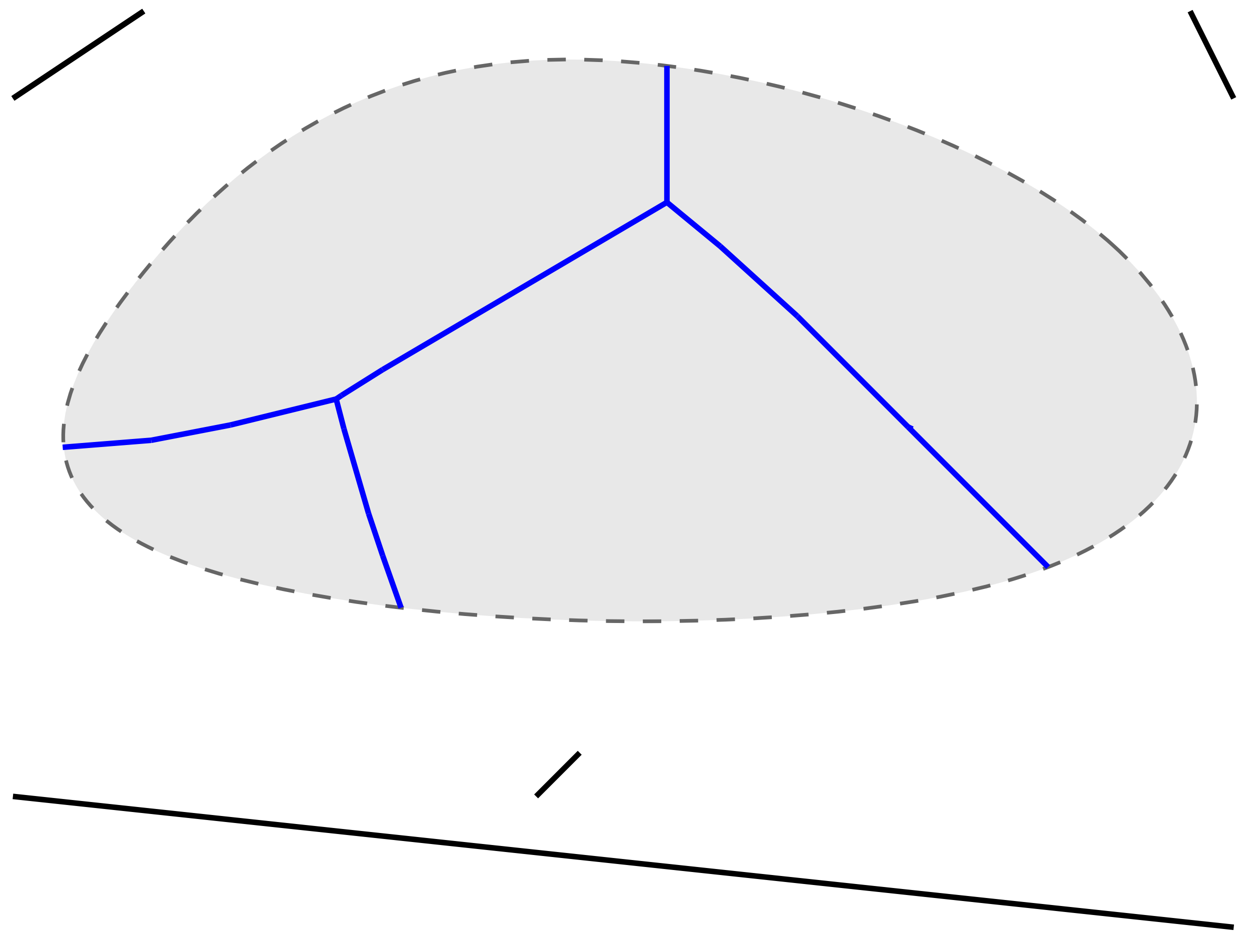
- In a different insertion order, we may need to split a region in two.
  - not a major problem in general.



A Voronoi-like structure

- We need to compute the structure on right, which is a Voronoi-like diagram
- A Voronoi diagram of **bits and pieces** of these segments

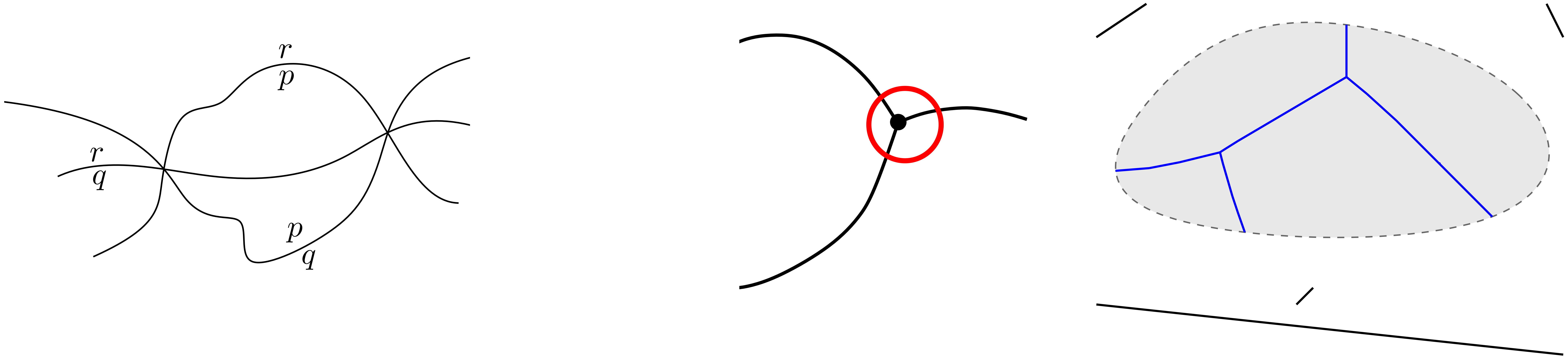




Voronoi-like graph

- We call **Voronoi-like** any **graph** on the arrangement of a bisector system whose vertices (other than its leaves) are **locally Voronoi.**

- A vertex is called **locally Voronoi** if it is a legal Voronoi vertex of 3 sites



- Any graph on an abstarct bisector arrangment whose non-leaf vertices are locally correct Voronoi vertices is a **Voronoi-like diagram**

Delaunay’s Theorem for AVDs

- **Delaunay’s theorem** (points, Euclidean metric) **:**
  A triangulation is **globally Delaunay** iff it is **locally Delaunay**.

- **Recent extension** [P. SoCG23]: Under a bisector system of classic AVDs**, any Voronoi-like graph** in the plane **is** the **Voronoi diagram** of the involved sites

  - If you have a graph whose vertices are legal Voronoi vertices of 3 sites, then this graph **is** the Voronoi diagram of the involved sites.

Voronoi like graphs

- Voronoi–like graphs are useful to hold **partial** (flexible) **Voronoi information**
- They are **as close as possible to being Voronoi diagrams** subject to possibly missing some faces.
- Extend **Delaunay's Theorem** from Euclidean points to abstract Voronoi diagrams and their duals
- **Applications**: simple (expected) linear-time algorithms for Voronoi **tees** and **forests**
  - **Site-deletion** in abstract Voronoi diagrams (and related concrete VDs)
  - **Farthest** abstract Voronoi diagram (given the order of Voronoi regions at infinity)
  - **Order-k** abstract Voronoi diagram – iterative construction
  - Updating a **Constraint Delaunay Triangulation** after a segment constraint insertion
  - Computing a **tree VD** in a **domain D**, given the order of Voronoi faces on **∂D**, $|\partial D|=O(1)$

Open Problem

Open problem:

- **Deterministic linear-time** technique for the same problems.
  Combine **Voronoi-like structures** and the technique of **[Aggarwal, Guibas Saxe, Shor, 89]**

- Recent progress to the affirmative

Thank you for your attention!