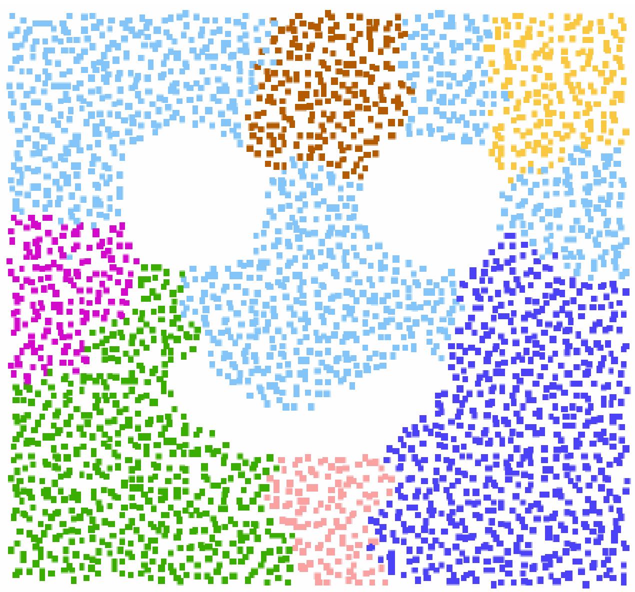
Shape Segmentation and Applications in Sensor Networks

Xianjin Zhu Rik Sarkar Jie Gao

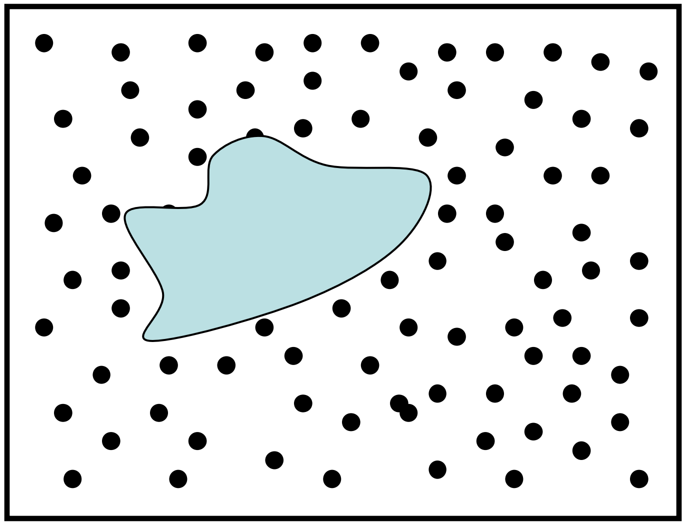


STONY BROOK UNIVERSITY

INFOCOM 2007

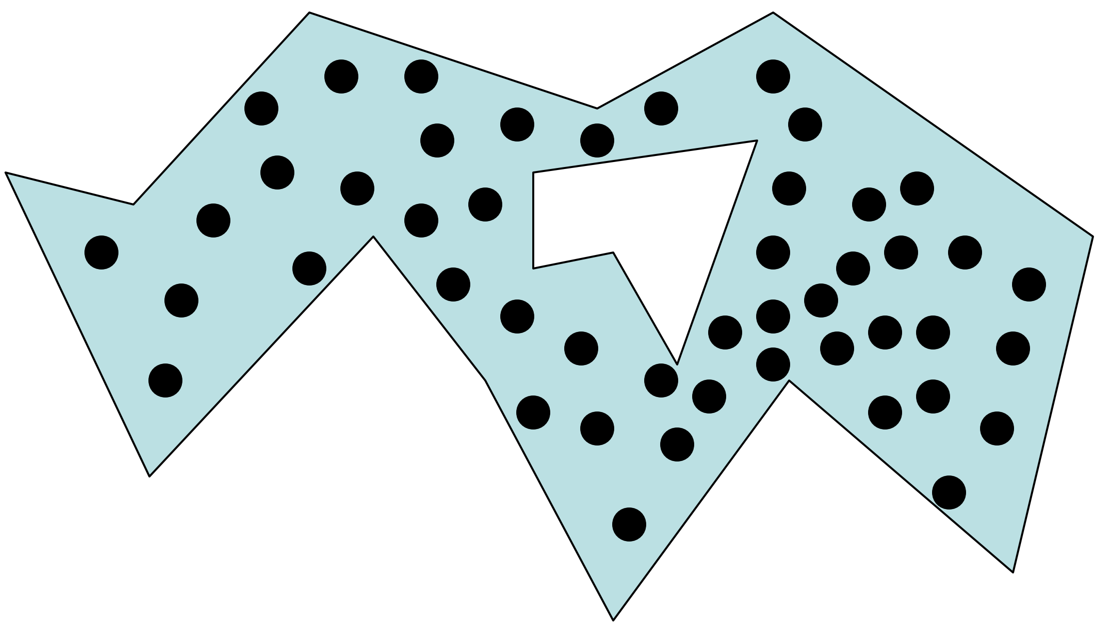
Motivation

- Common assumption: sensors are deployed uniformly randomly inside a simple region (e.g., square).
- In practice, can be complex.
  - Obstacles (lakes, buildings)
  - Terrain variation
  - Degradation over time



Sensor Distribution in Practice

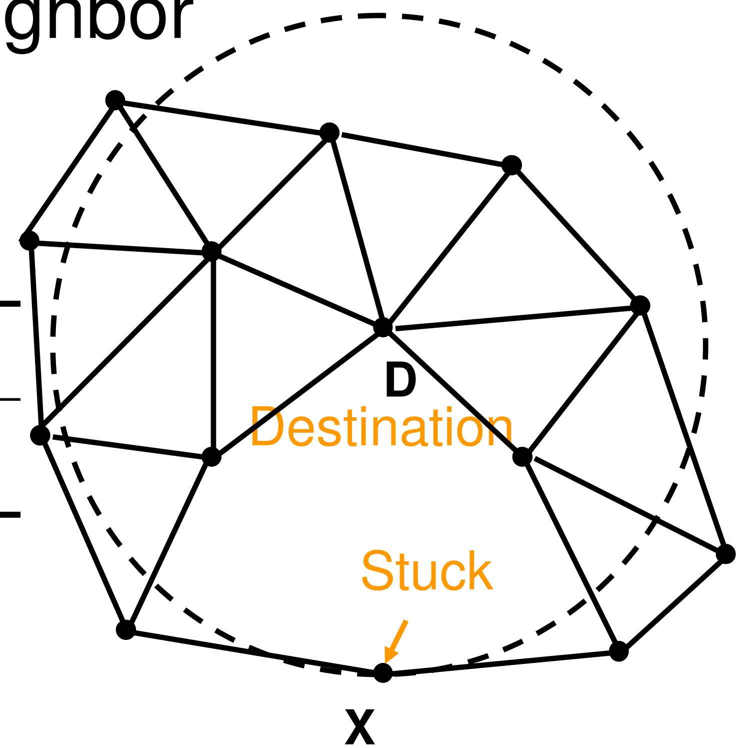
- Nodes are distributed in a geometric region with possible complex shape, with holes.



With holes or a complex shape…

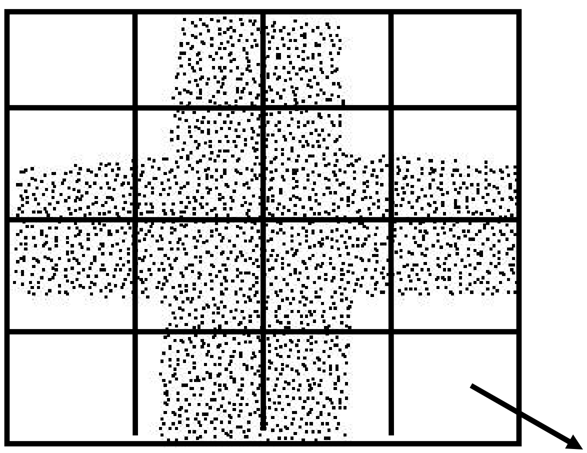
- Some protocols may fail:
  - Greedy forwarding: packets are greedily forward to the neighbor closest to the destination

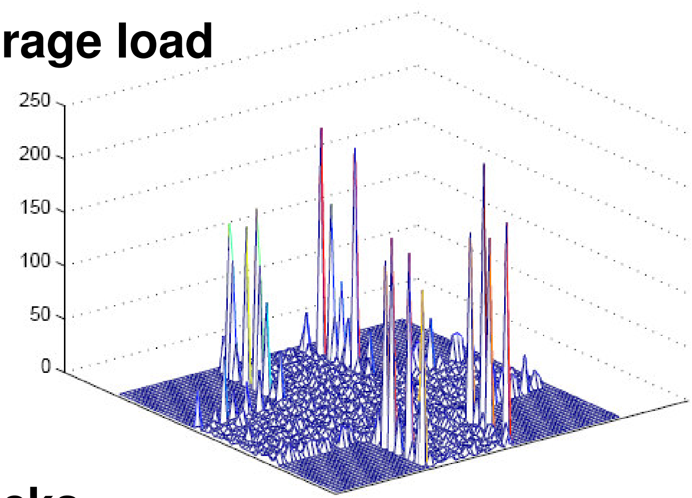
Dense uniform	Sparse, non-uniform
Works well	May get stuck



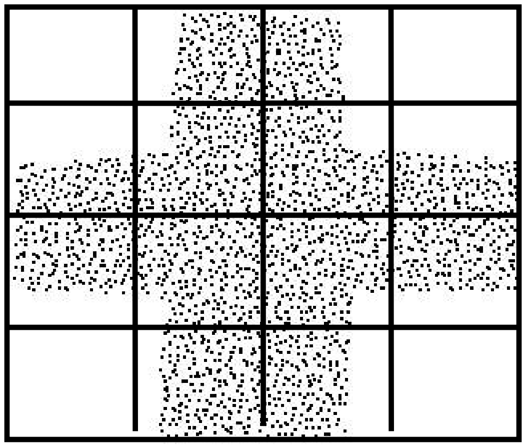
With holes or a complex shape…

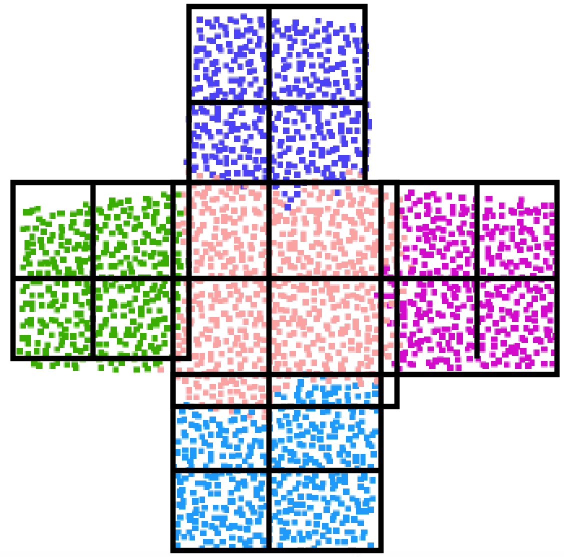
- Some protocols have degraded performance
  - Quad-tree type data storage hierarchy
    - Data is hashed uniformly to the quads



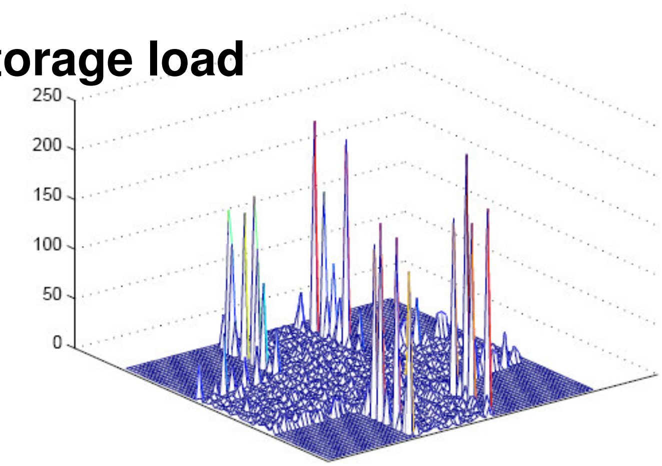


Quad-Tree Type Hierarchy



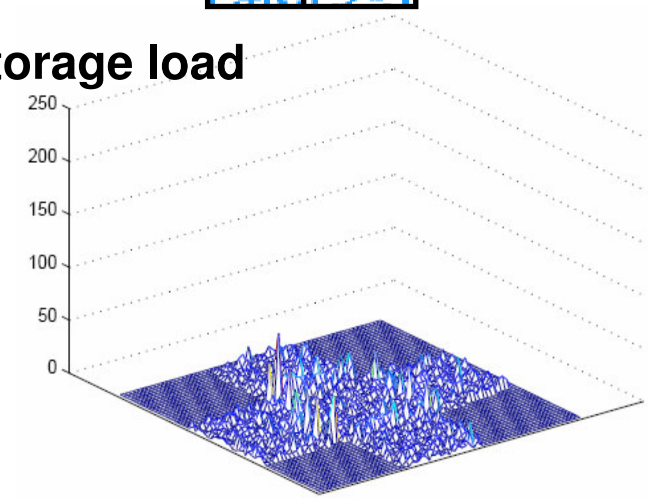


Storage load



w/o segmentation

Storage load

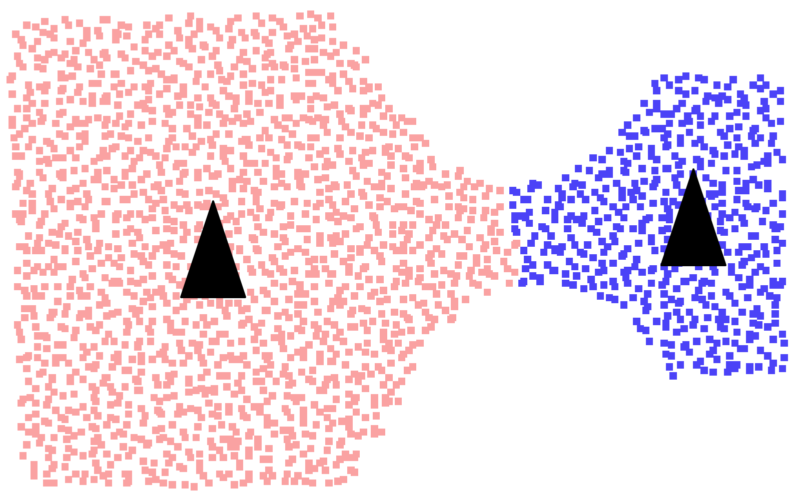


w/ segmentation

Lesson Learned

- Global geometric features affect many aspects of sensor networks.
  - Affect system performance.
  - Affect network design.

Place base stations and avoid traffic bottleneck

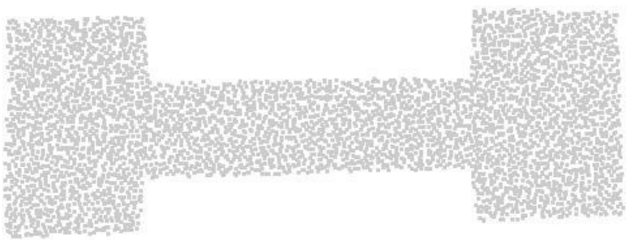


How to Handle Complex Shape?

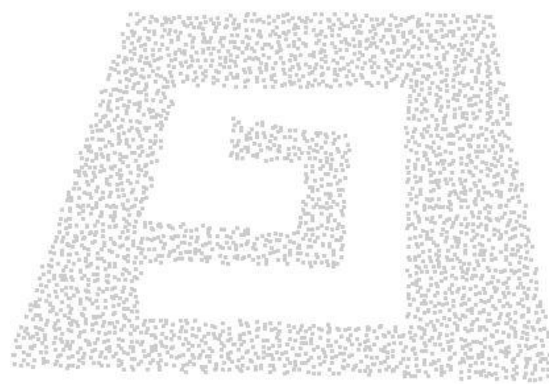
- Previous work
  - Build problem specific virtual coordinate system (e.g., for routing)
  - Redevelop every algorithm on virtual coordinate system
- Our approach: shape segmentation
  - A unified approach to handle complex geometry
  - Make existing protocols reusable

Sensor Field with Arbitrary Shape

- Density ranges from 7-12 neighbors/node

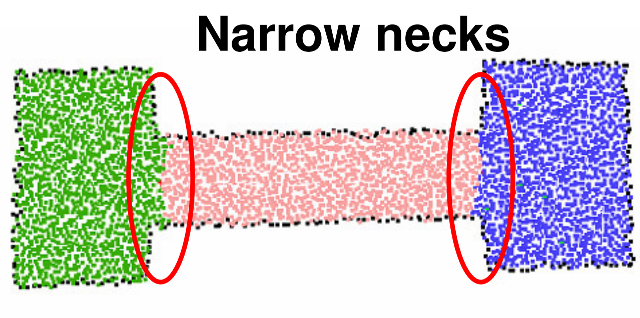


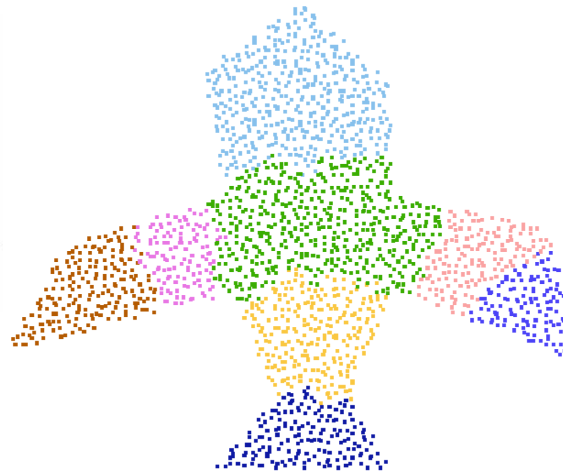


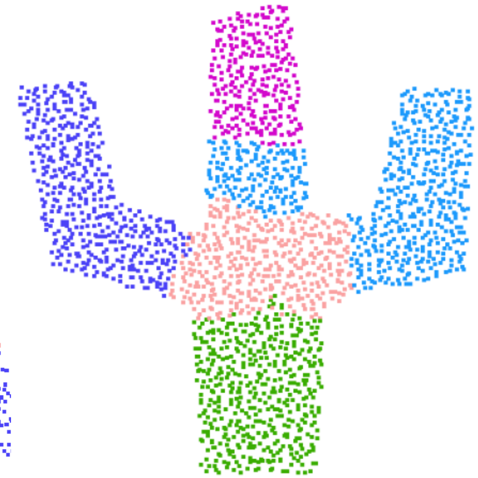


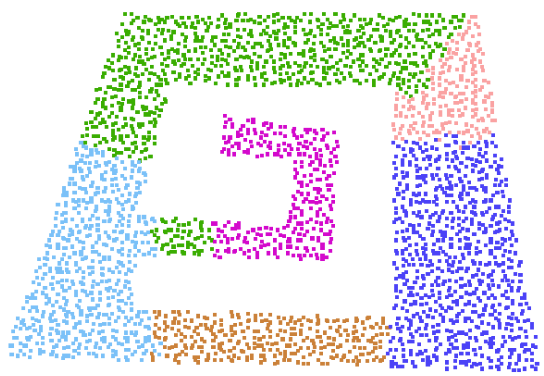

Simulation Results on Segmentation

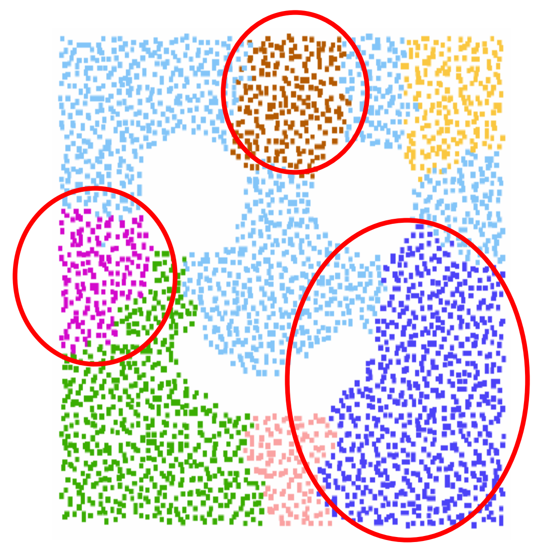
- Density ranges from 7-12 neighbors/node





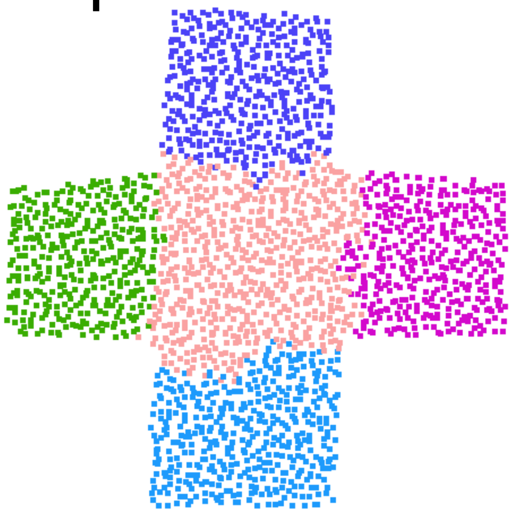






Our Approach: Shape Segmentation

- Segment the irregular field into "nice" pieces.
  - Each piece has no holes, and has a relatively nice shape
- Apply existing algorithms inside each piece.
  - Existing protocols are reusable
- Integrate the pieces together with a problem-dependent structure.

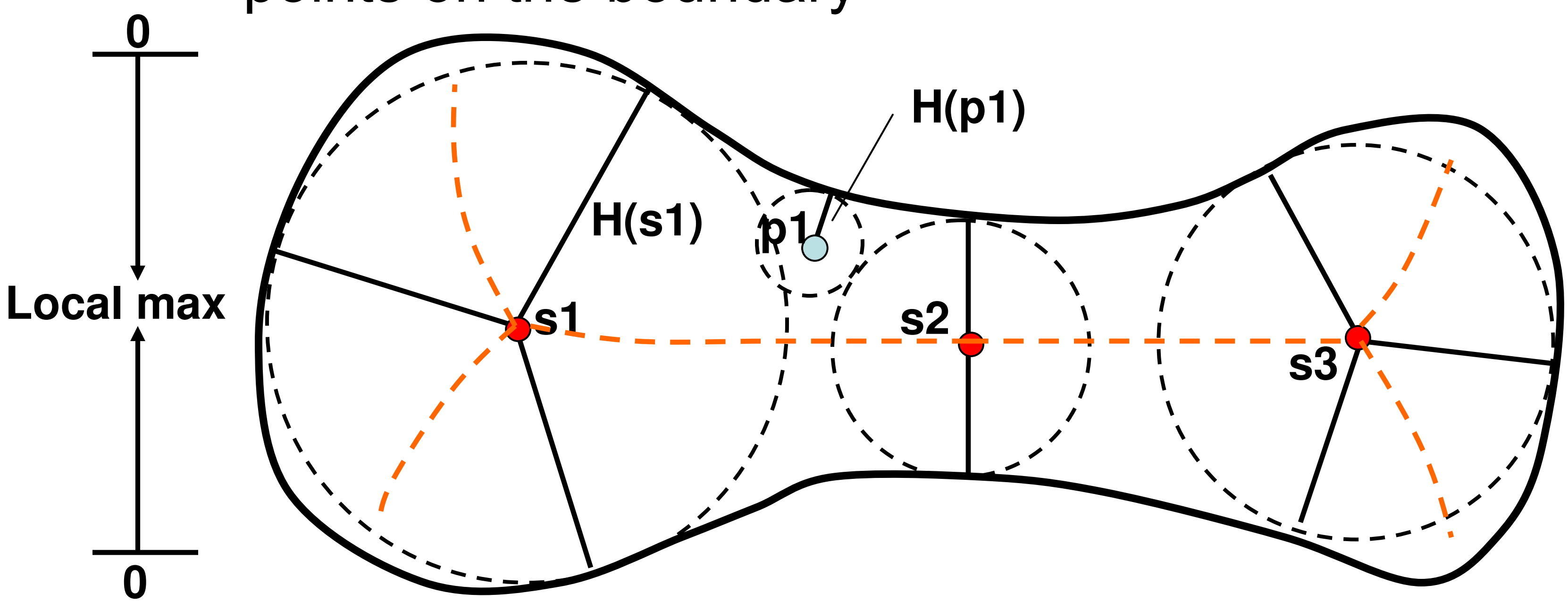


The rest of the talk …

- Segmentation algorithm
- Implementation issues

Segmentation with Flow Complex

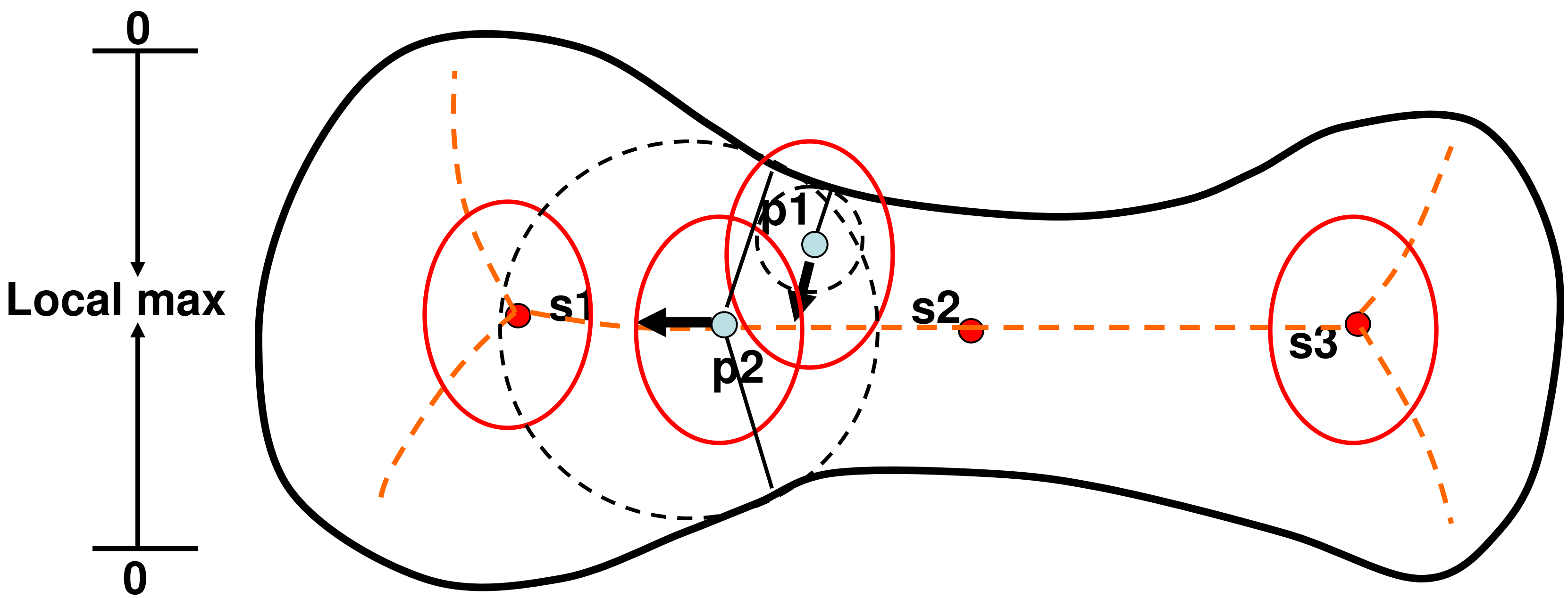
- Flow Complex in continuous domain
  - Distance function $h(x)=\min\{||x-p||^2: p \text{ on boundary}\}$
  - Medial axis: a set of points with at least two closest points on the boundary



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

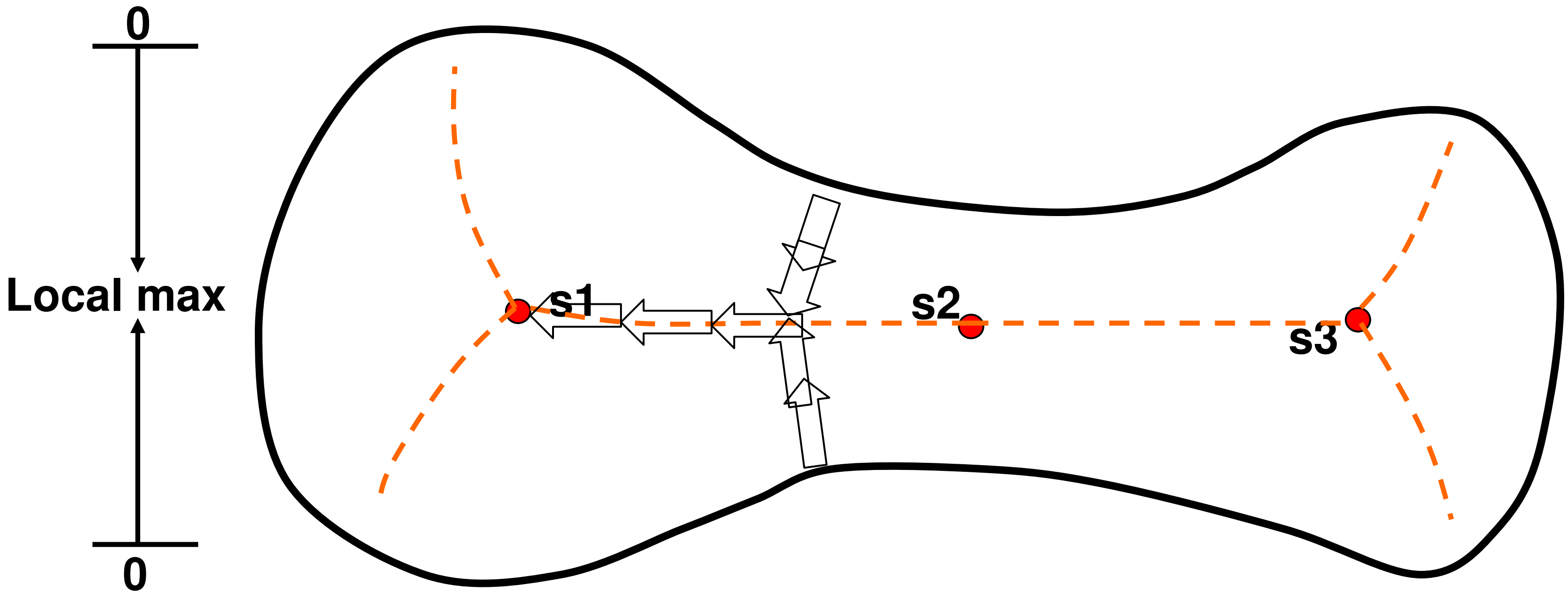
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

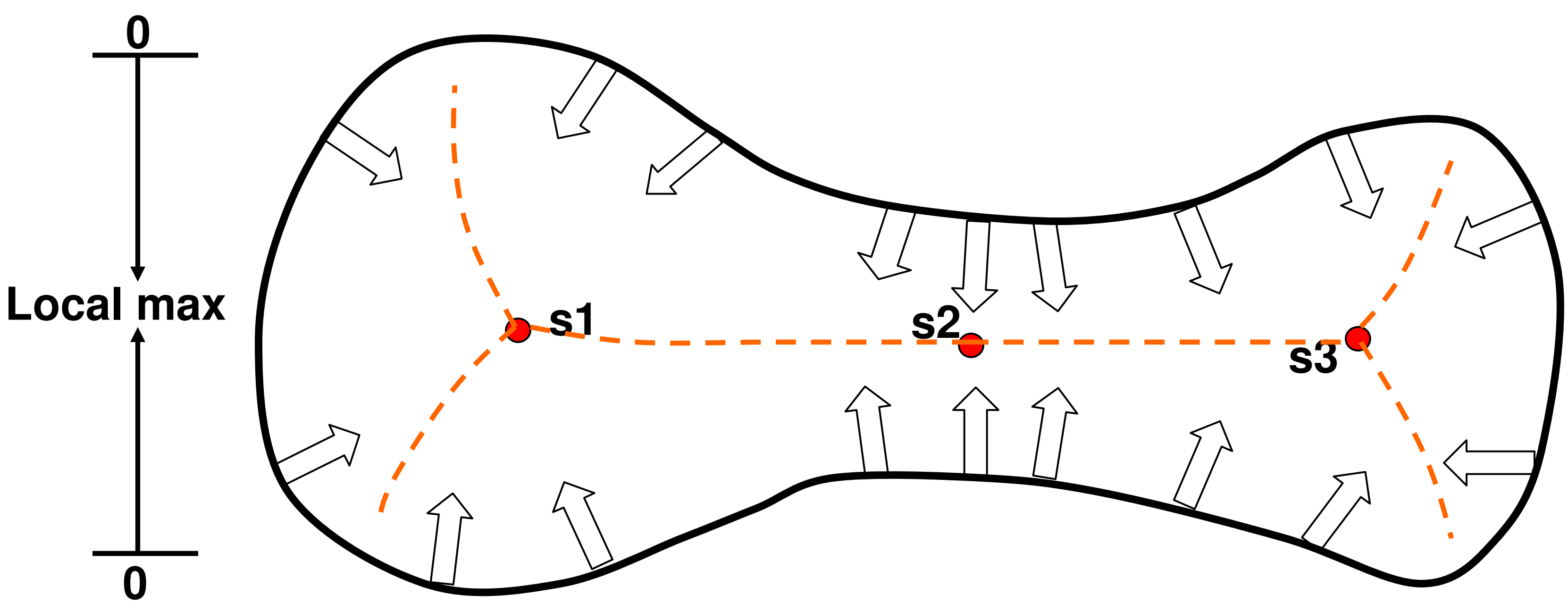
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS’03]

Segmentation with Flow Complex

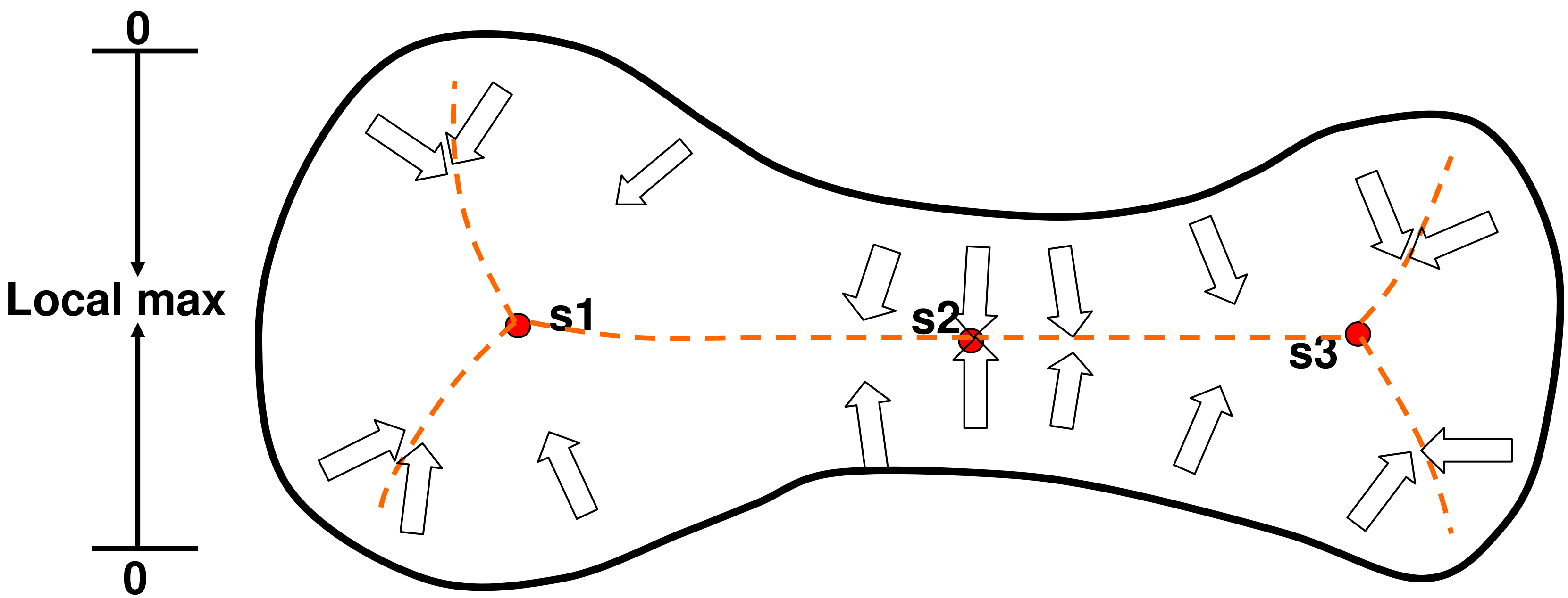
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

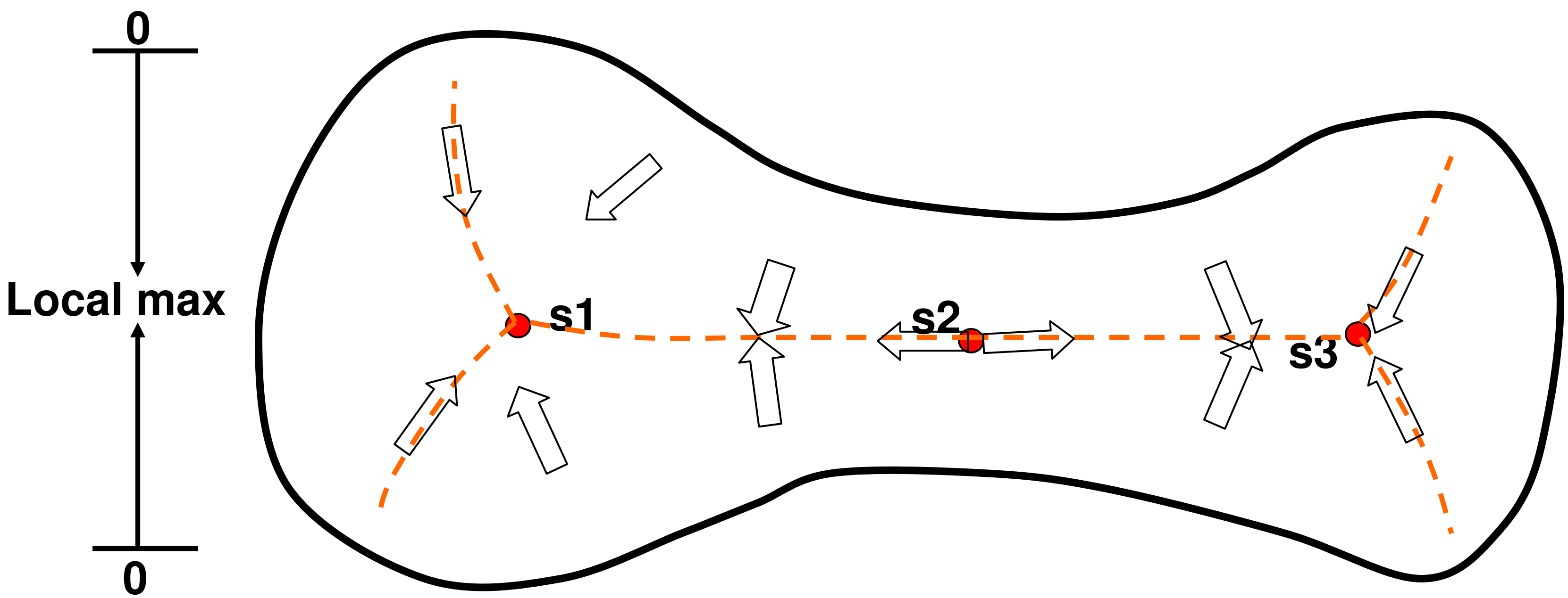
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

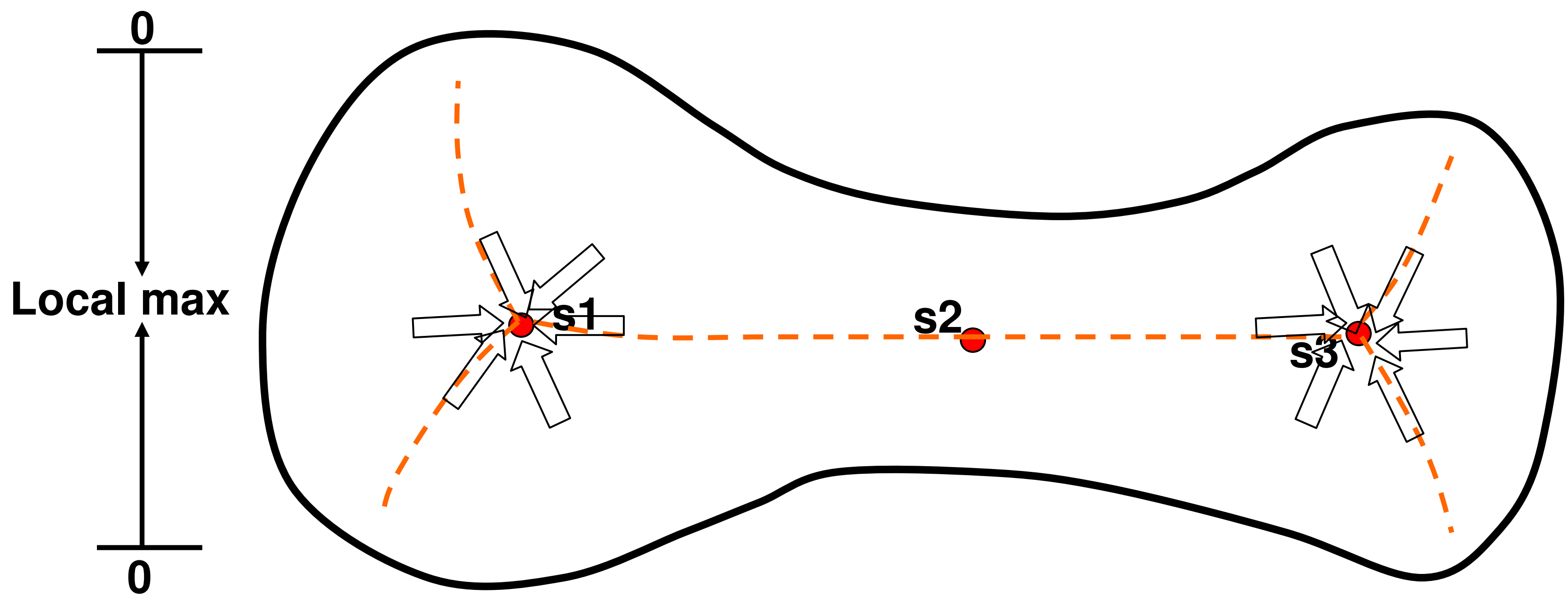
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

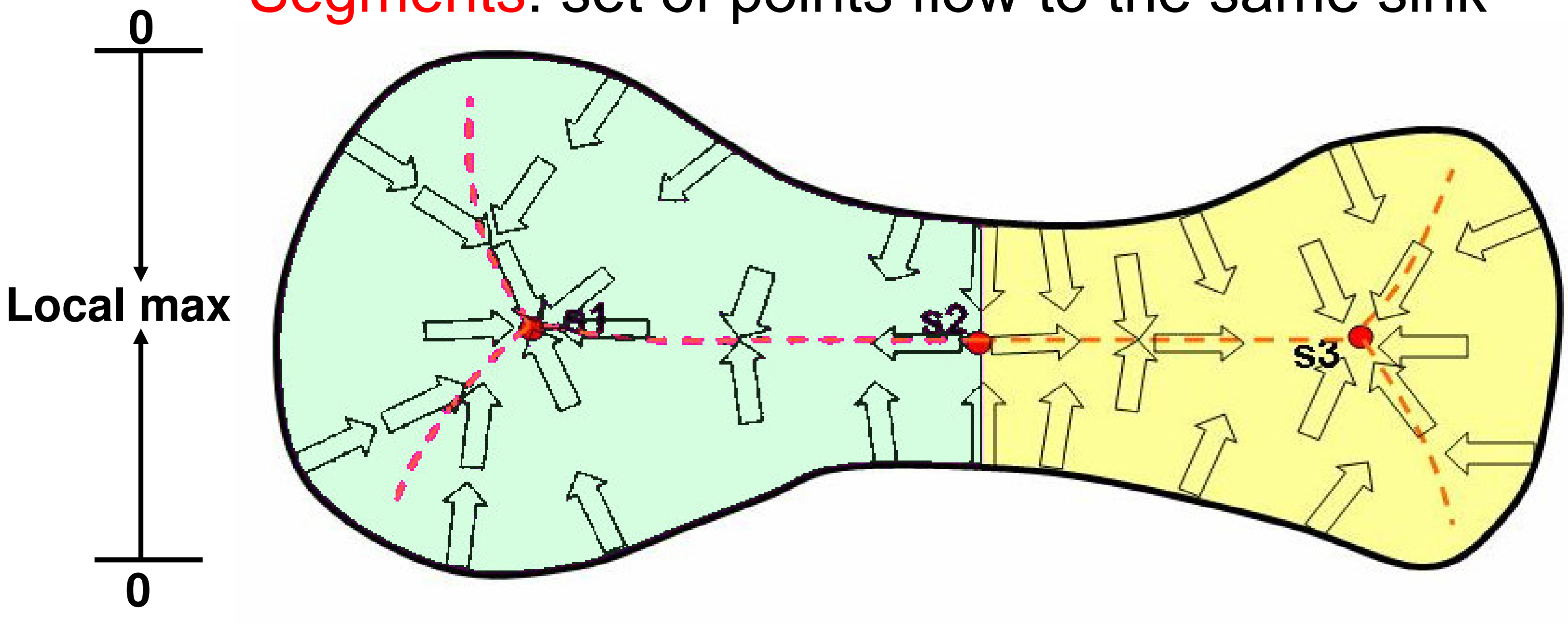
- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)



Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

Segmentation with Flow Complex

- Flow Complex in Continuous domain
  - Flow direction: the direction that h(x) increases fastest
  - Sinks: local maximum, no flow direction (s1 & s3 here)
  - Segments: set of points flow to the same sink

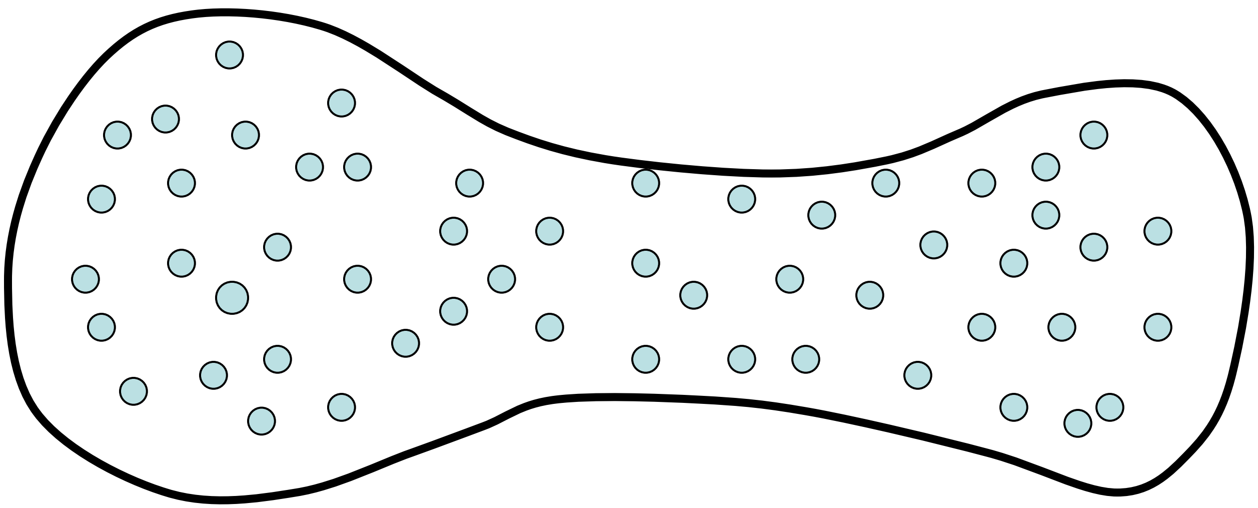


Naturally partition along narrow necks

Reference: flow complex [Dey, Giesen, Goswami, WADS'03]

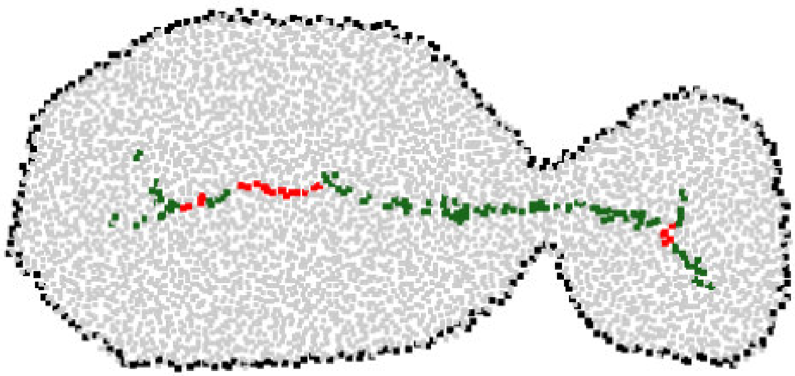
Implementation Challenges

- No global view, no centralized authority
- No location, only connectivity information
  - Distances are approximated by hop count
- Robust to inaccuracy, packet loss, etc.
- Goal: a distributed and robust segmentation algorithm.

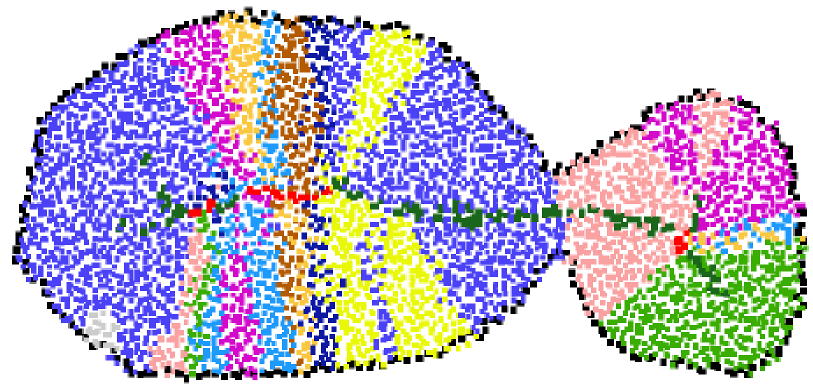


Algorithm Outline

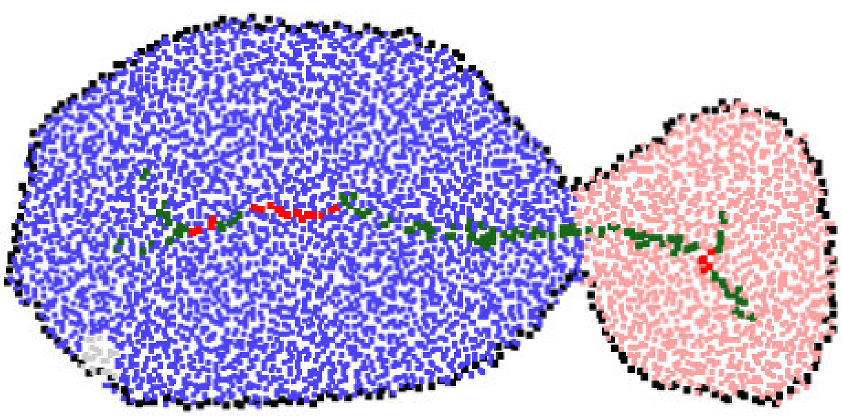
1. Compute the medial axis



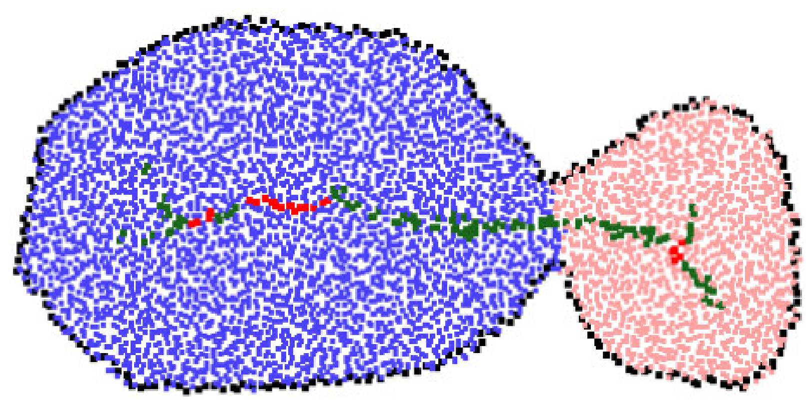
2. Compute the flow



3. Merge nearby sinks

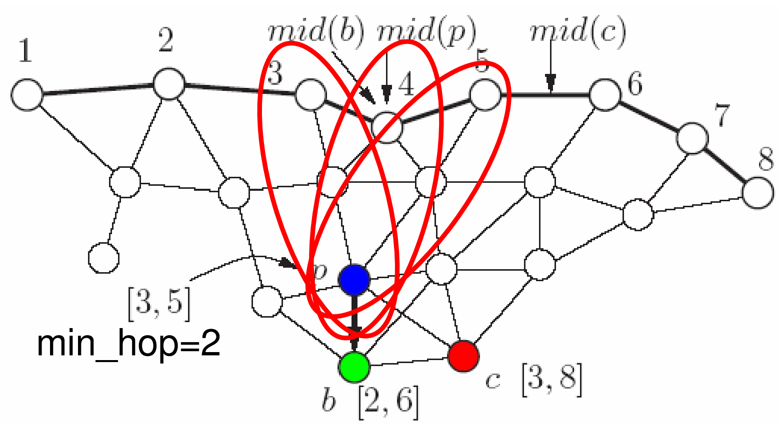


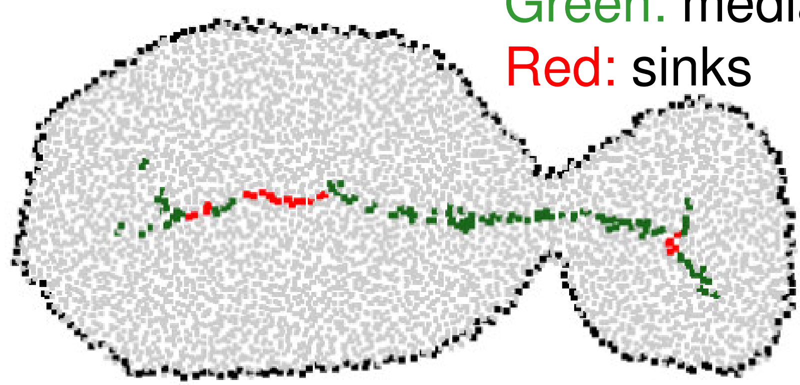
4. Final clean-up



Step 1: Compute the medial axis

- Boundary nodes flood inward simultaneously.
- Nodes record: minimum hop count &
  closest intervals on the boundary
- Medial axis: more than two closest intervals

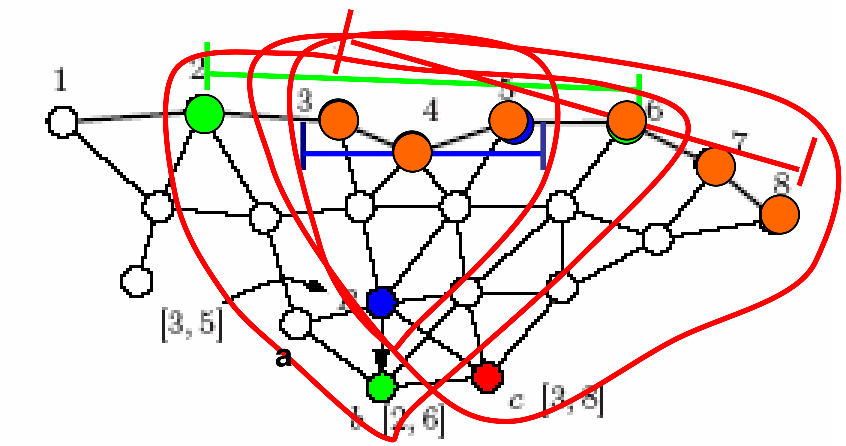


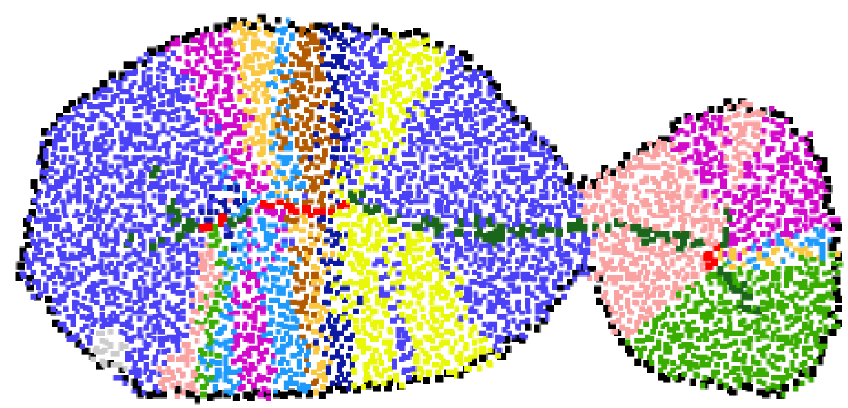


Reference: Boundary Detection [Wang, Gao, Mitchell, MobiCom'06]

Step2: Compute the flow

- Flow direction: a pointer to a neighbor with a higher hop count from the same boundary
  - Prefer neighbor with the most symmetric interval
- Sinks must be on the medial axis.
- Network is organized into forests, sinks are roots
- Nodes are classified into segments by their sinks.

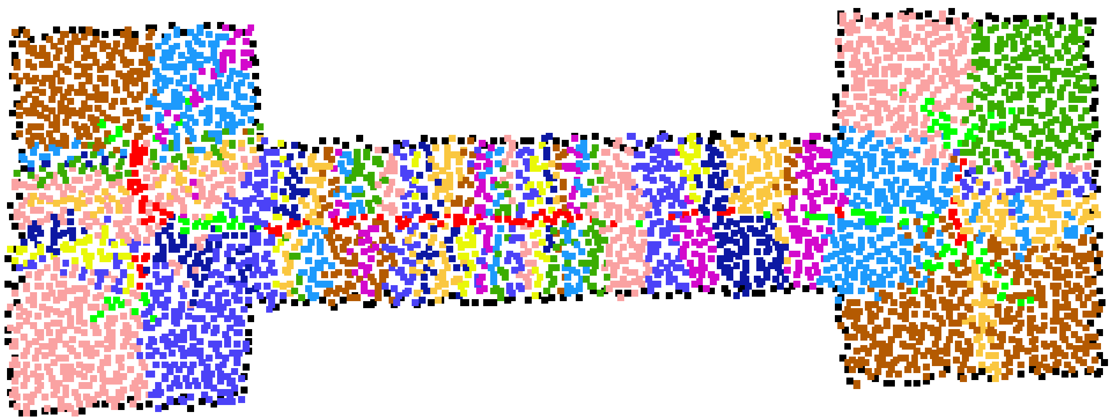




Too many segments!

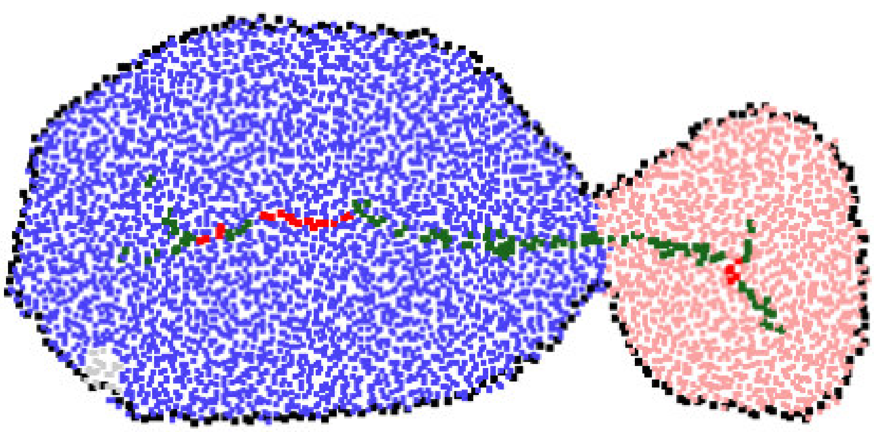
Example of Heavy Fragmentation

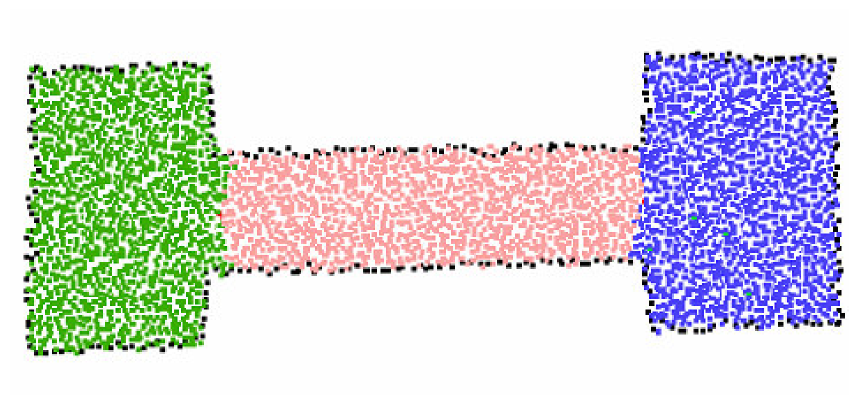
- Fragmentation problem becomes severe with parallel boundaries.



Step3: Merge nearby sinks

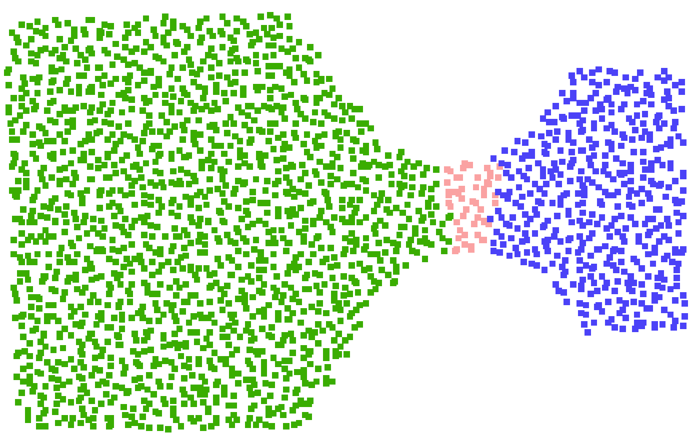
- Nearby sinks with similar hop count to the boundaries are merged (together with their segments).



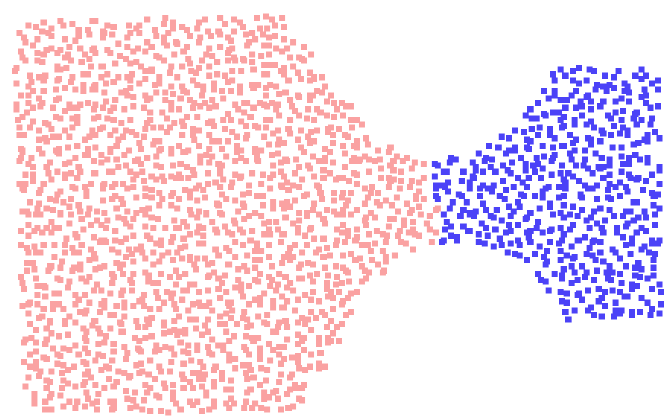


Step3: Merge nearby sinks

- Nearby sinks with similar hop count to the boundaries are merged (together with their segments).
  - Segmentation granularity: $|H_{max}-H_{min}| < t$



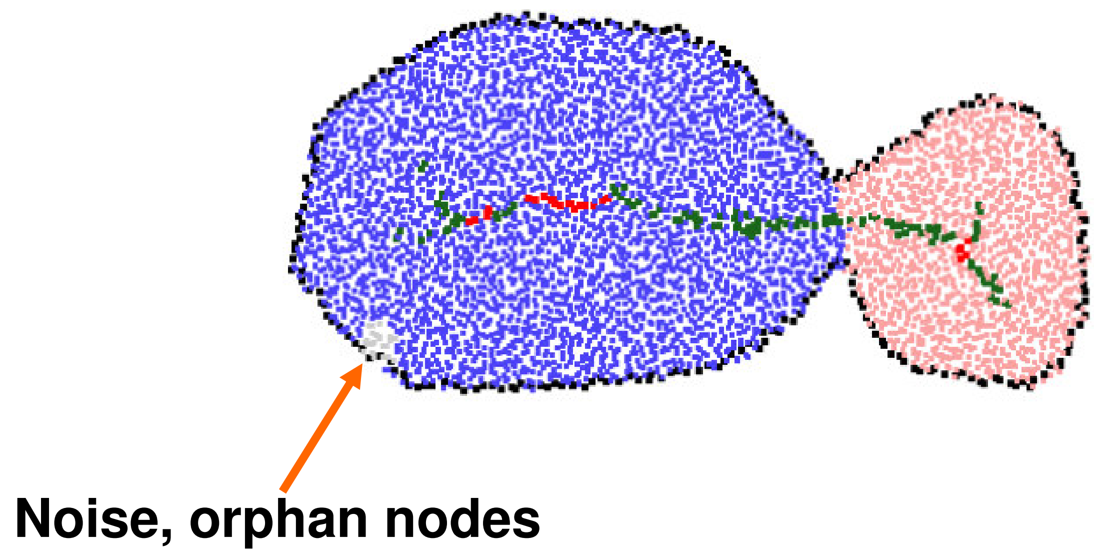
t=2



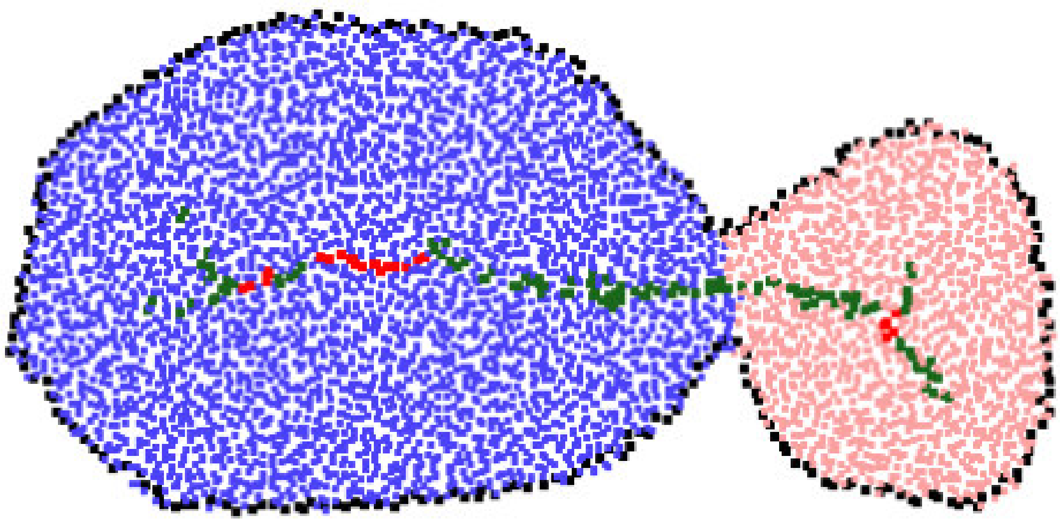
t=4

Step4: Final clean-up

- Merge orphan nodes with nearby segments
- Orphan nodes: local maximum and nodes that flow into them

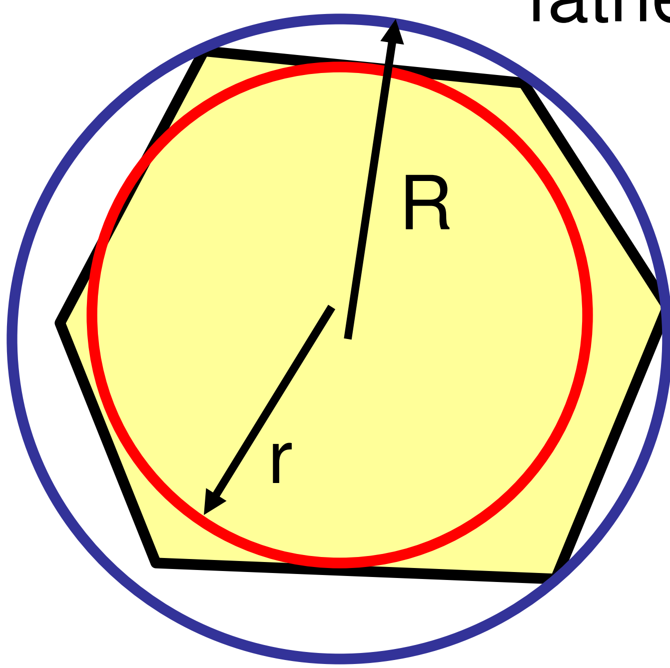


Final Result



Properties of Segmentation

- A few "fat" segments
- Further merging only hurts fatness



$$\text{fatness} = \frac{\text{max inscribing ball radius}}{\text{min enclosing ball radius}} = \frac{r}{R}$$

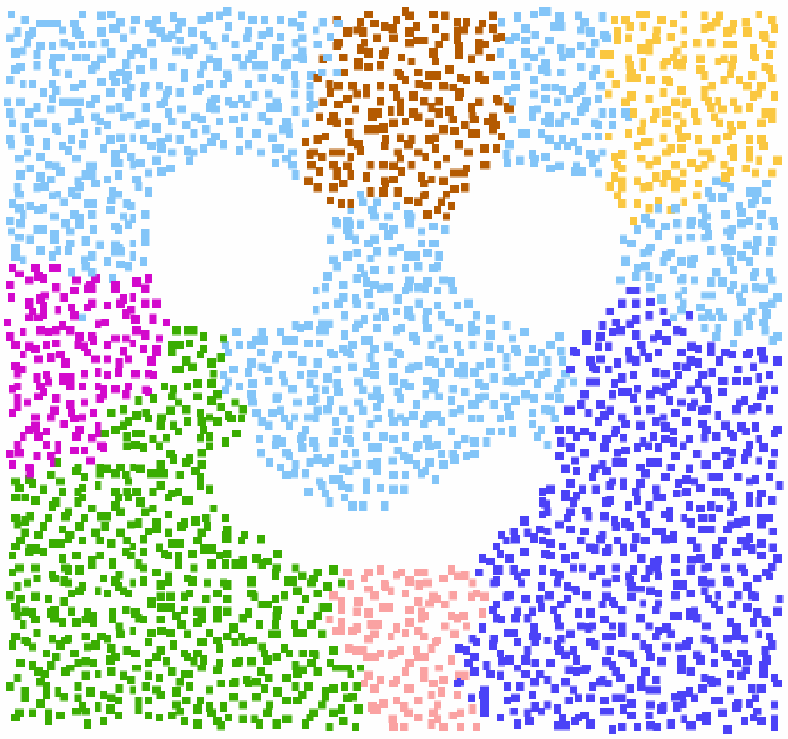
The bigger the fatter.

Conclusion

- A unified approach handling complex shape in sensor networks.
- A good example to extract high-level geometry from connectivity information.
- Network self-organizes by local operations.

Thank you!

- Questions ?



Email: {xjzhu, rik, jgao}@cs.sunysb.edu