

---

# Localization and Routing in Sensor Networks by Local Angle Information

---

Jehoshua Bruck

Jie Gao

Anxiao Andrew Jiang

California Institute of Technology



# Location information is important

---

- Sensor networks are data-centric.

Example: temperature=70F at position (16, 9).

- Location information can help routing, topology control, information storage, etc.

Example: geographical routing.



# Existing localization approaches

---

- Globally Positioning System (GPS).

Expensive, doesn't work indoor.

- Derive location information by local measurements.



# Anchor-based and anchor-free methods

---

## ■ Anchor-based methods

- Anchor nodes have GPS;
- Other nodes derive their locations by trilateration.

## ■ Anchor-free methods.

- Connectivity only;
- Distance estimation for all communication links.

Please see the paper for a detailed reference list.

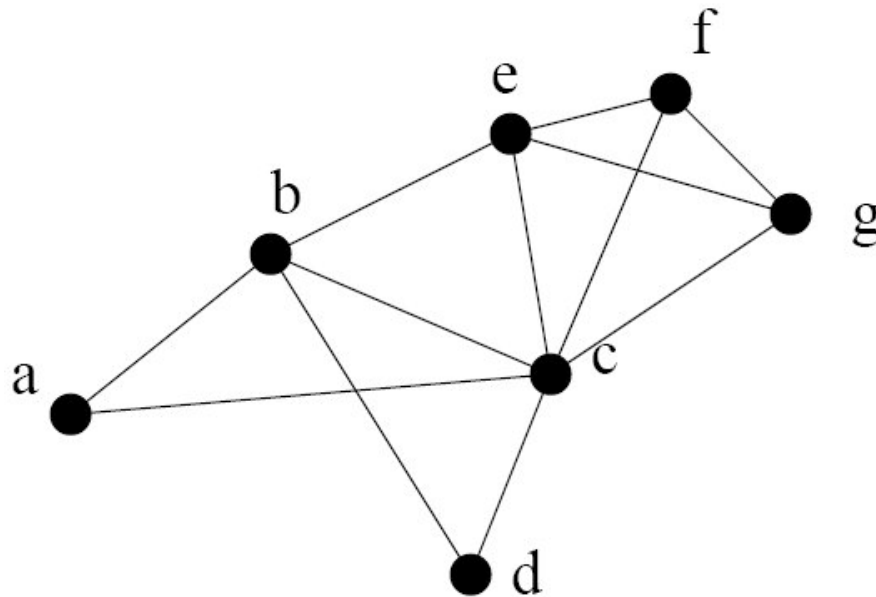
# Locality of communication

---

Unit disk graph model:  $\exists$  an edge  $uv$  iff  $|uv| \leq 1$ .

Quasi-Unit disk graph model:

$\exists uv$  if  $|uv| < \beta < 1$ ; no  $uv$  if  $|uv| > 1$ ; unclear otherwise.





# Localization by edge length information

---

A **valid embedding** by edge length information:

*Input:* A graph, and lengths of the edges.

*Output:* An embedding of the graph in the plane s.t.

1. Two nodes have an edge iff their distance  $\leq 1$ ;
2. The edge lengths are as specified.

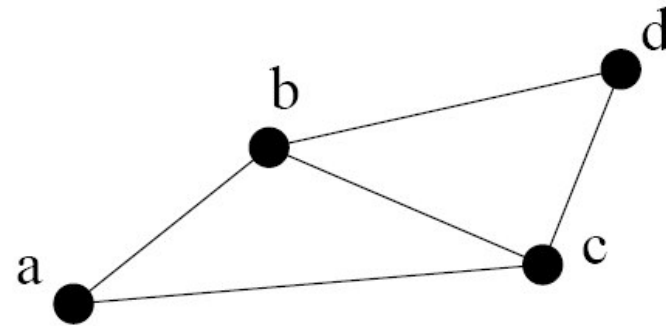
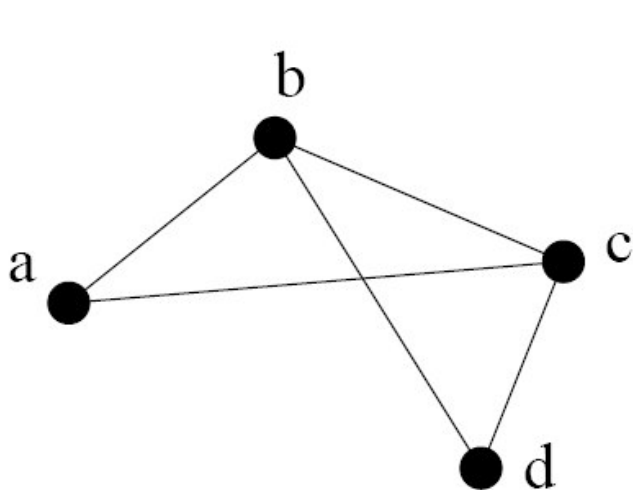
An  **$\alpha$ -approximate embedding** by edge length information:

Neighboring nodes  $\leq 1$ ;

Non-neighboring nodes  $\geq 1/\alpha$ ,  $\alpha > 1$ .

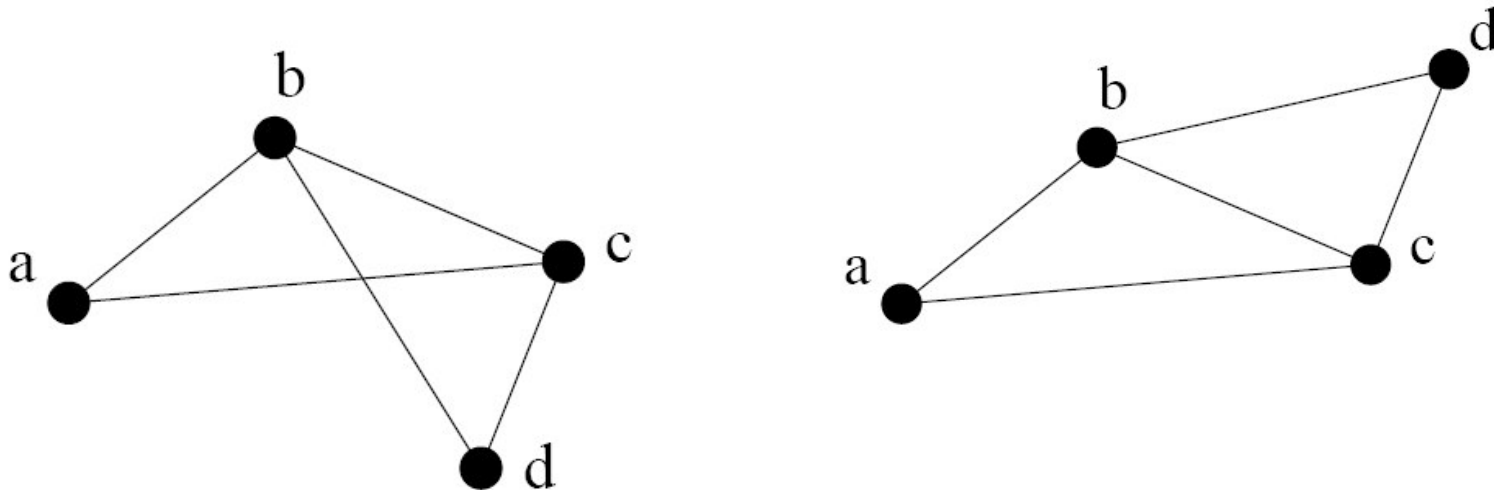
# Localization by edge length information

- In theory, localization is NP-hard [BK98][AGY04][BDHI04][KMW04].
- In practice, how to deal with folding is challenging.



# Localization by edge length information

- In theory, localization is NP-hard [BK98][AGY04][BDHI04][KMW04].
- In practice, how to deal with folding is challenging.

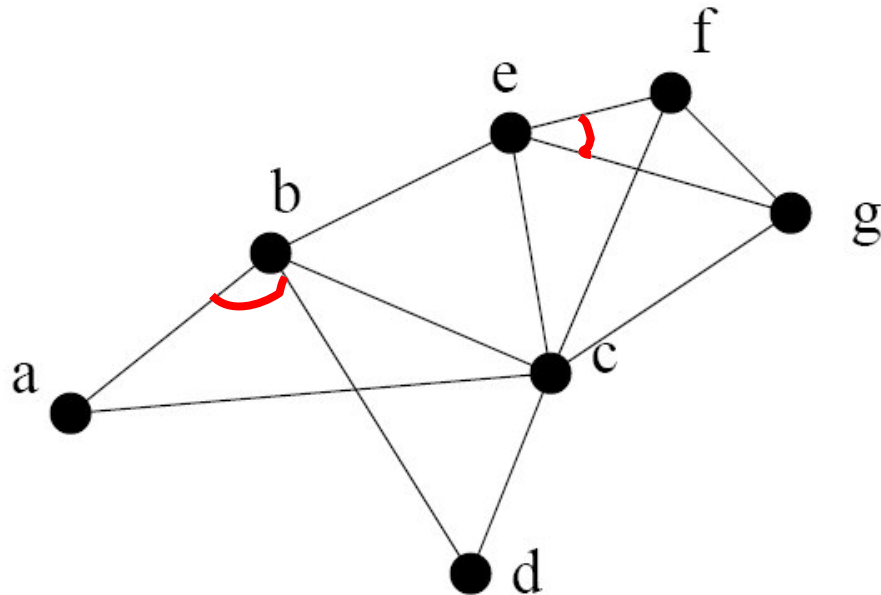


- What if we use angle info, instead of distance info?

# Localization by local angle information

- We study localization by local angle information.

Local angle: an angle between two adjacent edges.



Angle of Arrival or optical communication such as SmartDust.



# Localization by local angle information

---

A **valid embedding** by **local angle** information:

*Input:* A graph, and **local angles**.

*Output:* An embedding of the graph in the plane s.t.

1. Two nodes have an edge iff their distance  $\leq 1$ ;
2. The **local angles** are as specified.

An  **$\alpha$ -approximate embedding** by **local angle** information:

Neighboring nodes  $\leq 1$ ;

Non-neighboring nodes  $\geq 1/\alpha$ ,  $\alpha > 1$ .



# Overview of results

---

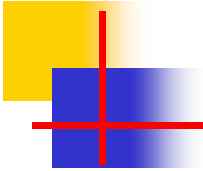
unit-disk graph,  $\xrightarrow{\text{hard}}$  embedding  
+local angles



# Overview of results

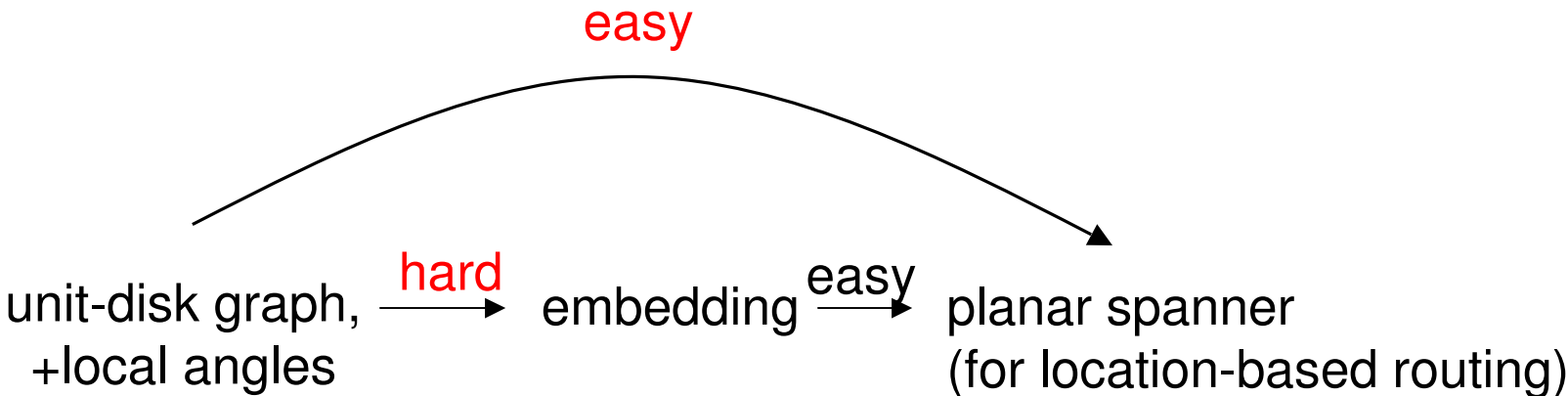
---

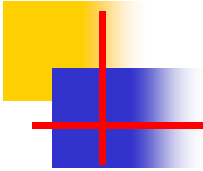
unit-disk graph, <sup>hard</sup> → embedding <sup>easy</sup> → planar spanner  
+local angles (for location-based routing)



# Overview of results

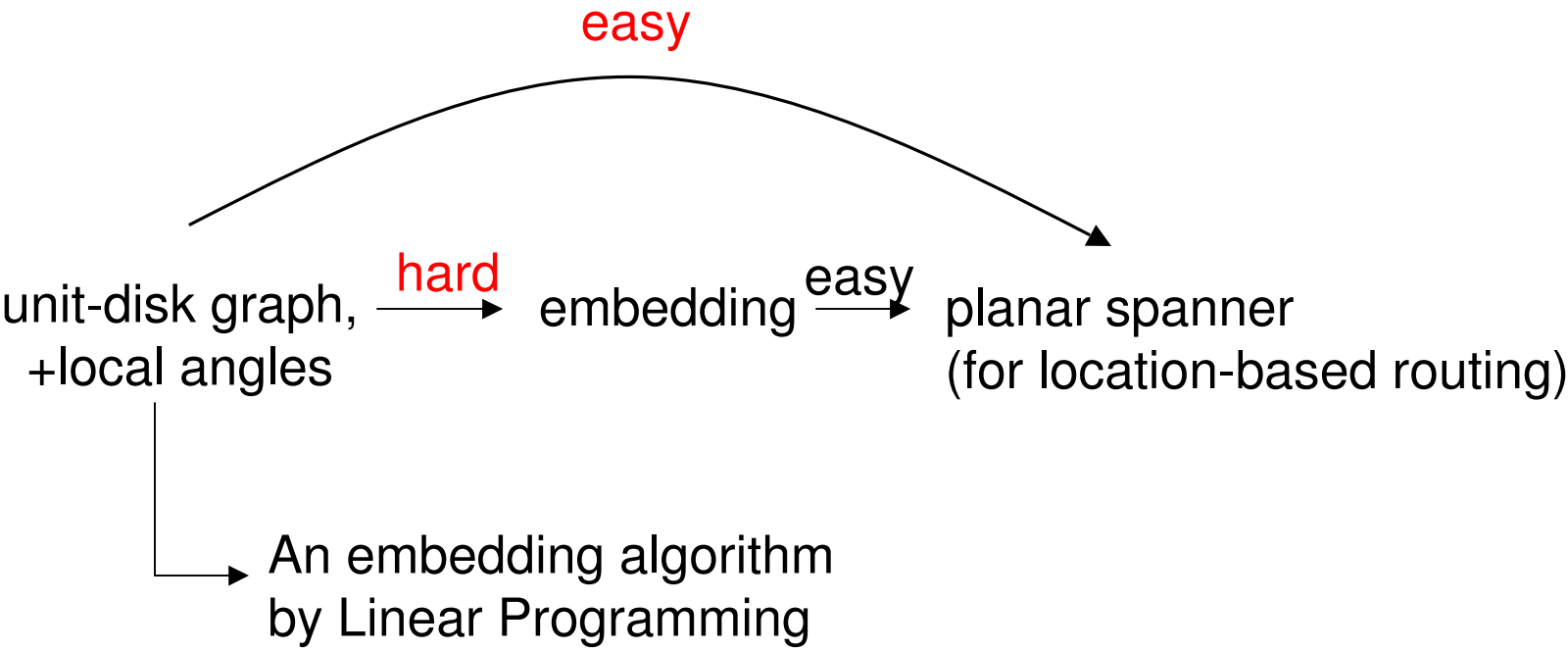
---

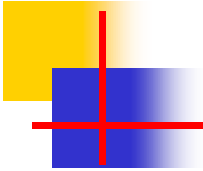




# Overview of results

---





# Part I

---

## NP hardness of embedding by angles

*Input:* A graph, and the values of local angles.



NP-hard

*Output:* A valid embedding.



# NP hardness of embedding

---

Basic idea: a reduction from the 3SAT problem.

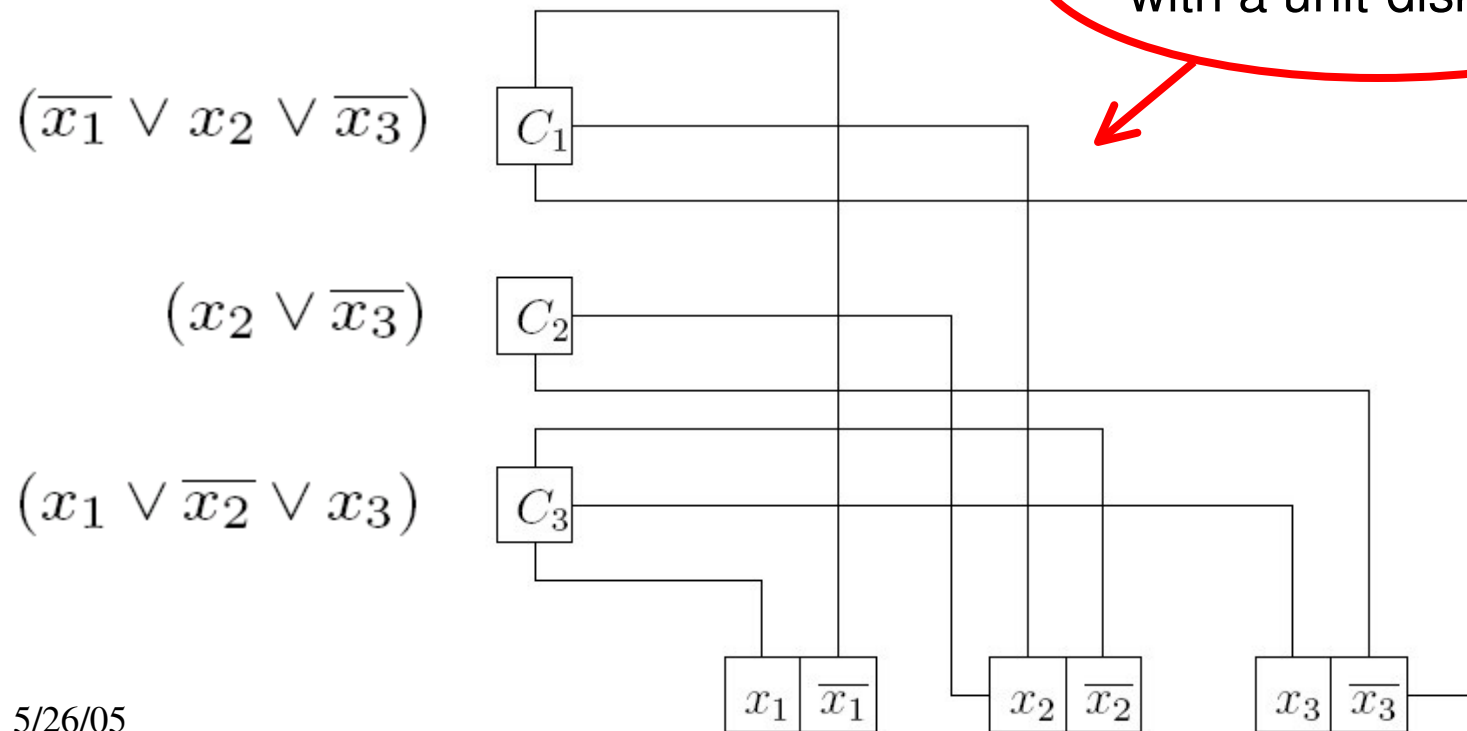
A 3SAT instance:

$$(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$$

# Formulate the 3SAT problem as a graph

3SAT instance:  $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$

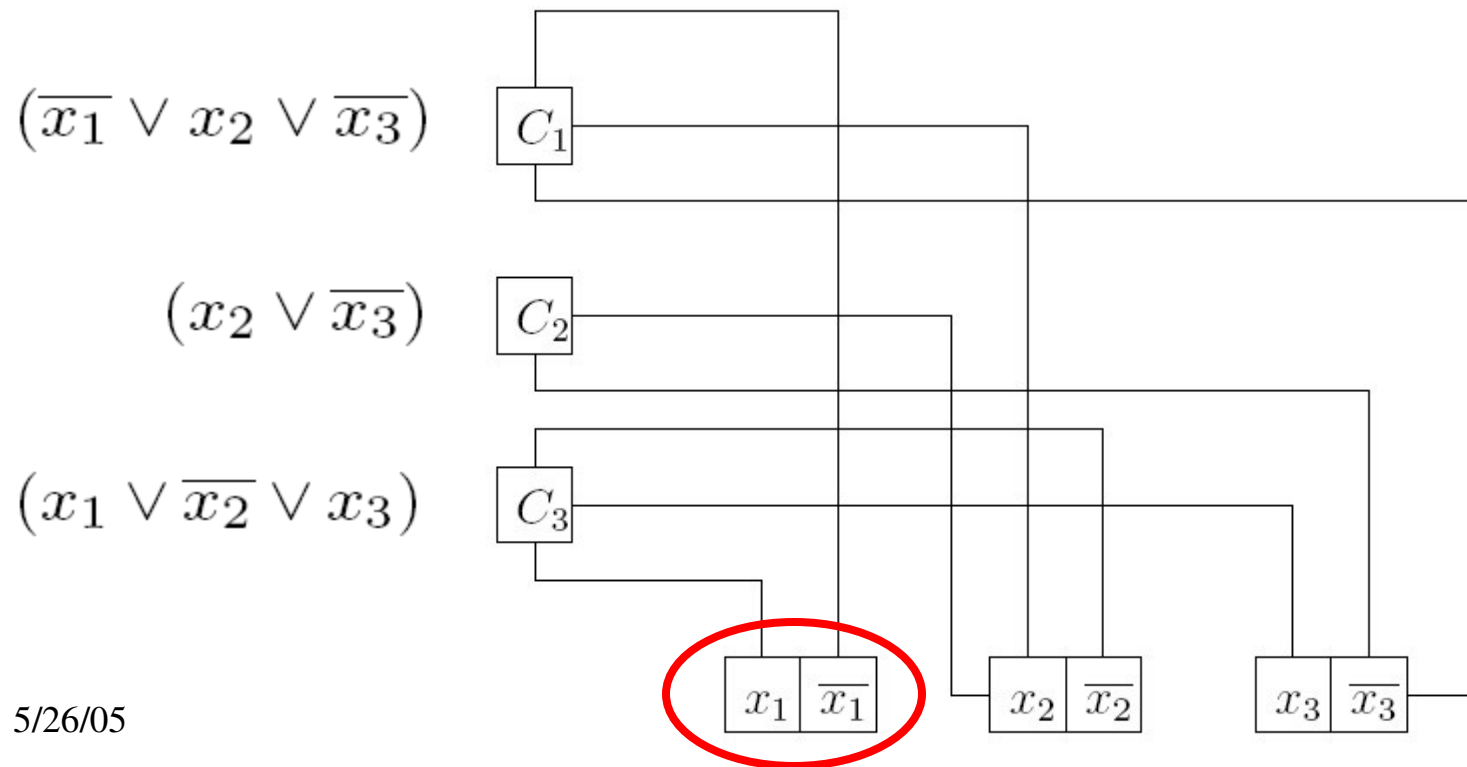
3SAT graph:



# Formulate the 3SAT problem as a graph

3SAT instance:  $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$

3SAT graph:





# Edge crossing can be detected

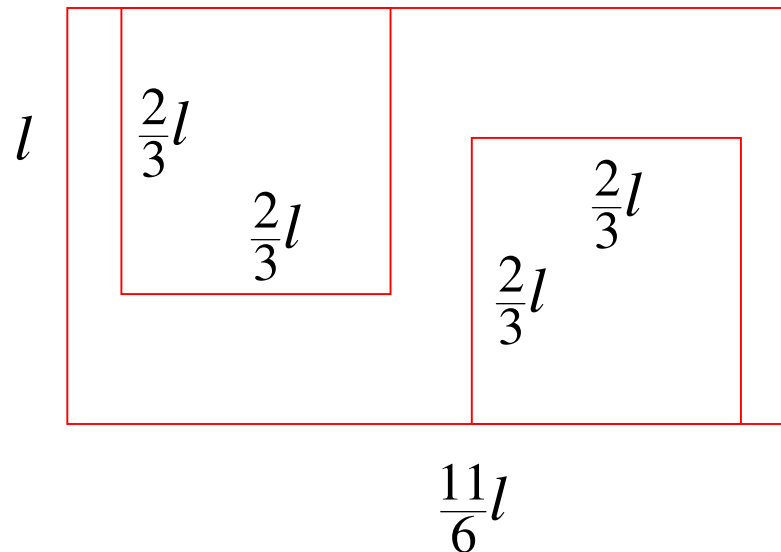
---

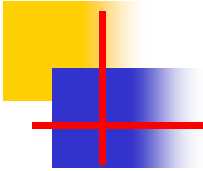
Lemma: By local angles of a unit-disk graph, we can determine all pairs of **crossing edges** in a valid embedding.

# Edge crossing can be detected

Lemma: By local angles of a unit-disk graph, we can determine all pairs of **crossing edges** in a valid embedding.

Consider a unit-disk graph, where the two 'teeth' do not cross:



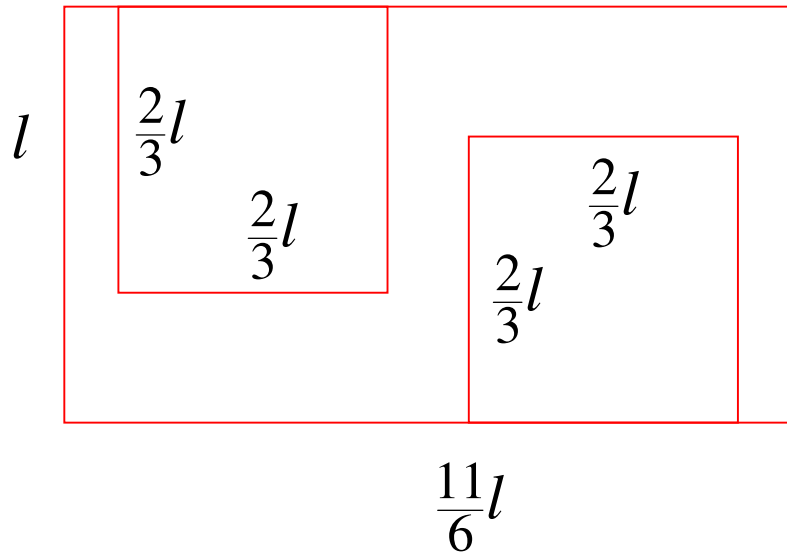


# 0/1 block

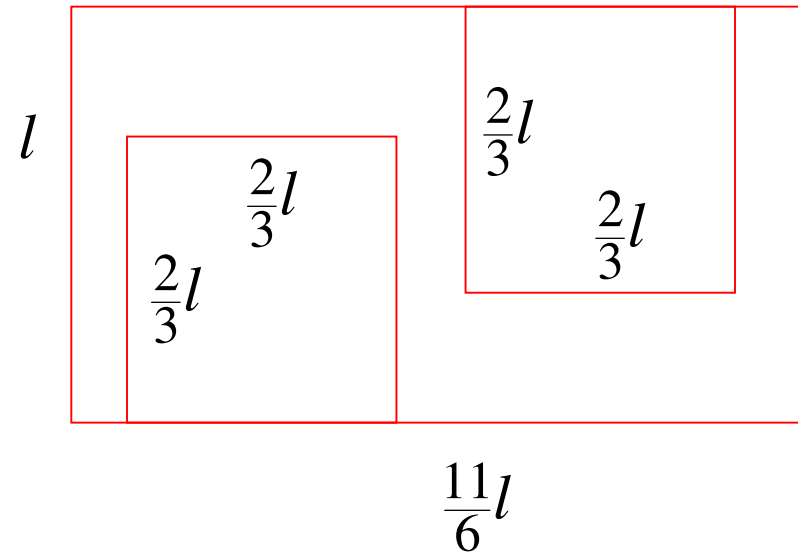
---

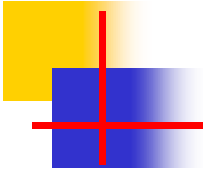
Consider a unit-disk graph, where the two 'teeth' do not cross:

*Case 1:*



*Case 2:*

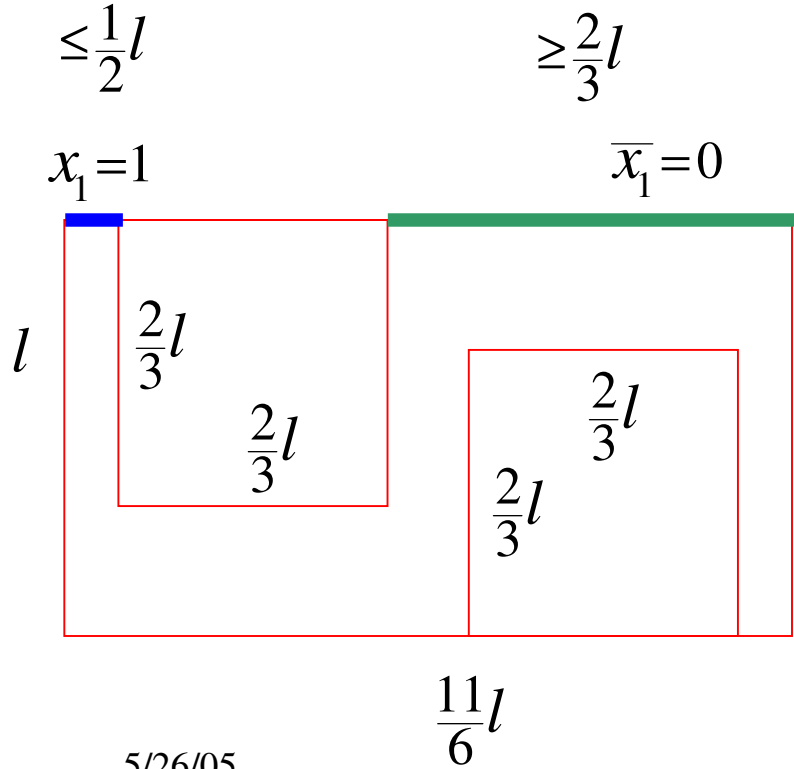




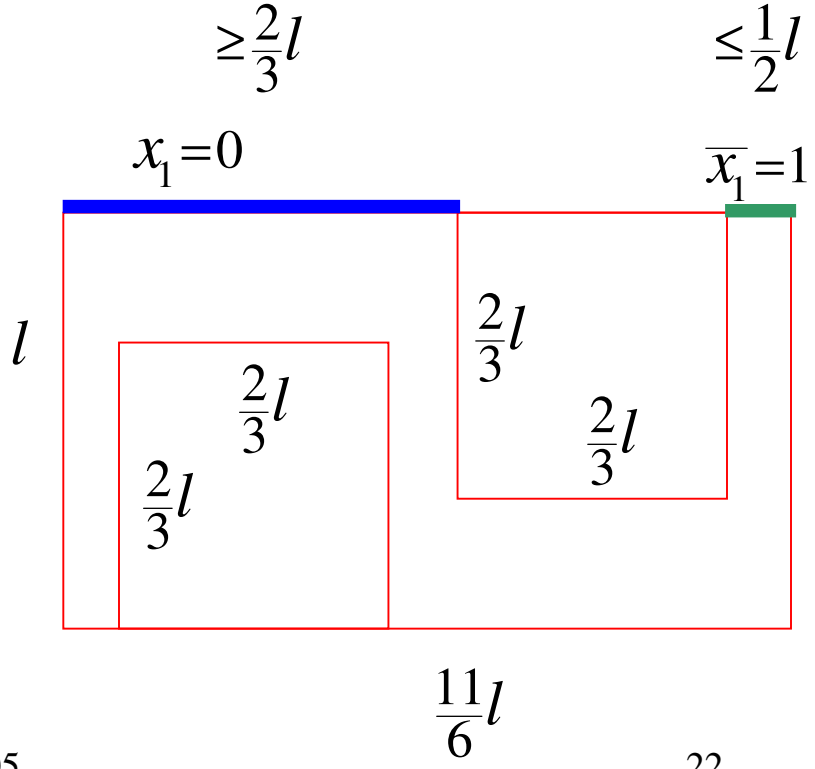
# 0/1 block

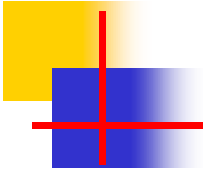
Consider a unit-disk graph, where the two 'teeth' do not cross:

Case 1:



Case 2:



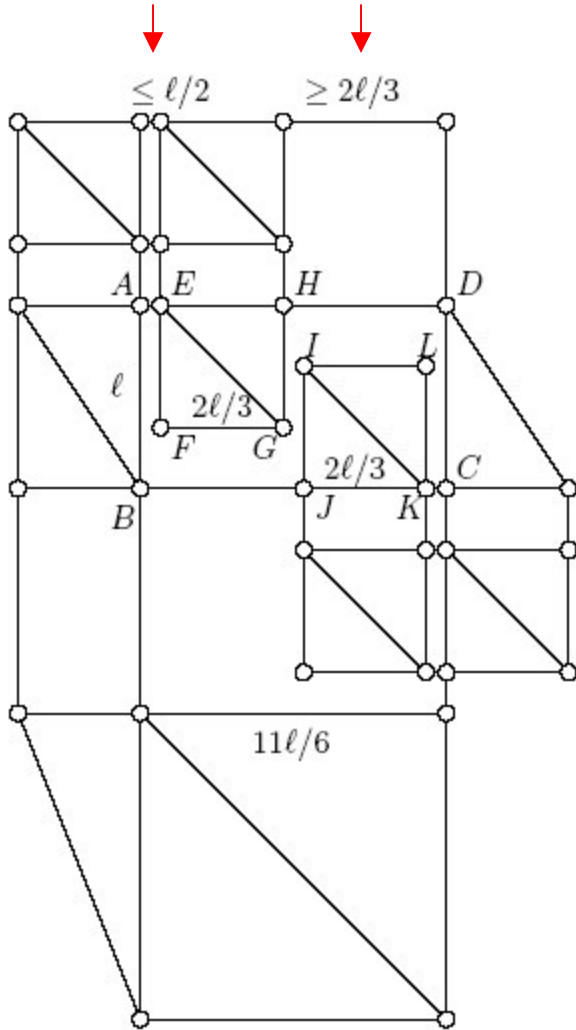


# 0/1 block --- true realization by UDG

Case 1:

$$x_1 = 1$$

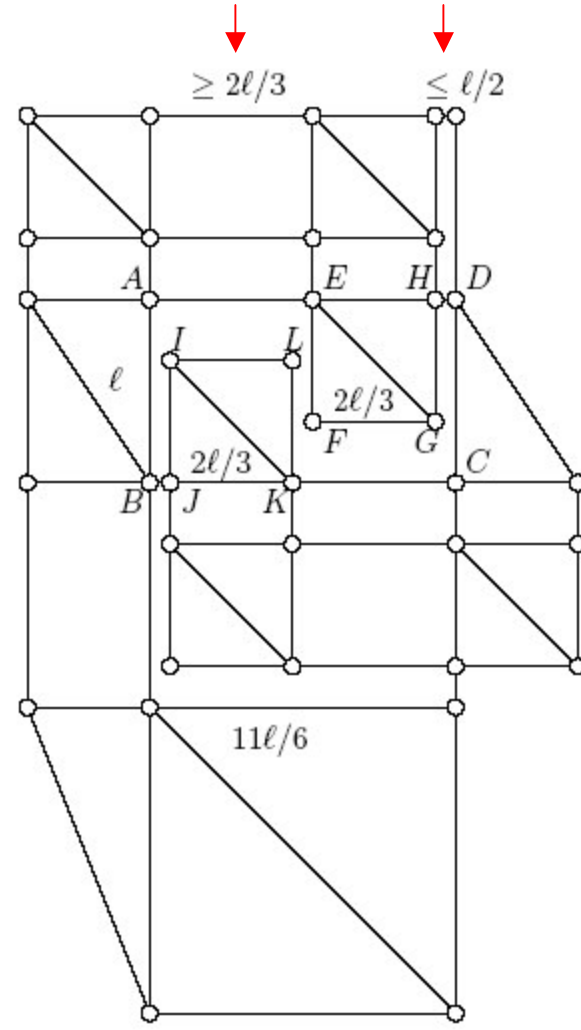
$$\bar{x}_1 = 0$$



Case 2:

$$x_1 = 0$$

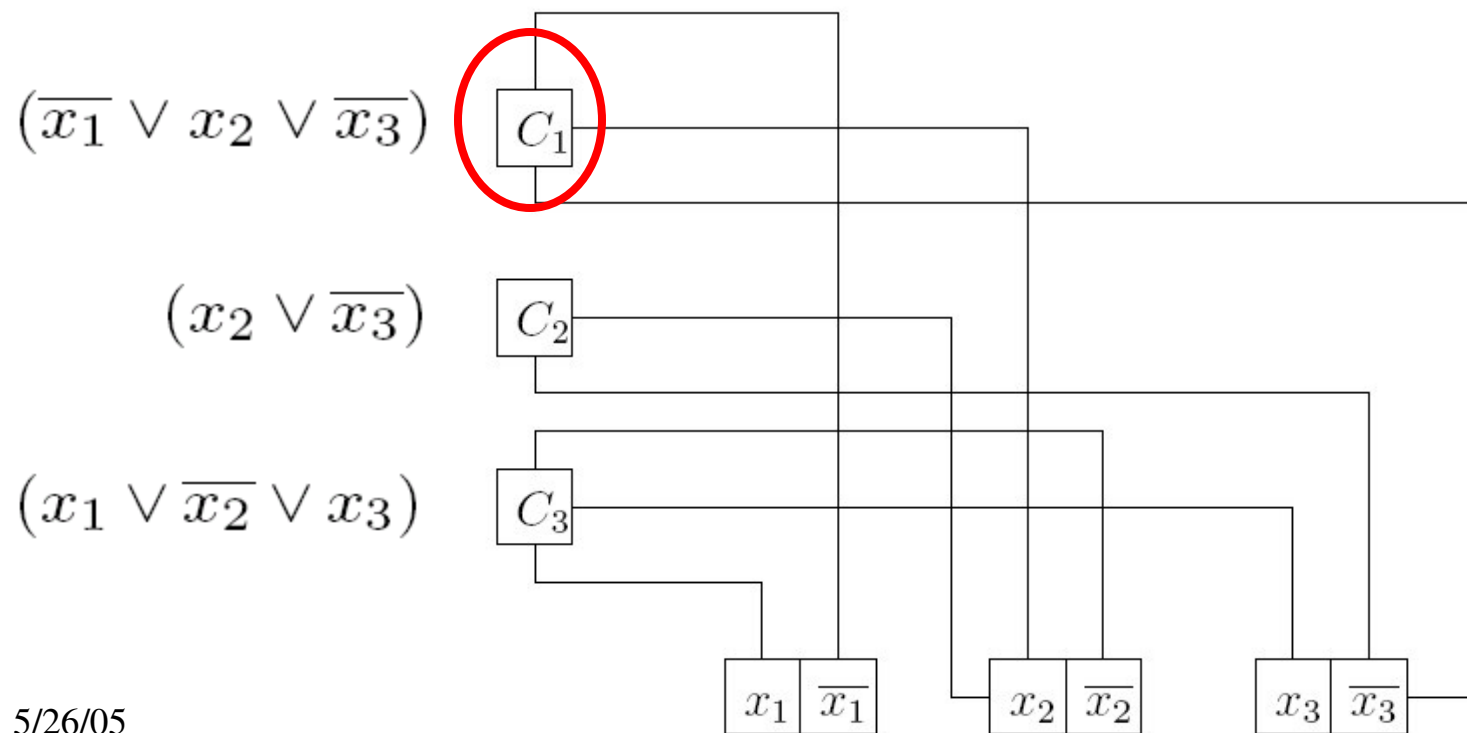
$$\bar{x}_1 = 1$$



# Formulate the 3SAT problem as a graph

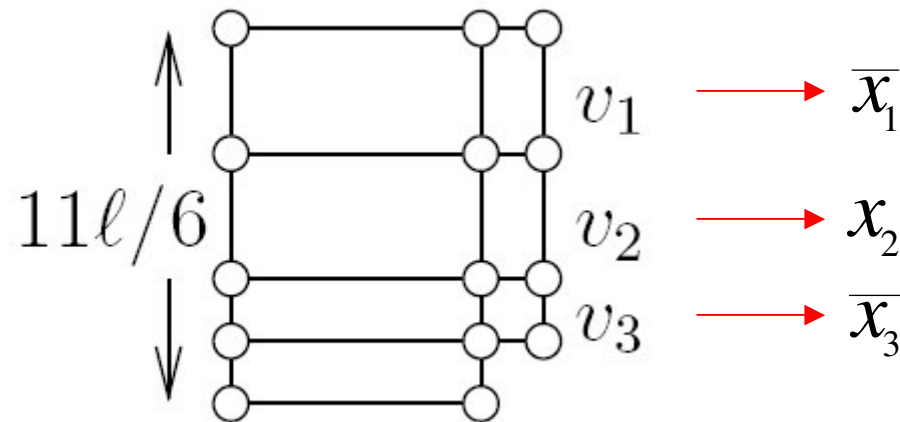
3SAT instance:  $(\overline{x_1} \vee x_2 \vee \overline{x_3}) \wedge (x_2 \vee \overline{x_3}) \wedge (x_1 \vee \overline{x_2} \vee x_3)$

3SAT graph:



# Realize a 3SAT clause by a UDG

3SAT clause:  $(\overline{x_1} \vee x_2 \vee \overline{x_3})$



$$v_1, v_2, v_3: \leq \frac{1}{2}l \quad \text{or} \quad \geq \frac{2}{3}l$$



# NP hardness of embedding

---

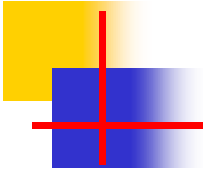
- Finding an embedding **without incorrect crossings** is NP-hard.

Therefore,

- Find a **valid** embedding is NP-hard.

Lemma: a  $\sqrt{2}$ -approx. embedding has no incorrect crossings.

- **$\sqrt{2}$ -approximate** embedding is also NP-hard.



## Part II

---

# An embedding method by Linear Programming

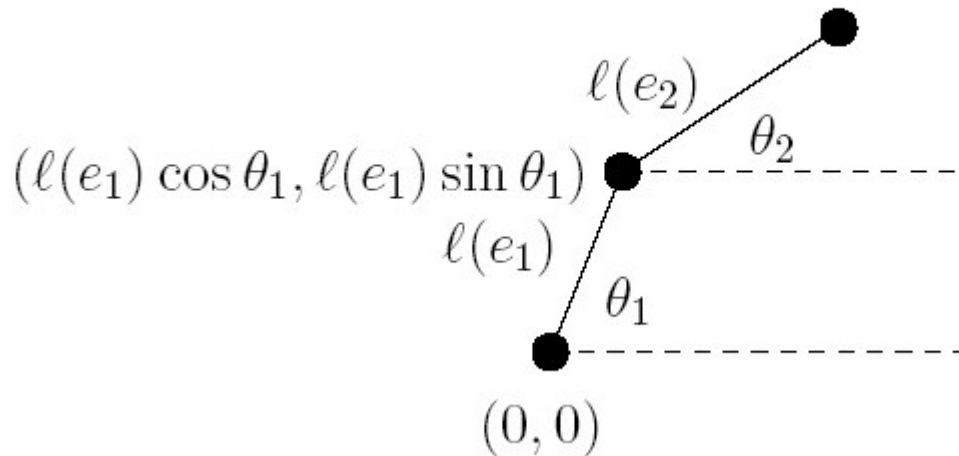
# An embedding method by LP

We formulate a linear programming problem:

**Variables:** lengths of edges,  $l(e)$ .

- Fix a node as the origin.
- All angles are measured against  $x$ -axis.

$$(l(e_1) \cos \theta_1 + l(e_2) \cos \theta_2, l(e_1) \sin \theta_1 + l(e_2) \sin \theta_2)$$





# An embedding method by LP

---

We formulate a linear programming problem:

**Variables:** lengths of edges,  $l(e)$ .

**Linear Constraints:**

1. Edge length constraint

$$0 \leq l(e) \leq 1$$

2. Cycle constraint: for a cycle with edges  $\{e_1, e_2, \dots, e_p\}$

$$\sum_{i=1}^p l(e_i) \cos \theta_{e_i} = 0,$$

$$\sum_{i=1}^p l(e_i) \sin \theta_{e_i} = 0.$$

# An embedding method by LP

We formulate a linear programming problem:

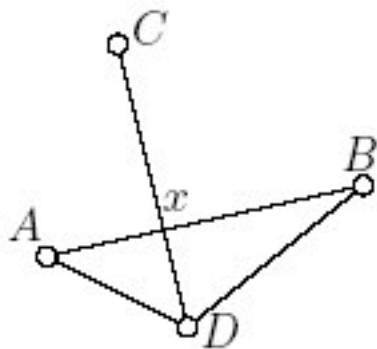
**Variables:** lengths of edges,  $\ell(e)$ .

**Linear Constraints:**

3.  $\exists$  edges AB, BC, and no AC

$$\ell(AB) + \ell(BC) > 1$$

4. Crossing edge constraint:

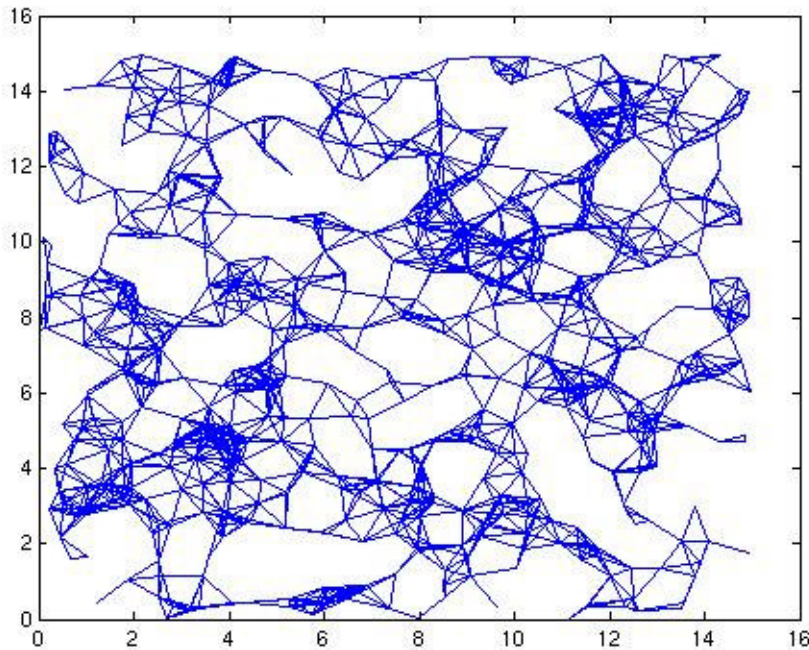


$$\ell(CD) \geq |xD| = \ell(AD) \frac{\sin \angle DAB}{\sin(\angle ADC + \angle DAB)}.$$

# Practical embedding using local angles

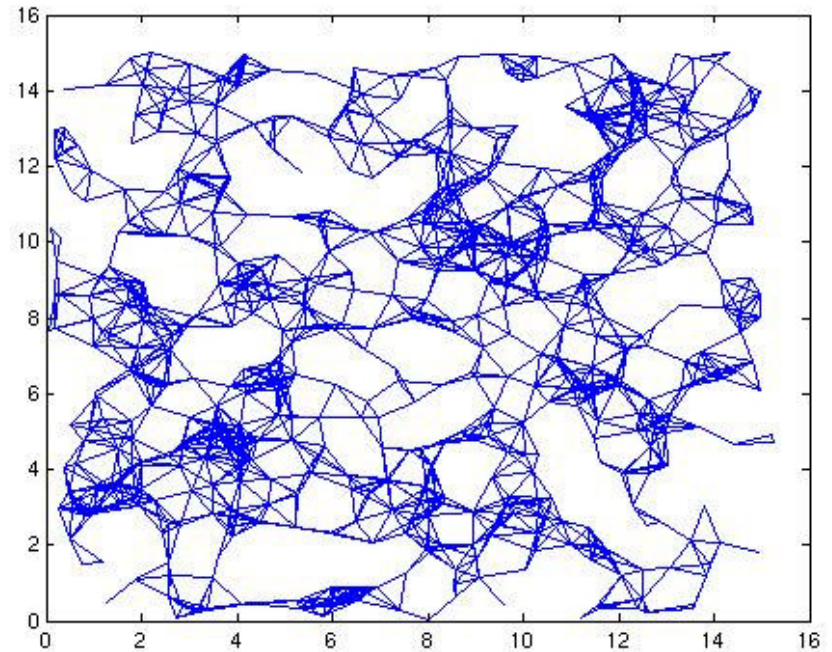
The LP doesn't necessarily produces a valid embedding, but it works well in practice.

True Network (600 nodes)



5/20/03

Embedding



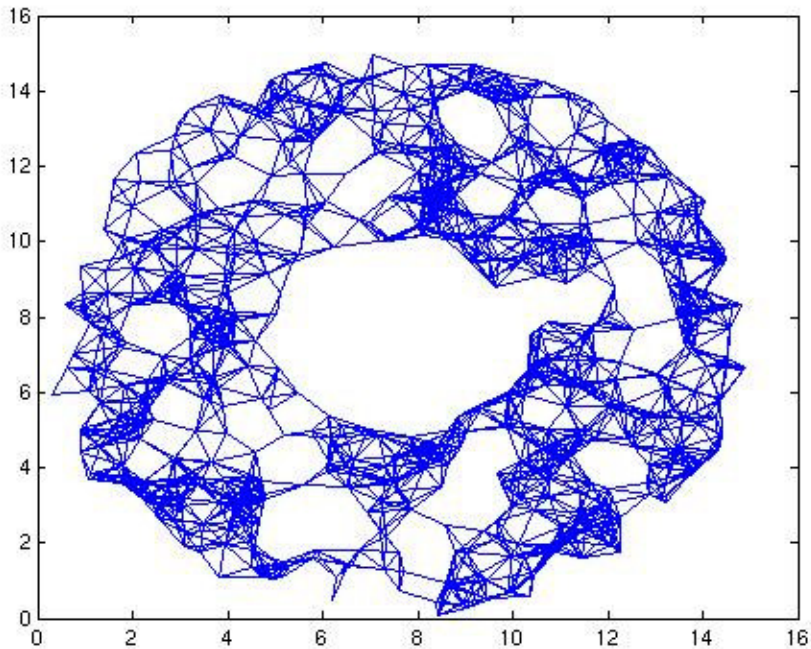
MODIFICUS

51

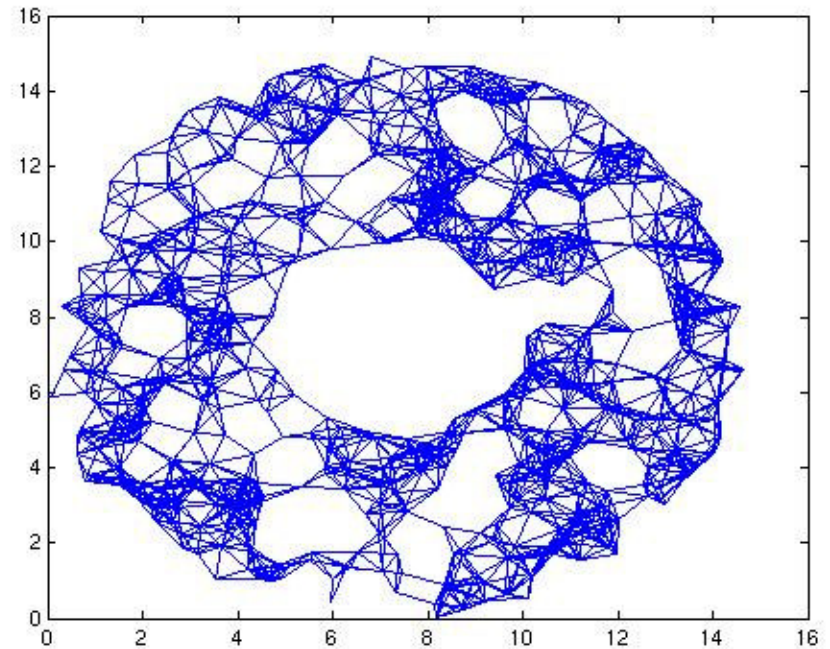
# Practical embedding using local angles

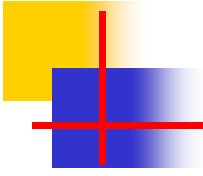
The LP doesn't necessarily produces a valid embedding, but it works well in practice.

True Network (600 nodes)



Embedding



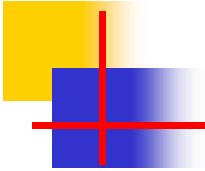


# Variations

---

Can be adapted to:

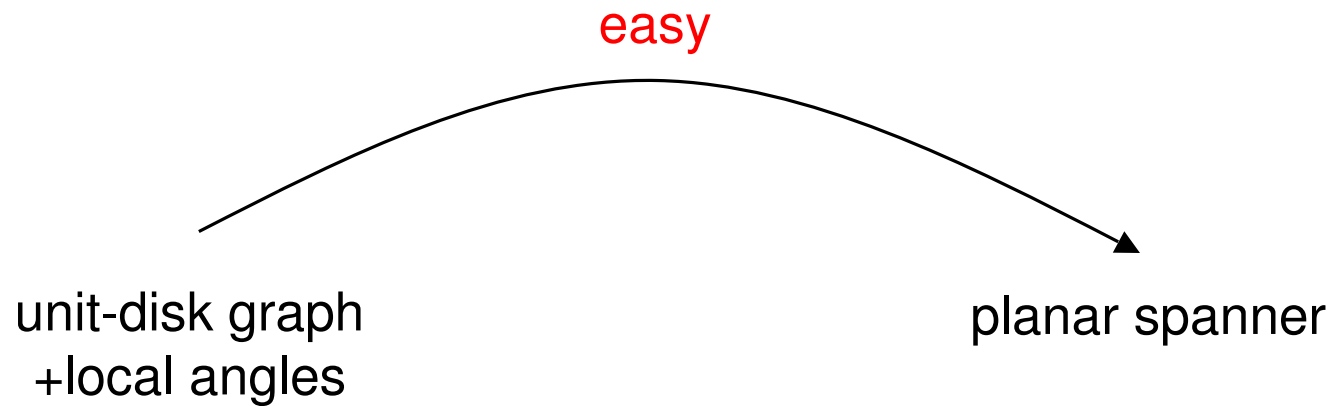
- Quasi unit disk graph model.
- Noisy angle measurements.



## Part III

---

Find a planar spanner by angles.





# Planar spanner

---

A planar spanner of a graph  $G$ :

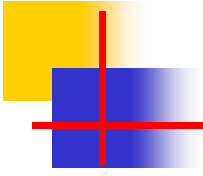
- a subgraph  $G'$
- $G'$  is planar – no crossing edges
- $\forall$  two nodes, the shortest-path distance between them in  $G'$  is at most a **constant** times that in  $G$ .



# Planar spanner construction by local angles

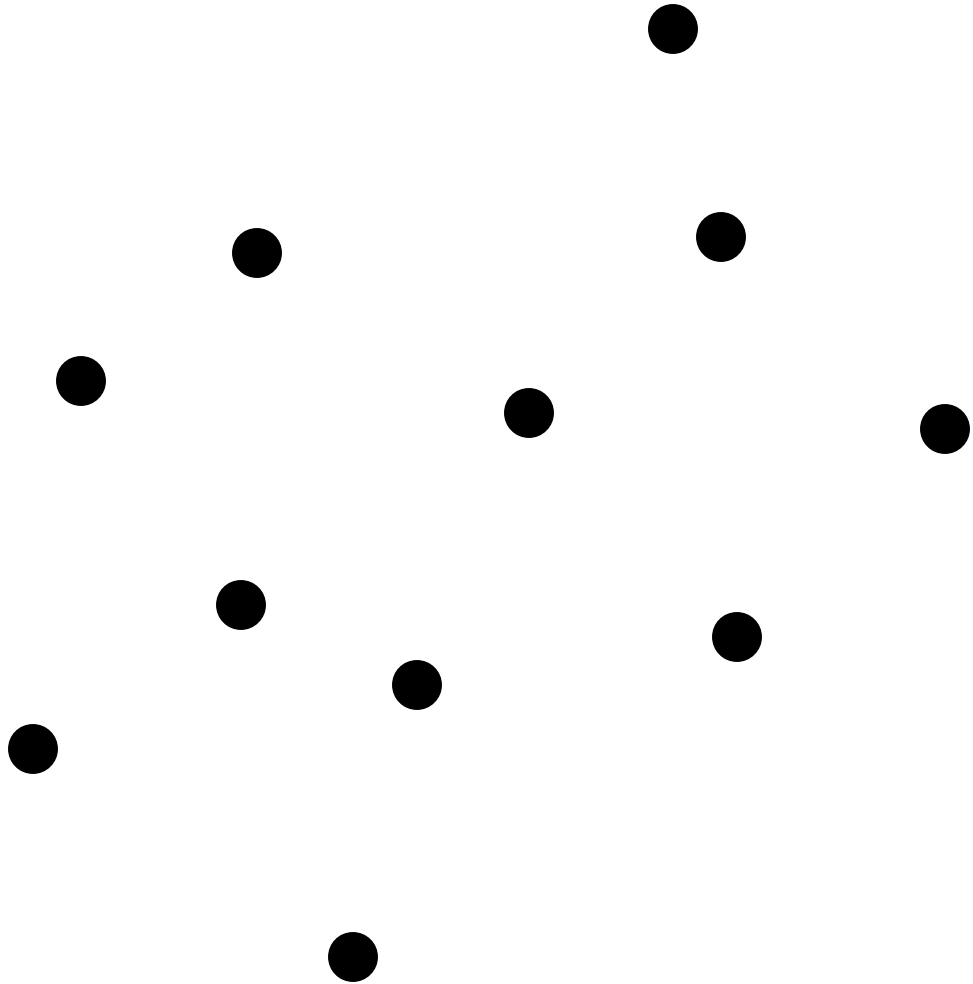
---

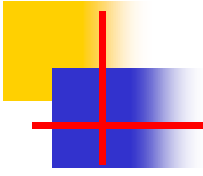
Basic idea: use the restricted Delaunay graph.



# Voronoi diagram

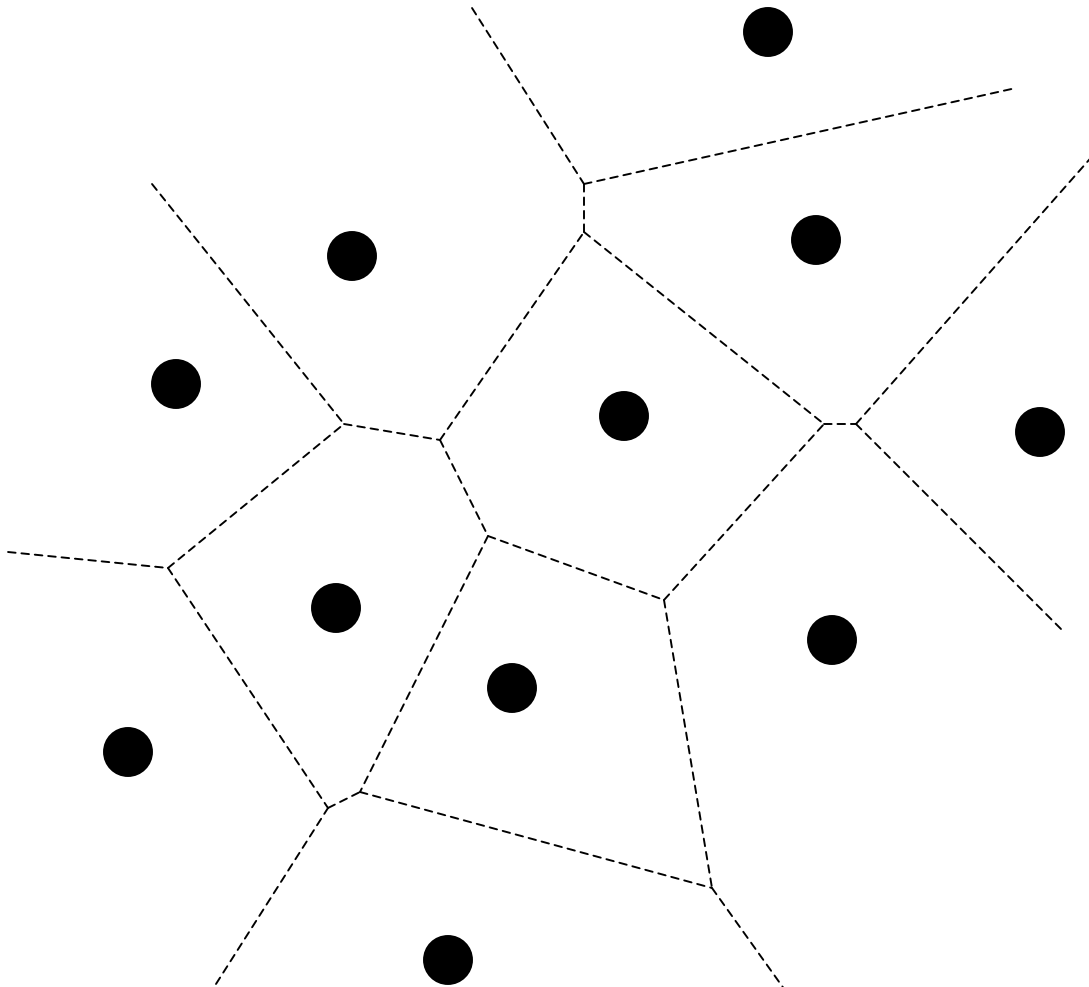
---





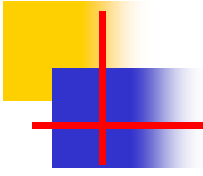
# Voronoi diagram

---



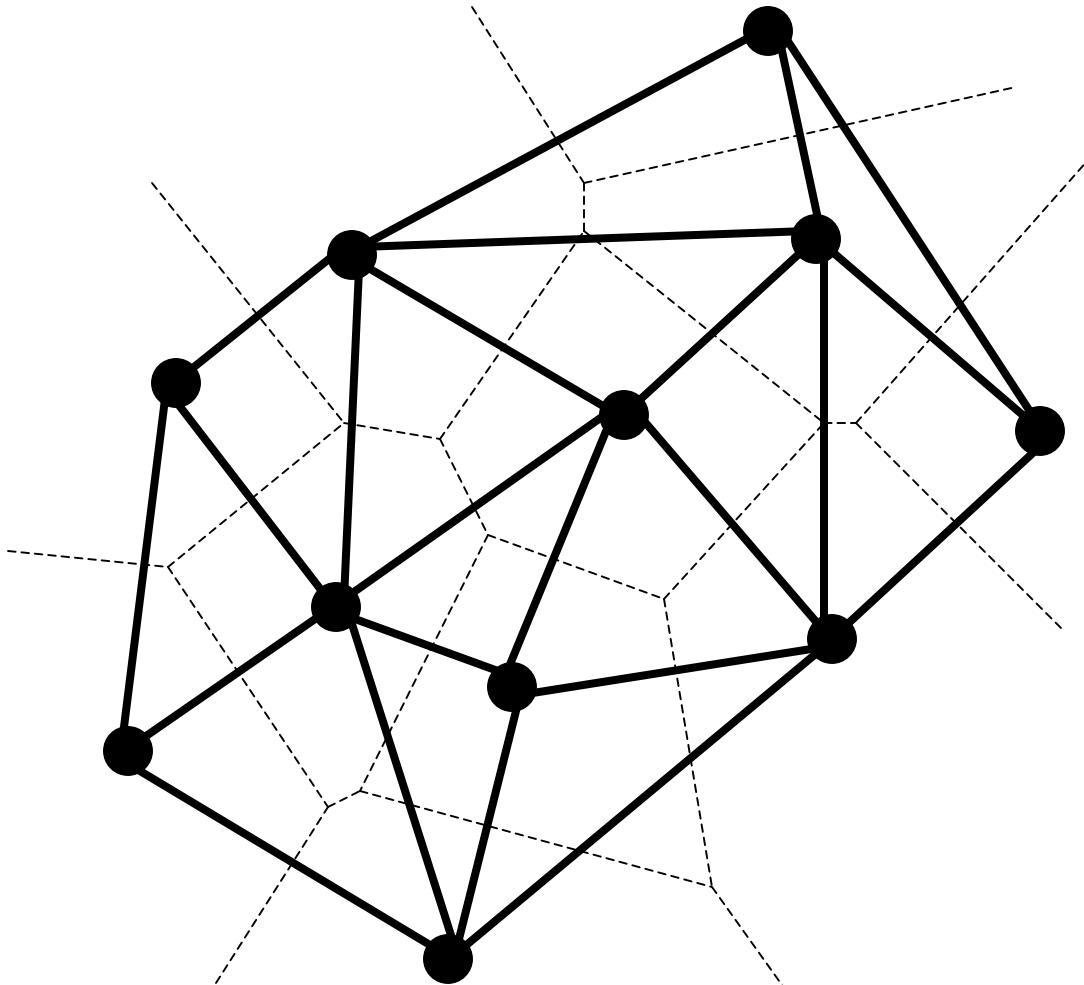
5/26/05

MobiHoc'05



# Delaunay triangulation

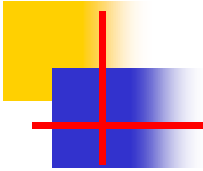
---



5/26/05

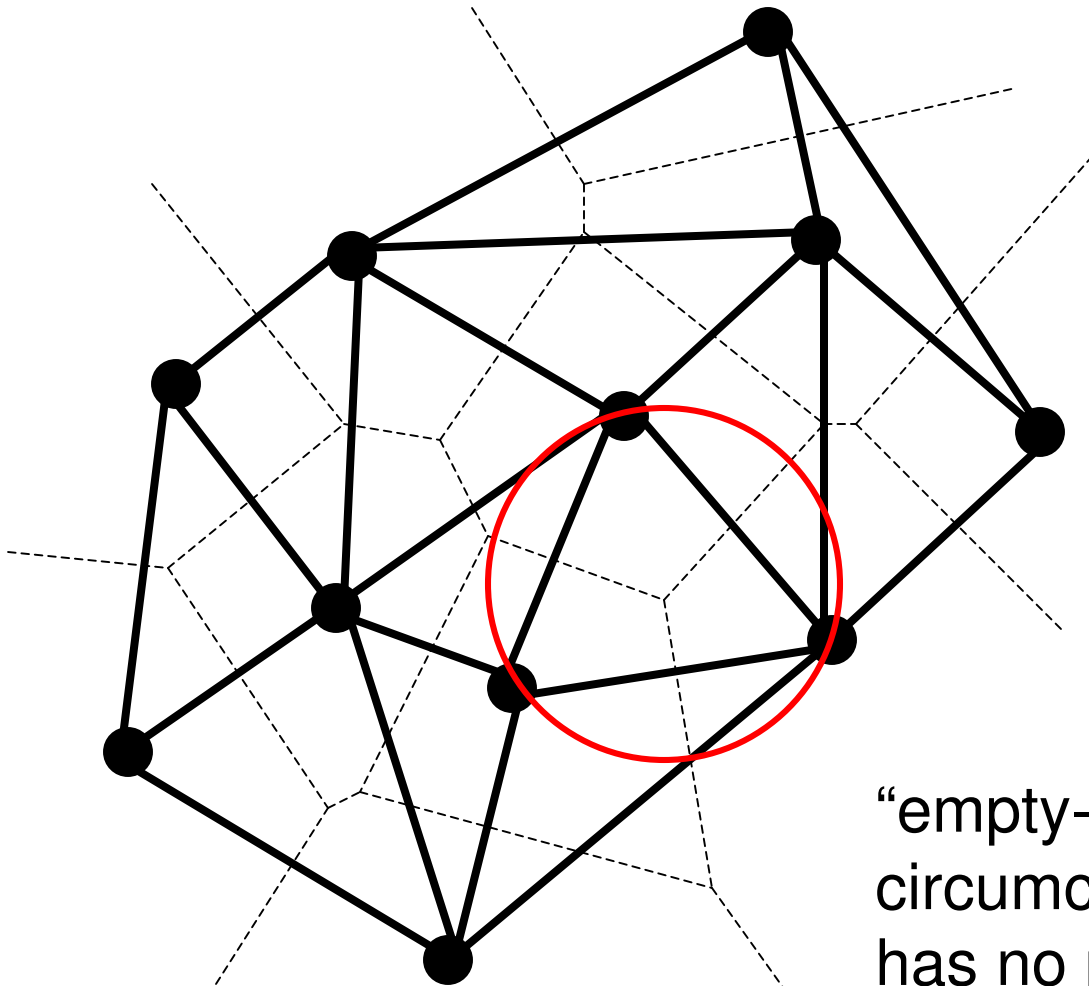
MobiHoc'05

39

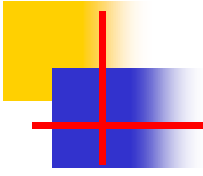


# Delaunay triangulation

---

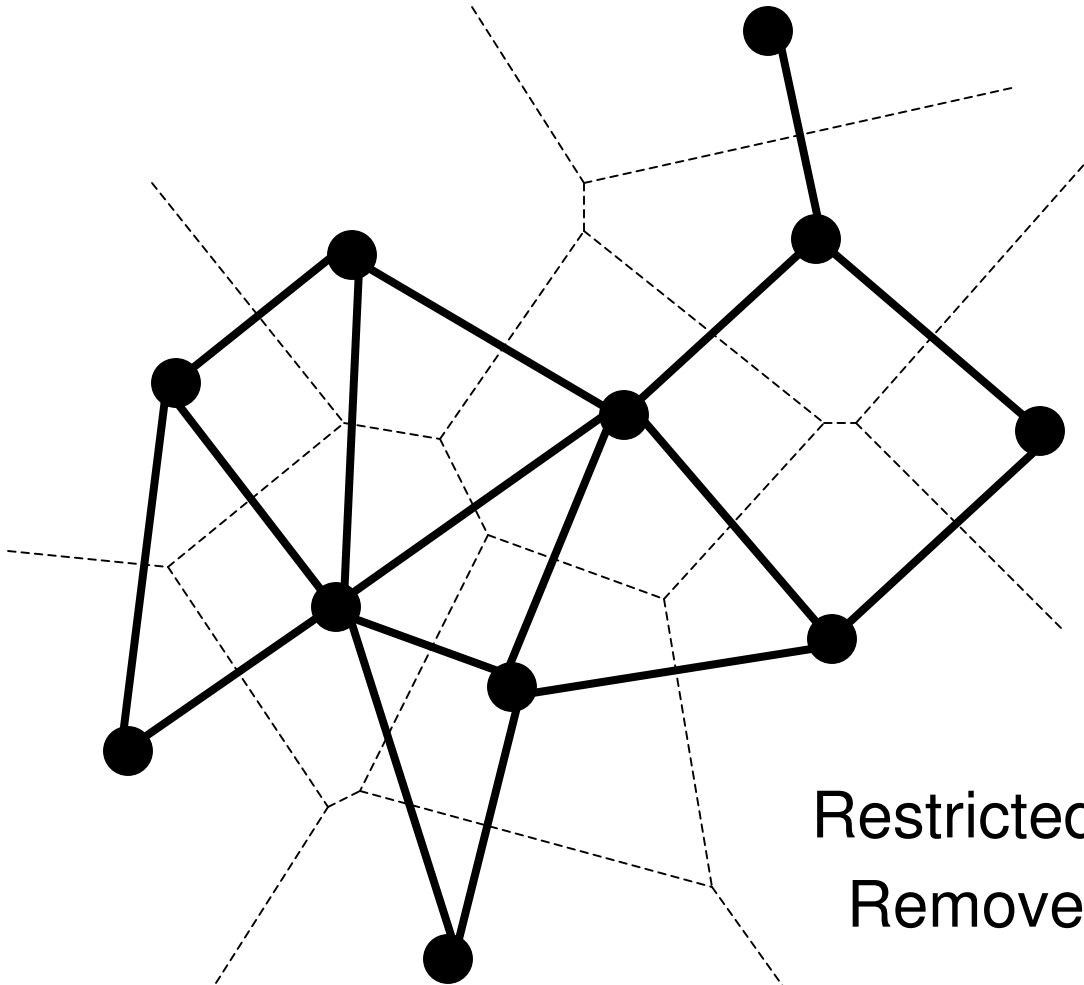


“empty-circle” property:  
circumcircle of a Delaunay  $\Delta$   
has no points inside.



# Restricted Delaunay graph

---



Restricted Delaunay graph:  
Remove edges longer than 1



# Restricted Delaunay graph

---

It's known that [GGHZZ01, LCW02]:  
A restricted Delaunay graph is a planar spanner of the  
unit-disk graph.



# Planar spanner construction

---

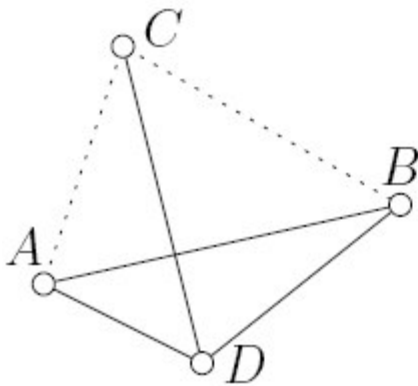
- Find a planar subgraph that contains all **short** ( $\leq 1$ ) Delaunay edges.

Key operation: If two edges in the unit-disk graph cross, **remove** the one that is **not** in the Delaunay triangulation.

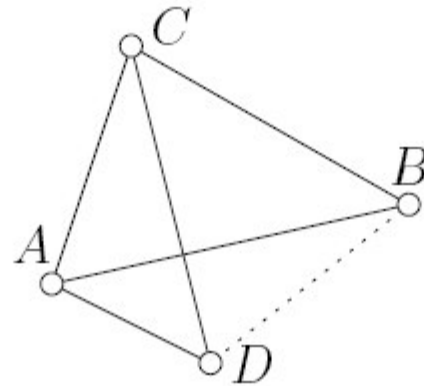
- How to tell that an edge is **not** in the Delaunay triangulation?

# Removing non-Delaunay edges

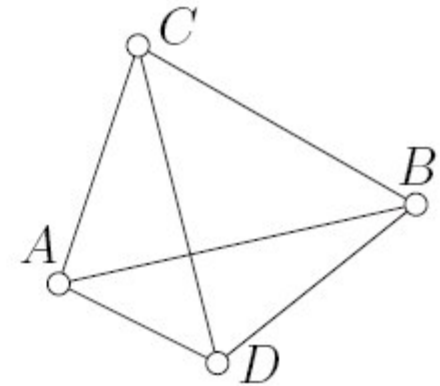
If two edges  $AB$ ,  $CD$  cross, there are only three cases:



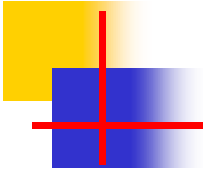
(i)



(ii)

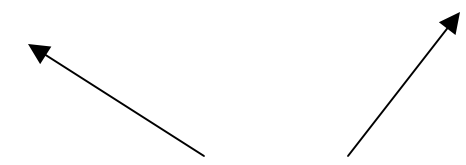
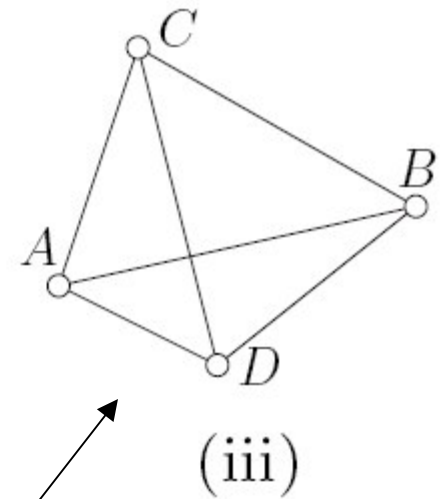
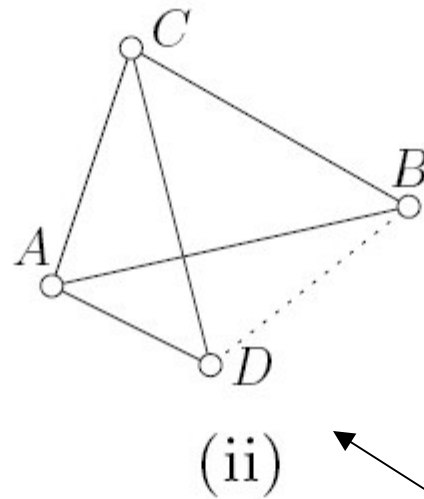
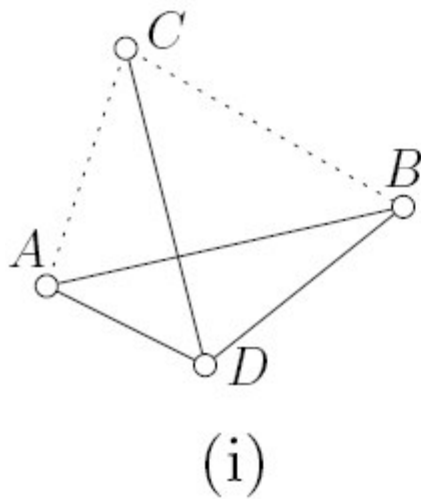


(iii)



# Removing the non-Delaunay edge

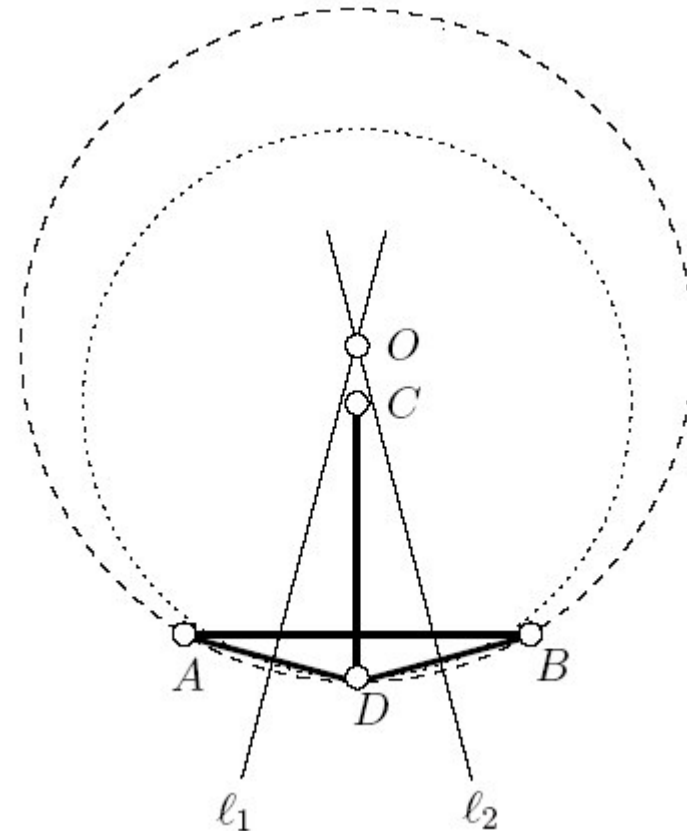
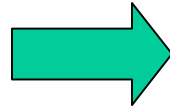
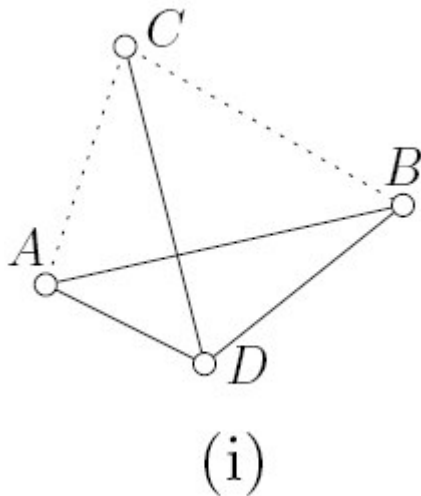
If two edges  $AB$ ,  $CD$  cross, there are only three cases:



The shape is fixed!  
Easy to tell which edge is not a Delaunay edge.

# Removing the non-Delaunay edge

Case (i) : Use the “empty-circle” test of Delaunay triangulation



Conclusion: The edge  $AB$  is not a Delaunay edge.



# Planar spanner by local angles information

---

Theorem: Given a unit-disk graph and local angles, one can construct a planar spanner subgraph with spanning ratio 2.42.

Note that we **don't** need to know the embedding.



# Conclusion

---

- What **CANNOT** be done by local angles:
  - Embedding of a UDG by angles is NP-hard!
  
- What **CAN** be done by local angles
  - Find a planar spanner subgraph;
  - An embedding method by LP.



## Future work

---

- In theory, develop an approximate embedding algorithm.
- Find a distributed localization algorithm by using local angles.