Virtual Coordinates for Ad Hoc and Sensor Networks

Thomas Moscibroda

Regina O'Dell

Mirjam Wattenhofer

Roger Wattenhofer

Unit Disk Graph Approximation

Fabian Kuhn

Thomas Moscibroda

Roger Wattenhofer

ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

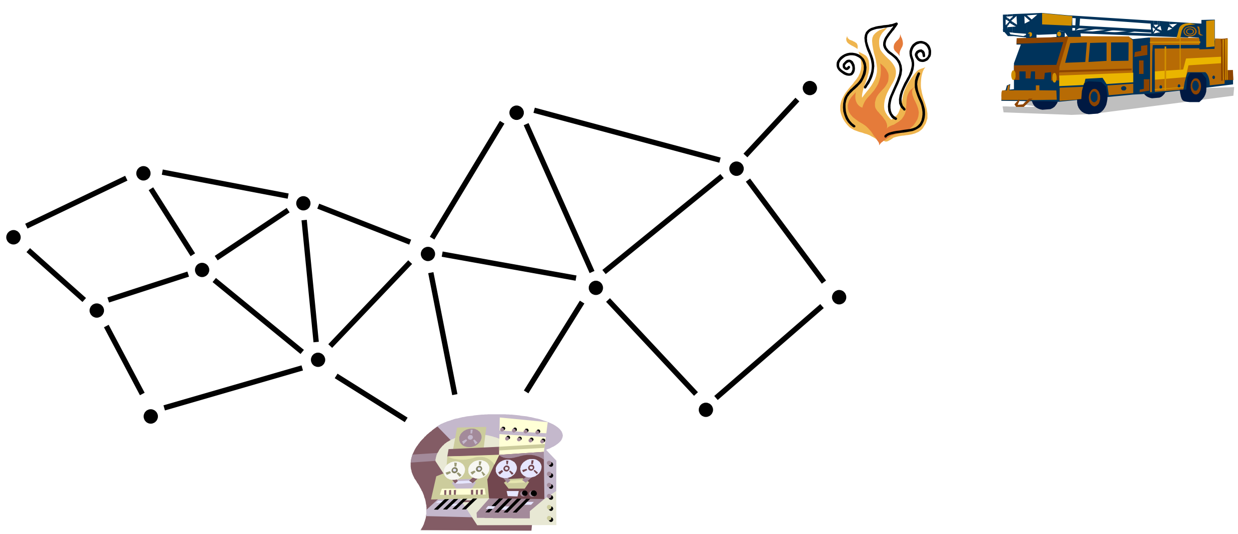
Distributed Computing Group

Ad Hoc and Sensor Networks

- Increasingly wide range of applications
  - Monitoring
  - Surveillance
  - Data-Gathering

⇒ **Position-awareness is key-issue**

- In sensor networks, positioning is indispensable

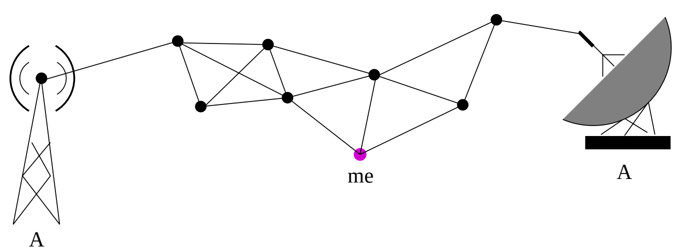


Positioning

- Attach GPS to each sensor node
  - Oftern undesirable or impossible
  - GPS receivers clumsy, expensive, and energy-inefficient

- Equip only a few designated nodes with a GPS
  - Anchor (landmark) nodes have GPS
  - Non-anchors derive their position through communication (e.g., count number of hops to different anchors)

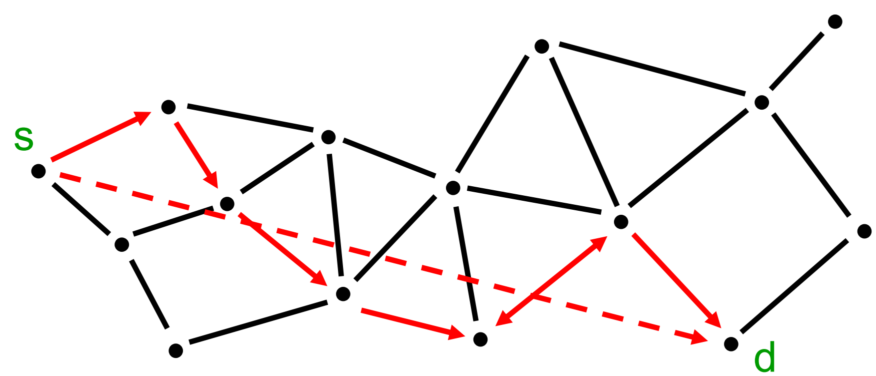
  → Typical positioning approach
  [Niculescu, Nath, Globecom 2001],...
  [Nagpal, Shrobe, Bachrach, IPSN 2003],...



Anchor density determines quality of solution

How about no anchors at all...?

- In absence of anchors...
  → ...nodes are clueless about real coordinates.
- For many applications, real coordinates are not necessary
  → Virtual coordinates are sufficient
  → Geometric Routing requires only virtual coordinates
    - Require no routing tables
    - Resource-frugal and scalable
    - GFG/GPSR [Bose et al., DIALM 1999][Karp, Kung, MOBICOM 2000]
    - GOAFR[Kuhn, Wattenhofer, Zhang, Zollinger, PODC 2003]



Virtual Coordinates

- Idea:

  Close-by nodes have similar coordinates

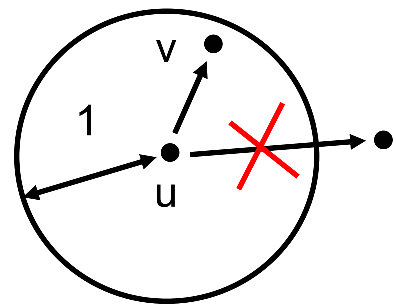
  Distant nodes have very different coordinates

→ Similar coordinates imply physical proximity!

- Applications
  - Geometric Routing
  - Locality-sensitive queries
  - Obtaining meta information on the network
  - Anycast services („*Which of the service nodes is closest to me?*“)
  - Internet mapping

Model

- Unit Disk Graph (UDG) to model wireless multi-hop network
  - Two nodes can communicate iff Euclidean distance is at most 1



- Sensor nodes may not be capable of
  - Sensing directions to neighbors
  - Measuring distances to neighbors

- Goal: Derive topologically correct coordinate information from connectivity information only.
  - Even the simplest nodes can derive connectivity information

Context

	Distance/Angle information	Connectivity information only
With Anchors	Positioning (Solution quality depends on anchor density)	
No Anchors	Distance/Angle based Virtual Coordinates	Connectivity based Virtual Coordinates

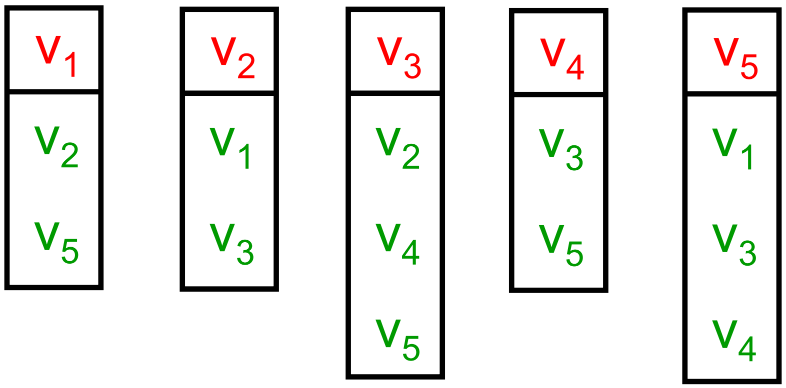
In this talk

Overview

- Introduction
  - Positioning
  - Virtual Coordinates
- Model and problem statement
- Upper Bound
- Lower Bound
- Conclusions

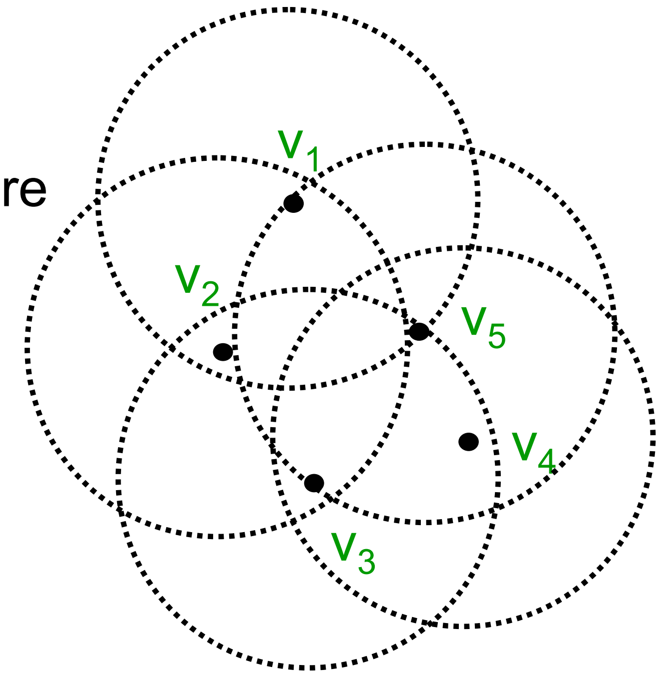
Virtual Coordinates ⟷ UDG Embedding

- Given the connectivity information for each node...



...and knowing the underlying graph is a UDG...

- ...find a UDG embedding in the plane such that all connectivity requirements are fulfilled! (→ Find a realization of a UDG)



This problem is NP-hard!

(Simple reduction to *UDG-recognition* problem, which is NP-hard)

[Breu, Kirkpatrick, Comp.Geom.Theory 1998]

UDG Approximation – Quality of Embedding

- Finding an exact realization of a UDG is NP-hard.
  → Find an embedding r(G) which approximates a realization.

- Particularly,
  → Map adjacent vertices (edges) to points which are close together.
  → Map non-adjacent vertices („non-edges“) to far apart points.

- Define quality of embedding q(r(G)) as:

Ratio between longest edge to shortest non-edge in the embedding.

Let ρ(u,v) be the distance between points u and v in the embedding.

$$q(r(G)) := \frac{\max_{\{u,v\}\in E} \rho(u,v)}{\min_{\{u',v'\}\notin E} \rho(u',v')}$$

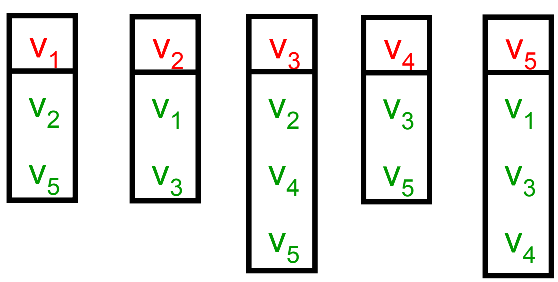
UDG Approximation

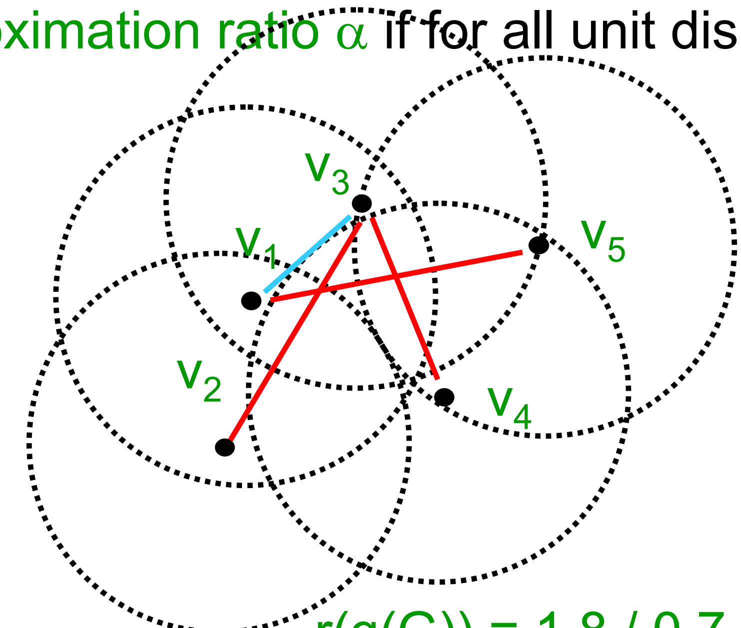
- For each UDG G, there exists an embedding r(G), such that, $q(r(G)) \leq 1$.
  (a realization of G)

$$q(r(G)) := \frac{\max_{\{u,v\}\in E} \rho(u,v)}{\min_{\{u',v'\}\notin E} \rho(u',v')}$$

- Finding such an embedding is NP-hard
- An algorithm ALG achieves approximation ratio α if for all unit disk graphs G, $q(r_{ALG}(G)) \leq \alpha$.

- Example:





r(q(G)) = 1.8 / 0.7 = 2.6

Previous work and our results

- There are a few virtual coordinates algorithms
  [Rao et al., MOBICOM 2003], [Shang et al., MOBIHOC 2003],
  [Biswas, Ye, IPSN 2004]
- All of them evaluated only by simulation on random graphs
- We give them first provable approximation algorithm

> Our algorithm achieves an approximation ratio of $O(\log^{2.5} n \sqrt{\log \log n})$, n being the number of nodes in G.

- Independently, it has been shown that there is no PTAS for the virtual coordinates problem. [Lotker, Martinez de Albeniz, Perennes, ADHOC-NOW 2004]
- We give the first actual lower bound on the approximability.

> There is no algorithm with approximation ratio better than $\sqrt{3/2} - \epsilon$, unless P=NP.

Overview

- Introduction
  - Positioning
  - Virtual Coordinates
- Model and problem statement
- Upper Bound
- Lower Bound
- Conclusions

Approximation Algorithm - Overview

- Four major steps

1. Compute metric on MIS of input graph → Spreading constraints
   (Key conceptual difference to previous approaches!)
2. Volume-respecting, high dimensional embedding
3. Random projection to 2D
4. Final embedding

UDG Graph G with MIS M.

↓

Approximate pairwise distances between nodes such that, MIS nodes are neatly spread out.

↓

Volume respecting embedding of nodes in $\boldsymbol{R}^n$ with small distortion.

↓

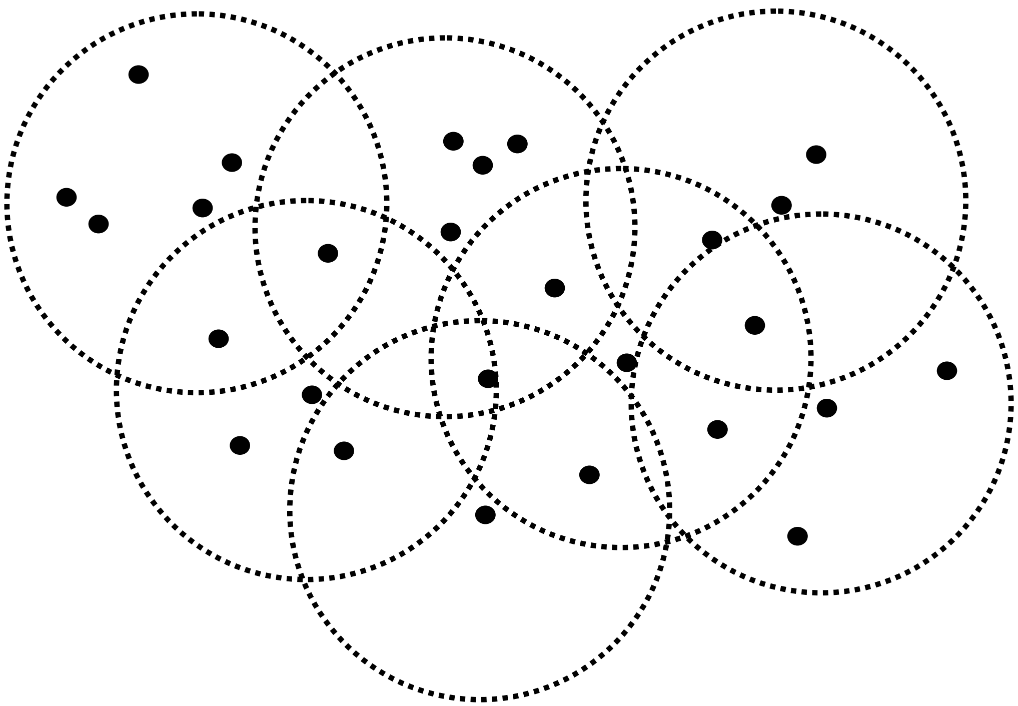
Nodes spread out fairly well in $\boldsymbol{R}^2$.

↓

Final *embedding of G in* $\boldsymbol{R}^2$.

Step 1 – Linear Constraints

- Problem: UDG conditions are inherently non-linear.
- Consider MIS in a UDG...



- ...in each region of radius R, there are at most $O(R^2)$ MIS nodes.

Step 1 – Spreading Constraints

- Set of non-edges (independent sets) must be sufficiently far apart.
- Idea: Use spreading constraints to compute approximate distances (metric) between MIS nodes!

[Even et al., FOCS 95]
[Vempala, FOCS 98]

$$\sum_{v \in IS} x_{uv} \leq c|IS|^{3/2} \quad \forall\, IS \subset V, \forall\, v \in V$$

→ Average distance of any set of k points from any given point v is $\Omega(k^{1/2})$.

→ In any region of radius R, there are at most $O(R^2)$ points.

→ Now, we have linear constraints!

Step 1 – Linear Program

$$
\begin{array}{rcll}
x_{uv} & \leq & 1 & \forall \{u,v\} \in E \\
x_{uv} & \leq & \sqrt{n} & \forall u,v \in V \\
x_{uv} & \geq & 0 & \forall u,v \in V \\
x_{uv} + x_{uk} & \geq & x_{vk} & \forall u,v,k \in V \\
\sum_{v \in IS} x_{uv} & \geq & \kappa |IS|^{3/2} & \forall IS \subset V, \forall u \in V
\end{array}
$$

- Feasible solution can be found in polynomial time.

  → Separation oracle

- Gives us metric on nodes.

  → Metric encodes UDG properties

Step 2 – Volume respecting embedding in $\boldsymbol{R}^n$

- Goal: Find embedding such that UDG metric is not distorted
- Problem: Direct embedding in 2D may have very large distortion

- Idea: Compute a volume respecting embedding into $\mathbf{R}^n$.
  [Feige, J. of Computer and System Sciences, 2000]

Volume respecting embeddings:

> Embedding that approximately maintains not only the length of edges, but also the volumes of all k-tuples.

Intuition: large volumes have large projections when being projected to a random lower dimensional subspace.

Step 2 – Volume respecting embedding in $\boldsymbol{R}^n$

- Consider an embedding f of the metric in $\mathbf{R}^N$

- Given set S of k vertices in $\mathbf{R}^N$
- EVol(S) is the volume of the simplex spanned by S
- Vol(S) is the maximum possible volume in any contracting embedding (edges are not allowed to become longer)

- By Feige, it is possible to obtain a (log n, $\log^2$n)-volume respecting embedding, that is,

$$\left(\frac{Vol(S)}{EVol(f(S))}\right)^{\frac{1}{\log n}} \leq \log^2 n$$

Step 3 – Random Projection

- Given positions $v_u^n \in \mathbf{R}^n$ for all nodes $u \in V$ (step 2)
- We now project them to $\mathbf{R}^2$.

Random Projection:

1. Independently choose two random vectors $l_1, l_2 \in \mathbf{R}^n$ of unit length (lines passing through origin).
2. For all $u \in V$, project $v_u^n \in \mathbf{R}^n$ to each line.
3. The $\mathbf{R}^2$ coordinates are $r_u^2 := (v_u^n \cdot l_1, v_u^n \cdot l_2)$.

Properties:

- When projecting a vector from $\mathbf{R}^N$ to a random line in $\mathbf{R}^N$, the length of the vector scales by roughly $1/\sqrt{N}$.
- The probability that a set of k points is projected to a small interval is inversely proportional to the volume of the points.

Step 3 – Random Projection

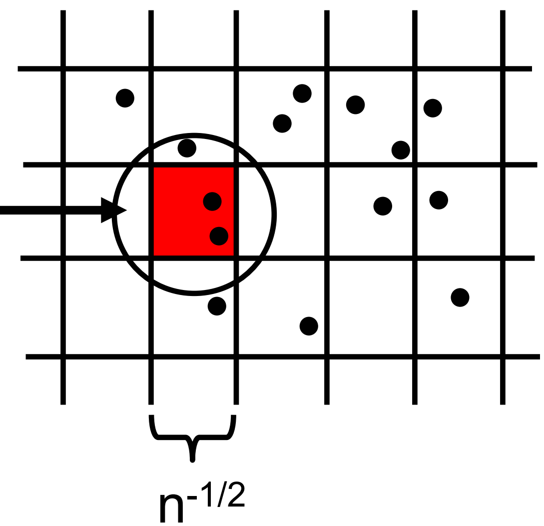
Properties:

- When projecting a vector from RN to a random line in RN, the length of the vector scales by roughly $1/\sqrt{N}$.
- The probability that a set of k points is projected to a small interval is inversely proportional to the volume of the points.

Together with the volume respecting embeddings,...

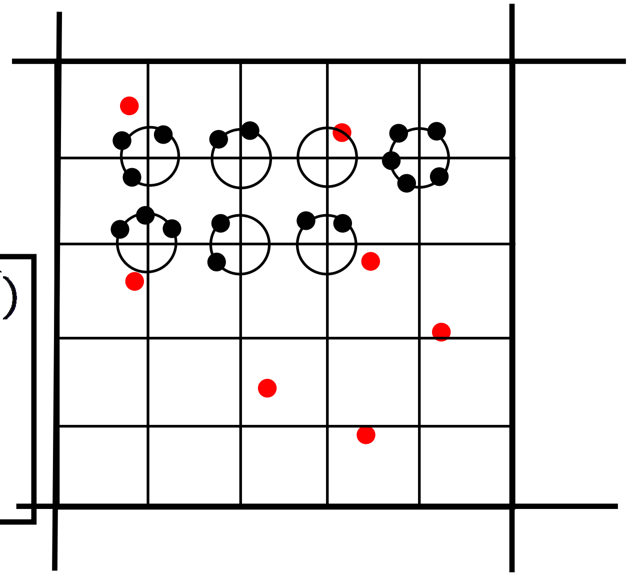
... randomly projected points spread quite well in 2D plane.

If we partition the plane into a grid with cell-width $1/\sqrt{n}$, at most $O(\log^4 n \cdot \log\log n)$ points lie in a cell w.h.p.



Step 4 – Final embedding

- Smallest non-edge must not be too short
  → Spread points within one cell evenly
- Compute a maximal independent set of nodes in each cell
  → Let M be the maximum cardinality of such a MIS in any cell
- For each cell,
  - Construct refined grid with width $1/\sqrt{nM}$.
  - Assign MIS nodes to grid points in this refined grid.
- All other (non-MIS) nodes in G are placed on circles around an arbitrary neighboring MIS node.



Maximum Edge Length $\in O(\log^{2.5} n \cdot \sqrt{\log\log n})$

Minimum Non-Edge Length $\geq$ 1/3

→ $O(\log^{2.5} n \cdot \sqrt{\log\log n})$ Approximation

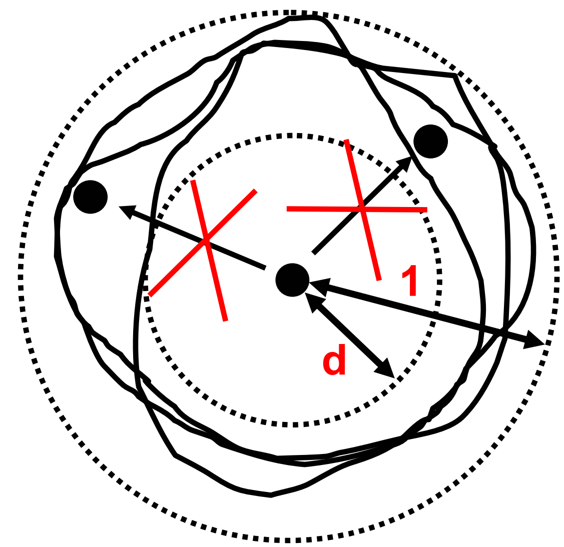
Overview

- Introduction
  - Positioning
  - Virtual Coordinates
- Model and problem statement
- Upper Bound
- Lower Bound
- Conclusions

Quasi Unit Disk Graph

- Definition Quasi Unit Disk Graph:

> Let $V \in \mathbf{R}^2$, and $d \in [0,1]$. The symmetric Euclidean graph $G=(V,E)$, such that for any pair $u,v \in V$
>
> - $\text{dist}(u,v) \leq d \Rightarrow \{u,v\} \in E$
> - $\text{dist}(u,v) > 1 \Rightarrow \{u,v\} \notin E$
>
> is called *d-quasi unit disk graph*.



[Barrière, Fraigniaud, Narayanan, DIALM 2001]

[Kuhn, Wattenhofer, Zollinger, DIALM 2003]

- Note that between d and 1, the existence of an edge is unspecified.

Reduction

- We want to show that finding an embedding with $q(r(G)) \leq \sqrt{3/2} - \epsilon$, where ε goes to 0 for n → ∞ is NP-hard.

- We prove an equivalent statement:

> Given a unit disk graph G=(V,E), it is NP-hard to find a realization of G as a d-quasi unit disk graph with $d \geq \sqrt{2/3} + \epsilon$, where ε tends to 0 for n→∞.

→ Even when allowing non-edges to be smaller than 1, embedding a unit disk graph remains NP-hard!

→ It follows that finding an approximation ratio better than $\sqrt{3/2} - \epsilon$ is also NP-hard.

Reduction

- Reduction from 3-SAT (each variable appears in at most 3 clauses)
- Given a instance C of this 3-SAT, we give a polynomial time construction of $G_C=(V_C, E_C)$ such that the following holds:

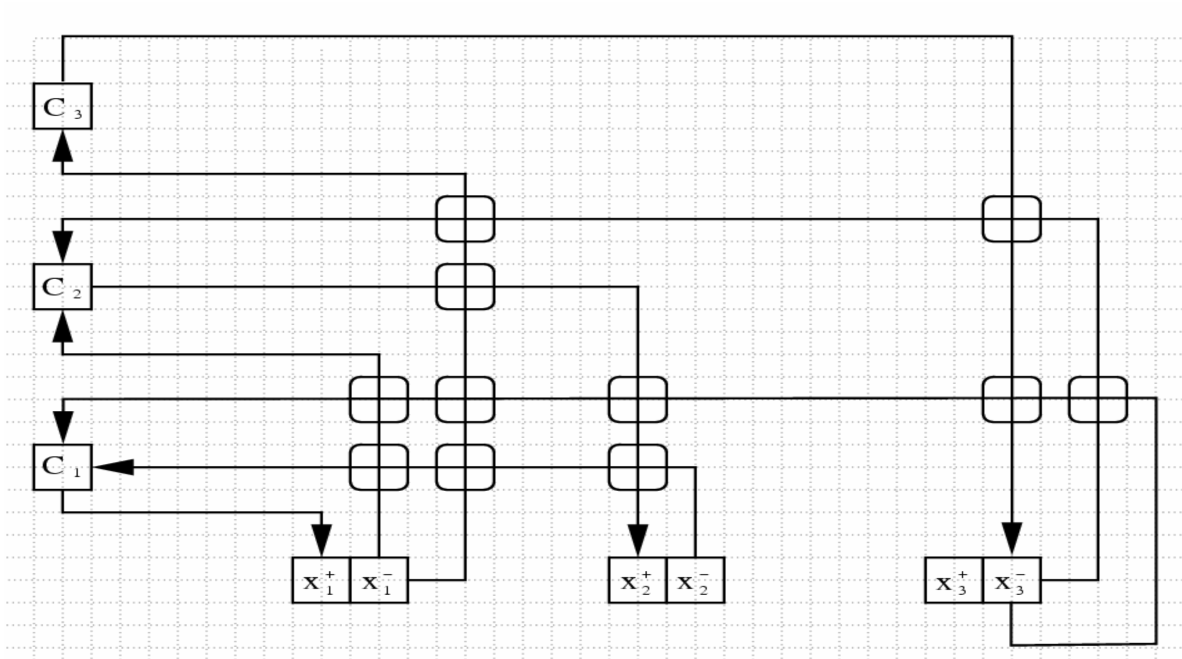
  – C is satisfiable $\Rightarrow$ G_C is realizable as a unit disk graph

  – C is not satisfiable $\Rightarrow$ G_C is not realizable as a d-quasi unit disk graph with $d \geq \sqrt{2/3} + \epsilon$

- Unless P=NP, there is no approximation algorithm with approximation ratio better than $\sqrt{3/2} - \epsilon$.

Proof-Idea

- Construct a grid drawing of the SAT instance.
- Grid drawing is *orientable* iff SAT instance is satisfiable.
- Grid components (clauses, literals, wires, crossings,...) are composed of nodes → Graph G_C.
- G_C is realizable as a d-quasi unit disk graph with $d \geq \sqrt{2/3} + \epsilon$ iff grid drawing is orientable.



Conclusion and Outlook

- Virtual coordinates problem is important!
- Natural formulation as unit disk graph embedding.
  → Clear-cut optimization problem.

Upper Bound : $\alpha \in O(\log^{2.5} n \sqrt{\log \log n})$
Lower Bound : $\alpha \geq \sqrt{3/2} - \epsilon$

→ **Gap between upper and lower bound is huge!**

Open Problems:

- Diminish gap between upper and lower bound
- Distributed Algorithm

Questions? Comments?

Questions?
Comments?

Distributed
Computing
Group

Fabian Kuhn, Thomas Moscibroda, Regina O‘Dell
Mirjam Wattenhofer, Roger Wattenhofer