
Good Quality Virtual Realization of Unit Disk Graphs

Sriram Pemmaraju

Imran Pirwani

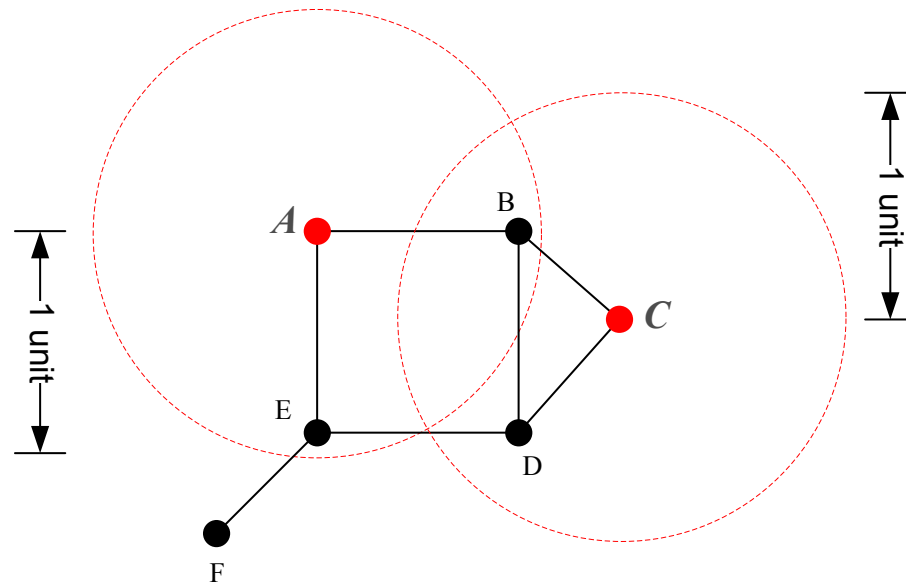
[sriram, pirwani]@cs.uiowa.edu

Department of Computer Science, The University of Iowa
Iowa City, USA

Unit Disk Graphs

A given graph $G = (V, E)$ is a *unit disk graph (UDG)* if there exists a map $\Phi : V \mapsto \mathbb{R}^2$ such that

$$\{u, v\} \in E \iff \|\Phi(u) - \Phi(v)\|_2 \leq 1$$



Recognizing a UDG is hard

INSTANCE: Given $G = (V, E)$ in general form, decide if it is a UDG.

- Breu-Kirkpatrick [1998] showed that recognizing whether a given graph is a UDG or not is NP-hard.
- This implies that computing a *realization* (a map Φ) for a UDG is also NP-hard.

Otherwise, we can find Φ and decide the UDG recognition problem by looking at “squares” of distance between pairs.

Problem Statement

Given a UDG, $G = (V, E)$, in general form (without geometry), compute $\Phi : V \mapsto \mathbb{R}^2$, such that:

$$\frac{\max_{\{u,v\} \in E} \{\|\Phi(u) - \Phi(v)\|_2\}}{\min_{\{u',v'\} \notin E} \{\|\Phi(u') - \Phi(v')\|_2\}}$$

is close to 1.

This measure is known as the *quality* of a realization and is due to Moscibroda et al. [DIALM-POMC 2004].

Hardness of the Problem

- If all edge lengths are known even then the problem is NP-hard. [Aspnes et al. 2004].
- If all the angles between pairs of incident edges is known then the problem is NP-hard. [Bruck et al. 2005].
- If all the angles between pairs of incident edges are known and “slightly noisy” edge lengths are known then the problem is NP-hard [Basu et al. 2006].
- If the smallest non-edge is constrained to be more than 1 unit long then any algorithm that guarantees a realization having quality within $\sqrt{3/2} - \varepsilon, \varepsilon > 0$ implies $P = NP$. [Kuhn et al. 2004].

Best Known Bound

The problem is related to the *minimum bandwidth problem* for graphs.

- Vempala [FOCS 1998] gave an $O(\log^3 n \sqrt{\log \log n})$ approximation.

Known positive results solve an exponential sized linear program by employing the *ellipsoid method*.

Our Results

- We give an $O(\log^{2.5} n)$ quality to the UDG realization problem for the Euclidean plane case.
- We also give an $O(1)$ quality to the UDG realization in $O(1)$ dimensions.

Feature: Our algorithm is purely combinatorial.

Overall Idea (Euclidean Plane case):

1. Compute a *growth restricted cluster graph* $G[C]$ of the UDG where each node in the cluster graph corresponds to a clique in G .
2. Compute a $(\log n, O(\sqrt{\log n}))$ *volume respecting* embedding of $G[C]$.
3. Use *random projection* method to project onto a randomly chosen Euclidean plane.
 - **Step 1. is new to the best of our knowledge.** (via the use of ideas developed by Raghavan-Spinrad, 2001.)
 - **Step 2. is combinatorial.** (via a combination of ideas developed by Lee-Krauthgamer 2003 and Satish Rao 1999).

Cluster Graph

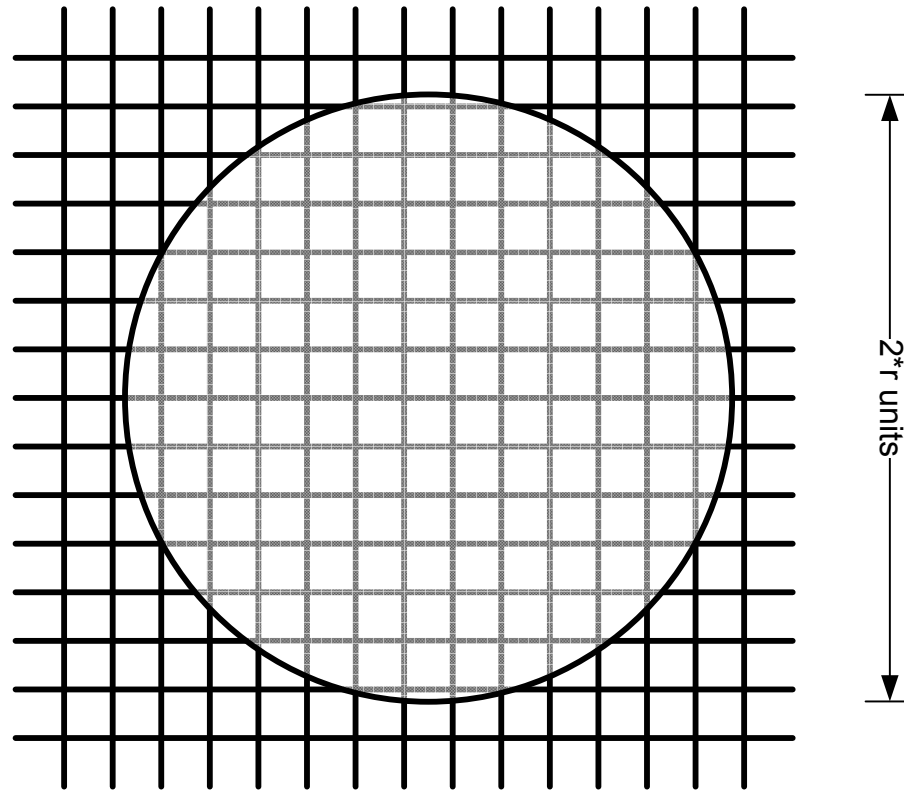
For a graph G , a cluster graph of G is another graph H .

Definition:

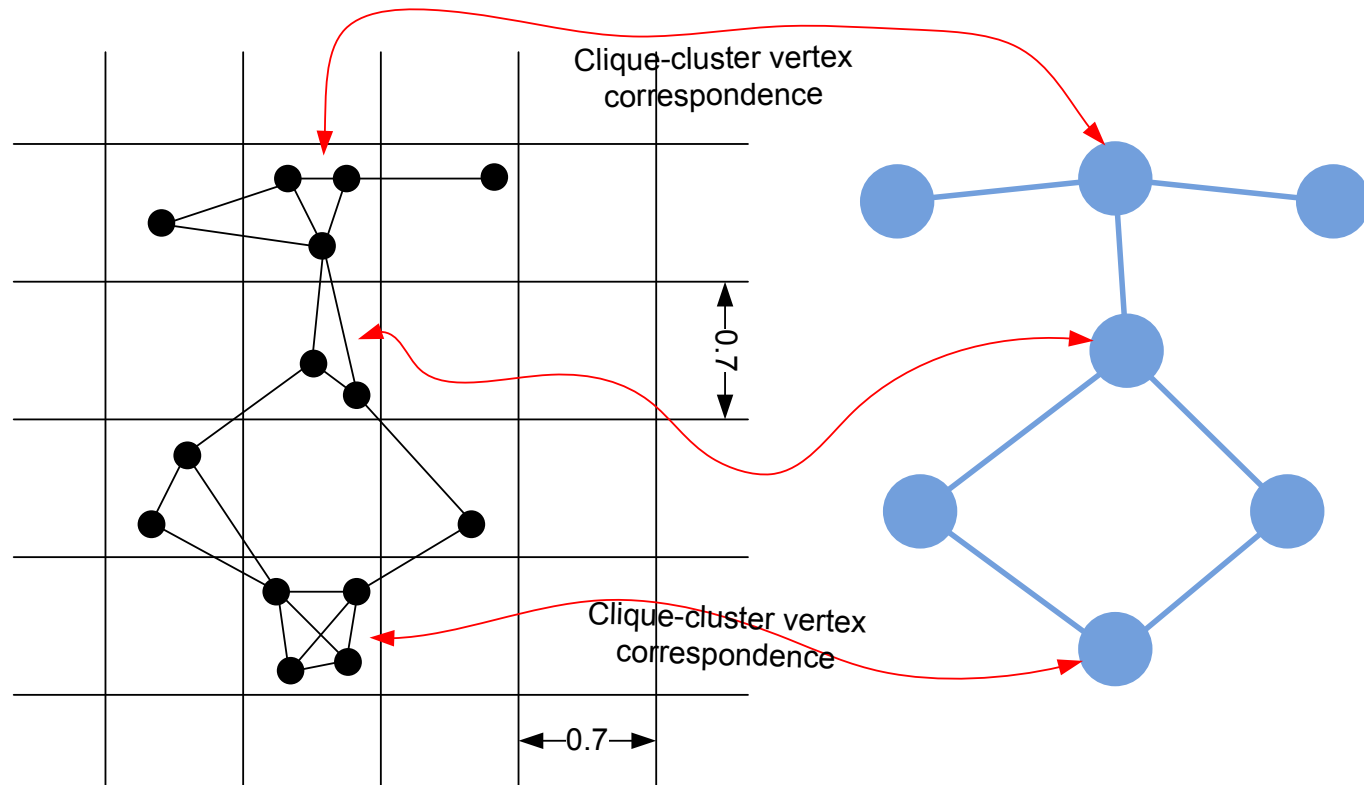
- For a clique partition of G , there are as many vertices in H as the size of the clique partition.
- Each vertex in H has a corresponding clique in the partition of G and vice-versa.
- There is an edge between two vertices in H iff there is an edge between the corresponding cliques in G .

Growth Restricted Property

Definition: A graph $G = (V, E)$ is called *growth restricted* if any “ball” of radius r centered at any vertex $v \in V$ contains at most $r^{O(1)}$ vertices.



Cluster Graph Construction:



The graph on the left is a UDG. Small grid induces a clique partition. The graph on the right is the associated cluster graph.

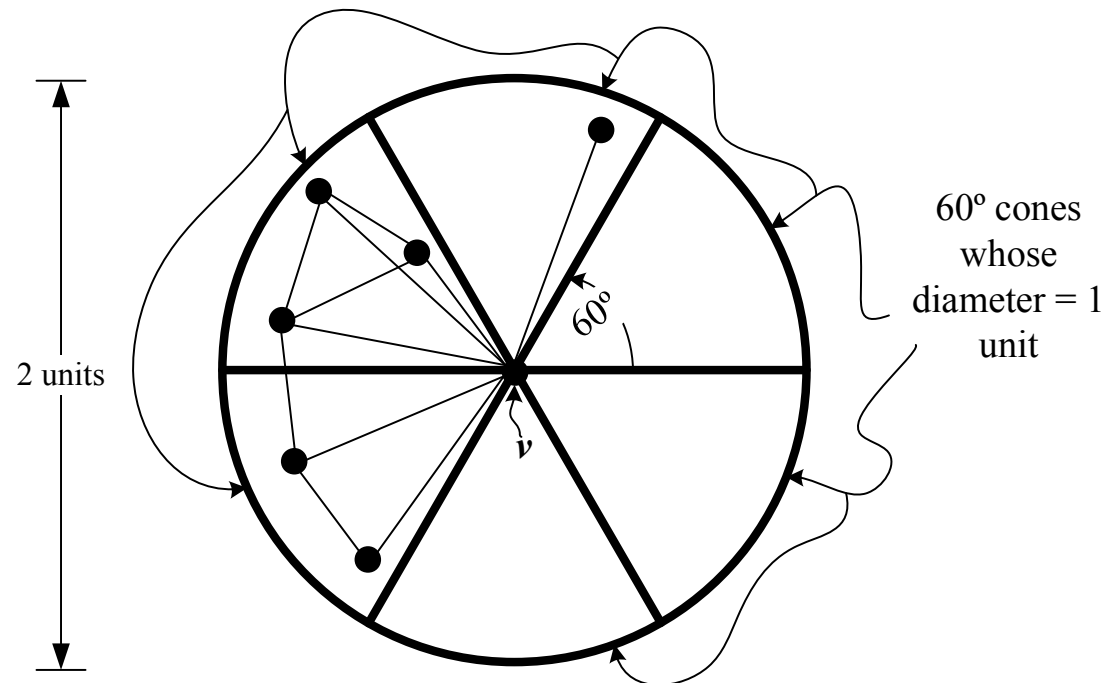
A Reasonable Idea:

1. For the UDG, G , compute an MIS I of G .
2. Every vertex attaches to a nearest $v \in I$. This partitions the vertex set into clusters $\{G_v\}$ of diameter 2.
3. Break up each cluster G_v into $O(1)$ cliques.

Steps 1 and 2 can be done very quickly even in a distributed setting. If Step 3 can be done easily also then we get a growth restricted cluster graph due to packing property of *independent sets* in UDGs.

$O(1)$ size Clique Partition of G_v

Note that G_v can be partitioned into at most 5 cliques so using the Steps 1-3 seem reasonable toward an $O(1)$ clique partition



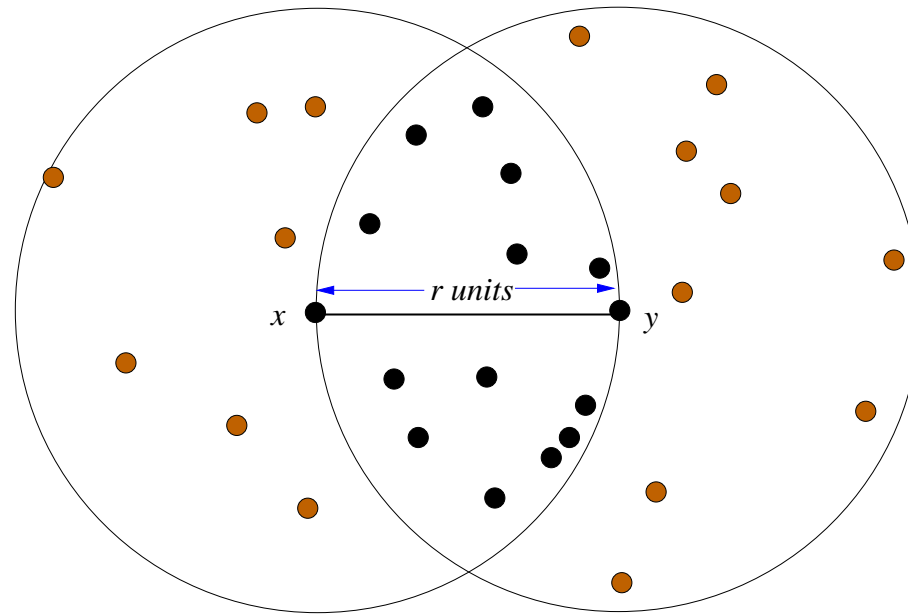
We would like to be able to guess these 5 cliques without geometry.

$\Omega(\log n)$ Bad Example for Greedy

- Raghavan-Spinrad [SODA 2001] present an algorithm that finds a maximum clique in a UDG without geometry.
- A reasonable “greedy” strategy is to use the R-S algorithm to find a maximum clique in G_v with the hope of constructing a small sized partition.
- Unfortunately, this does not produce an $O(1)$ sized clique partition as there is an easy example where using R-S as is repeatedly produces an $\Omega(\log n)$ sized clique partition.

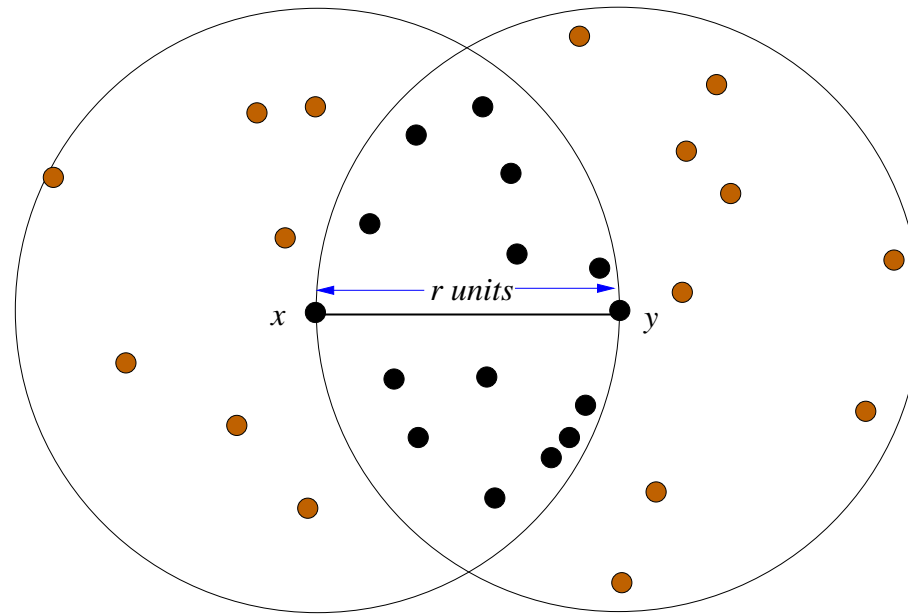
However, ideas developed by R-S come to our rescue in finding an $O(1)$ sized partition.

Raghavan-Spinrad for Max Clique



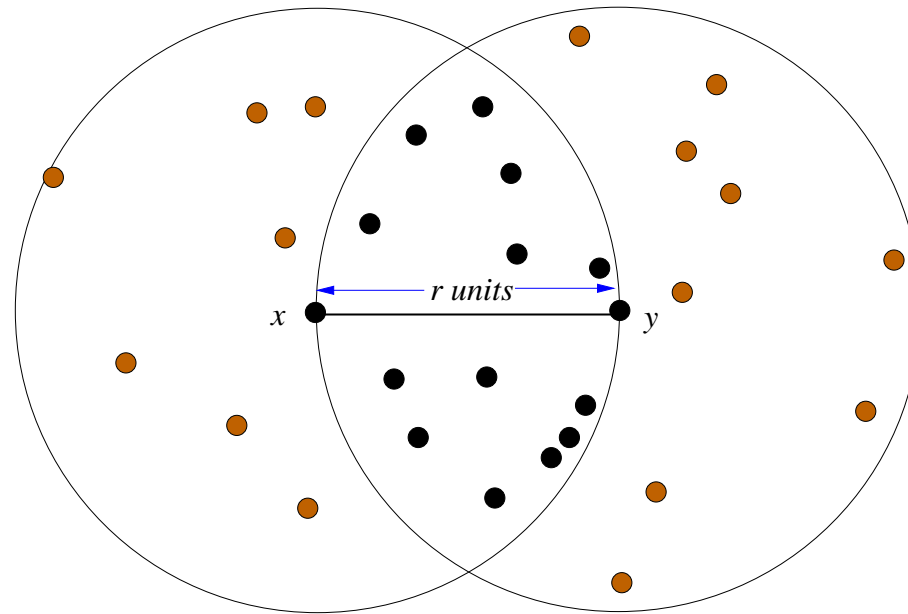
Observe that for any UDG, a maximum clique is contained completely inside the “lune” defined by the intersection of disks centered at x and y of radius $\|x - y\|_2$, where $\{x, y\}$ is a longest edge in the clique.

Raghavan-Spinrad for Max Clique



They note that vertices that fall in the top half of the “lune” must form a clique and those that fall in the bottom half of the “lune” must form a clique. In other words, vertices in “lune” form a *co-bipartite graph* (complement is a bi-partite graph).

Raghavan-Spinrad for Max Clique



So, to “catch” a maximum clique, one should ignore some edges and only consider an edge induced subgraph so that the common neighborhood of the edge $\{x, y\}$ in what remains is a co-bipartite graph in the original graph.

Raghavan-Spinrad for Max Clique

Raghavan-Spinrad [SODA 2001] devise an edge ordering $L = (e_1, e_2, \dots, e_m)$, $m = |E|$.

The ordering L has the property that the the common neighborhood of e_i , $N_L[i]$ in the edge induced subgraph of $(e_i, e_{i+1}, \dots, e_m)$ denoted $G_L[i]$, is *co-bipartite* (the compliment is a bi-partite graph) in G . They call this ordering a CNEEO (Co-bipartite Neighborhood Edge Elimination Ordering).

They show that all UDGs admit a CNEEO which can be constructed easily without geometry.

$O(1)$ clique Partition of G_v

Lemma: Let C be a clique in G_v and L a CNEEO of G_v . Then, there exists an i such that $C \in N_L[i]$.

Proof: Some edge of C appears first in the ordering $L = (e_1, e_2, \dots, e_m)$. Let us say that that edge is e_i . By definition, the common neighborhood of e_i , $N_L[i]$ induces a co-bipartite graph in G . Since e_i is the first edge of C in L , $N_L[i]$ contains C . Otherwise, that contradicts the fact that e_i is the first edge of C .

This lemma opens up the possibility of “catching” any clique that we want in G_v with at most 2 cliques.

How do we use this Lemma

Note that the neighborhood G_v of any vertex v in a UDG can be partitioned into at most 5 cliques. **So, we are looking for at most 5 cliques in G_v such that their removal makes G_v empty.** We would like to “catch” them!

Main Idea: Guess 5 edge sequences and check each to see if deleting the cliques that they are part of leaves G_v empty. For each guess and each edge in the guess, we have to consider the “right” subgraph that enables us to “catch” the right clique. This is where CNEEO helps.

The Algorithm

NBD-CLIQUE-PARTITION(G_v)

1. **for each** 5-edge sequence (f_1, f_2, \dots, f_5) of $E(G_v)$ **do**
2. $G_0 \leftarrow G_v$
3. **for** $j \leftarrow 1$ **to** 5 **do**
4. Compute a CNEEO L of G_{j-1}
5. $i \leftarrow$ rank of f_j in L
6. Partition $N_L[i]$ into two cliques C'_j and C''_j
7. $G_j \leftarrow G_{j-1} \setminus N_L[i]$
8. **if** $(G_5 = \emptyset)$ **return** $\{C'_j, C''_j \mid j = 1, 2, \dots, 5\}$

Using the Lemma, it is easy to see that the algorithm partitions G_v into at most 10 cliques.

$(\log n, O(\sqrt{\log n}))$ Vol. Resp. Embed.

1. Compute a *growth restricted cluster graph* $G[C]$ of the UDG where each node in the cluster graph corresponds to a clique in G .
2. Compute a $(\log n, O(\sqrt{\log n}))$ *volume respecting embedding* of $G[C]$.

We use a combination of ideas developed by Rao [SoCG 99] and Lee-Krauthgamer [STOC 03] to devise an embedding that has low *volume distortion*.

This then is used in the *random projection* method.

A little more standard work leads to final embedding in the plane.

Thank you!

Any questions?

Please email: pirwani@cs.uiowa.edu