

Good Quality Virtual Realization of Unit Disk Graphs ^{*}

Sriram V. Pemmaraju [†] Imran A. Pirwani [‡]

June 25, 2009

Abstract

The *quality* of an embedding $\Phi : V \mapsto \mathbb{R}^2$ of a graph $G = (V, E)$ into the Euclidean plane is the ratio of $\max_{\{u,v\} \in E} \|\Phi(u) - \Phi(v)\|_2$ to $\min_{\{u,v\} \notin E} \|\Phi(u) - \Phi(v)\|_2$. Given a graph $G = (V, E)$, that is known to be a *unit disk graph* (UDG), we seek algorithms to compute an embedding $\Phi : V \mapsto \mathbb{R}^2$ of best (smallest) quality. Note that G comes with no associated geometric information and in this setting, related problems such as recognizing if G is a UDG, are NP-hard. While any UDG that is not a clique has a 2-dimensional embedding with quality between $1/2$ and 1 , the adaptation of Vempala’s random projection approach (*FOCS 1998*) by Kuhn et al. (*Algorithmica*, to appear) provides the best quality bound of $O(\log^{3.5} n \cdot \sqrt{\log \log n})$.

This paper presents a simple, *combinatorial* algorithm for computing a $O(\log^{2.5} n)$ -quality 2-dimensional embedding of a given graph, that is known to be a UDG in the Euclidean plane. If the embedding is allowed to reside in higher dimensional space, we obtain improved results: a quality-2 embedding in $\mathbb{R}^{O(1)}$. Our key technical contribution is the construction of a “growth-restricted approximation” of the given UDG. While such a construction is trivial if the UDG comes with a geometric representation, we are not aware of any other algorithm that can perform this step without geometric information. Construction of a growth-restricted approximation permits us to bypass the standard and costly technique of solving a linear program with exponentially many “spreading constraints.” As a side effect of our construction, we get the first constant-factor approximation to the minimum clique partition problem for UDGs, given without a geometric representation.

Our problem is a version of the well known *localization* problem in wireless sensor networks, in which network nodes are required to compute virtual 2-dimensional Euclidean coordinates given little or (as in our case) no geometric information.

1 Introduction

A graph $G = (V, E)$ is a *unit disk graph* (UDG) if there is an embedding $\Phi : V \mapsto \mathbb{R}^2$ such that $\{u, v\} \in E$ iff $\|\Phi(u) - \Phi(v)\|_2 \leq 1$. Such an embedding Φ of G is called a *realization* of G . In this paper, we are interested in the problem of finding a realization Φ of a given UDG. It is unlikely that this problem has a polynomial-time algorithm because even the problem of recognizing if a given graph is a UDG is NP-hard [9]. Aspnes et al. [2] have shown that the problem of computing a realization of a given UDG is NP-hard even if all edge lengths between pairs of neighboring

^{*}A preliminary version appeared in the 15th Annual European Symposium on Algorithms (ESA), Eilat, Israel (2007), pp. 311-322.

[†]Department of Computer Science, The University of Iowa, Iowa City, IA 52242-1419, U.S.A. E-mail: sriram@cs.uiowa.edu.

[‡]Department of Computing Science, The University of Alberta, Edmonton, Alberta T6G 2E8, Canada. Supported by Alberta Ingenuity. E-mail: pirwani@cs.ualberta.ca.

vertices are known. The problem remains NP-hard when all angles between adjacent edges are known [10] and also when all angles plus slightly noisy, pairwise distances are known [3]. Given this situation, we consider the problem of computing an “approximate” realization of the given UDG. Let $G = (V, E)$ be a UDG. Let $\Phi : V \mapsto \mathbb{R}^2$ be an embedding of G into \mathbb{R}^2 . If G is not a clique, then the *quality* of the embedding Φ is defined as:

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

In case G is a clique, then the *quality* of Φ is simply $\max_{\{u,v\} \in E} \|\Phi(u) - \Phi(v)\|_2$. The specific optimization problem we consider is the following.

Given d , and a UDG $G = (V, E)$, find an embedding $\Phi : V \mapsto \mathbb{R}^d$ with best (smallest) quality.

We call this the *best quality embedding* problem. It is easy to see that every UDG has an embedding into \mathbb{R}^2 with best quality, between $1/2$ and 1 . This paper focuses on devising an approximation algorithm for the best quality embedding problem.

There exist other reasonable measures of the quality of UDG realization. For example, a measure of the quality of embedding considered in the context of *VLSI layout problem* [36, 13] is to obtain a realization that minimizes the *sum of edge lengths*. In the VLSI layout problem, the input is a graph G and the goal is to embed the vertices at distinct grid points in a d -dimensional grid so as to optimize some objective function over the set of edges under the embedding; one objective is to minimize the sum of (Euclidean) edge lengths obtained over all feasible embeddings. More precisely, compute Φ so as to minimize the following:

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

The problem of finding an embedding with the aim of minimizing the *sum of edge lengths* seems to be open. In this paper, we do not consider optimizing this objective.

Motivated by the *localization* problem in wireless sensor networks, Moscibroda et al. define quality as a measure of the “goodness” of a realization [27]. This paper [27] claims an $O(\log^{2.5} n \cdot \sqrt{\log \log n})$ -quality embedding for UDGs into \mathbb{R}^2 , but a subsequent full version of the paper [21] corrects this bound to $O(\log^{3.5} n \cdot \sqrt{\log \log n})^1$. The techniques employed by [27, 21] are an adaptation of the *random projection method* devised by Vempala [36] who gave a first poly-logarithmic approximation to the VLSI layout problem. The only hardness of approximation result for the best quality embedding problem, that we are aware of, is this [22]: there is no polynomial time algorithm (unless $P = NP$) that can compute a $(\sqrt{3/2} - o(1))$ -quality, 2-dimensional embedding of a given UDG in which non-adjacent vertices are required to be more than one unit away from each other.

As mentioned earlier, the best quality embedding problem for UDGs is motivated by the *localization problem* in wireless sensor networks [11, 29]. Low dimensional UDGs are typically used to

¹Kuhn et al. [21] use a weaker volume respecting embedding bound of $O(\log n \sqrt{\log \log n})$ due to Feige [14]. However, Krauthgamer et al. [18] give an optimal bound of $O(\log n)$ on the volume distortion of general metrics. Using the bound of Krauthgamer et al. [18], Kuhn et al.’s [21] bound on the quality of the embedding of a UDG improves to $O(\log^{3.5} n)$.

model wireless sensor networks. The localization problem requires each node in a wireless sensor network to compute its own coordinates with respect to some global coordinate system. For most sensing applications, it is critical that each node know its location, at least approximately. Knowledge of location information can also dramatically improve the performance of routing algorithms because it allows the use of *geometric routing* techniques [8, 16, 24]. One technological solution to the localization problem is to equip each node with a GPS receiver. However, this solution seems too costly, currently and in any case such a solution will have to deal with GPS errors and the fact that GPS service may be unavailable indoors. Another solution is to equip a few “anchor” nodes with GPS receivers [5] and have the rest of the nodes compute their coordinates by using the known coordinates of the anchor nodes. The main drawback of this approach is that for a good quality solution, the number of anchor nodes needed may be fairly high and furthermore they may have to be placed manually. Recent work has suggested that for geometric routing schemes, having “real” coordinates is not necessary; having *virtual coordinates* suffices to ensure prompt and guaranteed routing [30, 34]. In fact, virtual coordinates that are derived only from connectivity information are preferable as this information is robust to errors in radio signals. Motivated by the localization problem, we assume that our input graph is a UDG. While the best quality embedding problem allows the host space to have arbitrary dimensions (specified as d in the input), we are particularly interested in the case when $d = 2$. This is partly because two is the “intrinsic” dimension of a UDG and partly because such results are most relevant to the wireless sensor networks application when $d = 2$. Well-known geometric routing algorithms [8, 16, 24] only provide guaranteed delivery for networks in the plane or ones that lie in a thin slab in \mathbb{R}^3 [12].

1.1 Results and Techniques

In this paper, we present a combinatorial algorithm for computing an $O(\log^{2.5} n)$ -quality embedding of any UDG into \mathbb{R}^2 . Our result can be seen as improving Kuhn et al.’s bound [21]. However, the most important aspect of our algorithm is that it is combinatorial and seems amenable to local, distributed implementation. Our algorithm avoids the costly first step of Kuhn et al.’s algorithm in which an exponentially large linear program (LP), that imposes *spreading constraints* on the vertices, is solved via the *ellipsoid method*. Starting with an LP or a semi-definite program that imposes spreading constraints is a fairly common approach to solving vertex-ordering problems (such as the *minimum bandwidth* problem) [7, 13], and the VLSI layout problem [36]. We avoid the spreading constraints approach via a combinatorial algorithm for computing a “growth-restricted approximation” of the given UDG. This step may be of independent interest. Partitioning a network into clusters so that each cluster has edges to at most a constant number of other clusters, is a standard first step in many topology control and routing algorithms on wireless networks (see [24, 23] for representative examples). Thus far such clustering has been accomplished by the use of Euclidean coordinates or pairwise distances that are assumed to be available at nodes. Our algorithm can perform such a clustering without access to geometric information and can also be implemented efficiently in a distributed setting (in $O(\log^* n)$ rounds under the *LOCAL* model of computation [31]) as well. Our algorithm has three main steps which we outline in the next three subsections.

1.1.1 Constructing a Growth-Restricted Approximation

In the first step, we partition the given UDG into cliques such that when the cliques are contracted into vertices, we get a “growth-restricted approximation” of the given UDG. To describe this step

more precisely, we need some definitions. For any graph $G = (V, E)$ and for any pair of vertices $u, v \in V$, let $d_G(u, v)$ denote the shortest path distance between u and v . Let $B_G(v, r) = \{u \in V \mid d_G(u, v) \leq r\}$ denote the closed ball of radius r centered at v in G . Where G is clear from the context, we write $B(v, r)$ instead of $B_G(v, r)$. Define the *growth rate* of G to be

$$\rho_G = \inf\{\rho : |B_G(v, r)| \leq r^\rho \text{ for all } v \in V \text{ and all integers } r > 1\}.$$

A class \mathcal{G} of graphs is *growth-restricted* if there is some constant c such that for every graph G in \mathcal{G} , $\rho_G \leq c$. For any partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ of the vertex set V of G , the *cluster graph of G induced by \mathcal{C}* , denoted $G[\mathcal{C}]$, is obtained from G by contracting each C_i into a vertex. In Section 2, we show how to partition a given UDG $G = (V, E)$ into cliques $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ such that the cluster graph $G[\mathcal{C}]$ induced by the clique partition has constant growth rate. Note that if we can construct an α -quality embedding Φ of $G[\mathcal{C}]$ into \mathbb{R}^L , we can immediately get an α -quality embedding Φ' of G into \mathbb{R}^L : for each vertex v of G set $\Phi'(v) := \Phi(C_i)$, where C_i is the clique that contains v . This allows us to focus on the problem of obtaining a good quality embedding of growth-restricted graphs.

It is quite easy to obtain a clique-partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ with the desired properties if a realization of G is given. For example, given a UDG with 2-dimensional coordinates of vertices known, one can place an infinite grid of $\frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}}$ square cells on the plane and obtain a vertex partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ in which each part C_i is all vertices in a cell. Ties caused by points falling on cell boundaries can be broken arbitrarily. Due to the size of the cells each C_i is a clique and furthermore a simple geometric argument shows that there are $O(r^2)$ cells in the radius- $2r$ disk centered at the center of any cell. This suffices to show that in $H := G[\mathcal{C}]$, $|B_H(v, r)| = O(r^2)$, implying that ρ_H is bounded by a constant. See Figure 1(a) for an illustration.

In the absence of geometric information, it is not immediately clear how to obtain the desired clique partition. One possible approach is to start with a maximal independent set I of G and attach each $v \in V \setminus I$ to an arbitrary neighbor in I . For each $v \in I$, let S_v denote the set consisting of v along with neighbors which have attached to v . Let $\mathcal{S} = \{S_v \mid v \in I\}$ be the induced vertex partition of V . Since I is an independent set, in any realization Φ of G into \mathbb{R}^2 , $\|\Phi(u) - \Phi(v)\|_2 > 1$ for all $u, v \in I, u \neq v$. From this observation, one can deduce the fact that $H := G[\mathcal{S}]$ has bounded growth rate. However, the sets S_v are not cliques, even though the subgraphs they induce $H_v := G[S_v]$ have diameter at most 2. Furthermore, we know that each H_v is a graph induced by a neighborhood of a UDG, there exists a partition of H_v into at most 5 cliques. But, in the absence of geometry, how can we find a constant-sized partition of H_v ? In this context, the work of Raghavan and Spinrad [33] is relevant. They [33] present an algorithm that computes a maximum cardinality clique of an input UDG, given without any geometric information; one can use their algorithm as a subroutine in the following greedy approach: repeatedly find and remove a maximum size clique from H_v , until it becomes empty. Since we know that H_v can be partitioned into a constant c number of cliques, H_v contains a clique of size at least $|S_v|/c$ and therefore each step removes a $1/c$ fraction of vertices (or more) from H_v . This immediately implies that the greedy approach produces $O(\log n)$ cliques, where $n = |S_v|$. Unfortunately, this bound is tight and there is a simple example (see Figure 1(b)) showing that the greedy approach can lead to a clique-partition of S_v of size $\Omega(\log n)$.

To further motivate the problem, note that the problem of partitioning H_v into a constant number of cliques is equivalent to the problem of coloring $\overline{H_v}$ (the complement of H_v) with a constant number of colors. Computing a constant-sized coloring of a graph, even when it is known to have a coloring of constant size, seems quite hard. For example, the best approximation algorithm

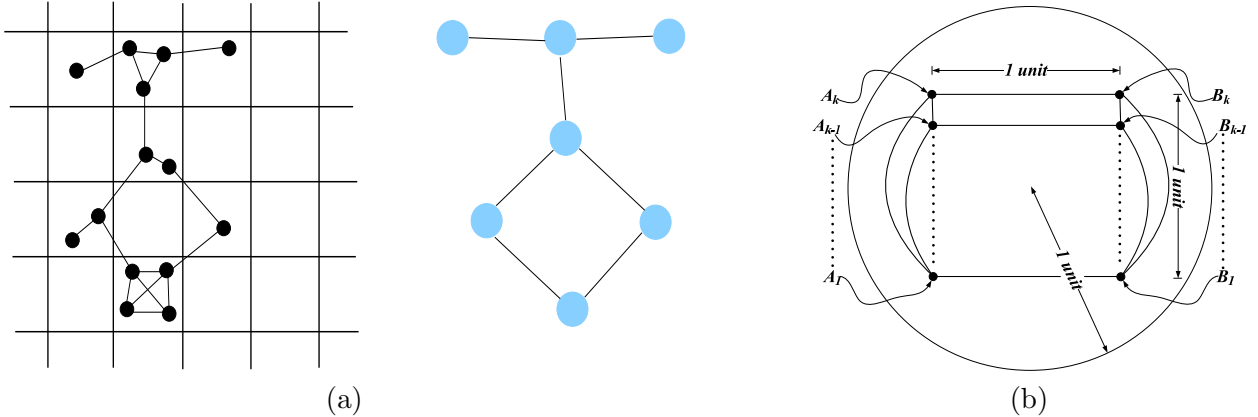


Figure 1: (a) A realization of a UDG G , partitioned by a grid of $\frac{1}{\sqrt{2}} \times \frac{1}{\sqrt{2}}$ square cells. The cluster graph $G[\mathcal{C}]$ induced by the partition \mathcal{C} is also shown. (b) A bad example for the greedy approach. Each A_i and each B_i is a set of 2^i points. A point in A_i is adjacent to all points in A_j , $1 \leq j \leq k$ and exactly the points in B_i . The adjacencies for points in B_i are symmetric. Thus the largest clique is $A_k \cup B_k$ and has size $2 \cdot 2^k$. Removing this clique leaves sets A_j and B_j , $1 \leq j \leq k - 1$ intact. Thus the algorithm uses k cliques to cover about 2^{k+2} points, yielding a lower bound of $\Omega(\log n)$ on the size of the clique cover produced by the greedy algorithm.

for coloring 3-colorable graphs uses $\tilde{O}(n^{3/14})$ colors [6]².

In Section 2 we present an algorithm for partitioning each H_v into a constant number of cliques; this involves extending ideas developed by Raghavan and Spinrad [33] in the context of finding a maximum clique in a UDG with no given realization. Since our problem arises in sensor network localization, it is worth pointing out that the overall construction of the growth-restricted cluster graph induced by cliques has a very simple distributed implementation. The first step of the construction is to compute a maximal independent set (MIS) of G , and a recent paper by Schneider and Wattenhofer [35] is relevant here. They [35] show how to compute a MIS on a class of graphs, *bounded growth graphs*, that contains UDGs in $O(\log^* n)$ rounds using only connectivity information. Using their distributed algorithm [35], one can compute a MIS of G without any geometry. Following this step, the remaining vertices join a neighboring independent vertex in $O(1)$ rounds, forming neighborhood partition of the vertex set. Next, the partition of neighborhoods, H_v , into constant number of cliques can be done again in $O(1)$ rounds since the diameter of each neighborhood is at most 2.

1.1.2 Constructing Volume Respecting Embeddings

The remaining two steps of our algorithm follow the approach introduced by Vempala [36] with some important differences due to the fact that our input graph is growth-restricted. Let $H := G[\mathcal{C}]$ be the cluster graph of G constructed in the previous step.

In the second step of our algorithm, we construct a *volume respecting embedding* of the shortest path metric of H . The notion of volume respecting embeddings was introduced by Feige [14] in the context of the minimum bandwidth problem. Let (X, d) be a metric space. An embedding $\Phi : X \mapsto \mathbb{R}^L$ is a *contraction* if $\|\Phi(u) - \Phi(v)\|_2 \leq d(u, v)$ for all $u, v \in X$. For a set T of k points

² Here the \tilde{O} notation hides factors that are logarithmic in n .

in \mathbb{R}^L , define $\text{Evol}(T)$ to be the $(k-1)$ -dimensional volume of the $(k-1)$ -dimensional simplex spanned by T , computed using the ℓ_2 norm. Note that $\text{Evol}(T) = 0$ if T is affinely dependent. For any finite metric space (X, d) , define $\text{Vol}(X)$ as $\sup_{\Phi} \text{Evol}(\Phi(X))$, where the supremum is over all contractions $\Phi : X \rightarrow \mathbb{R}^{|X|-1}$. Given an arbitrary metric space (X, d) , a contraction $\Phi : X \rightarrow \mathbb{R}^L$ is called (k, D) -volume respecting embedding if for every size- k subset $S \subseteq X$, we have:

$$\text{Evol}(\Phi(S)) \geq \left(\frac{\text{Vol}(S)}{D^{k-1}} \right)$$

Note that when $k = 2$, this condition reduces to $\|\Phi(u) - \Phi(v)\|_2 \geq d(u, v)/D$ for all $u, v \in X$. Thus, a volume respecting embedding is a generalization of the more commonly used notion of small distortion embeddings [26], in which only pairs of points are considered. A volume respecting embedding will be very useful for the next step our algorithm, in which a random projection of the vertices of H into \mathbb{R}^2 , will be performed. For the random projection step to spread points fairly well in \mathbb{R}^2 , we require sets of points to have large volumes, because intuitively, point sets with large volume will be spread out in their projection into a lower dimensional sub-space. For arbitrary metric spaces, Feige [14] presents a polynomial time algorithm to compute a $(k, O(\sqrt{\log n} \cdot \sqrt{k \log k + \log n}))$ -volume respecting embedding. We are interested in $k = \log n$ and therefore, Feige's algorithm yields a $(\log n, O(\log n \cdot \sqrt{\log \log n}))$ -volume respecting embedding. Krauthgamer et al. [18] give a novel embedding technique which they call *measured descent embedding*. This embedding improves on Feige's bound to obtain a $(k, O(\log n))$ -volume respecting embeddings for $1 \leq k \leq n$ for general metrics [18].

However, it is possible to improve on Krauthgamer et al. [18] bounds in our context where the graph has constant growth rate. Specifically, using a technique due to Krauthgamer and Lee [17] for growth-bounded graphs, we obtain an embedding of H into $O(1)$ -dimensional Euclidean space such that any edge has length at most 2 and every pair of non-neighbor vertices are at least one unit apart. This yields a metric whose *doubling dimension* is $O(1)$ (also known as a *doubling metric*). We then use the measured descent embedding of Krauthgamer et al. [18] to obtain a $(k, O(\sqrt{\log n}))$ -volume respecting embedding of H .

Now we mention a useful lemma due to Feige [14] that establishes a lower bound on $\text{Vol}(S)$. To show that an embedding $\Phi : X \mapsto \mathbb{R}^L$ is a (k, D) -volume respecting embedding of X , we need to show that $\text{Evol}(\Phi(S)) \geq \text{Vol}(S)/D^{k-1}$. $\text{Vol}(S)$ is difficult to compare against and so Feige defined the notion of a *tree volume* and showed a lower bound on $\text{Vol}(S)$ in terms of the tree volume of S . For any $S \subseteq X$, the *tree volume* of S , denoted $\text{Tvol}(S)$ the product of the edge lengths in a minimum spanning tree of S .

Lemma 1 (Feige [14]) *Let (X, d) be a metric space. For any size- k subset $S \subseteq X$, $\text{Tvol}(S) \leq 2^k(k-1)! \text{Vol}(S)$.*

1.1.3 Random Projections and Rounding

The third and the last step of our algorithm uses a slightly modified version of Vempala's random projection and rounding technique. A similar technique is used by Moscibroda et al. [27]. After computing a volume respecting embedding of H , we project the vertices onto a plane defined by two unit vectors chosen uniformly and independently at random. We state two lemmas due to Vempala [36] that show how the random projection step affects individual points and subsets of points, respectively. These lemmas are used in estimating the number of vertices of H that fall in a region of the plane.

Lemma 2 (Vempala [36]) Let $v \in \mathbb{R}^d$. For a random unit vector l , $c > 1$,

$$\Pr \left[|v \cdot l| \geq \frac{c}{\sqrt{d}} |v| \right] < \frac{1}{e^{c^2/4}}.$$

Lemma 3 (Vempala [36]) Let S be a set of vectors $v_1, v_2, \dots, v_k \in \mathbb{R}^d$. For a random unit vector l ,

$$\Pr[\max_i \{v_i \cdot l\} - \min_i \{v_i \cdot l\} \leq W] = O \left(\frac{W^{k-1} d^{\frac{k-1}{2}}}{(k-1)! \text{Evol}(S)} \right)$$

Assuming that H has a (k, D) -volume respecting embedding for sufficiently small D , we can use Lemma 3 to obtain an upper bound on the number of vertices of H that gets projected onto a small region in the plane. To get sufficient separation among pairs of non-neighboring vertices, we partition the plane using a fine enough grid so that every vertex of H can be mapped to a distinct grid point. Vempala [36] calls this the “rounding” step. Scaling by a factor proportional to the maximum number of vertices in the small region ensures that the minimum distance between any pair of vertices is at least 1. The fact that not too many vertices were projected into the region, in the first place, ensures that the scaling factor is not too large and allows us to show that edges do not get stretched too much. Finally, all vertices in the clique C_v in G that correspond to a vertex v in H are assigned the point to which v is mapped. Note that mapping all vertices in a clique in G to the same point does not affect the quality of embedding.

2 Constructing a Growth-Restricted Approximation

In this section, we show how to construct a clique partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ of a given UDG G so that $G[\mathcal{C}]$ has constant growth rate. Recall that $G[\mathcal{C}]$ is the graph obtained from G by contracting each C_i into a vertex. As mentioned in the introduction, our starting point is the following algorithm.

CLIQUE-PARTITION(G)

1. Compute a maximal independent set (MIS) I of G .
2. Associate each vertex $u \in V \setminus I$ to a neighbor in I . For each $v \in I$, let S_v consist of v and its associated vertices.
3. Partition each vertex subset S_v into a constant number of cliques.

The first two steps are simple and for the third step we makes use of ideas due to Raghavan and Spinrad [33]. Raghavan and Spinrad [33] present a “robust” algorithm for the problem of finding a maximum cardinality clique (henceforth, *maximum clique*) in a given UDG. Their algorithm is robust in the sense that it takes as input an arbitrary graph G and in polynomial time, (i) either returns a maximum clique in G or (ii) produces a certificate indicating that G is not a UDG. The existence of such an algorithm is surprising because both problems (a) recognizing whether a given graph is a UDG and (b) finding a maximum cardinality clique in a given graph, are NP-hard. The key idea underlying the Raghavan-Spinrad algorithm is the existence of a superclass \mathcal{G} of the class of UDGs such that in polynomial time one can determine if a given graph G is in \mathcal{G} or not. Furthermore, for any G in \mathcal{G} a maximum clique can be computed in polynomial time.

The superclass \mathcal{G} is the set of all graphs that admit a *cobipartite neighborhood edge elimination ordering* (CNEEO). Given a graph $G = (V, E)$ and an edge ordering $L = (e_1, e_2, \dots, e_m)$ of E , let $G_L[i]$ denote the spanning subgraph of G with edge set $\{e_i, e_{i+1}, \dots, e_m\}$. For each edge $e_i = \{x, y\}$ define $N_L[i]$ to be the set of common neighbors of x and y in $G_L[i]$. An edge ordering $L = (e_1, e_2, \dots, e_m)$ of $G = (V, E)$ is a CNEEO if for every edge e_i , $N_L[i]$ induces a cobipartite (i.e., the complement of a bipartite) graph. Raghavan and Spinrad prove three results: (1) If a graph G admits a CNEEO, then there is a simple greedy algorithm for finding a CNEEO of G . (2) Given a graph G and a CNEEO of G , a maximum clique in G can be found in polynomial time. (3) Every UDG admits a CNEEO. To see the second result, consider a maximum clique C in G . Let $L = (e_1, e_2, \dots, e_m)$ be a CNEEO of G and let e_i be the edge in $G[C]$ that occurs first in L . Then C is contained in the cobipartite graph $N_L[i]$. In fact, C is a maximum clique in $N_L[i]$. Using the fact that $N_L[i]$ is cobipartite and the fact that a maximum independent set in a bipartite graph can be computed in polynomial time, we can compute a clique of cardinality $|C|$ in $N_L[i]$ in polynomial time. Raghavan and Spinrad obtain the third result by showing that if we take a geometric representation of the given UDG G and order edges in non-increasing length order, we get a CNEEO of G . This follows fairly easily from the following geometric observation. Let $\{x, y\}$ be a line segment in the plane and let $r = \|x - y\|_2$. Then $\{x, y\}$ partitions the intersection of $Disk(x, r) \cap Disk(y, r)$ into two regions of diameter at most r . See Figure 2(a) for an illustration.

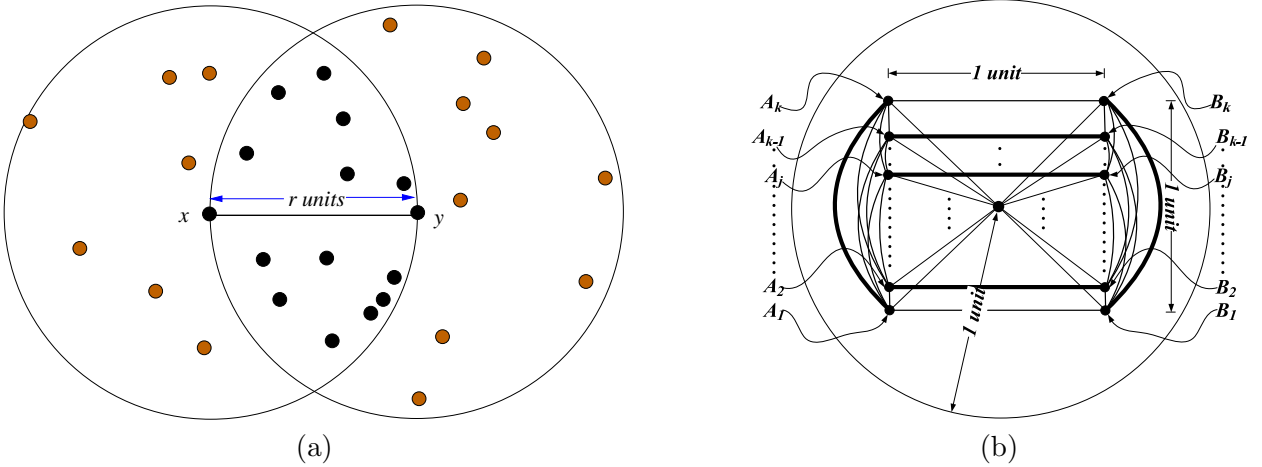


Figure 2: (a) Suppose that edge $\{x, y\}$ has rank i in an ordering of the edges of G in non-increasing length order. The points in the common neighborhood of x and y in $G_L[i]$ are exactly those in the lune shown above. A point outside the lune may be a common neighbor of x and y in G , but not in $G_L[i]$. The diameter of the upper and lower halves of the lune are r and therefore the vertices in each half induce a clique in $G_L[i]$. Hence, $N_L[i]$ induces a cobipartite graph. (b) A sample run of NBD-CLIQUE-PARTITION on the “bad example” for the greedy algorithm. The 5 thick lines correspond (in some order) to the 5-edge sequence guessed by NBD-CLIQUE-PARTITION in Step (1). Among the guessed edges, the edge between A_1 and A_k , say e_A , and the edge between B_1 and B_k , say e_B , are critical because the common neighborhood of the endpoints of e_A is the clique $\bigcup_i A_i$ and similarly, the common neighborhood of the endpoints of e_B is the clique $\bigcup_i B_i$. Note that in the guess as highlighted by the thick lines, the algorithm will produce a clique partition with at most 10 cliques independent of the order in which the 5 edges are processed.

The three results mentioned above lead to a polynomial-time algorithm that will successfully

report a maximum clique for every input UDG and for some graphs that are not UDGs (but admit a CNEEO). We now show how to use the CNEEO idea to implement Step (3) of the CLIQUE-PARTITION algorithm. Let $G_v := G[S_v]$ be the subgraph of G induced by S_v ; we call this H_v in the introduction. Since G_v is also a UDG, it admits a CNEEO. We start with a key lemma whose proof is straight-forward.

Lemma 4 *Let C be a clique in G_v and let $L = (e_1, e_2, \dots, e_m)$ be a CNEEO of G_v . There is an i , $1 \leq i \leq m$, such that $N_L[i]$ contains C .*

Proof: Let $e_i = \{x, y\}$ be the edge in $G_v[C]$ that occurs first in L . Recall the notation $N_L[i]$: this denotes the common neighborhood of the endpoints of edge e_i in the spanning subgraph of G_v containing only edges e_i, e_{i+1}, \dots, e_m . Since e_i is the first edge in C , $N_L[i]$, contains C . \square

Recall that the closed neighborhood of a vertex in a UDG can be partitioned into at most 5 cliques. Let C_1, C_2, \dots, C_5 be a clique partition of S_v . The implication of the above lemma is that even though we do not know the clique partition C_1, C_2, \dots, C_5 , we do know that for every CNEEO $L = (e_1, e_2, \dots, e_m)$ of G_v , there is an edge e_i such that $N_L[i]$ can be partitioned into two cliques that cover C_1 . This follows simply from the fact that L is a CNEEO and therefore the graph induced by $N_L[i]$ is cobipartite. This suggests an algorithm that starts by guessing an edge sequence (f_1, f_2, \dots, f_5) of G_v . Then the algorithm computes L , a CNEEO of G_v . The algorithm's first guess is "good" if f_1 is the edge in C_1 that occurs first in L . Suppose this is the case and further suppose that f_1 has rank i in L . Then C_1 is contained in $N_L[i]$. Therefore, when $N_L[i]$ is deleted from G_v we have a graph, say G'_v , that can be partitioned into 4 cliques, namely $C_j \setminus N_L[i]$ for $j = 2, 3, 4, 5$. For each j , let C'_j denote $C_j \setminus N_L[i]$ and let L' be a CNEEO of G'_v . The algorithm's second guess, f_2 , is "good" if f_2 is the edge in C'_2 that occurs first in L' . Letting i' denote the rank of f_2 in L' , we see that $N_{L'}[i']$ contains C'_2 . We then delete $N_{L'}[i']$ from G'_v to get a graph that can be partitioned into 3 cliques. Continuing in this manner we get a partition of G_v into 10 cliques. Below, the algorithm is described more formally.

```

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  $\{C'_j, C''_j \mid j = 1, 2, \dots, 5\}$ 

```

The correctness of NBD-CLIQUE-PARTITION follows from the fact that there is some edge sequence (f_1, f_2, \dots, f_5) of $E(G_v)$ for which G_5 is empty. Figure 2(b) shows a sample run of the algorithm. The following lemma states this claim.

Lemma 5 *Algorithm NBD-CLIQUE-PARTITION partitions G_v into at most 10 cliques.*

Proof: Since G_v is a UDG, it can be partitioned into at most 5 cliques. Let these cliques be C_1, C_2, C_3, C_4, C_5 . Lemma 4 states that given C_1 and a CNEEO, we can efficiently find a

subset of vertices that covers C_1 . Furthermore, the complement of the subgraph induced by this subset of vertices is bipartite and it is easy to efficiently partition such a subset into at most two cliques. However, we do not know C_1 . Instead, we guess an edge f_1 of C_1 such that the common neighborhood of the endpoints in the remaining edge induced subgraph contains C_1 and is co-bipartite in G_v . Guessing f_1 correctly and then removing such a common neighborhood also removes C_1 . This common neighborhood can be covered by at most 2 cliques. It remains to find the remaining 4 subsets each of which cover the remaining 4 cliques. Therefore, there exists a 5 edge sequence that is a correct such sequence. **NBD-CLIQUE-PARTITION** considers all such sequences so eventually, the condition at line “8.” of the algorithm is true and we end up with at most 10 clique partition of G_v . \square

The above algorithm can be optimized a bit to yield an 8-clique partition; when cliques C_1 , C_2 , and C_3 have been deleted (at which time we have output 6 cliques), we are left with a cobipartite graph that can be easily partitioned into 2 cliques. If G_v has m edges then the number of guesses verified by the algorithm is $O(m^5)$. Since a CNEEO of a graph can be computed polynomial time (if it exists) and since a cobipartite can be partitioned into two cliques in linear time, we see that the algorithm **NBD-CLIQUE-PARTITION** runs in polynomial time. Thus Step (3) of the **CLIQUE-PARTITION** algorithm can be implemented by calling the **NBD-CLIQUE-PARTITION** algorithm for each vertex $v \in I$.

Let $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ be the clique partition of G produced by the algorithm **CLIQUE-PARTITION**. Let $H := G[\mathcal{C}]$. We now prove that H is growth-restricted. For each vertex $c \in V(H)$, there is a corresponding vertex v in the MIS I of G . Specifically, c corresponds to a clique in G that was obtained by partitioning S_v for some vertex $v \in I$. Recall that S_v consists of v along with some neighbors of v . Denote by $i(c)$ the vertex in I corresponding to $c \in V(H)$. Consider an arbitrary 2-dimensional realization Φ of G and for any pair of vertices $x, y \in V$, let $|xy|$ denote $\|\Phi(x) - \Phi(y)\|_2$. Let $B_H(v, r) = \{u \in V(H) \mid d_H(v, u) \leq r\}$.

Lemma 6 *For any $u, v \in V(H)$ and $r \geq 0$, if $u \in B(v, r)$ then $|i(u)i(v)| \leq 3r$.*

Proof: Consider two neighbors in H , x and y . Let C_x and C_y denote the cliques in G that were contracted into x and y respectively. Since x and y are neighbors in H , there are vertices $x' \in C_x$ and $y' \in C_y$ that are neighbors in G . Also, because of the way the cliques are constructed, x' is $i(x)$ or a neighbor of $i(x)$ in G . Similarly y' is $i(y)$ or a neighbor of $i(y)$ in G . Since G is a UDG, by triangle inequality $|i(x)i(y)| \leq |i(x)x'| + |x'y'| + |y'i(y)| \leq 3$.

If $u \in B(v, r)$, then there is a uv -path P in H of length at most r . Corresponding to P there is a sequence of vertices in I starting with $i(u)$ and ending with $i(v)$ such that consecutive vertices in this sequence are at most 3 units apart in any realization. Therefore, by triangle inequality $|i(u)i(v)| \leq 3r$. \square

Lemma 7 *There is a constant α such that for any $v \in V(H)$, $|B(v, r)| \leq \alpha \cdot r^2$.*

Proof: Let X be the number of vertices in $B(v, r)$. By the previous lemma, for each $u \in B(v, r)$, there is a vertex $i(u) \in I$ such that $i(u) \in \text{Disk}(i(v), 3r)$. Here $\text{Disk}(i(v), 3r)$ denotes the disk of radius $3r$ centered at vertex $i(v)$ in some realization of G . Also, by Lemma 5, there are at most 10 vertices in H that have the same corresponding vertex in I . Therefore, the number of vertices in $\text{Disk}(i(v), 3r)$ needs to be at least $X/10$. Any pair of vertices in I are more than one unit apart (in Euclidean distance) from each other. By the standard packing argument, this implies that the ball $\text{Disk}(i(v), 3r)$ can contain at most $4 \cdot (3r + 1/2)^2$ points in I . Therefore, $X/10 \leq 4(3r + 1/2)^2$ and hence for some constant α , we have $X \leq \alpha \cdot r^2$. \square

Lemma 7 leads to the following theorem.

Theorem 1 *There is a polynomial time algorithm that takes as input the combinatorial representation of a UDG $G = (V, E)$ and constructs a clique partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ of G such that $G[\mathcal{C}]$ has constant growth rate.*

A side effect of our construction is that the constructed clique partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ is a constant-factor approximation to the *minimum clique partition* problem for a UDG, given without geometric representation. The minimum clique partition problem is formulated thus: Given a graph $G = (V, E)$, partition V into $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ such that each C_i is a clique and t is minimized. Our claim follows from the fact that the size of any independent set is a lower bound on the size of a minimum clique partition and our solution produces a clique partition whose size is at most 8 times the size of a maximal independent set. This is the first constant factor approximation to the minimum clique partition problem for UDGs expressed in general form, i.e., without a geometric representation. These observations lead to the following theorem.

Theorem 2 *There is an 8-approximation to the minimum clique partition problem on UDGs given in general form. Furthermore, this algorithm can be implemented in $O(\log^* n)$ rounds of distributed computation in the \mathcal{LOCAL} model.*

Recently, Pirwani and Salavatipour [32] have claimed a $(2 + \varepsilon)$ -approximation for the vertex weighted version of the minimum clique partition problem on UDGs expressed in general form for any $\varepsilon > 0$. While they [32] give a centralized algorithm for the weighted case, their algorithm has an easy $O(\frac{\log^* n}{\varepsilon^{O(1)}})$ round distributed implementation in the \mathcal{LOCAL} model [31] for the unweighted case.

3 Volume Respecting Embedding of Growth-Restricted Graphs

In this section, we show that by combining techniques due to Krauthgamer and Lee [17] with the volume respecting embedding of Krauthgamer et al. [18], we can derive a combinatorial algorithm for constructing a $(k, O(\sqrt{\log n}))$ -volume respecting embedding of any n -vertex growth-restricted graph and any $k \geq 2$. We emphasize the combinatorial nature of our algorithm because in previous approaches [36, 27, 21] this step involved solving an LP with exponentially many constraints via the ellipsoid method. We propose the following two steps:

1. Construct an embedding $\Phi : V \mapsto \mathbb{R}^{O(1)}$ using the algorithm of Krauthgamer and Lee [17, Theorem 6.1] such that for all pairs of vertices $u, v \in V(H)$, $u \neq v$, $\|\Phi(u) - \Phi(v)\|_2 \geq 1$ and for all edges $\{u, v\} \in E(H)$, $\|\Phi(u) - \Phi(v)\|_2 \leq 2$. This step also yields a quality-2 embedding in $O(1)$ -dimensional Euclidean space (Section 3.1).
2. Apply the measured descent embedding of Krauthgamer et al. [18, Theorem 1.5] to the metric obtained in ‘‘Step 1’’ to obtain a $(k, O(\sqrt{\log n}))$ -volume respecting embedding of H .

3.1 Quality-2 Embedding in $O(1)$ -Dimensions

Levin, together with Linial, London, and Rabinovich [26], made a conjecture (Conjecture 8.2 in [26]) that is quite relevant to the best quality embedding problem. Let \mathbb{Z}_∞^d be the infinite graph with vertex set \mathbb{Z}^d (i.e., the d -dimensional integral lattice) and an edge $\{u, v\}$ whenever $\|u - v\|_\infty = 1$. For any graph G , define $\text{dim}(G)$ to be the smallest d such that G occurs as a (not necessarily induced) subgraph of \mathbb{Z}_∞^d .

Conjecture 1 [Levin, Linial, London, Rabinovich] For any graph $G = (V, E)$ with growth rate ρ_G , G occurs as a (not necessarily induced) subgraph of $\mathbb{Z}_\infty^{O(\rho_G)}$. In other words, $\dim(G) = O(\rho_G)$.

Linial [25] introduced the following Euclidean analogue to this notion of dimensionality. For any graph G , define $\dim_2(G)$ to be the smallest d such that there is a mapping $\Phi : V \mapsto \mathbb{R}^d$ with the properties: (i) $\|\Phi(u) - \Phi(v)\|_2 \geq 1$ for all $u \neq v \in V$ and (ii) $\|\Phi(u) - \Phi(v)\|_2 \leq 2$ for all $\{u, v\} \in E$.

Lee and Krauthgamer [17] show that the specific bound on $\dim(G)$, mentioned in the above conjecture does not hold, by exhibiting a graph G for which $\dim(G) = \Omega(\rho_G \log \rho_G)$. They also prove a weaker form of the conjecture by showing that $\dim(G) = O(\rho_G \log \rho_G)$ for any graph G [17]. This proof relies on the Lovász Local Lemma, but can be turned into a polynomial time algorithm using standard algorithmic versions of the Lovász Local Lemma [4, 28]. Finally, they also prove that $\dim_2(G) = O(\rho_G \log \rho_G)$. This result, along with Theorem 1 leads to the following theorem.

Theorem 3 *There is a polynomial time algorithm, that takes as input a UDG and constructs an embedding of quality-2 in $O(1)$ -dimensional Euclidean space.*

3.2 $(k, O(\sqrt{\log n}))$ -Volume Respecting Embedding of Growth-Restricted Graphs

For the next step, we assume that we are given an embedding Φ of H into $\mathbb{R}^{O(1)}$ such that $\|\Phi(u) - \Phi(v)\|_2 \geq 1$ for all $u, v \in V(H)$, $u \neq v$, and $\|\Phi(u) - \Phi(v)\|_2 \leq 2$ for all $\{u, v\} \in E(H)$. Let $\Phi(H)$ denote the set of points in $\mathbb{R}^{O(1)}$ that Φ maps $V(H)$ into. Theorem 1.5 of Krauthgamer et al. [18] implies that every metric space (X, d) has a $(k, \sqrt{\alpha_X \cdot \log n})$ -volume respecting embedding into L_2 , where α_X is the doubling dimension on X . Since Φ maps H into $\mathbb{R}^{O(1)}$, the metric space $(\Phi(H), \|\cdot\|_2)$ has constant doubling dimension and therefore using this theorem [18] we obtain a $(k, O(\sqrt{\log n}))$ -volume respecting embedding of $(\Phi(H), \|\cdot\|_2)$. Scaling the volume respecting embedding down by a factor of 2 leads to the following theorem.

Theorem 4 *There is a polynomial time algorithm that constructs, with high probability, a $(k, O(\sqrt{\log n}))$ -volume respecting embedding, of any growth-restricted graph.*

4 $O(\log^{2.5} n)$ Quality Embedding in the Plane

In this section, we describe complete algorithm and prove that the constructed 2-dimensional embedding has quality $O(\log^{2.5} n)$. The first two steps of our algorithm are described in detail in the previous two sections. The last three steps respectively describe (i) a random projection in \mathbb{R}^2 , (ii) a “rounding” step, and (iii) constructing an embedding of the original input graph G from the embedding of the cluster graph $G[\mathcal{C}]$. The random projection and the subsequent “rounding” step are essentially the same as in Vempala’s algorithm [36] and are also used by [27, 21].

Step 1. Construct a clique partition $\mathcal{C} = \{C_1, C_2, \dots, C_t\}$ of the given UDG $G = (V, E)$ so that the induced cluster graph $H := G[\mathcal{C}]$ is growth-restricted. This is described in Section 2.

Step 2. Let $V(H) = \{v_1, v_2, \dots, v_n\}$. Construct a $(\log n, O(\sqrt{\log n}))$ -volume respecting embedding Φ of the shortest path metric of H , as described in Section 3. Let $u_i := \Phi(v_i)$ for $i = 1, 2, \dots, n$.

Step 3. Choose 2 random lines, ℓ_1 and ℓ_2 (independently and uniformly), passing through the origin. Project the point set $\{u_1, u_2, \dots, u_n\}$ onto each of the two lines, mapping each u_i to $(u_i \cdot \ell_1, u_i \cdot \ell_2)$. Denote each $(u_i \cdot \ell_1, u_i \cdot \ell_2)$ by w_i .

Step 4. Discretize the plane into grid, with each cell having dimensions $1/\sqrt{n} \times 1/\sqrt{n}$. Call each such grid cell an an *outer* grid cell. Let M be the maximum number of points w_i that fall in any outer cell after the random projection step. Subdivide each outer grid cell by an *inner* grid, with each inner grid cell having dimensions $1/\sqrt{n \cdot M} \times 1/\sqrt{n \cdot M}$. For each outer cell C , map each point that falls into C to grid points of the inner grid such that each w_i gets mapped to a distinct grid point. Finally, scale up all points by a factor of $\sqrt{n \cdot M}$ along both dimensions so that each inner grid cell has unit width.

Step 5. Since every vertex v_i in H is associated with a clique C_i in G , all vertices in C_i are assigned the coordinates assigned to v_i in Step (4), to get the final embedding of G .

4.1 Analysis of Approximation Guarantee

Here we show that the above algorithm, with high probability, yields a 2-dimensional embedding of quality $O(\log^{2.5} n)$. The analysis is similar to Vempala’s analysis [36] and that of Kuhn et al. [21] and Moscibroda et al. [27]. The proof of Lemma 10 is included mostly for the sake of completeness. An interesting aspect of our analysis is that a key technical lemma (Lemma 8), that was proved using “spreading constraints” by Vempala [36], follows quite easily, simply from the fact that we are working with a growth-restricted graph.

Lemma 8 *There is a constant β such that for any $v \in V(H)$, and any $S \subset V(H)$, $\sum_{u \in S} \left(\frac{1}{d(v,u)}\right)^2 \leq \beta \cdot \log |S|$.*

Proof: Lemma 7 tells us that for some constant α , $|B(v, r)| \leq \alpha \cdot r^2$. The sum $\sum_{u \in S} \left(\frac{1}{d(v,u)}\right)^2$ is maximized when, the largest possible subset $S_1 \subseteq S$ of vertices is at distance 1 from v , the largest possible subset $S_2 \subseteq S \setminus S_1$ of vertices is at distance 2 from v and so on. Thus, $|S_i| = \alpha \cdot (i^2 - (i-1)^2) = \alpha \cdot (2i-1)$ for $i = 1, 2, \dots$. Therefore,

$$\sum_{u \in S} \left(\frac{1}{d(v,u)}\right)^2 \leq \sum_{i \geq 1} \alpha \cdot \frac{(2i-1)}{i^2} \leq \sum_{i \geq 1} \frac{2\alpha}{i} \leq 2\alpha \cdot (\ln |S| + 1).$$

The last inequality follows from the fact that i can take on $|S|$ distinct values. Hence, for some constant β , $\sum_{u \in S} \left(\frac{1}{d(v,u)}\right)^2 \leq \beta \cdot \log |S|$. □

The above upper bound on the sum on inverse square distances from v is critically used to derive the following upper bound on sum of inverse squares of tree volumes of all size- k vertex subsets. We skip the proof of this claim, since it is identical to the proof of Lemma 5.4 in [21].

Lemma 9

$$\sum_{S \subset V(H), |S|=k} \left(\frac{1}{\text{Vol}(S)}\right)^2 \leq 2k! \cdot \left(\frac{\beta}{4}\right)^{k-1} \cdot n \cdot \log^{k-1} n.$$

Here β is the constant that appears in the previous lemma.

Lemma 10 *After Step (4), with high probability, the maximum number of vertices that fall in an outer grid cell is $O(\log^4 n)$.*

Proof: Consider an outer grid cell C . Note that C is defined by two length $1/\sqrt{n}$ intervals – one on line l_1 and the other on line l_2 . For any subset S of vertices, let X_S^i be the indicator random variable that is 1 if all vectors associated with the vertices of S fall in the interval corresponding to cell C on line l_i . Let N_C denote the number of size- k subsets $S \subseteq V(H)$ that have fallen into C (as a result of Step (3)). In Step (3), the value of k that was used is $\log n$; we will replace k by $\log n$ later in the proof.

$$\begin{aligned}
\mathbf{E}[N_C] &= \sum_{S \subseteq V(H), |S|=k} \mathbf{E}[X_S^1 \cdot X_S^2] \\
&= \sum_{S \subseteq V(H), |S|=k} \mathbf{E}[X_S^1] \mathbf{E}[X_S^2] && \text{(by independence of } X_S^1 \text{ and } X_S^2) \\
&= \sum_{S \subseteq V(H), |S|=k} \mathbf{Pr}[X_S^1 = 1]^2 \\
&\leq \gamma \cdot \sum_{S \subseteq V(H), |S|=k} \left(\frac{W^{k-1} n^{\frac{k-1}{2}}}{(k-1)! \text{Evol}(S)} \right)^2 && \text{(for constant } \gamma, \text{ using Lemma 3)} \\
&\leq \gamma \cdot \sum_{S \subseteq V(H), |S|=k} \left(\frac{1}{(k-1)! \text{Evol}(S)} \right)^2 && \text{(since } W = 1/\sqrt{n}) \\
&\leq \gamma \cdot \sum_{S \subseteq V(H), |S|=k} \left(\frac{(\delta \sqrt{\log n})^{k-1}}{(k-1)! \text{Vol}(S)} \right)^2 && \text{(for constant } \delta, \text{ using Theorem 4)} \\
&\leq \gamma \cdot \sum_{S \subseteq V(H), |S|=k} \left(\frac{(\delta \sqrt{\log n})^{k-1} \cdot 2^k}{\text{Tvol}(S)} \right)^2 && \text{(Lemma 1)} \\
&\leq \gamma \cdot \delta^{2(k-1)} \cdot (4 \log n)^k \cdot \sum_{S \subseteq V(H), |S|=k} \left(\frac{1}{\text{Tvol}(S)} \right)^2 \\
&\leq \gamma \cdot \delta^{2(k-1)} \cdot 8k! \cdot \beta^k \cdot n \cdot \log^{2k} n \leq n \cdot (\zeta \cdot k \cdot \log^2 n)^k && \text{(for constant } \zeta, \text{ by Lemma 9)}
\end{aligned}$$

Using Markov's inequality we get $\mathbf{Pr}[N_C \geq n^4 \cdot n \cdot (\zeta \cdot k \cdot \log^2 n)^k] \leq 1/n^4$. Since H has n vertices, the maximum distance between a pair of vertices in H is bounded above by n . Since the embedding in Step (2) is non-expansive, we can bound the total number of outer cells in each dimension by $n^{1.5}$, which leads to a total of n^3 outer cells. Using the union bound we get

$$\mathbf{Pr} \left[\begin{array}{l} \text{There exists an outer cell} \\ \text{containing } n^4 \cdot n \cdot (\zeta \cdot k \cdot \log^2 n)^k \\ \text{or more size-}k \text{ subsets of } V(H) \end{array} \right] \leq \frac{1}{n}$$

Hence, with probability at least $1 - 1/n$, every outer cell contains fewer than $n^4 \cdot n \cdot (\zeta \cdot k \cdot \log^2 n)^k$ size- k subsets of $V(H)$. Using the fact that if there are N subsets of size k (in an outer cell), then there are at most $kN^{1/k}$ points in it, we obtain that with probability at least $1 - 1/n$, every outer cell contains at most $k(n^5 \cdot (\zeta \cdot k \cdot \log^2 n)^k)^{1/k}$ points. Since the value of $k = \log n$, using the fact that $n^{5/\log n} = O(1)$, we get that with high probability, every outer cell has at most $O(\log^4 n)$ points. \square

Lemma 11 *With high probability, the maximum edge length in the final embedding is $O(\log^{2.5} n)$.*

Proof: For any edge $\{v_i, v_j\}$ in the cluster graph H , after Step (2), $\|u_i - u_j\|_2 \leq 1$, since the embedding constructed in Step (2) is non-expansive. After the random projection in Step (3), using the value of $c = 4\sqrt{\log n}$ in Lemma 2, we see that $\|w_i - w_j\|_2 < \frac{4\sqrt{\log n}}{\sqrt{n}}$ for all edges $\{v_i, v_j\} \in E(H)$, with probability at least $1 - 1/n$. Lemma 10 tells us that with high probability, each outer grid cell is divided into at most $O(\log^2 n)$ inner grid cells along each of the two dimensions of the outer grid cell. Therefore, the scaling in Step (4) by a factor of at most $\sqrt{n} \cdot \log^2 n$, will cause each edge to have length at most $O(\log^{2.5} n)$, with high probability. \square

Theorem 5 *With high probability, the quality of the embedding is $O(\log^{2.5} n)$.*

Proof: Follows immediately from the high probability upper bound on the maximum edge length proved in the above lemma and the fact that every pair of vertices are separated by at least unit distance. \square

5 Conclusions

The techniques used in this paper may not be able to provide an embedding of quality better than polylogarithmic in n . These techniques were introduced in the context of the bandwidth minimization problem [14] and since nothing better than $O(\text{polylog}(n))$ approximation is known for that problem, even on trees, perhaps we cannot hope for $o(\text{polylog}(n))$ quality embeddings of UDGs via these techniques. We have used the fact that our input graph has an $O(1)$ -quality embedding in a critical way, for e.g., in constructing a growth-restricted approximation of the input graph. Perhaps we can take advantage of this feature of the input graph in other ways as well. One possibility is that a completely different approach, possibly analogous to the dynamic programming approach used by Gurari and Sudborough [15] for bandwidth minimization of graphs with constant bandwidth, could be used to obtain a constant-quality embedding of UDGs.

Note that our algorithm makes no claims about the *sum of edge lengths* quality measure described in Section 1. For our problem, it was sufficient to construct a constant-sized neighborhood clique partition of the input UDG G in order to construct a growth-restricted approximation of G . Such a partition may yield two cliques that have many edges between them and it is possible that two such cliques are embedded far away from each other, contributing a large term to the objective function. It seems that a more sophisticated partitioning and embedding scheme is needed to avoid this possibility. We have not investigated this problem and leave it as a question for future research.

Raghavan and Spinrad [33] claim that their algorithm for finding a maximum clique in a UDG without any geometry can be easily generalized to *unit ball graphs* (UBGs) in d -dimensional Euclidean space. In their paper, they show that ordering the edges in non-increasing order is a CNEEO. While such an ordering is a CNEEO for UDGs, it is not a CNEEO for 3-dimensional UBGs. It is unclear as to how to generalize the Raghavan-Spinrad argument to higher dimensions. It is easy to see why a greedy ordering of edges by length does not produce a CNEEO even for $d = 3$. Afshani and Chan [1] also consider the problem of finding a maximum clique in 3-d UBGs and give several efficient constant-factor approximations to the maximum clique problem in 3-d UBGs with a realization. They note that the complexity of the clique problem for d -UBGs for $d > 2$, is unknown [1] even with the use of geometry.

In an orthogonal direction, it may be relevant to the wireless sensor networks application, if we could devise an efficient distributed algorithm (i.e., one that runs in polylogarithmic number of rounds) for the best quality embedding problem, without sacrificing the quality of the embedding.

We have made some progress toward this goal in this paper as shown by the fact that the cluster graph approximation step can be implemented in polylogarithmic number of rounds using ideas from recent papers [19, 20]. At this point, it is unclear how to devise distributed implementations for constructing a volume-respecting embedding and for and Vempala’s random projection step.

References

- [1] P. Afshani and T. M. Chan. Approximation algorithms for maximum cliques in 3d unit-disk graphs. In *CCCG*, pages 19–22, 2005.
- [2] J. Aspnes, D. K. Goldenberg, and Y. R. Yang. On the computational complexity of sensor network localization. In *ALGOSENSORS ’04: First International Workshop on Algorithmic Aspects of Wireless Sensor Networks*, pages 32–44, Turku, Finland, 2004. Springer-Verlag.
- [3] A. Basu, J. Gao, J. S. B. Mitchell, and G. Sabhnani. Distributed localization using noisy distance and angle information. In *MobiHoc ’06: Proceedings of the seventh ACM international symposium on Mobile ad hoc networking and computing*, pages 262–273, New York, NY, USA, 2006. ACM Press.
- [4] J. Beck. An algorithmic approach to the Lovasz local lemma. i. *Random Structures and Algorithms*, 2(4):343–366, 1991.
- [5] P. Biswas and Y. Ye. Semidefinite programming for ad hoc wireless sensor network localization. In *IPSN ’04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 46–54, New York, NY, USA, 2004. ACM Press.
- [6] A. Blum and D. Karger. An $\tilde{O}(n^{3/14})$ -coloring algorithm for 3-colorable graphs. *Information Processing Letters*, 61(1):49–53, 1997.
- [7] A. Blum, G. Konjevod, R. Ravi, and S. Vempala. Semi-definite relaxations for minimum bandwidth and other vertex-ordering problems. In *STOC ’98: Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 100–105, New York, NY, USA, 1998. ACM Press.
- [8] P. Bose, P. Morin, I. Stojmenovic, and J. Urrutia. Routing with guaranteed delivery in ad hoc wireless networks. *Wireless Networks*, 7(6):609–616, 2001.
- [9] H. Breu and D. G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Comput. Geom. Theory Appl.*, 9(1-2):3–24, 1998.
- [10] J. Bruck, J. Gao, and A. Jiang. Localization and routing in sensor networks by local angle information. In *MobiHoc ’05: Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, pages 181–192, New York, NY, USA, 2005. ACM Press.
- [11] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localization for very small devices, 2000.
- [12] S. Durocher, D. G. Kirkpatrick, and L. Narayanan. On routing with guaranteed delivery in three-dimensional ad hoc wireless networks. In Shrisha Rao, Mainak Chatterjee, Prasad Jayanti, C. Siva Ram Murthy, and Sanjoy Kumar Saha, editors, *ICDCN*, volume 4904 of *Lecture Notes in Computer Science*, pages 546–557. Springer, 2008.

- [13] G. Even, J. Naor, S. Rao, and B. Schieber. Divide-and-conquer approximation algorithms via spreading metrics. In *FOCS '95: Proceedings of the 36th Annual Symposium on Foundations of Computer Science (FOCS'95)*, page 62, Washington, DC, USA, 1995. IEEE Computer Society.
- [14] U. Feige. Approximating the bandwidth via volume respecting embeddings. *J. Comput. Syst. Sci.*, 60(3):510–539, 2000.
- [15] E. M. Gurari and I. H. Sudborough. Improved dynamic programming algorithms for bandwidth minimization and the mincut linear arrangement problem. *J. Algorithms*, 5(4):531–546, 1984.
- [16] B. Karp and H. T. Kung. GPSR: greedy perimeter stateless routing for wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 243–254, New York, NY, USA, 2000. ACM Press.
- [17] R. Krauthgamer and J. R. Lee. The intrinsic dimensionality of graphs. *Combinatorica*, 27(5):551–585, 2007.
- [18] R. Krauthgamer, J. R. Lee, M. Mendel, and A. Naor. Measured descent: A new embedding method for finite metrics. *Geometric And Functional Analysis*, 15(4):839–858, 2005.
- [19] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs. In *19th International Symposium on Distributed Computing (DISC), Cracow, Poland*, September 2005.
- [20] F. Kuhn, T. Moscibroda, T. Nieberg, and R. Wattenhofer. Local approximation schemes for ad hoc and sensor networks. In *DIALM-POMC '05: Proceedings of the 2005 joint workshop on Foundations of mobile computing*, pages 97–103, New York, NY, USA, 2005. ACM Press.
- [21] F. Kuhn, T. Moscibroda, R. O'Dell, M. Wattenhofer, and R. Wattenhofer. Virtual coordinates for ad hoc and sensor networks. To appear in *Algorithmica*, 2006.
- [22] F. Kuhn, T. Moscibroda, and R. Wattenhofer. Unit disk graph approximation. In *DIALM-POMC '04: Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pages 17–23, New York, NY, USA, 2004. ACM Press.
- [23] F. Kuhn, T. Moscibroda, and R. Wattenhofer. On the locality of bounded growth. In *PODC '05: Proceedings of the twenty-fourth annual ACM symposium on Principles of distributed computing*, pages 60–68, New York, NY, USA, 2005. ACM.
- [24] F. Kuhn, R. Wattenhofer, Y. Zhang, and A. Zollinger. Geometric ad-hoc routing: of theory and practice. In *PODC '03: Proceedings of the twenty-second annual symposium on Principles of distributed computing*, pages 63–72, New York, NY, USA, 2003. ACM Press.
- [25] N. Linial. Variation on a theme of Levin. In J. Matoušek, editor, *Open Problems, Workshop on Discrete Metric Spaces and their Algorithmic Applications*, 2002.
- [26] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.

- [27] T. Moscibroda, R. O’Dell, M. Wattenhofer, and R. Wattenhofer. Virtual coordinates for ad hoc and sensor networks. In *DIALM-POMC ’04: Proceedings of the 2004 joint workshop on Foundations of mobile computing*, pages 8–16, New York, NY, USA, 2004. ACM Press.
- [28] R. A. Moser and G. Tardos. A constructive proof of the general lovasz local lemma. *CoRR*, abs/0903.0544, 2009.
- [29] R. Nagpal, H. E. Shrobe, and J. Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *IPSN ’03: Proceedings of the second international symposium on Information processing in sensor networks*, pages 333–348, New York, NY, USA, 2003. ACM Press.
- [30] D. Niculescu and B. Nath. Ad hoc positioning system (APS). In *In Proceedings of Sixth Global Internet Symposium (GI2001) in conjunction with IEEE Globecom 2001*, pages 2926–2931, November 25–29 2001.
- [31] D. Peleg. *Distributed computing: a locality-sensitive approach*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2000.
- [32] I. A. Pirwani and M. R. Salavatipour. A PTAS for minimum clique partition in unit disk graphs, 2009.
- [33] V. Raghavan and J. Spinrad. Robust algorithms for restricted domains. In *SODA ’01: Proceedings of the twelfth annual ACM-SIAM symposium on Discrete algorithms*, pages 460–467, Philadelphia, PA, USA, 2001. Society for Industrial and Applied Mathematics.
- [34] A. Rao, C. Papadimitriou, S. Shenker, and I. Stoica. Geographic routing without location information. In *MobiCom ’03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 96–108, New York, NY, USA, 2003. ACM Press.
- [35] J. Schneider and R. Wattenhofer. A log-star distributed maximal independent set algorithm for growth-bounded graphs. In Rida A. Bazzi and Boaz Patt-Shamir, editors, *PODC*, pages 35–44. ACM, 2008.
- [36] S. Vempala. Random projection: A new approach to VLSI layout. In *FOCS ’98: Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, pages 389–395, Washington, DC, USA, 1998. IEEE Computer Society.