Good Quality Virtual Realization of Unit Ball Graphs *

Sriram V. Pemmaraju Imran A. Pirwani

April 13, 2007

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 ball graph* in fixed dimensional Euclidean space $\mathbb{R}^d$, 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 unit disk graph (UDG), are NP-hard. While any connected unit disk graph (UDG) has a 2-dimensional embedding with quality between 1/2 and 1, as far as we know, Vempala's random projection approach (*FOCS 1998*) provides the best quality bound of $O(\log^3 n \cdot \sqrt{\log\log n})$ for this problem.

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 UBG in fixed dimensional Euclidean space $\mathbb{R}^d$. If the embedding is allowed to reside in higher dimensional space, we obtain improved results: a quality-2 embedding in $\mathbb{R}^{O(d \log d)}$. The first step of our algorithm constructs a "growth-restricted approximation" of the given UBG. While such a construction is trivial if the UBG 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 a constant-factor approximation to the minimum clique cover problem for UBGs, described without geometry. The second step of our algorithm combines the probabilistic decomposition of growth-restricted graphs due to Lee and Krauthgamer (*STOC 2003*) with Rao's embedding algorithm for planar graphs (*SoCG 1999*) to obtain a $(k, O(\sqrt{\log n}))$-volume respecting embedding of growth-restricted graphs.

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 *d-dimensional unit ball graph (UBG)* if there is an embedding $\Phi : V \mapsto \mathbb{R}^d$ 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 essentially interested in the problem of finding a realization Φ of a given UBG. It is unlikely that this problem has a polynomial-time algorithm because even the problem of recognizing if a given graph is a 2-dimensional UBG (henceforth, a *unit disk graph* and in short a *UDG*) is NP-hard [7]. Recently, Aspnes et al. [1] 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 vertices are known. The problem remains NP-hard when all angles between adjacent edges are known [8] and also when all angles plus slightly noisy, pairwise distances are known [2]. Given this situation, we consider the problem of computing an "approximate" realization of the given UBG. Let $G = (E, V)$ be a d_1-dimensional UBG. Let $\Phi : V \mapsto \mathbb{R}^{d_2}$ be an embedding of G into $\mathbb{R}^{d_2}$. 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_1, d_2, and a d_1-dimensional UBG $G = (V, E)$, find an embedding $\Phi : V \mapsto \mathbb{R}^{d_2}$ with best (smallest) quality.

*The authors are in the Department of Computer Science, The University of Iowa, Iowa City, IA 52242-1419. E-mail addresses: `[sriram, pirwani]@cs.uiowa.edu`.

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

Motivated by applications in VLSI design, Vempala considers a similar problem [28]: given an undirected graph G, compute a one-to-one assignment of vertices of G to points of a 2-dimensional grid (with unit square cells) such that the maximum Euclidean length of an edge is minimized. Note that G is an arbitrary graph, not just a UBG. Using the random projection approach, Vempala obtains an embedding with maximum length $O(\log^3 n\sqrt{\log\log n})$[1]. Note that the embedding constructed by Vempala's algorithm has quality $O(\log^3 n\sqrt{\log\log n})$ because every pair of vertices are separated by at least one unit distance. Motivated by the *localization* problem in wireless sensor networks, Moscibroda et al. define quality as a measure the "goodness" of a realization [22]. This paper [22] 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 [17] corrects this bound to $O(\log^{3.5} n\cdot\sqrt{\log\log n})$. The only hardness of approximation result for the best quality embedding problem, that we are aware of, is this [18]: there is no polynomial time algorithm (unless $P = NP$) that can compute a $(\sqrt{3/2}-\varepsilon)$-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. Here ε goes to 0 as n, the number of vertices in the given UDG, tends to ∞.

As mentioned earlier, the best quality embedding problem for UBGs is motivated by the *localization problem* in wireless sensor networks [9, 23]. Low dimensional UBGs are typically used to 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 [6, 12, 19]. 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 results can have additional errors when used indoors. Another solution is to equip a few "anchor" nodes with GPS receivers [3] 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 [24, 26]. 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 d_1-dimensional UBG for small fixed values of d_1; $d_1 = 2, 3$ are probably the most relevant values for wireless sensor networks. While the best quality embedding problem allows the host space to have arbitrary dimensions (specified as d_2 in the input), we are particularly interested in the case when $d_2 = 2$. Even though our results generalize to higher dimensional host spaces, they are most relevant to the wireless sensor networks application when $d_2 = 2$. This is because currently, well-known geometric routing algorithms such as GPSR [12] only provide guaranteed delivery for networks in the plane. Thus, even though G may be a 3-dimensional UBG, we are mainly interested in obtaining in a 2-dimensional embedding of G.

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 d-dimensional UBG (for any fixed d) into $\mathbb{R}^2$. Our result can be seen as improving Vempala's bound [28] when the input is restricted to be a d-dimensional UBG. 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 Vempala'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) [5, 10]. We avoid the spreading constraints approach via a combinatorial algorithm for computing a "growth-restricted approximation" of the given UBG. This step may be of independent interest since it can be implemented efficiently (in polylogarithmic number of rounds under the $\mathcal{LOCAL}$ model of computation) in a distributed setting as well. In a related result, we point out that using the "growth-restricted approximation" of the UBG in combination with a recent result due to Krauthgamer and Lee [14] leads to a quality-2 embedding of any d-dimensional UBG into $\mathbb{R}^{O(d\log d)}$. Since we are primarily interested in small, fixed values of d, this is a quality-2 embedding into constant dimensional space. Our algorithm has three main steps.

[1]Due to a small error in Lemma 2 in Vempala's paper [28], the bound proved in the paper is $O(\log^4 n)$. However, by using a stronger version of Theorem 2 in his paper, one can obtain an $O(\log^3 n\sqrt{\log\log n})$ bound.

1.1.1 Constructing a Growth-Restricted Approximation

In the first step, we partition the given UBG into cliques such that when the cliques are contracted into vertices, we get a "growth-restricted approximation" of the given UBG. 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 $B_G(v, r) = \{u \in V \mid d_G(u, v) \leq r\}$[2] denote the closed ball of radius r centered at v in G. 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 $\mathcal{C} = \{C_1, C_2, \ldots\}$ 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 d-dimensional UBG $G = (V, E)$ into cliques $\mathcal{C} = \{C_1, C_2, \ldots\}$ 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 $\mathcal{C} = \{C_1, C_2, \ldots\}$ 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 $\mathcal{C} = \{C_1, C_2, \ldots\}$ in which each part C_i is all vertices in a cell. 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. The above argument generalizes in a straightforward way to d-dimensional space, where d-dimensional cubes of length $1/sqrtd$ can be used to partition the vertices.

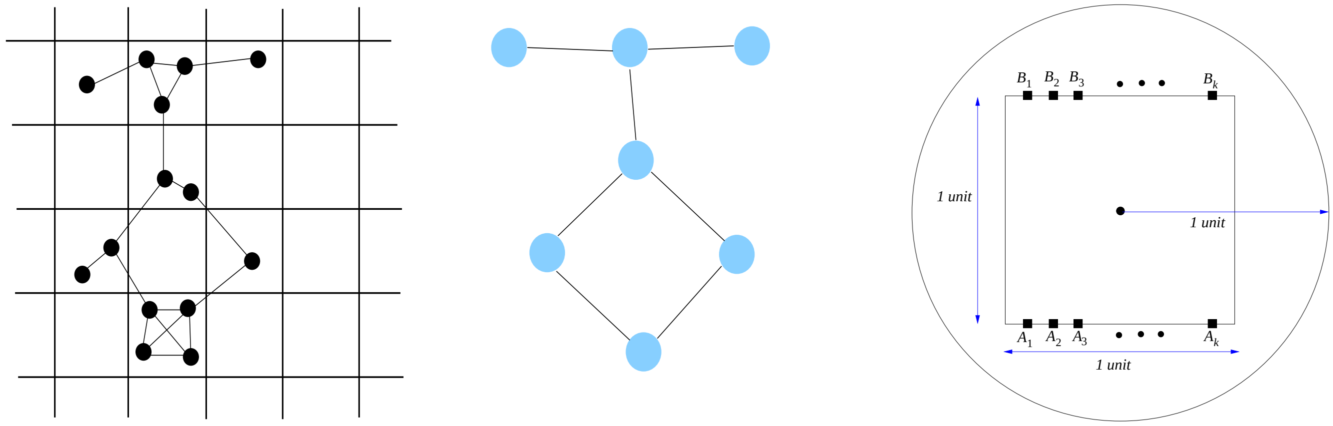
In the absence of geometric information, it is not immediately clear how to obtain the desired clique partition. One possible approach is to start with a maximal independent set I of G and attach each $v \in V \setminus I$ to an arbitrary neighbor in I. For each $v \in I$, let S_v denote the set consisting 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 $\mathbb{R}^d$, $\|\Phi(u) - \Phi(v)\|_2 > 1$ 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. We know that each H_v has a partition into $O(1)$ cliques. For example, if G is a UDG, there exists a partition of H_v into at most 5 cliques. But, how to find a constant-sized partition of H_v? Noteworthy is the work of Raghavan and Spinrad [25] that computes a maximum cardinality clique of an input UBG, 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 $\overline{H_v}$ 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 for coloring 3-colorable graphs uses $\tilde{O}(n^{3/14})$ colors [4][3].

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 [25] 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 recent papers by Kuhn et al. [15, 16] are relevant here. Kuhn et al. [15] show how to compute a MIS on a class of graphs, *bounded growth graphs*, that contains UBGs in $O(\log \Delta \cdot \log^* n)$ rounds using only connectivity information, where Δ is the maximum degree of G. Kuhn et al. [16] compute a $(1+\varepsilon)$ approximation to the maximum independent set (MaxIS) problem for any given $\varepsilon > 0$ in $O(\log \Delta \cdot \log^* n + \log^* n/\varepsilon^{O(1)})$ rounds of distributed computation also on the same class of graphs. Using either of the distributed algorithms [15, 16], 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.

[2]Where it is clear from the context, we write $B(u, v)$ instead of $B_G(u, v)$

[3]Here the $\tilde{O}$ notation hides factors that are logarithmic in n.



(a) (b)

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.

1.1.2 Constructing Volume Respecting Embeddings

The remaining two steps of our algorithm follow the approach introduced by Vempala [28], 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 [11] 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 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 $\Phi : X \to \mathbb{R}^{|X|-1}$. Given an arbitrary metric space (X, d), a contraction $\Phi : X \to \mathbb{R}^L$ 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 [21], in which only pairs of points are considered. A volume respecting embedding will be very useful for the last 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 [11] presents a polynomial time algorithms 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.

However, it is possible to improve on Feige's bounds in our context where the graph has constant growth rate. Specifically, we perform a probabilistic decomposition of H using a technique due to Krauthgamer and Lee [14] for growth-bounded graphs and then use the decomposition as a starting point for Rao's algorithm [27] for constructing a $(k, O(\sqrt{\log n}))$-volume respecting embedding of H. Rao [27] describes an algorithm for constructing a $(k, O(\sqrt{\log n}))$-volume respecting embedding of $K_{r,r}$-minor free graphs, for any fixed r. The starting point of Rao's algorithm is a probabilistic version of a graph decomposition due to Klein, Plotkin, and Rao [13] that works for $K_{r,r}$-minor free graphs. Note that H may contain a $K_{r,r}$-minor for any r, and therefore Rao's algorithm cannot be used as is. However, the Krauthgamer-Lee probabilistic decomposition [14] for growth-restricted graphs can be used to replace the Klein-Plotkin-Rao decomposition in the volume respecting embedding algorithm by Rao [27].

Now we mention a useful lemma due to Feige [11] that establishes a lower bound on $\text{Vol}(S)$. To show that an 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

Vol(S) in terms of the tree volume of S. For any $S \subseteq X$, the *tree volume* of S, denoted Tvol(S) the product of the edge lengths in a minimum spanning tree of S.

Lemma 1 (Feige [11]) *Let (X, d) be a metric space. For any size-k subset $S \subseteq X$, $Tvol(S) \leq 2^k (k-1)!\, Vol(S)$.*

1.1.3 Random Projections

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. [22]. 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 [28] 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 [28]) *Let $v \in \mathbb{R}^d$. For a random unit vector l, $c > 1$,*

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

Lemma 3 (Vempala [28]) *Let S be a set of vectors $v_1, v_2, \ldots, v_k \in \mathbb{R}^d$. For a random unit vector l,*

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

Next, we partition the plane into squares and then construct a fine enough square grid inside each square so that all vertices of H that fall into any square can be mapped to a distinct grid point. Vempala [28] calls this step "rounding". Finally, vertices in the clique C_v in G that correspond to v in H are assigned the grid point that v is mapped to.

2 Constructing a Growth-Restricted Approximation

In this section, we show how to construct a clique partition $\mathcal{C} = \{C_1, C_2, \ldots\}$ of a given d-dimensional UBG 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 [25]. For simplicity, we discuss only the 2-dimensional case; extension to d-dimensional UBGs is straightforward. Raghavan and Spinrad [25] 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 $L = (e_1, e_2, \ldots, e_m)$ of E, let $G_L[i]$ denote the spanning subgraph of G with edge set $\{e_i, e_{i+1}, \ldots, e_m\}$. 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 $L = (e_1, e_2, \ldots, e_m)$ 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 $L = (e_1, e_2, \ldots, e_m)$ 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. The three results mentioned above lead to a polynomial-time

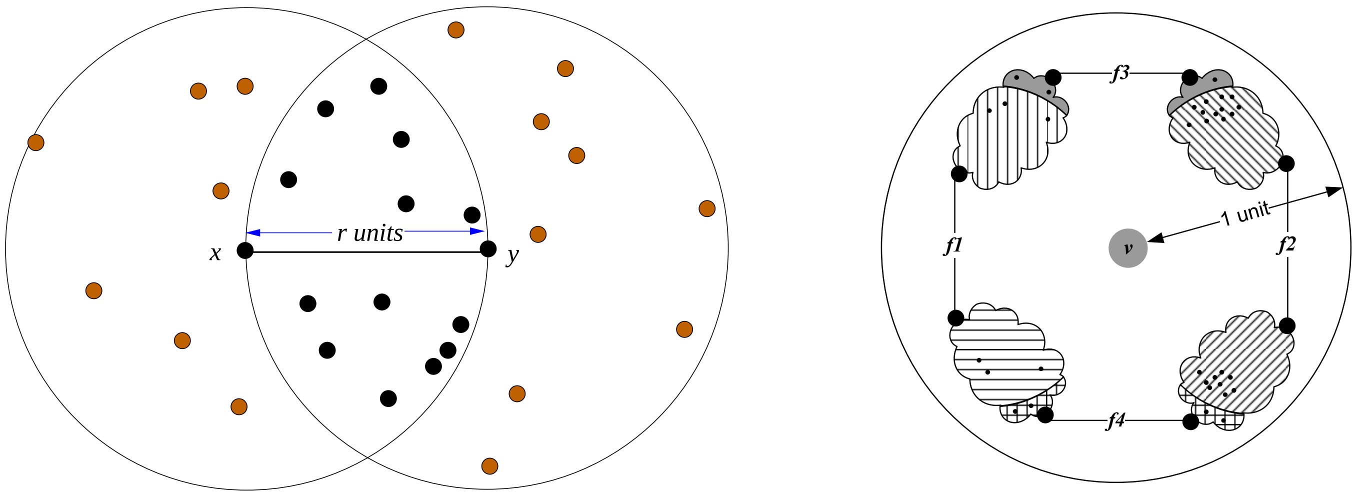


Figure 2: (a) Suppose that edge $\{x, y\}$ has rank i in an ordering of the edges of G in non-increasing length order. The points in the common neighborhood of x and y in $G_L[i]$ are exactly those in the lune shown above. A point outside the lune may be a common neighbor of x and y in G, but not in $G_L[i]$. 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`. Each "cloud" is a clique. Patterns in the clouds represent the cliques obtained; the a vertical line pattern is C_1', and the horizontal line pattern is C_1''; the down-sloping pattern is C_2' and the up-sloping one is C_2''; the shaded pattern is C_3', and the checkered pattern is C_4' for a totoal of 6 cliques.

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. Since G_v is also a UDG, it admits a CNEEO. We start with a simple lemma.

Lemma 4 *Let C be a clique in G_v and let $L = (e_1, e_2, \ldots, e_m)$ 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 $e_i, e_{i+1}, \ldots$. Since e_i is the first edge in C, $N_L[i]$, contains C. □

Recall that the closed neighborhood of an vertex in a UDG can be partitioned into at most 5 cliques. Let $C_1, C_2, \ldots, C_5$ be a clique partition of S_v. The implication of the above lemma is that even though we do not know the clique partition $C_1, C_2, \ldots, C_5$, we do know that for every CNEEO $L = (e_1, e_2, \ldots, e_m)$ 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 $(f_1, f_2, \ldots f_5)$ 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 $(f_1, f_2, \ldots f_5)$ of $E(G_v)$ **do**

$G_0 \leftarrow G_v$
for $j \leftarrow 1$ **to** 5 **do**
    Compute a CNEEO L of G_{j-1}
    $i \leftarrow$ rank of f_j in L
    Partition $N_L[i]$ into two cliques C'_j and C''_j
    $G_j \leftarrow G_{j-1} \setminus N_L[i]$
if $(G_5 = \emptyset)$ **return** $\{C'_j, C''_j \mid j = 1, 2, \ldots, 5\}$

The correctness of `NBD-CLIQUE-PARTITION` follows from the fact that there is some edge sequence $(f_1, f_2, \ldots f_5)$ of $E(G_v)$ for which G_5 is empty. Figure 2(b) shows a sample run of the algorithm. We state without proof the following lemma.

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

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 $\mathcal{C} = \{\mathcal{C}_1, \mathcal{C}_2, \ldots\}$ 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 $B(v, r) = \{u \in V(H) \mid d_H(v, u) \leq r\}$ denote the closed ball of radius r, centered at v, with respect to shortest path distances in H.

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 a neighbor of $i(x)$ and y' is a neighbor of $i(y)$. 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 $|i(u)i(v)| \leq 3r$. □

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 $i(u) \in Disk(i(v), 3r)$. Here $Disk(i(v), 3r)$ denotes the disk of radius $3r$ centered at vertex $i(v)$ in any 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 $Disk(i(v), 3r)$ 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 $Disk(i(v), 3r)$ 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 a constant α, we have $X \leq \alpha \cdot r^2$. □

Lemma 7 can be generalized to higher dimensions to yield the following theorem.

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

A side effect of our construction is that the constructed clique partition $\mathcal{C} = \{C_1, C_2, \ldots\}$ is a constant-factor approximation to the *minimum clique cover* problem for any UBG in fixed dimensional space. This follows from the fact that the size of any independent set is a lower bound on the size of a minimum clique cover and our solution produces a clique partition whose size is within a constant of the size of a maximal independent set.

3 Embeddings of Growth-Restricted Graphs

In this section, we show that by combining techniques due to Krauthgamer and Lee [14] with Rao's technique [27], we can derive a simple, 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 earlier papers [28, 22], this step involved solving an LP with exponentially many constraints via the ellipsoid method. Rao [27] presents an algorithm for computing a $(k, O(\sqrt{\log n}))$-volume respecting embedding for planar graphs. This algorithm has two steps. Let G be a given planar graph and let r be an integer satisfying $1 \leq r \leq diam(G)$.

Step 1. Decompose G into clusters using a probabilistic version of the Klein-Plotkin-Rao decomposition [13] for $K_{3,3}$-minor free graphs, that takes r as a parameter. Since planar graphs are $K_{3,3}$-minor free, the Klein-Plotkin-Rao decomposition successfully decomposes G. Due to a result proved by Klein et al. [13], each cluster resulting from this decomposition is guaranteed to have weak diameter bounded above by $O(r)$.

Step 2. Each vertex v in G is assigned a weight that is (roughly speaking) a function of the distance between v and the boundary of the cluster it belongs to.

Each vertex weight forms one coordinate in the embedding for that vertex. For each value of r, the above two steps are repeated $\alpha k \log n$ times, where α is a large enough constant. Finally, the process is repeated for $r = 1, 2, 4, \ldots, diam(G)$. This yields a total of $\alpha k \log n \cdot \log(diam(G))$ coordinates, thus leading to an embedding in $O(k \log^2 n)$-dimensional space.

In our case, the input graph is not guaranteed to be $K_{t,t}$-minor free for any t and therefore the Klein-Plotkin-Rao decomposition algorithm cannot be used. But, since our input graph is growth-restricted, we replace Step 1 above, by the probabilistic decomposition due to Krauthgamer and Lee [14] that exploits the growth-restricted nature of the graph.

Let $G = (V, E)$ be a graph with growth rate ρ. Krauthgamer and Lee [14] present the following simple partitioning procedure for G, parameterized by an integer $r > 0$, that will be the first step of our algorithm. For any $M > 0$, let $\mathrm{T}exp(r, M)$ denote the probability distribution obtained by taking the continuous exponential distribution with mean r, truncating it at M, and rescaling the density function. The resulting distribution has density function

$$p(z) = \frac{e^{M/r}}{r(e^{M/r} - 1)} \cdot e^{-z/r} \quad \text{for any } z \in (0, M).$$

Let $V = \{v_1, v_2, \ldots, v_n\}$. For each vertex v_j, independently choose a radius r_j by sampling the distribution $\mathrm{T}exp(r, 8\rho r \ln r)$. Now define $S_j = B(v_j, r_j) \setminus \cup_{i=1}^{j-1} B(v_i, r_i)$. Thus S_j is the set of all vertices in the ball $B(v_j, r_j)$ that are not contained in any of the earlier balls $B(v_i, r_i)$, $1 \leq i \leq j-1$. It is clear that $\mathcal{C} = \{S_1, S_2, \ldots, S_n\}$ partitions G such that the weak diameter of each S_i is at most $16\rho r \ln r$.

Now we state a key lemma from Krauthgamer and Lee [14] that allows us to use Rao's technique. For any $u \in V$ and $x \geq 0$, let $\mathcal{E}_u^x$ denote the event that $B(u, x)$ is split between multiple clusters in $\mathcal{C}$.

Lemma 8 *Let $u \in V$, $r \geq 16\rho$, and $x \geq 0$. Then $\mathbf{Pr}[\mathcal{E}_u^x] \leq 10x/r$.*

The implication of this lemma is that for any vertex u, with probability at least a constant, a "large" ball centered at u is completely contained in one of the clusters in $\mathcal{C}$. This implication can be stated more precisely, as follows. For each cluster $C \in \mathcal{C}$, define the *boundary of* C, denoted ∂C, as the subset of vertices in C that have neighbors (in G) outside of C. Let B denote the set of vertices that are in the boundary of some cluster in $\mathcal{C}$. Define a *δ-good vertex* to be a vertex that is at least δ hops away from any vertex in B. From Lemma 8 we can immediately derive the following.

Lemma 9 *Let $u \in V$ and $r \geq 16\rho$. With probability at least $\frac{1}{2}$, u is $\frac{r}{20}$-good.*

The above lemma (which is similar to Lemma 3 in Rao [27]) leads to the second stage of the algorithm. Consider the graph $G - B$. For each connected component Y in $G - B$, pick a *rate* α independently and uniformly at random from the interval $[1, 2]$. To each edge in Y, assign as weight the rate α corresponding to Y. Thus all edges in a connected component in $G - B$ have the same weight. To all other edges in G (i.e., those that are incident on vertices in B) assign the weight 0. Finally, to each vertex u in G assign a weight that is the length of a shortest path from u to a vertex in B. Note that the shortest paths are computed in the weighted version of G. Thus all vertices in B will get assigned a weight 0, whereas vertices in $G - B$ will get assigned a positive weight. In fact, it is easy to verify that the weight assigned to a vertex in $G - B$ is at least 1 and at most $16\rho r \ln r$. Exactly, as in Rao [27], we can derive the following lemma.

Lemma 10 *The weight of any δ-good node ranges uniformly over an interval I of length at least δ. Moreover, the choice is independent of anything in a different component.*

As mentioned before, for each vertex v, the weight of v forms one coordinate of v. For each value of r, the above algorithm is repeated $\alpha k \log n$ times for a large enough constant α. Since Lemma 9 holds only for values of $r \geq 16\rho$, we repeat this the entire process for $r = 1, 16\rho, (16\rho)^2, \ldots, diam(G)$. For any constant ρ, this is essentially the same as using values of $r = 1, 2, 4, \ldots, diam(G)$ as Rao does [27]. Lemmas 9 and 10 are the two key results needed for the rest of Rao's analysis [27] to go through. Therefore, we get the following result.

Theorem 2 *There is a polynomial time algorithm that constructs a $(k, O(\sqrt{\log n}))$-volume respecting embedding, with high probability, of any growth-restricted graph.*

3.1 Constant Quality Embedding in Constant Dimensions

Levin, together with Linial, London, and Rabinovich [21], made a conjecture (Conjecture 8.2 in [21]) that is quite relevant to the best quality embedding problem. Let $\mathbb{Z}^d_\infty$ 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 $\dim(G)$ to be the smallest d such that G occurs as a (not necessarily induced) subgraph of $\mathbb{Z}^d_\infty$.

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 $\mathbb{Z}^{O(\rho_G)}_\infty$. In other words, $dim(G) = O(\rho_G)$.*

Linial [20] 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 $\{u, v\} \in V$.

Lee and Krauthgamer [14] 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 [14]. This proof relies on the Lovász Local Lemma, but can be turned into a polynomial time algorithm using standard techniques for turning proofs based on the Lovász Local Lemma into algorithms. 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 d-dimensional UBG and constructs an embedding of quality-2 in $O(d \log d)$ dimensional Euclidean space.*

4 Embedding into 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 [28].

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

Step 2. Let $V(H) = \{v_1, v_2, \ldots, v_n\}$. 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 $i = 1, 2, \ldots, n$.

Step 3. Choose 2 random lines, ℓ_1 and ℓ_2 (independently), passing through the origin. Project the point set $\{u_1, u_2, \ldots, u_n\}$ 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 dimension $1/\sqrt{n} \times 1/\sqrt{n}$. Call each such grid cell an an *outer* grid cell. Let M be the maximum number of points w_i that fall in any cell after the random projection step. Subdivide each outer grid cell into a finer grid, with each cell having dimensions $1/\sqrt{n \cdot M} \times 1/\sqrt{n \cdot M}$. For each outer cell C, map all points that fall into C to grid points of the finer grid. Finally, scale up all grid points by a factor of $\sqrt{n \cdot M}$ along both dimensions so that each 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 [28] and the proof of Lemma 13 is included mostly for the sake of completeness. An interesting aspect of our analysis is that a key technical lemma (Lemma 11), that was proved using "spreading constraints" by Vempala [28], follows quite easily, simply from the fact that we are working with a growth-restricted graph.

The key technical lemma, that we prove below, gives an upper bound on the sum of inverse squares of distances from a vertex. In [17] and in [28], this lemma followed from explicitly imposed "spreading constraints." In our case, the lemma follows easily from the fact that the cluster graph H is growth-restricted.

Lemma 11 *There is a constant β such that for any $v \in V(H)$, and any $S \subset V(H)$, $\sum_{u\in S} \frac{1}{d^2(v,u)} \leq \beta \cdot \log |S|$.*

Proof: Lemma 7 tells us that for some constant α, $|B(v,r)| \leq \alpha \cdot r^2$. The sum $\sum_{u\in S} \frac{1}{d^2(v,u)}$ 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 $i = 1, 2, \ldots$. Therefore,

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

Hence, for some constant β, $\sum_{u\in S} \frac{1}{d^2(v,u)} \leq \beta \cdot \log |S|$. □

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 [17].

Lemma 12

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

Here β is the constant that appears in previous lemma.

Lemma 13 *After Step (4), with high probability, the maximum number of vertices that fall in an outer grid cell is $O\left(\log^4 n\right)$.*

Proof: Consider an outer grid cell C. 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 cell C along 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.

$$\begin{aligned}
\mathbf{E}[N_C] = \sum_{S\subset V(H),|S|=k} \mathbf{E}[X_S^1 \cdot X_S^2] &= \sum_{S\subset V(H),|S|=k} \mathbf{E}[X_S^1]\mathbf{E}[X_S^2] && \text{(by independence of } X_S^1 \text{ and } X_S^2\text{)} \\
&= \sum_{S\subset V(H),|S|=k} \mathbf{Pr}[X_S^1 = 1]^2 \\
&\leq \gamma \sum_{S\subset V(H),|S|=k} \left(\frac{W^{k-1} n^{\frac{k-1}{2}}}{(k-1)!\mathrm{Evol}(S)}\right)^2 && \text{(for constant } \gamma\text{, using Lemma 3)} \\
&\leq \gamma \sum_{S\subset V(H),|S|=k} \left(\frac{1}{(k-1)!\mathrm{Evol}(S)}\right)^2 && \text{(since } W = 1/\sqrt{n}\text{)} \\
&\leq \gamma \sum_{S\subset V(H),|S|=k} \left(\frac{\left(\delta\sqrt{\log n}\right)^{k-1}}{(k-1)!\mathrm{Vol}(S)}\right)^2 && \text{(for constant } \delta\text{, using Theorem 2)} \\
&\leq \gamma \sum_{S\subset V(H),|S|=k} \left(\frac{\left(\delta\sqrt{\log n}\right)^{k-1} \cdot 2^k}{\mathrm{Tvol}(S)}\right)^2 && \text{(Lemma 1)} \\
&\leq \gamma \cdot \delta^{2(k-1)} \cdot (4\log n)^k \cdot \sum_{S\subset V(H),|S|=k} \frac{1}{\mathrm{Tvol}^2(S)} \\
&\leq \gamma \cdot \delta^{2(k-1)} \cdot 8k! \cdot \beta^k \cdot n \cdot \log^{2k} n \leq n \cdot (\zeta \cdot k \cdot \log^2 n)^k && \text{(for constant } \zeta\text{, by Lemma 12)}
\end{aligned}$$

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

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

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 $k(n \cdot (\zeta \cdot k \cdot \log^2 n)^k)^{1/k}$ points. Since the value of $k = \log n$, using the fact that $n^{1/\log n} = O(1)$, we get that with high probability, every outer cell has at most $O(\log^4 n)$ points. □

Lemma 14 *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 $1 - 1/n$. Thus, each edge $\{v_i, v_j\}$ spans at most $4\sqrt{\log n}$ outer grid cells along each of the two dimensions of the grid. Lemma 13 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), will cause each edge to have length at most $O(\log^{2.5} n)$, with high probability. □

Theorem 4 (Main Result) *With high probability, the quality of the embedding is $O(\log^{2.5} n)$.*

Proof: Follows immediately from the 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. □

5 Conclusions

It seems as though the techniques used in this paper may not be able to provide an embedding of quality better than polylogarithmic in n. So significantly new techniques may have to be developed in order to obtain a constant-quality 2-dimensional embedding. Any of these techniques would probably benefit from starting with the growth-restricted approximation, described in this paper.

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 best quality embedding problem, without sacrificing the quality of the embedding. We have made some progress towards 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 [15, 16]. As future work, we intend to investigate possible distributed implementations of Rao's algorithm [27] for constructing a volume-respecting embedding and Vempala's random projection step [28].

References

[1] James Aspnes, David Kiyoshi Goldenberg, and Yang Richard 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.

[2] Amitabh Basu, Jie Gao, Joseph S. B. Mitchell, and Girishkumar 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.

[3] Pratik Biswas and Yinyu 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.

[4] Avrim Blum and David Karger. An $\tilde{O}(n^{3/14})$-coloring algorithm for 3-colorable graphs. *Information Processing Letters*, 61(1):49–53, 1997.

[5] Avrim Blum, Goran Konjevod, R. Ravi, and Santosh 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.

[6] 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.

[7] Heinz Breu and David G. Kirkpatrick. Unit disk graph recognition is NP-hard. *Comput. Geom. Theory Appl.*, 9(1-2):3–24, 1998.

[8] Jehoshua Bruck, Jie Gao, and Anxiao (Andrew) 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.

[9] N. Bulusu, J. Heidemann, and D. Estrin. GPS-less low cost outdoor localization for very small devices, 2000.

[10] 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.

[11] Uriel Feige. Approximating the bandwidth via volume respecting embeddings. *J. Comput. Syst. Sci.*, 60(3):510–539, 2000.

[12] Brad 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.

[13] Philip Klein, Serge A. Plotkin, and Satish Rao. Excluded minors, network decomposition, and multicommodity flow. In *STOC '93: Proceedings of the twenty-fifth annual ACM symposium on Theory of computing*, pages 682–690, New York, NY, USA, 1993. ACM Press.

[14] Robert Krauthgamer and James R. Lee. The intrinsic dimensionality of graphs. In *STOC '03: Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pages 438–447, New York, NY, USA, 2003. ACM Press.

[15] Fabian Kuhn, Thomas Moscibroda, Tim Nieberg, and Roger Wattenhofer. Fast Deterministic Distributed Maximal Independent Set Computation on Growth-Bounded Graphs. In *19th International Symposium on Distributed Computing (DISC), Cracow, Poland*, September 2005.

[16] Fabian Kuhn, Thomas Moscibroda, Tim Nieberg, and Roger 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.

[17] Fabian Kuhn, Thomas Moscibroda, Regina O'Dell, Mirjam Wattenhofer, and Roger Wattenhofer. Virtual coordinates for ad hoc and sensor networks. To appear in *Algorithmica*, 2006.

[18] Fabian Kuhn, Thomas Moscibroda, and Roger 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.

[19] Fabian Kuhn, Roger Wattenhofer, Yan Zhang, and Aaron 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.

[20] N. Linial. Variation on a theme of Levin. In J. Matoušek, editor, *Open Problems, Workshop on Discrete Metric Spaces and their Algorithmic Applications*, 2002.

[21] Nathan Linial, Eran London, and Yuri Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15:215–245, 1995.

[22] Thomas Moscibroda, Regina O'Dell, Mirjam Wattenhofer, and Roger 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.

[23] Radhika Nagpal, Howard E. Shrobe, and Jonathan 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.

[24] Dragos Niculescu and Badri 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.

[25] Vijay Raghavan and Jeremy 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.

[26] Ananth Rao, Christos Papadimitriou, Scott Shenker, and Ion 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.

[27] Satish Rao. Small distortion and volume preserving embeddings for planar and euclidean metrics. In *SCG '99: Proceedings of the fifteenth annual symposium on Computational geometry*, pages 300–306, New York, NY, USA, 1999. ACM Press.

[28] Santosh 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.