ℓ_2^2 Spreading Metrics For Vertex Ordering Problems (Extended Abstract)

Moses Charikar*
Princeton University

MohammadTaghi Hajiaghayi †
MIT

Howard Karloff ‡
AT&T Labs—Research

Satish Rao §
UC Berkeley

Abstract

We design approximation algorithms for the vertex ordering problems MINIMUM LINEAR ARRANGEMENT, MINIMUM CONTAINING INTERVAL GRAPH, and MINIMUM STORAGE-TIME PRODUCT, achieving approximation factors of $O(\sqrt{\log n}\log\log n)$, $O(\sqrt{\log n}\log\log n)$, and $O(\sqrt{\log T}\log\log T)$, respectively, the last running in time polynomial in T (T being the sum of execution times). The technical contribution of our paper is to introduce "ℓ_2^2 spreading metrics" (that can be computed by semidefinite programming) as relaxations for both undirected and directed "permutation metrics," which are induced by permutations of $\{1, 2, ..., n\}$. The techniques introduced in the recent work of Arora, Rao and Vazirani can be adapted to exploit the geometry of such ℓ_2^2 spreading metrics, giving a powerful tool for the design of divide-and-conquer algorithms. In addition to their applications to approximation algorithms, the study of such ℓ_2^2 spreading metrics as relaxations of permutation metrics is interesting in its own right. We show how our results imply that, in a certain sense we make precise, ℓ_2^2 spreading metrics approximate permutation metrics on n points to a factor of $O(\sqrt{\log n}\log\log n)$.

1 Introduction

We consider approximation algorithms for vertex ordering problems such as MINIMUM LINEAR ARRANGEMENT, MINIMUM STORAGE-TIME PRODUCT, and MINIMUM CONTAINING INTERVAL GRAPH. In these problems, the goal is to find a permutation of the vertices of a given graph or digraph $G(V, E)$ (i.e., a bijection σ between V and $\{1, 2, \ldots, |V|\}$, possibly satisfying some side constraints, so as to minimize some function of the permutation. In the MINIMUM LINEAR ARRANGEMENT problem, the input is an undirected graph and the goal is to minimize the sum, over edges $\{i, j\}$, of the weight of the edge times the "undirected permutation distance" $|\sigma(j) - \sigma(i)|$. In the (unit execution times special case of) MINIMUM STORAGE-TIME PRODUCT, the input is a directed acyclic graph and the goal is to find a permutation such that $\sigma(i) < \sigma(j)$ if $(i, j) \in E$, so as to minimize the sum, over arcs (i, j), of the weight of the arc times $\sigma(j) - \sigma(i) + 1$.(More generally, each job has an execution time, and the goal is to minimize the sum, over arcs (i, j), of the weight of (i, j) times the sum of the execution times of jobs falling at or after i and at or before j in σ.) MINIMUM CONTAINING INTERVAL GRAPH is the problem of finding the smallest interval supergraph of a given undirected graph. Somewhat surprisingly (see Section 4), an equivalent goal is to find a permutation σ of V minimizing the sum over nodes i of $\max_{j:\{i,j\}\in E}\{\max\{\sigma(j) - \sigma(i), 0\}\}$.

The best previously known approximation factors for these problems were obtained by Rao and Richa [15]. They obtained $O(\log n)$ approximations for MINIMUM LINEAR ARRANGEMENT and MINIMUM CONTAINING INTERVAL GRAPH (where n denotes the number of vertices in the graph) and a $O(\log T)$ approximation for MINIMUM STORAGE-TIME PRODUCT (where T denotes the sum of processing times of the jobs). We improve these to $O(\sqrt{\log n}\log\log n)$ for MINIMUM LINEAR ARRANGEMENT and MINIMUM CONTAINING INTERVAL GRAPH and to $O(\sqrt{\log T}\log\log T)$ for MINIMUM STORAGE-TIME PRODUCT (the last algorithm running in time polynomial in T).

It is convenient to make two definitions.

DEFINITION 1.1. *A* permutation metric (X, d) *is a metric on the set X for which there is a bijection $\sigma : X \to \{1, 2, \ldots, |X|\}$ such that for all $x, y \in X$, $d(x, y) = |\sigma(x) - \sigma(y)|$.*

DEFINITION 1.2. *A* directed permutation metric (X, d) *is a function on the set X for which there is a bijection $\sigma : X \to \{1, 2, \ldots, |X|\}$ such that for all $x, y \in X$,*

*Email: moses@cs.princeton.edu. Supported by NSF ITR grant CCR-0205594, DOE Early Career Principal Investigator award DE-FG02-02ER25540, NSF CAREER award CCR-0237113 and an Alfred P. Sloan Fellowship.

†Email: hajiagha@theory.csail.mit.edu.

‡Email: howard@research.att.com.

§Email: satishr@cs.berkeley.edu. Partially supported by NSF award CCR-0105533.

$d(x, y) = \max\{\sigma(y) - \sigma(x), 0\}$ *(and so is not really a metric).*

The technical contribution of the paper is to introduce a class of relaxations for permutation metrics, which we call "ℓ_2^2 spreading metrics." Given any of these optimization problems, we can find a (near) optimal ℓ_2^2 spreading metric by solving an SDP. We use ideas from the recent work of Arora, Rao and Vazirani [2] to exploit the geometry of these ℓ_2^2 spreading metrics. This gives us powerful tools to design divide-and-conquer algorithms for these optimization problems. We also show how ideas from the recent work of Agarwal, Charikar, Makarychev and Makarychev [1] can be used to find *directed* spreading metrics from configurations of vectors produced by the SDP. We show that these tools can be used together with the algorithmic framework in Rao and Richa [15]. Independently of our work, Feige and Lee [6] obtained similar results for MINIMUM LINEAR ARRANGEMENT.

The study of such ℓ_2^2 spreading metrics as relaxations of permutation metrics is interesting in its own right. We formalize this connection and show how our results imply that ℓ_2^2 spreading metrics on n points are $O(\sqrt{\log n} \log \log n)$ approximable by convex combinations of permutation metrics.

1.1 Related work. Given a graph $G = (V, E)$, we are often interested in finding a small edge separator whose removal from the graph leaves two sets of vertices of roughly equal size (say, of size at most $2|V|/3$), with no edges between these two sets. Balanced separators of small size are important in several contexts, e.g., [12, 10, 3]. Perhaps more importantly, finding balanced separators of small size is a major primitive for many graph algorithms, and in particular, for those that are based on the divide-and-conquer paradigm [13, 3, 11].

The approximation ratio of $O(\log n)$ for balanced separators is based on the work of Leighton and Rao [11]. They presented an algorithm based on linear programming that (pseudo-)approximates the minimum edge separator within a ratio of $O(\log n)$. Among other applications, this provides a $O(\log^2 n)$ approximation for MINIMUM FEEDBACK ARC SET (in directed graphs) and an $O(\log^2 n)$-approximation algorithm for MINIMUM CUT LINEAR ARRANGEMENT. Using the ideas of Leighton and Rao, Hanson [8] presented a $O(\log^2 n)$-approximation algorithm for MINIMUM LINEAR ARRANGEMENT and even more generally for the problem of graph embeddings in d-dimensional meshes. Ravi et al. [16] further extended these polylogarithmic approximation algorithms to MINIMUM STORAGE-TIME PRODUCT and MINIMUM CONTAINING INTERVAL GRAPH.

Seymour [17] pioneered a new divide-and-conquer graph-decomposition approach, not based on balanced cuts, for directed graphs. More specifically, he gave a $O(\log n \log \log n)$-approximation algorithm for MINIMUM FEEDBACK ARC SET. Even, Naor, Rao, and Schieber [5] extended the approach used by Seymour to obtain polynomial-time, $O(\log n \log \log n)$-approximation algorithms for MINIMUM LINEAR ARRANGEMENT and MINIMUM CONTAINING INTERVAL GRAPH and an $O(\log T \log \log T)$-approximation algorithm for MINIMUM STORAGE-TIME PRODUCT. In fact, they showed similar approximation results for a broader class of graph optimization problems which satisfy their *approximation paradigm*, i.e., problems for which there exists a *spreading metric* and which can be handled by their divide-and-conquer approach. Intuitively, a spreading metric on a graph is an assignment of lengths to either its edges or vertices so that subgraphs for which the optimization problem is nontrivial are spread apart in the associated metric space. In a later paper, Even, Naor, Rao, and Schieber [4] extended the spreading metric techniques to graph partitioning problems and obtain simpler recursions that yield a logarithmic approximation factor for balanced cuts and multiway separators (and not the other problems in [5]). Rao and Richa [15] improved the approximation factor to $O(\log n)$ for MINIMUM LINEAR ARRANGEMENT and MINIMUM CONTAINING INTERVAL GRAPH and to $O(\log T)$ for MINIMUM STORAGE-TIME PRODUCT. Furthermore, Rao and Richa [15] showed that if the graph is planar (or, more generally, if the graph excludes K_r as a minor, for a fixed r), then we can obtain a $O(\log \log n)$-approximation factor for MINIMUM LINEAR ARRANGEMENT, using a variation of the algorithm for the general case. An example due to Alon and Seymour [17] shows that there exists a logarithmic gap between the spreading metric volume and the true optimal value for certain instances of MINIMUM LINEAR ARRANGEMENT; hence the result of Rao and Richa [15] gives a tight bound (up to constants) between the spreading metric volume and the true optimal value of this problem in general graphs.

2 Minimum Linear Arrangement

The MINIMUM LINEAR ARRANGEMENT problem is the following: Given an undirected multigraph $G(V, E)$ on some ordered set V having n nodes, and nonnegative weights $w(x, y) \in \mathbf{N}$ for all edges $\{x, y\} \in E$, the goal is find a permutation $\sigma : V \to \{1, 2, \ldots, n\}$ that minimizes

$$\sum_{\{x,y\} \in E, x<y} w(x, y)|\sigma(y) - \sigma(x)|.$$

Via the scaling and rounding technique used by [15, 5], at the cost of only negligibly increasing the approximation ratio, we can and will reduce the case of general binary weights to the case of a multigraph in which each edge has weight 1. Hence, for the sequel, we will take the input to consist of an unweighted multigraph, though our notation, e.g., "$\{x, y\} \in E$," won't necessarily reflect the fact that parallel edges are allowed.

Roughly speaking, our approach for MINIMUM LINEAR ARRANGEMENT is as follows. First, we write a vector (semidefinite) program which is similar to the linear program written by Even et al. [5], enhanced by the addition of the so-called "triangle-inequality" constraints. For this SDP, we are able to use the seminal results of [2] (and the subsequent paper by Lee [9]), which can be viewed as a line embedding, to show that the solution of the semidefinite relaxation can be rounded to a solution to MINIMUM LINEAR ARRANGEMENT whose cost is larger by at most a factor of $O(\sqrt{\log n}\log\log n)$. In our rounding algorithm, we exploit a technique similar to that of Rao and Richa [15] for obtaining a $O(\log\log n)$-approximation algorithm on graphs excluding K_r as a minor, for fixed r. Our algorithm proceeds in rounds and in each round we have a "cut step," which corresponds to the series of cuts performed during the round, and a "recursive step," which consists of recursing on the connected components that result from the cut step. As in Rao and Richa's algorithm, our algorithm uses the divide-and-conquer technique of Seymour [17], which (roughly) makes a tradeoff between the depth and the breadth of the recursion. However, in Rao and Richa's algorithm the structure of graphs excluding K_r as a minor plays a crucial role and we need to replace these parts by new parts from [2] appropriate for our application.

More precisely, we consider the following semidefinite programming problem for MINIMUM LINEAR ARRANGEMENT on a graph $G = (V, E)$, $V = \{1, 2, ..., n\}$: Find $d(x, y)$ satisfying

$$d(x,y) = ||v_x - v_y||_2^2 \qquad (2.1)$$

for all x, y, for some vectors $v_1, v_2, ..., v_n$ in $\mathbf{R}^n$, so as to minimize

$$\sum_{\{x,y\}\in E, x<y} d(x,y),$$

subject to the constraints that

$$\sum_{y\in S} d(x,y) \geq (|S|^2-1)/4 \quad \forall S\subseteq V, x\in S, \qquad (2.2)$$

$$d(x,z) \leq d(x,y) + d(y,z) \quad \forall x,y,z\in V. \qquad (2.3)$$

Recall that d is called an ℓ_2^2*-metric* if, as here, there are vectors $v_1, v_2, ..., v_n$ such that $d(x,y) = ||v_x - v_y||_2^2$ *and* d satisfies the triangle inequality. Hence this SDP merely asks one to find an ℓ_2^2-metric which minimizes the sum of edge lengths subject to a "spreading constraint": for all $S \subseteq V$ and $x \in S$, $\sum_{y\in S} d(x,y) \geq (|S|^2-1)/4$.

Here is why (2.1),(2.2),(2.3) is a relaxation. A feasible solution to MINIMUM LINEAR ARRANGEMENT on an n-node graph is a bijection $\sigma : V \to \{1, 2, ..., n\}$. Take any feasible solution σ. For vertex x, with $\sigma(x) = i$, define v_x to be half the vector which has $+1$'s in positions $1, 2, ..., i$ and -1's in positions $i+1, i+2, ..., n$.[1] Then define $d(x,y) = ||v_x - v_y||_2^2$. If $\sigma(x) = i$ and $\sigma(y) = j$, then $d(x,y) = (1/4)(4|i-j|) = |i-j|$. It follows that $\sum_{\{x,y\}\in E, x<y} d(x,y)$ is exactly the cost of the linear arrangement σ. Now, given a set Z of distinct integral points on the real line, the sum of the distances between any one point z in Z and the others is at least $(|Z|^2-1)/4$ (a tight bound when $Z = \{1, 2, ..., |Z|\}$, $|Z|$ odd, and $z = (|Z|+1)/2$). Hence (2.2) is valid. Clearly (2.3) is valid.

[1] We also could use 0's and 1's in the construction of v_x.

Motivated by the linear program in [5], our exponentially large SDP is solvable in polynomial time, for there is a separation oracle. Indeed, it's the same separation oracle used in [5]: For each vertex x, order the vertices in order of increasing distance from x. For each $l = 1, 2, ..., n$, determine if the set S consisting of the l closest vectors to x satisfies (2.2). If for some x and l, the constraint is violated, we have a violated constraint; otherwise, there is no violated constraint at all.

Let $sdpopt(G)$ be the infimum of the objective function value of all semidefinite matrices which exactly satisfy all the constraints. Then in time polynomial in the number of bits in the input and $\lg(1/\epsilon)$, one can find a positive semidefinite matrix which exactly satisfies the constraints the value of the objective function of which is within ϵ of $sdpopt(G)$. (However, any algorithm which finds the vectors $\{v_x\}$ must introduce further error, since, given a positive semidefinite matrix $A = (a_{xy})$, there are sometimes no rational vectors $\{v_x\}$ such that $v_x \cdot v_y = a_{xy}$.)

Fix any feasible solution d and $\{v_x\}$ to the SDP and let w be its value. We will use a lemma that follows from arguments in [2] (and improvements in [9]) about ℓ_2^2 metrics. It refers to the following definition.

DEFINITION 2.1. *Given a metric* (S, d)*, a* Δ-separated pair *is a pair of sets* $X, Y \subseteq S$ *such that for all* $x \in X$ *and* $y \in Y$*,* $d(x,y) \geq \Delta$.

LEMMA 2.1. *There are real* $c, \epsilon > 0$ *and a polynomial-time algorithm that, given a* k*-point* ℓ_2^2 *metric,* $k \geq 2$*, given by vectors* $\{v_x : x \in S\}$ *satisfying (2.1), (2.2), and (2.3), produces a* Δ*-separated pair of sets* X *and* Y*,* Δ *integral, where* $\Delta \geq c(k/\sqrt{\lg k})$ *and* $|X|, |Y|$ *are at least* ϵk.

We only sketch the ideas in the proof, deferring the proof to the full version of the paper.

Sketch. For every point $x \in S$, the spreading constraints ensure that there exist at least $k/2$ points y with $d(x,y)$ at least $k/8$. Thus, at least $1/2$ of the ordered pairs (x,y) of points in S have $d(x,y)$ at least $k/8$. We pick random vector u whose coordinates are independent normal $N(0,1)$ random variables, and consider the "projections" $\{v_x \cdot u\}$, i.e., think of this as a placement of vertices on the line with point x at position $v_x \cdot u$. ("Projection" is in quotes because u is not a unit vector.) With constant probability, there exists an interval of length $\Omega(\sqrt{k})$ with $\Omega(k)$ points to the left of it (call this set L) and $\Omega(k)$ points to the right of it (call this set R). We now delete pairs of points (x, y) such that $x \in R, y \in L$ and $||v_x - v_y||_2^2 < k/(c\sqrt{\lg k})$ for some large constant c. These are pairs that are "stretched" by factor $\Omega(\log^{1/4} k)$ in the projection. Now the arguments in [2] (see Definition 4) and the strengthening in [9] (implied by [9, Theorem 4.1]) show that the number of disjoint stretched pairs cannot be larger than a small constant fraction of the number of nodes with high probability. This then implies that a constant fraction of L and R survive the deletion phase with constant probability. This in turn yields the well-separated sets claimed in the lemma. ∎

Our algorithm follows the algorithmic framework developed by Rao and Richa [15]; in fact, this is the framework they use to design a $O(\log \log n)$-approximation algorithm for MINIMUM LINEAR ARRANGEMENT on n-node planar graphs. Let $N = \lfloor 2/\epsilon \rfloor$ and C be some real exceeding $32/c$, c, ϵ as defined in Lemma 2.1.

Given the procedure described in Lemma 2.1, we describe a recursive algorithm for MINIMUM LINEAR ARRANGEMENT as follows. Take any vectors $\{v_x\}$ which are feasible for the SDP; let w be the value of the SDP when those vectors are used. (We construct the vectors $v_1, v_2, ..., v_n$ only once, at the beginning.) Procedure MLA can recurse in two different ways. If there is a small (edge) cut, the algorithm breaks the graph up into two pieces. Otherwise, it breaks the graph up into *many* (not just three or more) pieces.

Set $S = V$ and call $MLA(S)$.

Procedure $MLA(S)$**:**

1. Let $k = |S|$. If $k \leq N$ or if the subgraph induced by S has no edges, output any linear arrangement and halt.

2. Use the k vectors $\{v_x : x \in S\}$ and Lemma 2.1 to produce a Δ-separated pair of sets X and Y, Δ integral and $\Omega(k/\sqrt{\log k})$, with $|X|$ and $|Y|$ both $\Omega(k)$.

3. For each $\delta \in \{0, 1, 2, ..., \Delta - 1\}$, define $N_\delta(X) = \{y \in S : \exists x \in X \text{ such that } d(x, y) \leq \delta\}$ and $E_\delta(X)$ to be the set of edges between $N_\delta(X)$ and $S - N_\delta(X)$.

   (The $E_\delta(X)$ are probably not pairwise disjoint. We prove in Lemma 2.3 that the sum of $|E_\delta(X)|$ over $\delta = 0, 1, 2, ..., \Delta - 1$ is at most twice the value w of the SDP.)

4. Check if $|E_\delta(X)|$ is at most $w/(\Delta \lg k)$ for any δ in $\{0, 1, 2, ..., \Delta - 1\}$. If so, produce two subproblems, one consisting of the subgraph induced by $N_\delta(X)$, the other consisting of the subgraph induced by the remaining nodes $S - N_\delta(X)$. Recursively order the nodes in the two subproblems and output the concatenation of the orderings (and halt).

5. Otherwise, group the $\delta \in \{0, 1, 2, ..., \Delta - 1\}$ according to the value of $|E_\delta(X)|$. Define
   (2.4)
   $$D_i = \{\delta : 2^i \left\lceil \frac{w}{\Delta \lg k} \right\rceil < |E_\delta(X)| \leq 2^{i+1} \left\lceil \frac{w}{\Delta \lg k} \right\rceil\}$$
   for $i = 0, 1, 2, 3, ...$ (We start at $i = 0$ because step 4 failed.) Among those $i \in \{0, 1, 2, ..., 1 + \lceil \lg \lg k \rceil\}$, let I be the value of i with the largest D_i.

6. Let $D_I = \{\delta_1, \delta_2, \delta_3, ..., \delta_s\}$, $\delta_1 < \delta_2 < \delta_3 < \cdots < \delta_s$. Produce subproblems corresponding to the subgraphs induced by $Z_1 := N_{\delta_1}(X), Z_2 := N_{\delta_2}(X) - N_{\delta_1}(X), Z_3 := N_{\delta_3}(X) - N_{\delta_2}(X), ..., Z_s := N_{\delta_s}(X) - N_{\delta_{s-1}}(X)$ and $Z_{s+1} := S - N_{\delta_s}(X)$. Let Z be the disjoint union of the subgraphs induced by $Z_1, Z_2, ..., Z_{s+1}$.

   Recursively call $MLA(Z)$ and rearrange the output placement, if necessary, so that the placement of Z_i uses $|Z_i|$ nodes and occurs immediately to the left of that of Z_{i+1}.

End

LEMMA 2.2. *If* $\{x, y\} \in E$, *and* $i \leq j$ *such that* $x \in N_i(X), y \notin N_{j-1}(X)$, *then* $d(x, y) \geq (1/2)(j - i)$.

Proof. If $d(x, y) \leq j - 1 - i$, then $x \in N_i(X)$ and (2.3) imply that $y \in N_{i+(j-1-i)}(X) = N_{j-1}(X)$, a contradiction. Therefore $d(x, y) > j - i - 1$. Also, $d(x, y) \geq 3/4$ by (2.2) with $S = \{x, y\}$. Hence $d(x, y) > (1/2)(j - i)$. ∎

LEMMA 2.3. $\sum_{\delta=0}^{\Delta-1} |E_\delta(X)| \leq 2w$.

Proof. $\sum_{\delta=0}^{\Delta-1} |E_\delta(X)| = \sum_{e=\{x,y\} \in E, x<y} |\{\delta :$ e crosses cut $(N_\delta(X), S - N_\delta(X))| \leq \sum_{e=\{x,y\} \in E, x<y} 2d(x, y)$, by Lemma 2.2, which equals $2w$. ∎

LEMMA 2.4. $\sum_{i=0}^{1+\lceil \lg \lg k \rceil} |D_i| \geq \Delta/2$.

Proof. Consider $\delta \in \{0, 1, 2, ..., \Delta - 1\}$ and choose i such that $\delta \in D_i$. $|E_\delta(X)| \geq 2^i w/(\Delta \lg k)$. If $i \geq 2 + \lceil \lg \lg k \rceil$, then $|E_\delta(X)| \geq 4w/\Delta$. Call δ *bad* if the corresponding i is at least $2 + \lceil \lg \lg k \rceil$, and *good* otherwise. By Lemma 2.3, $2w \geq \sum_{\delta=0}^{\Delta-1} |E_\delta(X)| \geq \sum_{\text{bad } \delta} |E_\delta(X)| \geq (4w/\Delta)(\text{number of bad } \delta)$. Hence the number of bad δ is at most $\Delta/2$. Hence $\Delta - \Delta/2 \leq$ number of good $\delta = \sum_{i=0}^{1+\lceil \lg \lg k \rceil} |D_i|$. ∎

The following corollary follows easily from Lemma 2.4.

COROLLARY 2.1. *The value* I *chosen in step 5 satisfies* $|D_I| \geq (\Delta/2)/(2 + \lceil \lg \lg k \rceil) \geq \Delta/(4 \lg \lg k)$ *for sufficiently large* k.

DEFINITION 2.2. *Say an edge is* removed *if it appears in the subproblem on S but not in any of the recursively generated subproblems.*

LEMMA 2.5. *Suppose $e = \{x, y\}$ is a removed edge, with $x \in N_{\delta_l}(X)$ and $y \notin N_{\delta_{m-1}}(X)$, $l < m$. Then $d(x, y) \geq (1/2)(m - l)$.*

Proof. The proof of Lemma 2.2 shows that $d(x, y) > \delta_{m-1} - \delta_l$. However, $\delta_1 < \delta_2 < \cdots < \delta_s$ are distinct integers. It follows that $\delta_q - \delta_p \geq q - p$ if $p \leq q$. Hence $d(x, y) > (m-1) - l$. Since every edge has length at least $3/4$, we conclude that $d(x, y) \geq \max\{3/4, (m-l)-1\} \geq (1/2)(m - l)$. ■

Now Corollary 2.2 follows trivially:

COROLLARY 2.2. *For any edge $\{x, y\}$, $d(x, y)$ is at least half the number of j such that x or y is in $N_{\delta_j}(X)$ and the other is not.*

For the analysis alone, it helps to work with integral upper bounds on the distances, in lieu of the actual real-valued distances. For any nonnegative integers k, l, let $C(k, l)$ be the maximum cost of a linear arrangement produced by our algorithm when run on (possibly disconnected) graphs $G = (V, E)$ with at most k vertices and a given feasible vector solution $\{v_x\}$ such that $d(x, y) = ||v_x - v_y||_2^2$ satisfies $\sum_{\{x,y\} \in E, x<y} \lceil d(x, y) \rceil \leq l$. Note that $\lceil d(x, y) \rceil \leq 2d(x, y)$, since $d(x, y) \geq 3/4$ if $x \neq y$.

THEOREM 2.1. *There is a constant C such that $C(k, l) \leq l(C\sqrt{\lg k} \lg \lg k)$ for all $l \geq 0$ and for all $k \geq 3$.*

Proof. By induction on $k + l$. The basis, the details of which we omit, is the case in which $k + l \leq N$. Let k, l be arbitrary positive integers. Let G be any graph with more than k vertices and let $\{v_x\}$ be a feasible vector solution with $\sum_{\{x,y\} \in E, x<y} \lceil d(x, y) \rceil \leq l$.

Notice that we can recurse in two possible ways, in steps 4 and 6.

If we recurse in step 4, we generate two graphs G_1 and G_2, G_i having k_i vertices, $k_1 + k_2 = k$, and $k_1, k_2 \leq (1 - \epsilon)k$, because the first subproblem (V_1, E_1) avoids Y and the second problem (V_2, E_2), X. Furthermore, the given vectors induce feasible solutions to the two problems. Since $\sum_{\{x,y\} \in E, x<y} \lceil d(x, y) \rceil \leq l$, if $l_1 = \sum_{\{x,y\} \in E_1, x<y} \lceil d(x, y) \rceil$ and $l_2 = \sum_{\{x,y\} \in E_2, x<y} \lceil d(x, y) \rceil$, then $l_1 + l_2 \leq l$. By induction, the costs of the solutions returned by the two subproblems are at most $C(k_1, l_1)$ and $C(k_2, l_2)$. We conclude that the overall cost of the placement of G obtained by our algorithm is at most $C(k_1, l_1) + C(k_2, l_2) + C'(l/\sqrt{\lg k})$ for a constant $C' > 0$, where $k_1, k_2 \leq (1 - \epsilon)k$ for a fixed $\epsilon > 0$, $k_1 + k_2 = k$, and $l_1 + l_2 \leq l$, by Lemma 2.1, induction, the fact that each of the at-most-$(w/(\Delta \lg k))$ removed edges has stretch at most k and $w \leq l$, and the fact that Δ is $\Omega(k/\sqrt{\lg k})$.

In step 6, we get a list of at least $\Delta/(4 \lg \lg k)$ cuts which partition the problem into subproblems, the jth having $k_j := |Z_j|$ vertices. Where I is the index defined in step 5, each of the cuts has (unweighted) size at least $\alpha := 2^I w/(\Delta \lg k)$. (Since $\alpha > 0$ (otherwise $w = 0$) and the cut size is integral, each cut has size at most 1, also.) Since each of the edges in the cuts is removed, by Corollary 2.2 the sum of the ceilings of the edge lengths in all the subproblems (i.e., the sum of $\lceil d(x, y) \rceil$ over edges $\{x, y\}, x < y$, in all subproblems) decreases by at least half the sum of the (unweighted) sizes of the cuts (even though they're not pairwise disjoint). Each cut removed has at least $\max\{1, \alpha\}$ edges, and there are at least $\Delta/(4 \lg \lg k)$ of them, by Corollary 2.1. Moreover, from Corollary 2.2 and the fact that Δ is at least $ck/\sqrt{\lg k}$ for some constant $c > 0$ (by Lemma 2.1), the decrease in weight is at least $(1/2) \max\{1, \alpha\} \cdot [(ck/\sqrt{\lg k})/(4 \lg \lg k)]$, which is at least $ck/(8(\lg \lg k)\sqrt{\lg k})$.

Now we have to bound from above the cost of embedding the removed edges. Each removed edge $e = \{x, y\}$ satisfies $x \in Z_l$, $y \in Z_m$, for some $l < m$, after possibly switching x and y. The cost in the final solution of embedding e (its "stretch") is at most $|Z_l| + |Z_{l+1}| + |Z_{l+2}| + \cdots + |Z_m| = k_l + k_{l+1} + k_{l+2} + \cdots + k_m \leq (k_l + k_{l+1}) + (k_{l+1} + k_{l+2}) + \cdots + (k_{m-1} + k_m)$. It follows that we can bound the overall cost of embedding these edges by the sum, over the cuts $(N_{\delta_j}(X), S - N_{\delta_j}(X))$, of the number of edges in the cut multiplied by $k_j + k_{j+1}$, i.e., by $\sum_{j=1}^{s} |E_{\delta_j}(X)|(k_j + k_{j+1}|$, which is at most $\sum_{j=1}^{s} (2\alpha)(k_j + k_{j+1})$ (by the upper bound in (2.4)), hence at most $(2\alpha)(2k) = 4k\alpha$.

The result is a (possibly disconnected) graph G' with exactly k vertices for which the sum of $\lceil d(x, y) \rceil$ over edges $\{x, y\}$, $x < y$, in G' is at most $\lceil l - (1/8)c(k\alpha)/((\lg \lg k)\sqrt{\lg k}) \rceil \geq 0$ and also at most $l - 1$, the latter since the decrease in weight is at least $ck/(8(\lg \lg k)\sqrt{\lg k}$, which far exceeds 1; our additional cost is at most $4k\alpha$. Thus, in this case, we infer that the cost of the placement of G obtained by our algorithm is at most $C(k, \min\{l-1, \lceil l - (1/8)c\alpha k/((\lg \lg k)\sqrt{\lg k}) \rceil\}) + 4k\alpha$.

Take now any graph G on, say, K vertices, along with vectors and $d(x, y)$'s satisfying $\sum_{\{x,y\} \in E, x<y} \lceil d(x, y) \rceil \leq L$, and such that $K > N$.

First, consider $T_2 := C(K, \min\{L - 1, \lceil L - (1/8)c\alpha K/((\lg \lg K)\sqrt{\lg K}) \rceil\}) + 4\alpha K$ for some $\alpha > 0$ and note that $K + \min\{L - 1, \lceil L - (1/8)c\alpha K/((\lg \lg K)\sqrt{\lg K}) \rceil\} \leq K + L - 1$. Use the fact that the "added" term $4\alpha K$ is $O(\sqrt{\lg K} \lg \lg K)$ times the decrease in the L term: $T_2 \leq (L - (1/8)c(\alpha K)/((\lg \lg K)\sqrt{\lg K}))C\sqrt{\lg K} \lg \lg K + 4\alpha K = LC\sqrt{\lg K} \lg \lg K - (1/8)c\alpha KC + 4\alpha K$. We want C to satisfy $-(1/8)c\alpha KC + 4\alpha K \leq 0$, which it does, since $C \geq 32/c$. Notice that the value of α doesn't matter; what matters is the factor by which the "added" term exceeds the decrease in w.

Second, consider $T_1 := C(K_1, L_1) + C(K_2, L_2) + C'L/\sqrt{\lg K}$ for $K_1, K_2 \leq (1-\epsilon)K$ and $L_1 + L_2 \leq L$; since for $i = 1, 2$, $K_i + L_i < K + L$ and since $K > N$ implies that both K_i's are at least 3, we can apply induction. Hence $T_1 \leq L_1 C\sqrt{\lg K_1}\lg\lg K_1 + L_2 C\sqrt{\lg K_2}\lg\lg K_2 + C'L/\sqrt{\lg K} \leq L_1 C\sqrt{\lg K_2}\lg\lg K_2 + L_2 C\sqrt{\lg K_2}\lg\lg K_2 + C'L/\sqrt{\lg K}$ (assuming without loss of generality that $K_1 \leq K_2$) $\leq LC\sqrt{\lg K_2}\lg\lg K_2 + C'L/\sqrt{\lg K} \leq LC\sqrt{\lg((1-\epsilon)K)}\lg\lg K + C'L/\sqrt{\lg K} = LC\sqrt{\lg K + \lg(1-\epsilon)}\lg\lg K + C'L/\sqrt{\lg K} \leq LC[\sqrt{\lg K} + (\lg(1-\epsilon))/(2\sqrt{\lg K})]\lg\lg K + C'L/\sqrt{\lg K}$ (because $\sqrt{a+b} \leq \sqrt{a} + b/(2\sqrt{a})$ if $a, a+b > 0$). That last expression is equal to $L(C\sqrt{\lg K}\lg\lg K) + LC(\lg\lg K)(\lg(1-\epsilon))/(2\sqrt{\lg K}) + C'L/\sqrt{\lg K} \leq L(C\sqrt{\lg K}\lg\lg K) + (L/\sqrt{\lg K})[(\lg(1-\epsilon))C(\lg\lg K)/2 + C']$, and it is easy to see that the bracketed quantity in the last equation is negative (by $K \geq 3$ and the definition of C). Now we just use the fact that $\lceil d(x,y)\rceil \leq 2d(x,y)$ to infer that $\sum_{\{x,y\}\in E, x<y}\lceil d(x,y)\rceil \leq 2w$. ■

The polynomiality of the algorithm relies on (1) the fact that recursing in step 4 of the algorithm leads to two subproblems with at most $(1-\epsilon)k$ vertices apiece; (2) the fact that each cut removed in step 6 has at least one edge; and (3) the fact that the input graph is unweighted.

3 Minimum Storage-Time Product

The MINIMUM STORAGE-TIME PRODUCT problem is the following: Let $G(V, E)$ be an acyclic directed graph with arc weights $w(e)$ and positive integral node weights $\tau(x)$. Every vertex represents a task and the weight $\tau(x)$ is the execution time for the task x. The weight w_e on arc $e = (x, y)$ represents the number of units of storage required to save intermediate results generated by task x until they are consumed at task y. MINIMUM STORAGE-TIME PRODUCT is the problem of finding a topological ordering of the tasks such that the total storage-time product of the process is minimized. In other words, the goal is to find a topological ordering of the nodes $\sigma : V \rightarrow \{1, \ldots, n\}$ that minimizes $\sum_{(x,y)\in E}\left(w(x,y)\sum_{z:\sigma(x)\leq\sigma(z)\leq\sigma(y)}\tau(z)\right)$

In order to present our SDP relaxation for this problem, we need to introduce our relaxation for directed permutation metrics. Consider a "directed metric" $d(x, y)$ satisfying the following conditions:

$$d(x,y) \geq 0 \quad \text{for all } x, y \in V$$

$$d(x,y) = \frac{1}{2}(||v_x - v_y||^2 - v_0 \cdot (v_x - v_y))$$

for all $x, y \in V$, for some v_0;

$$\sum_{y\in S}(d(x,y) + d(y,x)) \geq \frac{|S|^2 - 1}{4}$$

for all $S \subseteq V, x \in S$; and

$$d(x,z) \leq d(x,y) + d(y,z) \quad \text{for all } x, y, z \in V.$$

(The last condition implies, by adding $d(x,z) \leq d(x,y) + d(y,z)$ to $d(z,x) \leq d(z,y) + d(y,x)$, that $||v_x - v_z||^2 \leq ||v_x - v_y||^2 + ||v_y - v_z||^2$, for all $x, y, z \in V$.)

We refer to such a "metric" as a *directed ℓ_2^2 spreading metric*. We use such a metric as a relaxation of a directed permutation metric where distances measured from left to right are distances along the line, but distances measured from right to left are 0. Here is why a directed ℓ_2^2 spreading metric is a relaxation of a directed permutation metric. Take any mapping σ of the n vertices to distinct positions $\{1, 2, \ldots, n\}$ on the line. Consider the vector solution corresponding to σ, defined as before: for vertex x with $\sigma(x) = i$, define v_x to be half the vector which has $+1$'s in positions $1, 2, \ldots, i$ and -1's in positions $i+1, \ldots, n$. The vector v_0 is the vector with $+1$'s in all coordinates. If $\sigma(x) = i$ and $\sigma(y) = j$, then $d(x,y) = (1/2)(||v_x - v_y||^2 - v_0 \cdot (v_x - v_y)) = |i - j|$ if $i \leq j$ and $d(x,y) = 0$ if $i \geq j$.

Given a directed ℓ_2^2 spreading metric $d(x,y)$ given by vectors $\{v_x : x \in S \cup \{0\}\}$, we will refer to $||v_x - v_y||^2 = d(x,y) + d(y,x)$ as the *underlying* undirected (spreading) metric.

We prove a directed analogue of Lemma 2.1. This is an extension, to directed spreading metrics, of techniques developed recently in [1] for directed semimetrics.

LEMMA 3.1. *There is a polynomial-time algorithm that, given a k-point directed ℓ_2^2 spreading metric given by vectors $\{v_x : x \in S \cup \{0\}\}$ satisfying $d(x,y) = \frac{1}{2}(||v_x - v_y||^2 - v_0 \cdot (v_x - v_y))$, produces sets X, Y of size $\Omega(k)$ each, such that for all $x \in X, y \in Y$, $d(x,y)$ is $\Omega(k/\sqrt{\log k})$*

The proof is a reduction of this directed case to the undirected case of Lemma 2.1.

Proof. Consider the underlying undirected metric $||v_x - v_y||^2 = d(x,y) + d(y,x)$. This is an ℓ_2^2 spreading metric. Using Lemma 2.1, we can find sets $X', Y' \subseteq S$ such that $|X'|$ is $\Omega(k)$, $|Y'|$ is $\Omega(k)$, and for all $x \in X', y \in Y'$, $||v_x - v_y||^2$ is $\Omega(k/\sqrt{\log k})$. Let m be the median value in the set $\{v_0 \cdot v_x : x \in X'\}$. Let $X^+ = \{x \in X' : v_0 \cdot v_x \geq m\}$ and let $X^- = \{x \in X' : v_0 \cdot v_x \leq m\}$. Let $Y^+ = \{y \in Y' : v_0 \cdot v_y \geq m\}$ and let $Y^- = \{y \in Y' : v_0 \cdot v_y \leq m\}$. Note that $|X^+| \geq |X'|/2$ and $|X^-| \geq |X'|/2$. If $|Y^+| \geq |Y'|/2$, we set $(X, Y) = (X^-, Y^+)$; otherwise, we set $(X, Y) = (Y^-, X^+)$. Note that this ensures that $|X|, |Y|$ are $\Omega(k)$. From the guarantee on X', Y', we know that for all $x \in X, y \in Y$, $||v_x - v_y||^2$ is $\Omega(k/\sqrt{\log k})$. Further, by the choice of X, Y, for all $x \in X, y \in Y$, $v_0 \cdot v_x \leq v_0 \cdot v_y$. Recall that $d(x,y) = \frac{1}{2}(||v_x - v_y||^2 - v_0 \cdot (v_x - v_y))$. For $x \in X, y \in Y$, note that the first term in the expression for $d(x,y)$ is $\Omega(k/\sqrt{\log k})$ and the second term is nonpositive. This proves the lemma. ■

Consider the following relaxation for a special case of STORAGE TIME PRODUCT, where all tasks have unit execution time. We later show how the general problem can be solved using this unit-time problem as a subroutine.

$$\min \sum_{(x,y)\in E} w(x,y) \cdot (d(x,y)+1)$$

subject to the constraints that

$$d(x,y) \text{ is a directed } \ell_2^2 \text{ spreading metric, and}$$
$$d(y,x) = 0 \text{ for all } (x,y) \in E.$$

Note that the objective function is a constant plus a weighted sum of distances $d(x,y)$. We can adapt the algorithm for MINIMUM LINEAR ARRANGEMENT to approximate MINIMUM STORAGE-TIME PRODUCT using the solution to the SDP relaxation. We use the well-separated sets X, Y guaranteed by Lemma 3.1 to partition the problem into subproblems and recurse on them. As in the earlier algorithm, we either find a very small (directed) cut between X and Y, and divide the vertex set into two sets using it, or we make a large number of parallel cuts, dividing the problem into several subproblems. The difference compared to MINIMUM LINEAR ARRANGEMENT is the presence of directed arcs in the digraph G, which constrain the placement of vertices: for every arc (s,t), vertex s must be placed to the left of vertex t. The solution produced by the algorithm indeed satisfies these ordering constraints. This is because the SDP constraints ensure that for every arc (s,t), $d(t,s) = 0$. Hence, if (s,t) is an arc, then $d(x,s) \leq d(x,t) + d(t,s) = d(x,t)$, and therefore for any set X, $d(X,s) \leq d(X,t)$, where $d(X,s) = \min_{x\in X} d(x,t)$. The partitioning into subproblems is based on distance from the set X which is guaranteed to exist by Lemma 3.1: the sets $N_\delta(X)$ have the property that if $t \in N_\delta(X)$, then so is s. It follows that in each step, if the algorithm puts s and t into different sets, then s is to the left of t. We omit the details since they are similar to those in the previous section.

THEOREM 3.1. *There is a polynomial-time $O(\sqrt{\log n}\log\log n)$-approximation algorithm for* MINIMUM STORAGE-TIME PRODUCT *with unit-time jobs.*

Unfortunately, for lack of space, we defer to the full version the reduction from the general case to the unit-time jobs case. That reduction gives us

THEOREM 3.2. *There is a $O(\sqrt{\log T}\log\log T)$-approximation algorithm for* MINIMUM STORAGE-TIME PRODUCT *that runs in time polynomial in T, where T is the sum of the execution times of all jobs.*

We suspect that it is possible to improve the running time of the approximation algorithm from polynomial in T to polynomial in n.

4 Minimum Containing Interval Graph

An *interval graph* is a graph whose vertices can be mapped to intervals on the line such that any two vertices are adjacent if and only if their corresponding intervals overlap. MINIMUM CONTAINING INTERVAL GRAPH is this problem: Given a graph $G = (V,E)$, find an interval supergraph $\bar{G}$ of G, on the same vertex set, with the fewest edges.

LEMMA 4.1. *[14] For any graph $G = (V,E)$ without isolated vertices, the minimum of $|\bar{E}|$ over interval supergraphs $\bar{G} = (V,\bar{E})$ of G equals the minimum, over bijections σ from V to $\{1,2,...,|V|\}$, of*

$$\sum_{x\in V} \max_{y:\{x,y\}\in E} \{\max\{\sigma(y)-\sigma(x),0\}\}.$$

It follows that, instead of explicitly computing an interval supergraph of G, we can instead find an ordering σ of V (i.e., a bijection of V with $\{1,2,...,|V|\}$) which approximately minimizes the sum, over x in V, of the maximum, over neighbors y in G of x, of $\max\{0, \sigma(y)-\sigma(x)\}$. Hence we use the following relaxation for MINIMUM CONTAINING INTERVAL GRAPH:

$$\min \sum_{x\in V} \max_{y:\{x,y\}\in E} d(x,y)$$
$$d(x,y) \text{ is a directed } \ell_2^2 \text{ spreading metric.}$$

Since directed ℓ_2^2 spreading metrics are relaxations of directed permutation metrics, this SDP relaxation is indeed a valid relaxation for MINIMUM CONTAINING INTERVAL GRAPH.

We round the SDP solution via the following procedure, a variant of the algorithm for MINIMUM LINEAR ARRANGEMENT. Here, as before, we recurse in two different ways, breaking the graph up in the first case into two pieces, and in the second, into many (not just at least three). Unlike the MINIMUM LINEAR ARRANGEMENT case, in both cases here "separating" sets of *vertices* are *removed* from the graph, not to appear in any subproblem. Another difference between this algorithm and the MINIMUM LINEAR ARRANGEMENT one is that here, the decision as to which case we're in depends on the size of a set of *vertices*, whereas in MINIMUM LINEAR ARRANGEMENT the decision depended on the size of a set of *edges*.

Take any vectors $\{v_x\}$ which are feasible for the SDP; let w be the value of the SDP when those vectors are used. (We construct the vectors $v_1, v_2, ..., v_n$ only once, at the beginning.) Set $S = V$ and call $MCIG(S)$.

Procedure $MCIG(S)$**:**

1. Let $k = |S|$. If $k \leq N$ or if the subgraph induced by S has no edges, output any ordering of S and halt.

2. Use the k vectors $\{v_x : x \in S\}$ and Lemma 3.1 to produce a pair of sets X and Y such that $d(x, y) \geq \Delta$ for all $x \in X, y \in Y$, where Δ is integral and at least $ck/\sqrt{\lg k}$, and $|X|$ and $|Y|$ are both $\Omega(k)$.

3. For each $\delta \in \{0, 1, 2, ..., \Delta - 1\}$, define $N_\delta(X) = \{y \in S : d(X, y) \leq \delta\}$, where $d(X, y) = \min_{x \in X}\{d(x, y)\}$. Define $Bdry_\delta(X) = \{y \in N_\delta(X) : \exists z \in S - N_\delta(X)$ such that $\{y, z\} \in E\}$.

   (Note that $Bdry_\delta(X)$ is a set of *vertices*.)

4. Check if $|Bdry_\delta(X)|$ is at most $w/(\Delta \lg k)$ for any δ in $\{0, 1, 2, ..., \Delta - 1\}$. If so, choose such a δ and produce two subproblems, one consisting of the subgraph induced by $Z_1 := N_\delta(X) - Bdry_\delta(X)$, the other consisting of the subgraph induced by the set $Z_2 := S - N_\delta(X)$. Call $MCIG(Z)$ for $Z = Z_1, Z_2$ to produce orderings $\sigma(Z_1)$ and $\sigma(Z_2)$ respectively. Define $\sigma(Bdry_\delta(X))$ to be an arbitrary ordering of $Bdry_\delta(X)$. Produce the final ordering for S by concatenating the orderings as follows: $\sigma(Bdry_\delta(X)), \sigma(Z_1), \sigma(Z_2)$ (and halt).

   (Note that we place the vertices of the boundary *before* the vertices of the two sets separated by the boundary.)

5. Otherwise, group the δ in $\{0, 1, 2, ..., \Delta - 1\}$ according to the value of $|Bdry_\delta(X)|$. Define

   $$D_i = \{\delta : 2^i[w/(\Delta \lg k)] < |Bdry_\delta(X)| \leq 2^{i+1}[w/(\Delta \lg k)]\}, \tag{4.5}$$

   for $i = 0, 1, 2, 3, ...$ (We start at $i = 0$ because step 4 failed.) Among those $i \in \{0, 1, 2, ..., 1 + \lceil \lg \lg k \rceil\}$, let I be the value of i with the largest D_i.

6. Let $D_I = \{\delta_1, \delta_2, \delta_3, ..., \delta_s\}$, $\delta_1 < \delta_2 < \delta_3 < \cdots < \delta_s$. Let $Z_1 := (N_{\delta_1}(X)) - Bdry_{\delta_1}(X), Z_2 := (N_{\delta_2}(X) - N_{\delta_1}(X)) - Bdry_{\delta_2}(X), Z_3 := (N_{\delta_3}(X) - N_{\delta_2}(X)) - Bdry_{\delta_3}(X), \ldots, Z_s := (N_{\delta_s}(X) - N_{\delta_{s-1}}(X)) - Bdry_{\delta_s}(X), Z_{s+1} := S - N_{\delta_s}(X)$ and let Z be the disjoint union of the subgraphs induced by $Z_1, Z_2, ..., Z_{s+1}$. (Since the boundary vertices have been removed, there are, in fact, no edges of G between vertices in different Z_i's anyway.)

   Recursively call $MCIG(Z)$. From the ordering returned, compute the implied orderings $\sigma(Z_1), \sigma(Z_2), ..., \sigma(Z_{s+1})$ of $Z_1, Z_2, ..., Z_{s+1}$, respectively. Let $R_i = (N_{\delta_i}(X) - N_{\delta_{i-1}}(X)) \cap Bdry_{\delta_i}(X)$ for $i = 2, 3, ..., s + 1$ and let $R_1 = N_{\delta_1}(X) \cap Bdry_{\delta_1}(X)$. Produce the final ordering for S by concatenating the orderings as follows: $\sigma(R_1), \sigma(Z_1); \sigma(R_2), \sigma(Z_2); \ldots; \sigma(R_s), \sigma(Z_s); \sigma(Z_{s+1})$, where we define $\sigma(R_i)$ to be an arbitrary ordering of R_i.

   (Note that we are placing the boundary vertices R_i *before* the nonboundary vertices Z_i.)

End

LEMMA 4.2. *If $x \in Bdry_i(X)$ and $x \in Bdry_{j-1}(X)$ for $i < j$, then $\exists y \in S$ such that $\{x, y\} \in E$, $y \notin N_{j-1}(X)$ and $d(x, y) \geq (1/2)(j - i)$. Further $x \in Bdry_\delta(X)$ for all $i \leq \delta \leq j - 1$.*

Proof. Since $x \in Bdry_i(X)$, $x \in N_i(X)$. Also, since $x \in Bdry_{j-1}(X)$, by the definition of $Bdry_{j-1}(X)$ there exists $y \in S$ such that $y \notin N_{j-1}(X)$ and $\{x, y\} \in E$. Arguing as in Lemma 2.2, $d(x, y) \geq (1/2)(j - i)$. Also the existence of y implies that $x \in Bdry_\delta(X)$ for all $i \leq \delta \leq j - 1$. ∎

Let $D(x) = \{\delta : x \in Bdry_\delta(X)\}$.

LEMMA 4.3. *For any $x \in S$, $\max_{y \in S : \{x,y\} \in E} d(x, y) \geq (1/2)|D(x)|$.*

Proof. Let $i = \min D(x)$, $j = 1 + \max D(x)$. Then $x \in Bdry_i(X)$ and $x \in Bdry_{j-1}(X)$. By the previous lemma, $x \in Bdry_\delta(X)$ for all $i \leq \delta \leq j - 1$. Hence $D(x) = \{i, i + 1, i + 2, ..., j - 1\}$. Also, by the previous lemma, there exists $y \in S$ such that $\{x, y\} \in E$ and $d(x, y) \geq (1/2)(j - i) = (1/2)|D(x)|$. Hence $\max_{y \in S : \{x,y\} \in E} d(x, y) \geq (1/2)|D(x)|$. ∎

LEMMA 4.4. *$\sum_{\delta=0}^{\Delta-1} |Bdry_\delta(X)| \leq 2w$.*

Proof. $w = \sum_{x \in S} \max_{y : \{x,y\} \in E} d(x, y) \geq \sum_x (1/2)|D(x)|$. Hence $\sum_{\delta=0}^{\Delta-1} |Bdry_\delta(X)| = \sum_x |D(x)| \leq 2w$. ∎

LEMMA 4.5. *$\sum_{i=0}^{1+\lceil \lg \lg k \rceil} |D_i| \geq \Delta/2$.*

Proof. Consider $\delta \in \{0, 1, 2, ..., \Delta - 1\}$ and choose i such that $\delta \in D_i$. $|Bdry_\delta(X)| \geq 2^i w/(\Delta \lg k)$. If $i \geq 2 + \lceil \lg \lg k \rceil$, then $|Bdry_\delta(X)| \geq 4w/\Delta$. Call δ *bad* if the corresponding i is at least $2 + \lceil \lg \lg k \rceil$, and *good* otherwise. By Lemma 4.4,

$$2w \geq \sum_{\delta=0}^{\Delta-1} |Bdry_\delta(X)| \geq \sum_{\text{bad } \delta} |Bdry_\delta(X)|| \geq (4w/\Delta)(\text{number of bad } \delta).$$

Hence the number of bad δ is at most $\Delta/2$. Hence $\Delta - \Delta/2 \leq$ number of good $\delta = \sum_{i=0}^{1+\lceil \lg \lg k \rceil} |D_i|$. ∎

COROLLARY 4.1. *The value I chosen in step 5 satisfies $|D_I| \geq (\Delta/2)/(2 + \lceil \lg \lg k \rceil) \geq \Delta/(4 \lg \lg k)$ for sufficiently large k.*

LEMMA 4.6. *For any $x \in S$, $\max_{y \in S:\{x,y\} \in E} d(x,y) \geq (1/2)|\{\delta \in D_I : x \in Bdry_\delta(X)\}|$.*

Proof. Note that $\{\delta \in D_I : x \in Bdry_\delta(X)\} \subseteq D(x)$. The lemma now follows from Lemma 4.3. ■

THEOREM 4.1. *The procedure above produces an interval graph with $O(w\sqrt{\log k}\log\log k)$ edges.*

Proof. For the analysis alone, it helps to work with integral upper bounds on the distances, in lieu of the actual real-valued distances. For any nonnegative integers k, l, let $I(k,l)$ be the maximum number of edges in an interval graph produced by our algorithm when run on (possibly disconnected) graphs $G = (V,E)$ with at most k vertices and a given feasible vector solution $\{v_x\}$ such that $d(x,y) = ||v_x - v_y||_2^2$ satisfies $\sum_{x \in V} \max_{y:\{x,y\} \in E} \lceil d(x,y) \rceil \leq l$. Note that $\lceil d(x,y) \rceil \leq 2d(x,y)$, since $d(x,y) \geq 3/4$ if $x \neq y$.

Say that the *weight* of a vertex x is $\max_{y \in S:\{x,y\} \in E} d(x,y)$.

Note that we can recurse in two ways, in steps 4 and 6. If we recurse in step 4, we have a set $Bdry_\delta(X)$ of vertices of size at most $w/(\Delta \lg k)$. Each vertex x in $Bdry_\delta(X)$ has weight at most k, so that the total contribution of vertices in $Bdry_\delta(X)$ is at most $k(w/(\Delta \lg k)) \leq w/(c\sqrt{\lg k})$.

No vertex in Z_1 has an edge to a vertex in $S - N_\delta(X)$; hence there are no $Z_1 - Z_2$ edges. All edges incident to vertices in Z_1 go to other vertices in Z_1 or *leftward* to vertices in $Bdry_\delta(X)$, and the latter edges don't affect the contribution of Z_1 to the cost of the layout. An analogous statement holds for the vertices of Z_2.

Let $k_1 = |Z_1|$ and $k_2 = |Z_2|$. Let $l = \sum_{x \in S} \max_{y \in S:\{x,y\} \in E} \lceil d(x,y) \rceil$. Let $l_1 = \sum_{x \in Z_1} \max_{y \in Z_1:\{x,y\} \in E} \lceil d(x,y) \rceil$ and let $l_2 = \sum_{x \in Z_2} \max_{y \in Z_2:\{x,y\} \in E} \lceil d(x,y) \rceil$. Clearly $l_1 + l_2 \leq l$. The cost of our placement, by induction, is at most $I(k_1, l_1) + I(k_2, l_2) + w/(c\sqrt{\lg k})$, where $k_1, k_2 \leq (1-\epsilon)k$ for a fixed $\epsilon > 0$, $k_1 + k_2 \leq k$, $w_1 + w_2 \leq w$.

In step 6, we get a list of at least $\Delta/(4 \lg \lg k)$ sets $Bdry_\delta(X)$. Each of the vertex sets has size satisfying $|Bdry_\delta(X)| \geq \alpha := 2^I w/(\Delta \lg k)$ and $|Bdry_\delta(X)| \geq 1$, where I is the index defined in step 5. Since the vertices x in $\cup_{\delta \in D_I} Bdry_\delta(X) = \cup_{j=1}^s Bdry_{\delta_j}(X)$ are removed from the subproblems produced in this step, by Lemma 4.6 the total weight of the subproblems decreases by at least $\sum_{x \in \cup_{j=1}^s Bdry_{\delta_j}(X)} \left(\max_{y \in S:\{x,y\} \in E} d(x,y)\right) \geq \sum_{x \in \cup_{j=1}^s Bdry_{\delta_j}(X)} (1/2)|\{\delta \in D_I : x \in Bdry_\delta(X)\}| \geq (1/2)\sum_{j=1}^s |Bdry_{\delta_j}(X)|$. Each $|Bdry_{\delta_j}(X)| \geq \alpha$, for $j = 1, 2, ..., s$. There are $s \geq \Delta/(4 \lg \lg k) \geq 2$ values of j, by Corollary 4.1. From the fact that Δ is at least $ck/\sqrt{\lg k}$ (by Lemma 2.1), the decrease in weight is at least $(1/2)\alpha s \geq (1/2)\alpha \cdot [(ck/\sqrt{\lg k})/(4 \lg \lg k)] = (1/2)c\alpha k/(4 \lg \lg k \sqrt{\lg k})$, but also at least $(1/2)s \geq 1$.

Now we have to bound from above the cost associated with vertices not in Z. Let $k_j = |R_j \cup Z_j| = |N_{\delta_j}(X) - N_{\delta_{j-1}}(X)|$ for all j. Consider a vertex $x \in \cup_{\delta \in D_I} Bdry_\delta(X) = \cup_{j=1}^s Bdry_{\delta_j}(X)$. By the definition of $D(x)$ and the proof of Lemma 4.3, there are i, i' such that $x \in Bdry_\delta(X)$ if and only if $i \leq \delta \leq i' - 1$. Hence we can choose l, m such that $x \in Bdry_{\delta_l}(X), Bdry_{\delta_{l+1}}(X), Bdry_{\delta_{l+2}}(X), \ldots, Bdry_{\delta_{m-1}}(X)$ yet $x \notin Bdry_{\delta_h}(X)$ for any other $h \in \{1, 2, ..., s\}$. Since x is incident to no vertex outside of $N_{\delta_m}(X)$, the cost paid for x in the final ordering, i.e., the maximum "stretch" of an edge $\{x,y\}$ in which y is placed to the right of x, is at most $|R_l \cup Z_l| + |R_{l+1} \cup Z_{l+1}| + \cdots + |R_m \cup Z_m| = k_l + k_{l+1} + k_{l+2} + \cdots + k_m \leq (k_l + k_{l+1}) + (k_{l+1} + k_{l+2}) + \cdots + (k_{m-1} + k_m)$. It follows that we can bound the overall cost paid for these vertices by $\sum_{j=1}^s |Bdry_{\delta_j}(X)|(k_j + k_{j+1})$, which is at most $\sum_{j=1}^s (2\alpha)(k_j + k_{j+1})$ (by the definition of α and the upper bound in (4.5)), and hence at most $(2\alpha)(2k) = 4k\alpha$.

Thus our total cost is at most $I(k, \min\{l - 1, \lceil l - (1/2)c\alpha k/(4 \lg \lg k \sqrt{\lg k}) \rceil\}) + 4\alpha k$. It follows that in general, $I(k,l)$ satisfies $I(k,l) \leq \max\{I(k_1,l_1) + I(k_2,l_2) + O(w/\sqrt{\lg k}), I(k, \min\{l - 1, \lceil l - (1/8)c\alpha k/(4 \lg \lg k \sqrt{\lg k}) \rceil\}) + 4\alpha k\}$, where, in the first expression, $k_1, k_2 \leq (1-\epsilon)k$ for a fixed $\epsilon > 0$, $k_1 + k_2 = k$, and $l_1 + l_2 \leq l$.

Since this is the same recurrence we got in Theorem 2.1, it has the same solution. ■

5 Remarks On Relaxations of Permutation Metrics

Note that a permutation metric or a directed permutation metric has an underlying permutation σ. For a given permutation σ, we refer to the associated permutation metric or the directed permutation metric by d_σ. (The context will clarify whether the metric is undirected or directed.)

We say that a metric (X, d) is *c-approximable by permutation metrics* if there is a probability distribution $\mathcal{D}$ on permutation metrics d_σ such that for all x, y, $E_{\sigma \in \mathcal{D}}[d_\sigma(x,y)] \leq c \cdot d(x,y)$. Here $E_{\sigma \in \mathcal{D}}[d_\sigma(x,y)]$ denotes the expected distance between x and y in a permutation metric drawn from the distribution. Similarly, we say that a directed metric (X,d) is *c-approximable by directed permutation metrics* if there is a probability distribution $\mathcal{D}$ on directed permutation metrics d_σ such that for all x, y, $E_{\sigma \in \mathcal{D}}[d_\sigma(x,y)] \leq c \cdot d(x,y)$.

Our results imply that ℓ_2^2 spreading metrics on n points are $O(\sqrt{\log n}\log\log n)$-approximable by permutation metrics and directed ℓ_2^2 spreading metrics on n points are $O(\sqrt{\log n}\log\log n)$ approximable by directed permutation metrics, as we now show. In fact, the problem of approximating a metric via a probability distribution over permutation metrics is the LP dual of (maximizing the integrality ratio for) MINIMUM LINEAR ARRANGEMENT. Given a metric $d(x, y)$, consider the problem of finding the best approximation of d via a convex combination of permutation metrics. This can be expressed as an LP with a variable x_σ for every permutation σ of X as follows:

$$\begin{aligned} & \min c \\ \sum_\sigma x_\sigma &\geq 1; \\ c \cdot d(x,y) - \sum_\sigma x_\sigma d_\sigma(x,y) &\geq 0 \quad \text{for all } x, y \in X; \\ x_\sigma &\geq 0 \quad \text{for all } \sigma. \end{aligned}$$

The dual of this is given by the following:

$$\max z$$

$$\sum_{x,y} d(x,y) w(x,y) \leq 1;$$

$$\sum_{x,y} d_\sigma(x,y) w(x,y) \geq z$$

for all permutation metrics d_σ;

$$w(x,y) \geq 0$$

for all $x, y \in X$.

(Assuming the metric is symmetric, we can replace "$x, y \in X$" by "$x, y \in X, x < y$," twice.)

Note that the dual is this problem: given $d(x, y)$, find weights $w(x, y)$ on pairs (x, y) (unordered pairs in the undirected case, ordered pairs in the directed case) to produce an instance of MINIMUM LINEAR ARRANGEMENT for which the integrality ratio of the solution given by $d(x, y)$ is maximum. If the metric (X, d) is an ℓ_2^2 spreading metric, then the value of the dual is $O(\sqrt{\log n} \log \log n)$ (by Theorem 2.1). This means that the value of the primal is $O(\sqrt{\log n} \log \log n)$. The same argument goes through for a directed ℓ_2^2 spreading metric.

In fact, using our algorithms for MINIMUM LINEAR ARRANGEMENT and its directed variant, we can construct a probability distribution on permutation metrics that approximates a given ℓ_2^2 spreading metric.

6 Open Problems

It would be interesting to investigate whether these ℓ_2^2 spreading metrics can be used to solve other problems dealing with permutation metrics. This could lead to improved approximation algorithms for embedding a metric into a line, a problem that has received some attention recently. Another interesting direction is whether there are good computable relaxations for tree metrics. Such relaxations could have application to a variety of problems. Finally, extending our directed ℓ_2^2 spreading metrics so as to improve the best known approximation factor of $O(\log n \log \log n)$ for MINIMUM FEEDBACK ARC SET [17, 7] remains an important and challenging open problem.

References

[1] A. Agarwal, M. Charikar, K. Makarychev, and Y. Makarychev, "$O(\sqrt{\log n})$-approximation algorithms for Min Uncut, Min 2CNF deletion and directed cut problems," in *Proc. 37th STOC*, 573-581, 2005.

[2] S. Arora, S. Rao, and U. Vazirani, "Expander flows, geometric embeddings, and graph partitioning," in *Proc. 36th STOC*, 222-231, 2004. Full version available at `http://www.cs.princeton.edu/~arora/pubs/arvfull.pdf`.

[3] S. Bhatt and F. T. Leighton, "A framework for solving VLSI graph layout problems," *Journal of Computer and System Sciences* 28(2), 300-343, 1984.

[4] G. Even, J. Naor, S. Rao, and B. Schieber, "Fast approximate graph partitioning algorithms," *SIAM Journal on Computing* 28(6), 2187-2214, 1999.

[5] G. Even, S. Naor, S. Rao, and B. Schieber, "Divide-and-conquer approximation algorithms via spreading metrics," *JACM* 47(4), 585-616, 2000.

[6] U. Feige and J. Lee, "Improved approximation algorithms for minimum linear arrangement," *manuscript*, 2005.

[7] G. Even, J. Naor, B. Schieber, and M. Sudan, "Approximating minimum feedback sets and multicuts in directed graphs," *Algorithmica* 20(2), 151-174, 1998.

[8] M. Hansen, "Approximation algorithms for geometric embeddings in the plane with applications to parallel processing problems," in *Proc. 30th FOCS*, 604-609, 1989.

[9] J. Lee, "On distance scales, embeddings, and efficient relaxations of the cut cone," in *Proc. 16th SODA*, 92-101, 2005.

[10] F. T. Leighton, *Complexity Issues in VLSI: Optimal Layout for the Shuffle-Exchange Graph and Other Networks*, MIT Press, Cambridge, MA, 1983.

[11] F. T. Leighton and S. Rao, "Multi-commodity max-flow min-cut theorems and their use in designing approximation algorithms," *JACM* 46(6), 787-832, 1999.

[12] C. E. Leiserson, "Area-efficient graph layouts (for VLSI)," in *21st FOCS*, 270-280, 1980.

[13] R. J. Lipton and R. E. Tarjan, "Applications of a planar separator theorem," *SIAM Journal on Computing* 9(3), 615-627, 1980.

[14] G. Ramalingam and C. Pandu Rangan, "A unified approach to domination problems on interval graphs," *Information Processing Letters* 27, 271-274, 1988.

[15] S. Rao and A. Richa, "New approximation techniques for some ordering problems," in *Proc. 9th SODA*, 211-218, 1998.

[16] R. Ravi, A. Agrawal, and P. Klein, "Ordering problems approximated: single-processor scheduling and interval graph completion," *Proc. 18th ICALP*, 751-762, 1991.

[17] P. D. Seymour, "Packing directed circuits fractionally," *Combinatorica* 15(2), 281-288, 1995.