Bypassing the Embedding: Approximation schemes and Compact Representations for Low Dimensional Metrics

Kunal Talwar[*]
Computer Science Division
University of California, Berkeley
kunal@cs.berkeley.edu

ABSTRACT

The *doubling dimension* of a metric is the smallest k such that any ball of radius $2r$ can be covered using 2^k balls of radius r. This concept for abstract metrics has been proposed as a natural analog to the dimension of a Euclidean space. If we could embed metrics with low doubling dimension into low dimensional Euclidean spaces, they would inherit several algorithmic and structural properties of the Euclidean spaces. Unfortunately however, such a restriction on dimension does not suffice to guarantee embeddibility in a normed space.

In this paper we explore the option of bypassing the embedding. In particular we show the following for low dimensional metrics:

- Quasi-polynomial time $(1+\epsilon)$-approximation algorithm for various optimization problems such as TSP, k-median and facility location.
- $(1+\epsilon)$-approximate distance labeling scheme with optimal label length.
- $(1+\epsilon)$-stretch polylogarithmic storage routing scheme.

1. INTRODUCTION

Starting with the seminal work of Linial, London and Rabinovich [35], the confluence of the rich mathematical theory of metric spaces and theoretical computer science has proved very fruitful for both fields. Metric embeddings have proved to be extremely useful tools for algorithm design, apart for being beautiful mathematical objects in their own right.

One of the motivations for embedding a metric space into, say, a Euclidean space is to use the simple and well studied properties of the Euclidean space for algorithm design. However, there are metrics which require a logarithmic distortion for such an embedding.

Restrictions on the geometry have been proposed in various contexts, as a metric analog to the low dimension of a normed space. The doubling dimension of a metric is the smallest k such that any ball of radius $2r$ can be covered by 2^k balls of radius r. Following [24], we say a family of metrics is *doubling* if it has constant doubling dimension. Euclidean metrics of dimension k have doubling dimension $\Theta(k)$, and the definition is robust to constant factor distortions (see Proposition 3). Moreover, submetrics of a metric have a smaller dimension and this notion generalizes the growth restriction assumption in [29] ([24] Prop. 1.2). Such low dimension restrictions are natural in several practical applications such as peer-to-peer networks (see e.g. [37]) and data analysis (where the input data lies on a low dimensional manifold [43]).

If we could, for example, embed a doubling metric into a low dimensional Euclidean space with a small distortion, we would be able to exploit the well studied structural and algorithmic properties of these spaces. However, as shown by Laakso [33], Lang and Plaut [34] and by Gupta, Krauthgamer and Lee [24], these restrictions do not suffice to ensure low distortion embeddings into Euclidean spaces.

In this work, we explore the option of bypassing the embedding step. A restriction on the doubling dimension itself is sufficient in several cases to prove some nice properties of the metric space. We show that this is the case for efficient optimization, distance labels and compact routing schemes.

Our first result deals with approximation schemes for NP-hard optimization problems on doubling metrics. For low dimensional Euclidean metrics, such schemes are known for several optimization problem; perhaps the most well-known being the ones for the traveling salesman problem (TSP) due to Arora [2] and Mitchell [36]. We extend the approximation scheme for Euclidean TSP to arbitrary doubling metrics. As one would expect, the techniques extend to a host of other optimization problems such as k-median, facility location and steiner tree.

Unfortunately, the dependence on the dimension is worse than in the Euclidean case, and we are only able to show a quasi-polynomial time approximation scheme (with running time $2^{(\log n)^{O(d)}}$ for constant ϵ). We leave open the question of whether this dependence in necessary; if e.g. the traveling salesman problem is Max-SNP hard on any family of metrics with doubling dimension $O(\log n/\log\log n)$, that would be strong evidence of the inherence of quasipolynomiality. At the same time, our results suggest that these optimization problems are unlikely to be Max-SNP hard on doubling

[*] Research partially supported by the NSF via grants CCR-0121555, CCR-0105533 and ITR grant 331494.

STOC'04, June 13–15, 2004, Chicago, Illinois, USA.
Copyright 2004 ACM 1-58113-852-0/04/0006 ...$5.00.

metrics.

Next we look at some problems in compact representations of metrics. In the the *distance labeling* problem, we want to assign *labels* to vertices of a metric X so that the distance between any pair of vertices can be approximated by inspecting their labels, without any additional information. We show how to assign $O(\log \Delta)$ bit labels to points in a doubling metric (where $\Delta = \frac{d_{\max}}{d_{\min}}$ is the aspect ratio of the metric), so that the distances can be estimated within a multiplicative error of $(1+\epsilon)$. The dependence on the doubling dimension is exponential, and we show that such a dependence is in fact inevitable. Note that for a normed space, the coordinates themseleves serve as distance labels of size $O(d \log \Delta)$. This points to the richness of doubling metrics, as compared to low dimensional normed spaces.

We also show an efficient routing scheme for graphs that induce doubling metrics. Our scheme requires only polylogarithmic storage at each node and routes on paths that are guaranteed to be no longer than $(1+\epsilon)$ times the shortest path. This contrasts with general networks where a polynomial storage is necessary to get any constant bound on the stretch [40]. Our routing scheme is stack based and requires stacks of logarithmic depth.

1.1 Techniques

A crucial ingredient of our algorithms is a split-tree, which gives a hierarchical decomposition of the metric. We show how to get such a decomposition with several desirable properties, much like the (randomly shifted) grid in Euclidean spaces. Another property of doubling metrics we repeatedly use is the existence of a small *nets* (see definition below) which play a role similar to grid points in Euclidean algorithms. In fact, several known results for Euclidean metrics can be easily generalized to doubling metrics using the above substitutions! (E.g. some recent results in [28, 25]).

We reinterpret our (and previous) approximation schemes in terms of probabilistic embeddings into low tree width graphs. This unified view leads to derandomization of the approximation schemes.

We also show how to construct a *well-separated pair decomposition* of a doubling metric. A pair of sets (A, B) is called well-separated if the distance between A and B is much larger than the diameters of A and B. A well-separted pair decomposition consists of (a small number of) well separated pairs, such that any pair of points is represented in some pair of sets. We show how such well-separated pairs can be used to compactly represent the distance information in the the metric. Our techniques seem new and may be of independent interest.

1.2 Related Work

The definition of doubling dimension is due to Gupta, Krauthgamer and Lee [24] and inspired by a definition due to Assouad [6]. A closely related definition was previously studied by Clarkson [15]. Karger and Ruhl [29] studied a stonger notion of growth restriction.

Efficient nearest neigbour algorithms for under such restrictions have been studied by Clarkson [15], Karger and Ruhl [29], Hildrum et.al. [26] and Krauthgamer and Lee [31, 32].

Approximation schemes for various optimization problems on Euclidean metrics have been studied, e.g. in [2, 36, 3, 5, 42, 30]. For more details on approximation schemes for Euclidean metrics, see the recent survey by Arora [4].

Distance labels were first studied by Peleg [39], and further work has appeared in [22, 20, 21]. Gavoille et.al. [22] showed several lower bounds on the lengths of such labels, e.g. $\Omega(n)$ for general graphs, $\Omega(\log^2 n)$ for trees and $\Omega(n^{\frac{1}{3}})$ for planar graphs. The relevance of distance labeling schemes for communication networks has been discussed in [39]. Some potential applications that have been suggested are "memory-free" routing, bounded broadcast protocols and topology update mechanisms.

On the other hand, distance labels fall into the general class of compact graph representation problems, and approximate distance oracles have been studied, amongst others by Thorup and Zwick [44]. Such oracles for the geometric setting have been explored in [23]. Compact routing schemes have been widely studied, e.g. [40, 7, 8, 16, 17, 45]. For a general overview of efficient routing schemes, see [38].

2. DEFINITIONS AND PRELIMINARIES

We first define some terms and notation used, and show some simple properties of doubling metrics.

A *metric* (X, d) is a set X endowed with a distance function $d : X \times X \to \Re_0$, satisfying $d(x, x) = 0$, symmetry and triangle inequality.

The *neighbourhood* $B(x, r)$ of a point $x \in X$ for a real number r is the set $\{y \in X : d(x, y) \leq r\}$. The *doubling dimension* of a metric (X, d) is the smallest constant k_X such that any neighbourhood $B(x, 2r)$ is contained in the union of at most 2^{k_X} neighbourhoods $B(y_i, r)$. We shall often refer to k_X as the dimension and denote it by k when X is obvious from context. We say a family of metrics is *doubling* if there is a constant C such that $k_X \leq C$ for every X in the family.

For a finite subset $Y \subseteq X$, the *aspect ratio* of Y is defined to be $\frac{\max_{y,y' \in Y} d(y,y')}{\min_{y,y' \in Y} d(y,y')}$.

An *r-cover* of X is a set of points Y such that for any point $x \in X$, there is a point $y \in Y$ such that $d(x, y) \leq r$. A set of points Y is an *r-packing* if for any $y, y' \in Y$, $d(y, y') > 2r$.

We say a set Y is an *r-net* in X if it is both an r-cover of X and an $\frac{r}{2}$-packing. Note that such a net can be constructed greedily, since any minimal r-cover is an r-net.

Preliminaries. We shall be using the following important property of doubling metrics, proved in [24]:

PROPOSITION 1
Let (X, d) be a doubling metric and $Y \subseteq X$ with aspect ratio Δ. Then $|Y| \leq 2^{k_X \cdot \lceil \log_2 \Delta \rceil}$.

PROOF. (sketch) Recursively apply definition. □

An approximate converse of the above is proved, e.g. in [32].

PROPOSITION 2
Let (X, d) be a doubling metric. Then there is a subset $Y \subseteq X$ with aspect ratio 4 such that $|Y| \geq 2^{k_X}$.

PROOF. (sketch) By definition, there is a ball $B(x, 2r)$ that requires 2^{k_X} balls $B(y_i, r)$ to be covered. The points y_i satisfy the required property. □

The last two propositions thus give an approximately equivalent definition of doubling dimension: A metric has doubling dimension $O(k)$ if and only if the largest 4-uniform submetric is of size 2^k. This also leads to an $O(n^2 2^{k_X} \log \Delta)$ time algorithm (based on greedy k-center approximation) for approximating the doubling dimension of any metric.

We can now derive the following *robustness* property of the doubling dimension:

Proposition 3
[Robustness] *Let (X, d) and (Y, d') be metric spaces such that there is a bijective map $f : X \rightarrow Y$ satisfying $d(x_1, x_2) \leq d'(f(x_1), f(x_2)) \leq D \cdot d(x_1, x_2)$. Then $k_Y \leq 2k_X \cdot \lceil \log 4D \rceil$.*

Proof. By proposition 2, it suffices to show that any submetric Z in (Y, d') with aspect ratio 4 has at most $2^{k_X \cdot \lceil \log 4D \rceil}$ points. By the bilipschitz condition, the preimage $f^{-1}(Z)$ in X has aspect ratio $4D$. By proposition 1, the claim follows. □

This implies that distortion roughly preserves doubling dimension. Hence a distorted version of a 2 dimensional Euclidean metric has constant doubling dimension.

3. RANDOMIZED SPLITTREES

An important ingredient of our results is the construction of a split-tree, which plays a role analogous the the shifted quadtree in Euclidean spaces.

A *decomposition* of the metric X is a partitioning of X into subsets, which we call *clusters*. A *hierarchical decomposition* is a sequence of decompositions $\mathcal{P}_0, \mathcal{P}_1, \ldots, \mathcal{P}_l$, such that every cluster in $\mathcal{P}_i$ is the union of clusters in $\mathcal{P}_{i-1}$. Given such a decomposition, the clusters in $\mathcal{P}_i$ are referred to as level i clusters. A *complete hierarchical decomposition* will be one where $P_l = \{X\}$ and $P_0 = \{\ \{x\} : x \in X\}$, i.e. the level l cluster is the whole metric and each point is its own level 0 cluster.

A *split-tree* of the metric stems naturally from a complete hierarchical decomposition - the root node corresponds to the single level l cluster X and the children of a level i cluster are the level $(i-1)$ clusters comprising it. The leaves are the singletons.

We shall show how to construct a complete hierarchical decomposition (split-tree) with the following properties:

(1) The number of levels is $\delta + 2$.

(2) Each level i cluster has diameter at most 2^{i+1}

(3) Each level i cluster is the union of at most $2^{O(k)}$ level $(i-1)$ clusters.

(4) For any pair of points (u, v) the probability that u and v fall in different level i clusters is at most $O(k) \cdot \frac{d(u,v)}{2^i}$.

We note that for points in $\Re^k$, the shifted quadtree has all the above properties.

Algorithm. We now describe our algorithm for this hierarchical decomposition. This is close to the decomposition algorithm used in [24], and is based on techniques used in [12, 18, 19].

Without loss of generality, the smallest distance in the metric is 1, and let $\Delta = 2^\delta$ be the diameter. We first construct a nested sequence $X = Y_0 \supseteq Y_1 \supseteq Y_2 \ldots \supseteq Y_\delta$ of sets such that Y_i is a 2^{i-2}-net of Y_{i-1}. It follows, by triangle inequality that Y_i is then a 2^{i-1}-cover of X, and still a 2^{i-3}-packing. Given this sequence of sets, the algorithm works as follows.

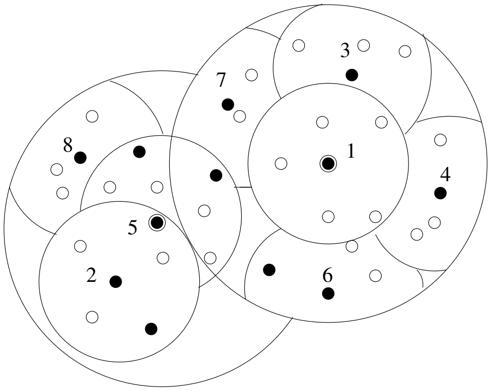

Figure 1: A sample decomposition. The dark points are in Y_i and the circled dark points in Y_{i+1}. The numbers indicate the permutation π.

```
Algorithm Decompose
1. Pick $\rho$ uniformly at randmom in $[\frac{1}{2}, 1)$
2. Pick a random permutation $\pi$ of $X$
3. Let $\mathcal{P}_{\delta+1} = \{X\}$.
4. For $i = \delta$ downto 0
4.1     Let $r_i = 2^i \rho$.
4.2     For each cluster $C_l \in \mathcal{P}_{i+1}$
4.2.1       For each $y \in Y_i$, define a new level $i$ cluster
                $C_l^y = \{x \in C_l : x \in B(y, r_i)$ and
                $(\forall z \in Y_i : \pi(z) < \pi(y))[x \notin B(z, r_i)]\}$
```

In words, at level i each point in Y_i comes in order of π and cuts out a cluster C_l^y consisting of all unassigned points in a ball of radius r_i around it. See figure 1.

Properties. Since Y_i is a 2^{i-1}-cover of X and $r_i \geq 2^{i-1}$, each $x \in X$ is contained in some $B(y, r_i)$ and hence the clustering $\mathcal{P}_i$ defines a partitioning of X. By constuction this partitioning is hierarchical. Moreover, since $r_0 < 1$, the lowest level clusters are singletons. Each level i cluster has radius at most 2^i and hence diameter no more than 2^{i+1}. Thus we get a complete hierarchical clustering satisfying (1) and (2).

Consider a level $(i+1)$ cluster $C_j \subseteq B(z, 2^{i+1})$ for some z. Any $y \in Y_i$ such that $B(y, 2^i) \cap C_j$ is nonempty must satisfy $d(y, z) \leq 3 \cdot 2^i$. Moreover, Y_i is a 2^{i-3} packing. By proposition 1, there can be at most 12^k such y's. Since each level i cluster in C_j is centered at some such y, property (3) follows.

Property (4) requires a slightly longer argument. It follows from arguments in [18, 19, 24]; we sketch a proof here. Consider a pair of points $u, v \in X$. Let us arrange the points in Y_i in increasing order of $\min\{d(u, y), d(v, y)\}$, say $y_1, y_2, \ldots,$. We say y cuts the pair (u, v) if exactly one of u and v falls in $B(y, r_i)$ and $B(z, r_i) \cap \{u, v\}$ is empty for all z satisfying $\pi(z) < \pi(y)$. For y_j to cut (u, v), r_i must fall in $[d(y, u), d(y, v)]$ and conditioned on this, $\pi(y_j)$ should be smaller than $\pi(y_k)$ whenver $k < j$. Thus the probability the y_j cuts (u, v) is at most $(\frac{d(u,v)}{2^{i-1}})(\frac{1}{j})$. Moreover, assuming w.l.o.g. $d(u, v) \leq 2^i$, by proposition 1, the number of y_j's that lie within distance 2^i of (u, v) is at most 12^k. Hence the overall probability of (u, v) being cut is at most

$$\begin{aligned} \sum_{j=1}^{12^k} \quad \Pr[y_j & \quad \text{cuts } (u,v)] \\ & \leq \quad \sum_{j=1}^{12^k} (\frac{d(u,v)}{2^{i-1}})(\frac{1}{j}) \\ & \leq \quad O(k)\frac{d(u,v)}{2^i} \end{aligned}$$

Theorem 4
The algorithm described above results in a split-tree satisfying properties (1)-(4).

4. APPROXIMATION SCHEMES

To get approximation schemes for various optimization problems, we use a randomized split tree decomposition, followed by a divide and conquer dynamic program approach.

4.1 The traveling salesman problem

In this section, we outline the approximation scheme for the traveling salesman problem on doubling metrics. The algorithm is a divide and conquer algorithm, along the lines of the algorithm for Euclidean spaces given by Arora [2]. The split tree decomposition replaces the quadtree decomposition used in the previous algorithms. The Euclidean algorithms use points on a the lower dimensional boundary of the clusters as *portals*. We instead use points in a suitably fine cover of the clusters as portals.

Let T be a split tree decomposition with depth $\delta = \log \Delta$ and a degree K at each node. For each cluster A in the tree, let N_A be a set of *portals*. We say a tour is *portal respecting* if it enters and leaves A only through points in N_A. We say a tour is *r-light* if it crosses the boundary of A at most r times. We first show how to compute the best r-light portal respecting tour with respect to a split tree and a set of portals. This is a dynamic programming algorithm, along the lines of [2]. We call a tour (path) *valid* with respect to a split tree and a set of portals, if it is both r-light and portal respecting. We call a set of paths valid if each path in the set is valid. We say a collection of paths is *A-spanning* if each point in A is visited by at least one of the paths.

The dynamic program Consider a node A in the decomposition. Any valid tour enters and leaves the set A at most r times and does so only at portals. Thus the part of the tour in A consists of an A-spanning valid collection of at most r paths, starting and ending at portals. Let (s_i, e_i) be the starting and ending points of the r^{th} segment. A *configuration* is defined to be a set of r such pairs of portals; clearly there are atmost $(N_A)^{2r}$ of them. Moreover, we have the independence necessary for writing a dynamic program: the portion of any optimum valid tour inside A is the optimum A-spanning valid collection of upto r paths, for some configuration C.

We now show how to compute, for a node A in the tree, and for each of the $|N_A|^{2r}$ configurations of entering/leaving A, the optimum A-spanning valid collection of upto r paths starting and ending at (s_i, e_i). Let $A_1, \ldots, A_K$ be the children of A in T, and suppose we have computed, for every possible configuration C_i of A_i, the best A_i-spanning valid collection of paths respecting configuration C_i.

Consider any combination $C_1, \ldots, C_K$ of configurations for $A_1, \ldots, A_K$ respectively. To compute the best C respecting combination of $C_1, \ldots, C_K$, we need to consider all ways of stitching together all the path segments in the C_i's to get r C-respecting paths. This is essentially an instance of the following problem: Given a weighted graph G and a set of r pairs of points (s_i, e_i), find the least cost G-spanning collection of paths starting and ending at (s_i, e_i). The path segments in C_i's are the internal vertices in G and there are at most Kr of them. Thus this is an optimization problem on a small graph and by enumerating all possibilities, we can compute the optimal one in time $(Kr)!$.

Thus for a configuration C, we can try out each of the combinations of the C_i's and compute the optimal one in time $\Pi_i |N_{A_i}|^{2r} \cdot (Kr)!$. Assuming each N_A is at most m, this is at most $(mKr)^{2Kr}$. The total number of table entries is at most m^{2r}, and there are only $O(n\delta)$ nodes in the tree. Thus we get an overall running time of $n\delta(mKr)^{O(Kr)}$.

Theorem 5
Given a split-tree decomposition with degree K and depth δ and a set of upto m portals for each cluster, the best r-light portal-respecting tour of X can be computed in time $n\delta(mKr)^{O(Kr)}$.

Structure Theorem We now argue that for a suitable choice of m and r, the expected cost of the best valid tour is within $(1 + \epsilon)$ times the optimal tour.

We show how to convert any tour $\mathcal{T}$ to a valid tour $\mathcal{T}''$, at small expected cost, where the expectation is taken over the randomness used in the decomposition. To get a valid tour, we need to make the tour portal respecting, and r-light. We do these in order. First consider an edge $e = (u, v)$ in the optimal tour $\mathcal{T}$. The probability that u and v fall in different level i clusters is at most $kd(u, v)/2^i$. We shall take a $\beta \cdot 2^i$ net to be our set of portals in a level i cluster. When u and v do fall in different level i clusters, we need to adjust the tour so that it enters and leaves the level i clusters at portals. Since the nearest portals to u and v are at a distance at most $\beta 2^i$, the overhead of this portalizing is at most $4\beta 2^i$, and this is incurred with probability $O(k)d(u, v)/2^i$. Thus the expected overhead is at most $\sum_i k(d(u, v)/2^i)4\beta 2^i = 4\beta k\delta d(u, v)$. Taking $\beta = \frac{\epsilon}{4k\delta}$, this overhead can be kept to at most $\epsilon d(u, v)$, and hence the overall expected cost of the resulting portal respecting tour $\mathcal{T}'$ is no more than $(1 + \epsilon)OPT$. Note that the number of portals m is then $\beta^{-k} = (4k\delta/\epsilon)^k$.

We now show how to convert $\mathcal{T}'$ to a r-light tour $\mathcal{T}''$. Unfortunately, unlike [3], we are currently unable to show how to do this for a constant r. As a result our running time is quasipolynomial (much like Arora's first result [2]). We also note that since we can only manage to make the tour r-light for $r \approx m$, the result has little significance. However, we hope the techniques used may prove useful in eventually getting a PTAS.

We first show the following simple lemma:

Lemma 6
(Small spanning trees lemma) *Let Y be a collection of r points in metric with doubling constant k and let L be the diameter of Y. Then the minimum spanning tree on Y has cost no more than $4L \cdot r^{1-\frac{1}{k}}$.*

Proof. We show how to construct a spanning tree of small cost. The minimum spanning tree (MST) is only cheaper. Consider the closest pair of points (u,v) in X. We shall add the edge (u,v) and recurse on $V \setminus \{u\}$. We now show that the closest pair of points should be close.

By the doubling property, Y can be covered by 2^k balls of diameter $L/2$, and recursing, by 2^{ik} balls of diameter $L/2^i$. For $i = \lceil(\frac{1}{k}\log_2 r)\rceil - 1$, this means that Y can be covered by (strictly) less than r balls of diameter $L/2^i = 2L \cdot r^{-\frac{1}{k}}$. Thus by the pigeonhole principle, the closest pair of points is closer than $4L \cdot r^{-\frac{1}{k}}$.

Thus, if $f(r)$ is the worst MST cost for r points, we have shown that $f(r) \le f(r-1) + 4Lr^{-\frac{1}{k}}$. The claim follows. □

We now use the patching lemma from [2]. Consider any tour that crosses a set A, r' times. The patching lemma shows that number of crossings can be reduced to 2, at an additional cost of $4MST_{r'}$, where $MST_{r'}$ is an upper bound on the cost of the minimum spanning tree on r' points in A. Let r_A be the number of times the tour $\mathcal{T}'$ crosses the set A. Whenever, $r_A > r$, we reduce the number of crossings to 2, by using the patching lemma. The cost of this is at most $8L_A \cdot r_A^{1-\frac{1}{k}}$, where the diameter of A is L. We do this for all sets A in the tree. The total cost of this is

$$\begin{aligned}\sum_{A\in T} 8L_A \cdot r_A^{1-\frac{1}{k}} &= \sum_{i=1}^{\delta} \sum_{A \text{ at level } i} 8L_A \cdot r_A^{1-\frac{1}{k}} \\ &\le 8\sum_{i=1}^{\delta} \sum_{A \text{ at level } i} 2^i r_A \cdot r^{-\frac{1}{k}} \\ &= 8\sum_{i=1}^{\delta} 2^i r^{-\frac{1}{k}} \sum_{A \text{ at level } i} r_A\end{aligned}$$

Each edge (u,v) in $\mathcal{T}$, in expectation, contributes $kd(u,v)/2^i$ to the number of level i crossings. Thus the expected number of level i crossings is at most $kOPT/2^i$. Note that the patchings can be done in a top down manner and patching inside a set A does not affect the tour outside A, and thus cannot possibly increase the number of crossings. Moreover, the patching at level i does not increase the number of crossings at level $(i-1)$ or smaller. Thus the expected cost of patching is at most $\sum_{i=1}^{\delta}(2^i r^{-\frac{1}{k}}) \cdot (kOPT/2^i) = (\delta k r^{-\frac{1}{k}})OPT$. Taking $r = (\delta k/\epsilon)^k$, we get a tour $\mathcal{T}''$ with expected cost at most $(1+\epsilon)OPT$.

Theorem 7
Given a tour τ on n points in (X,d) and a random split-tree, if $m \approx r \approx (k\delta/\epsilon)^k$, there is a valid tour τ'' such that

$$E[cost(\tau'')] \le (1+\epsilon)cost(\tau)$$

where the expectation is taken over random choice of the split-tree.

Finally, we show how to bound δ. Let us first scale the metric so that the diameter is n/ϵ. Now consider a 1-net Y of X, and consider a new instance defined by Y. Note that the optimum of this new instance is only smaller. We shall approximate this instance to within $(1+\epsilon)$. It is easy to see that given a tour of Y, a tour of X can be derived at an additional overhead of at most $2n$. Since OPT is at least the diameter of X, this overhead is at most $2\epsilon OPT$, and hence we get a $(1+O(\epsilon))$-approximation to the traveling salesman problem on X.

The net, on the other hand, has aspect ratio n/ϵ and hence δ for this instance is $\log(n/\epsilon)$. Since we require $m \approx r \approx (k\delta/\epsilon)^k$, the overall running time can be bounded by $n(k\log(n/\epsilon)/\epsilon)^{O((k\log(n/\epsilon)/\epsilon)^k)} = n2^{(\frac{k\log n}{\epsilon})^{O(k)}}$.

Theorem 8
There is a randomized algorithm that, with high probability, computes a tour of cost at most $(1+\epsilon)$ times the optimum traveling salesman tour and runs in time $n2^{(\frac{k\log n}{\epsilon})^{O(k)}}$

4.2 Derandomization via small sample space

In this section, we show how to derandomize the above algorithm by enumeration. To this end, we first show how to implement the randomized split-tree decomposition using very little randomness.

The algorithm in section 3 uses randomness at two places: in choosing the value ρ and in choosing the random permutation π. Since there are $\binom{n}{2}$ distances in the metric, there are at most that many relevant values of ρ and hence $O(\log n)$ bits of randomness suffice to pick ρ.

The permutation π requires a little more work. A look at the proof of property (4) in section 3 shows that a sufficient condition for the argument to hold is that for any set $Z \subseteq X$, $|Z| \le 12^k$ and any fixed $z \in Z$,

$$Pr_\pi[min_{w\in Z}\{\pi(w)\} = \pi(z)] = \frac{1}{|Z|}$$

Let $K = 12^k$. It is easy to see that if we use K-wise independent random 0-1 strings of length $O(\log n)$ to generate π [1], this property is guaranteed.

Moreover, we can afford to lose a constant factor in the separation probability in property (4). Thus it suffices to ensure that for any set $Z \subseteq X, |Z| \le K$ and any fixed $z \in Z$,

$$Pr_\pi[min_{w\in Z}\{\pi(w)\} = \pi(z)] \le \frac{2}{|Z|}$$

This is precisely the notion of *approximate restricted min-wise independent family* of permutations studied in [11]. They show how to construct such families using only $O(K + (\log K \log\log n))$ random bits (improved to $O(\log n)$, independent of K in [27]).

Thus we have shown how to implement the split-tree decomposition using only $O(\log n + K + \log K \log\log n)$ bits of randomness. Thus enumerating over all possible coin tosses gives us a deterministic algorithm with only a polynomial overhead in running time.

4.3 Probabilistic Embeddings and Derandomization

In this section, we interpret the approximation scheme above in a different light, leading to a different approach to derandomization. We first define the notion of a probabilistic embedding [1, 9].

Definition 1 *Given a metric (X,d), we say a distribution $\mathcal{D}$ over metrics α-*probabilistically approximates *d if for every $x,y \in X$,*

- *For every metric $d' \in \mathcal{D}$, $d'(x,y) \ge d(x,y)$*
- *$E_{d'\in\mathcal{D}}[d'(x,y)] \le \alpha \cdot d(x,y)$.*

[1] i.e. pick a string for each vertex; π is determined by the sorted order of these strings; w.h.p. there are no ties.

A graph G has tree-width at most κ if the vertices of G can be decomposed into a tree-like structure of sets of vertices, with each set having cardinality at most $\kappa + 1$. More formally,

Definition 2 *A tree-decomposition of a graph G is a pair $(T, \mathcal{X})$, where T is a tree and $\mathcal{X} = (X_t, t \in V(T))$ is a family of subsets of $V(G)$ with the following properties:*

1. $\cup_{t \in V(T)} X_t = V(G)$
2. *For every edge $e = (x, y) \in E(G)$ there exists $t \in V(T)$ such that $x, y \in X_t$*
3. *for $t, t', t'' \in V(T)$, if t' is on the path of T between t and t'' then $X_t \cap X_{t''} \subseteq X_{t'}$.*

The width of the tree-decomposition $(T, \mathcal{X})$ is defined to be $\max_{t \in V(T)}(|X_t| - 1)$.

The *tree width* of a graph is the minimum possible width of a tree decomposition of the graph. Note that under this definition, the family of tree width 1 graphs is exactly the family of trees.

We say a metric d' is a *tree width κ-metric* if it is the shortest path metric on a graph of tree width at most κ. We now show the following:

Theorem 9
Let (X, d) be a metric with doubling dimension k and aspect ratio $\Delta = 2^\delta$. Given any $\epsilon > 0$, (X, d) can be $(1 + \epsilon)$ probabilistically approximated by a family of tree width κ-metrics for $\kappa \leq 2^{O(k)} \lceil (\frac{4k\delta}{\epsilon})^k \rceil$.

Proof. We first construct the split-tree decomposition as in the section 3. We pick a suitably fine cover of each cluster in this decomposition, and use only these points (portals) to connect the points inside the cluster to points outside the cluster. More precisely, start with a complete graph on X and delete an edge leaving a cluster C unless its endpoint inside the cluster is a portal. If we had m portals per cluster, and maximum degree K in the split-tree, we would end up with a graph with tree width at most mK. Moreover, since this graph has been formed by deleting some edges from a graph representing the original metric, the distances can only increase.

The argument about the expected distortion being $(1+\epsilon)$ closely mimics the portalizing cost analysis in the TSP section. We omit the details from this extended abstract. □

The traveling salesman problem (and several other optimization problems) can be solved using dynamic programming in time polynomial in n and κ^κ on graphs of tree width κ. Moreover, as argued in [9, 10, 19], an α-probabilistic embedding couple with an exact algorithm for metrics from the host space leads to an efficient randomized α-approximation algorithm for the original metric. This gives an alternate view of the quasi-polynomial approximation scheme for the traveling salesman problem.

Derandomization This also leads to an alternate way to derandomize the approximation scheme along the lines of Charikar et.al. [14]. The idea is to obtain a distribution over a small number of low tree width metrics. Thus enumerating over all trees in the support gives a deterministic algorithm.

First note that the problem of probabilistically embedding a metric (X, d) into a family $\mathcal{L}$ of metrics (such that $d'(x, y) \geq d(x, y)$ for all $x, y \in X$, for all $d' \in \mathcal{L}$) can be written as the following linear program (with infinitely many variables)

$$\begin{aligned}
&\text{minimize } \alpha \\
&\text{subject to} \\
\sum_{d' \in \mathcal{L}} p_{d'} d'(x, y) \quad &\leq \alpha \cdot d(x, y) \quad \forall x, y \in X \\
\sum_{d' \in \mathcal{L}} p_{d'} \quad &= 1 \\
p_{d'} \quad &\geq 0 \qquad \forall d' \in \mathcal{L}
\end{aligned}$$

Since the above program has only $n^2 + 1$ constraints, it follows that there exists a solution with small support. This problem can also be looked at as a packing problem and thus the linear programming solution can be well approximated using the techniques of Plotkin, Shmoys and Tardos [41]. Charikar et.al. [14] show how to adopt the PST algorithm to give a distribution over $O(n \log n)$ metrics, if one is given a subroutine to solve the following dual problem:

Dual problem: Given metric (X, d) and weights w_{xy} on pairs of vertices, find a metric d' from the family $\mathcal{L}$ such that $\sum_{x,y \in X} w_{xy} \cdot d'(x, y) \leq \alpha \sum_{x,y \in X} w_{xy} \cdot d(x, y)$.

The algorithm outlined in the proof of theorem 9 gives a randomized algorithm for the above problem, which can be easily derandomized via the method of conditional expectation (see e.g. [12, 18, 19]).

4.4 Other problems

We note that the same techniques used here apply to other optimization problems as well. The minimum steiner tree problem has an approximation scheme similar to the one for traveling salesman above, with a quasipolynomial running time. The dynamic program for k-median and facility location problems proceeds along the lines of Arora, Raghavan and Rao [5], and gives a similar running time. We omit the details from this extended abstract.

5. WELL-SEPARATED PAIRS

Consider a metric with aspect ratio $\Delta = 2^\delta$, and assume, without loss of generality, that the smallest distance is at least 1. We assume we are given a split-tree as obtained in section 3. [2]

Let T be a split tree for X. Note that T is a tree with nodes labeled by subsets of X, with singletons at the leaves, and the set X at the root. Recall that a cluster A at level i has diameter at most 2^{i+1}. For any node A in the tree, we shall let $l(A)$ denote this upper bound on the diameter; for a singleton, we set $l(\{x\}) = 0$. We say a pair of clusters $\{A, B\}$ is *t-well separated* if $d(A, B) \geq t \cdot \max\{l(A), l(B)\}$.

We now show how to obtain a well separated pairs decomposition from this split tree decomposition. This follows an argument similar to that for the Euclidean case used by Callahan and Kosaraju [13].

We start by formally defining a well separated pair decomposition. The definition uses a separation parameter t.

Definition 3 *Let A and B be subsets of a metric X. The interaction product of A and B is defined as*

$$A \otimes B = \{\{x, y\} : x \in A \text{ and } y \in B \text{ and } x \neq y\}$$

[2] In fact, we only need properties (1)-(3) here, and a simple deterministic construction suffices.

Note that $X \otimes X$ is the set of all pairs of distinct points in X.

Definition 4 *A set* $\{\{A_1, B_1\}, \ldots, \{A_s, B_s\}\}$ *is said to be a* t-well separated pair decompostion *of* $A \otimes B$ *if*

1. $A_i \subseteq A$ *and* $B_i \subseteq B$ *for* $i = 1, 2, \ldots, s$.
2. A_i *and* B_i *are disjoint for each* i.
3. $(A_i \otimes B_i)$ *and* $(A_j \otimes B_j)$ *are disjoint for* $i \neq j$.
4. $\cup_{i=1}^{s}(A_i \otimes B_i) = A \otimes B$.
5. $d(A_i, B_i) \geq t \cdot \max\{l(A_i), l(B_i)\}$ *for each* i.

The algorithm is recursive.

```
Procedure Well-separate($A, B$)
1. If $\{A, B\}$ is $t$-well separated, the return $\{\{A, B\}\}$.
2. If $A = B = \{x\}$ for some $x \in X$, return $\Phi$.
3. Assume $l(A) \geq l(B)$ (swap if necessary).
4. Let $A_1, \ldots, A_K$ be the children of $A$ in the tree $T$.
5. Return $\cup_i$ Well-separate($A_i, B$)
```

It is easy to check that calling *Well-separate(X, X)* gives a well separated pair decomposition $\mathcal{D}$ of X. Moreover, for each pair $\{A, B\}$ in $\mathcal{D}$, both A and B are nodes in T. We first show the following simple but useful proposition.

Proposition 10
If the algorithm above makes a call to Well-separate(A, B), *then*

- $\frac{1}{2}l(A) \leq l(B) \leq 2l(A)$.
- $d(A, B) \leq 2(t + 1) \min\{l(A), l(B)\}$.

Proof. We show the claim by induction on the depth of the recursion. At the base, clearly the pair (X, X) satisfies the two properties. Now assume that the pair (A, B) satisfies the properties, and the algorithm splits A. Then we shall show that for any a child A' of A, (A', B) satisfies the property. First note that since we split A, we have $l(A) \geq l(B)$ and $d(A, B) \leq t \cdot l(A)$. Since $l(A') = l(A)/2$, we get $2l(A') \geq l(B)$. Inductively we have $l(B) \geq \frac{1}{2}l(A) = l(A')$. Thus the first property holds.

Moreover by triangle inequality, $d(A', B) \leq d(A, B) + l(A)$, since the diameter of A is at most $l(A)$. Let $l(A) = r$. Then $d(A', B) \leq tr + r = (t + 1)r$. Hence $d(A', B) \leq 2(t + 1) \min\{l(A'), l(B)\}$. Hence the claim. □

We now argue that for each node A in the tree T, there is a small number of B such that the pair $\{A, B\} \in \mathcal{D}$. We show something stronger.

Lemma 11
For any node A in T, the number of nodes B such that a call to Well-separate(X, X) *leads to a call to* Well-separate(A, B) *is at most* $(c(t + 3))^{(k_X)}$, *for some universal constant c.*

Proof. Consider a node A in T with $l(A) = 2 \cdot 2^i = r$. From the previous proposition, each such B lies within distance $2(t + 1)r$ from A. Moreover, $l(B) \in [r/2, 2r]$ and hence B is at level $(i - 1), i$ or $(i + 1)$. Consider a particular level. The centers of the B's all come from Y_{i-1} and are all at a distance closer than $2(t + 3)r$ from the center of A. Thus by proposition 1, there are at most $(16(t + 3))^k$ such B's. Hence the claim. □

Thus we have shown how to obtain a t-well separated pair decomposition containing at most $(c(t + 3))^k n\delta$ pairs. In the next section, we shall use this to construct compact representations of the metric.

6. COMPACT REPRESENTATIONS

6.1 Approximate Distance Labeling

First note that for each pair $\{A, B\}$ in the decomposition, $d(A, B)$ is a $(1 + \frac{2}{t})$ approximation to the distance between any pair $(a, b) : a \in A, b \in B$. Thus if $\{A, B\}$ is well separated, $d(A, B)$ is a good estimate for $d(a, b)$ for any $a \in A, b \in B$.

Labels For a node A in the tree, we first describe how to name it. The root has an empty string for its name. The name of the i^{th} child of a node A is formed by concatenating the name of A with a binary representation of i. Note that since each node has at most $K = 2^{O(k)}$ children, i can be described using $O(k)$ bits. Since the depth of the tree is δ, we get a $O(k\delta)$ length name *name(A)* for each node. Note that under this naming scheme, names of the ancestors of a node are prefixes of its own name.

For a node A in the tree, let $N_A = \{B : \{A, B\} \in \mathcal{D}\}$. We define *sublabel(A)* for each node A to contain the pair (*name(B)*,$d(A, B)$) for each node B in N_A. Since each N_A has size $t^{O(k)}$, each node has a sublabel of $O(kt^{O(k)}\delta)$ bits. Thus if we stored for each $x \in X$, a label *label(x)* consisting of the sublabels of each of the $O(\delta)$ sets it occurs in, we can computea $(1 + \epsilon)$ approximation to the distance between a and b given *label(a)* and *name(b)*. This is an $O(kt^{O(k)}\delta^2)$ length distance labeling.

Shorter Labels We now argue how to obtain even shorter distance labels. We shall try to shrink the sublabels of each node in the tree. Note first that for a pair $(A, B) \in \mathcal{D}$, the distance $d(A, B)$ is at least $t{\cdot}l(A)$ by the separation property. On the other hand by proposition 10, $d(A, B) \leq 2(t{+}1)l(A)$. Thus $d(A, B)$ lies in the interval $[t \cdot l(A), 4t \cdot l(A)]$ (since $t \geq 1$). Hence, $(\frac{d(A,B)}{t \cdot l(A)})$ is a small constant, and can be specified upto an error of $(1{+}\epsilon)$ by using only $O(\log \frac{1}{\epsilon})$ bits.

Shrinking the identities of the sets in N_A is more subtle, and is done at a more global level. Since we cannot store the whole name, we store in some sense, a trace or a transcript of the run of the algorithm *Well-separate*. For each node A, we argued that there are a constant number of nodes B such that we called *Well-separate(A, B)*. Now note that from the run of the algorithm, it follows that if we ever call *Well-separate(A,B)*, then for some ancestor B' of B (possibly B itself), we must have called *Well-separate($p(A), B'$)*. Moreover, from proposition 10, it follows that this ancestor should be at most three levels above B. Thus, if we knew the set of nodes B' such that we called *Well-separate($p(A), B'$)*, of which there are at most $t^{O(k)}$, we would get a set of at most $t^{O(k)}K^3$ candidates for B such that we ever call *Well-separate(A, B)* (where $K = 2^{O(k)}$ is the maximum degree of any node in T). Thus, using a constant number of bits, we can select the appropriate $t^{O(k)}$ nodes that are paired up with A (say by storing the characteristic vector, with lexicographic order on the $t^{O(k)}K^3$ nodes), and store the answer for each. Thus storing $O(t^{O(k)}K^3)$ bits for each sublabel of a node suffices. Since k and t are constants, the overall label length is then $O(\log\Delta)$ for a doubling metric with aspect ratio Δ.

Theorem 12
For a metric X with doubling dimension k and any $\epsilon > 0$, there exists a $(1+\epsilon)$-approximate distance labeling scheme with label length $((\frac{c}{\epsilon})^k k \log\frac{\Delta}{\epsilon})$ bits per node.

Corollary 13
For a metric as above, there is an $(1+\epsilon)$-approximate distance oracle with $O(n(\frac{c}{\epsilon})^k k \log\frac{\Delta}{\epsilon})$ bits of storage which can answer approximate distance queries in time $O((\frac{c}{\epsilon})^k k \log\frac{\Delta}{\epsilon})$.

6.2 Shortest Path Oracles

Consider the following problem: Given a graph $G = (V, E)$ with distances d on edges, preprocess the graph so that shortest path queries can be quickly answered. On general graphs, just computing the shortest path on demand using Dijkstra's algorithm leads to a $O(m)$ sized data structure where answering queries takes time $O(m + n\log n)$. On the other hand, by computing the all pair shortest paths, one can construct an $\tilde{O}(n^2)$ size data structure to answer such queries in $\tilde{O}(l)$ time, where l is the number of hops in the answer.

For graphs whose shortest path metric is doubling, we show how to do better: we show a data structure of size $n(\log n)^{O(1)}$ that returns a $(1+\epsilon)$-approximate shortest path in time $l(\log n)^{O(1)}$ time. When $m = \omega(n)$ or when l is $o(n)$ this significantly improves on the naive approaches above.

The basic idea is to augment the labels of the previous section with some path information. More precisely, for well-separated pair (A, B), let $a, w_1, w_2, \ldots, w_p, b$ be the shortest A-B path, where $a \in A$, $b \in B$ and $w_i \in V \setminus (A \cup B)$. We now augment the pair (*name*(B),d(A,B)) in *sub-label(A)* with *name(a)*, *name(w_1)*, *name(w_q)*, *name(w_{q+1})*, *name(w_p)* and *name(b)*, where q is chosen so that $d(a, w_q) \le \frac{d(A,B)}{2} \le d(a, w_{q+1})$.

To find the path from $u \in A$ to $v \in B$, we

1. Recursively find a path from u to a.
2. Recursively find a path from w_1 to w_q.
3. Recuresively find a path from w_{q+1} to w_p.
4. Recursively find a path from b to w.
5. Concatenate them appropriately.

We shall now show that this path has length no more than $(1+\epsilon)$ times the shortest path, for an appropriate choice of the separation parameter t.

Lemma 14
Let $t = \frac{3\delta}{\epsilon}$ for any $0 < \epsilon < \frac{1}{2}$. For any pair (u,v), the procedure described above returns a path of length no more than $(1+\epsilon)$ times the shortest path.

Proof. We claim that for any $u, v \in X$, the path from u to v returned by the above procedure has length atmost

$$d'(u,v) \quad\le\quad d(u,v)(1 + \frac{3\log(d(u,v))}{t})$$

We shall establish this by induction on $\lfloor \log_2 d(u,v) \rfloor$. In the base case, $d(u,v) \le 1$. So (u,v) must be a single edge and the claim trivially holds.

Inductively, we know that the $d'(u,a)$ and $d'(b,v)$ are each bounded by $\frac{d(u,v)}{t}(1 + \frac{3\log(d(u,v)/t)}{t})$. Since $t \ge 6\delta$, we get $d'(u,a) + d'(b,v) \le 3\frac{d(u,v)}{t}$.

On the other hand, the selected paths from w_1 to w_q and w_{q+1} to w_p inductively have lengths at most $(1+\frac{3\log(d(u,v)/2)}{t})$ times their shortest distances. The total cost of the selected path is

$$\begin{aligned}
d'(u,v) \quad &\le\quad d'(u,a) + d(a,v_1) + d'(v_1,v_q) + d(v_q,v_{q+1}) \\
&\qquad + d'(v_{q+1},v_p) + d(v_p,b) + d'(b,v) \\
&\le\quad \frac{3d(u,v)}{t} + d(a,v_1) + d(v_1,v_q) \\
&\qquad + d(v_q,v_{q+1}) + d(v_{q+1},v_p) + d(v_p,b) \\
&\qquad + (1 + \frac{3(\log d(u,v)/2)}{t})(d(v_1,v_q) + d(v_{q+1},v_p)) \\
&\le\quad d(u,v)(1 + \frac{3\log d(u,v)}{t}).
\end{aligned}$$

Hence the induction holds.

The claim follows by plugging in the value of t and noting that the diameter of the metric is at most 2^δ. □

Plugging in the value of t in Lemma 11, we get $O(n\delta(\frac{3\delta}{\epsilon})^k)$ well separated pairs in all, leading to a data structure of total size $O(nk(\frac{3\log\Delta}{\epsilon})^k(\log^2\Delta))$. Optimizing the parameters (number of intermediate hops stored), we can improve the bound to $O(nk(\frac{6\log\Delta}{\epsilon k})^k(\log^2\Delta))$.

Theorem 15
Given a weighted graph $G = (V, E)$ inducing a metric with doubling dimension k and aspect ratio Δ, it can be preprocessed to give a data structure of size $O(nk(\frac{6\log\Delta}{\epsilon k})^k(\log^2\Delta))$, such that $(1+\epsilon)$-approximate shortest paths can then be computed in time $\tilde{O}(l)$ time, where l is the number of hops in the path output by the algorithm.

6.3 Compact Routing Schemes

Note that the scheme described in the previous section is very local in nature. To route from u to v, we used the augmented label of u and the name of v to figure out $a, w_1, w_q, w_{q+1}, w_p$ and b. Thus if each node in a network stored its label, we can use a stack based routing protocol. A node u, when it sees v on the top of the stack, pops it out and pushes $(v, b, w_p, w_{q+1}, w_q, w_1, a)$ onto the stack and continues. If it has an edge to v, it just pops off v and send the packet along the edge to v.

The maximum stack depth is easily seen to be $O(\log\Delta)$ (with each entry being a name of $O(\log\Delta)$ bits). The amount of routing information each node needs to store is only poly-logarithmic. The stretch is guaranteed to be at most $(1+\epsilon)$.

Theorem 16
Given a weighted graph $G = (V, E)$ inducing a metric with doubling dimension k and aspect ratio Δ on V, there exists a routing scheme with stretch $(1+\epsilon)$ that requires only $O(k(\frac{6\log\Delta}{\epsilon k})^k(\log^2\Delta))$ bits of routing information per node, and has $O(k\log^2\Delta)$ bit headers.

This improves on a result of Thorup and Zwick [45] for general graphs where they get stretch $(2k-1)$ and $\tilde{O}(n^{\frac{1}{k}})$ storage per node for any integer $k \ge 2$ (The minimum stretch is thus 3).

6.4 Lower Bounds

We note that our distance labels have size exponential in the doubling dimension k. We now show that this dependence is necessary. Note that in contrast, for a d-dimensional normed space, the coordinates (or rather, an appropriate truncation) serve as approximate distance labels of length linear in the dimension.

THEOREM 17
For any k and any large enough n there exists an n-point metric X with doubling dimension k such that any $2 - \epsilon$ approximate distance oracle requires storage $\Omega(n2^k)$.

PROOF. Consider the set of all 1-2 metrics over $K = 2^k$ points, i.e. the set of all metrics on the set $\{1, 2, \dots, n\}$ such that $d(i, j) = 1$ or 2 for any $i \neq j$. It is easy to check that every distance function in this set is a metric. Moreover, the number of such metrics is $2^{\binom{K}{2}}$.

To get an asymptotic family with fixed doubling dimension, one can place such metric on a large line metric, i.e. consider the metric over points $[K]\times[m]$, where $d((i, j), (i', j')) = M|j-j'|$ if $j \neq j'$ and chosen from $\{1, 2\}$ otherwise, for some large enough M. The doubling dimension of this metric is still k (while, it has $n = Km$ points).

Now consider any $(2-\epsilon)$-approximate distance oracle with R bits of storage. The number of such oracles is 2^R. Since the distortion between any two distinct metrics in the original class is 2, any $(2 - \epsilon)$-distance labeling must result in a distinct oracle for each metric in the class. Thus $2^R \geq 2^{m\binom{K}{2}}$, and hence R must be at least $mK(K - 1)/2 = \Omega(n2^{k_X})$. □

COROLLARY 18
Any $(2 - \epsilon)$-approximate distance labeling scheme requires label length exponential in the doubling dimension.

COROLLARY 19
There exists a metric X such that any distortion $(2 - \epsilon)$ embedding of X into ℓ_p^d, must have $d = 2^{\Omega(k_X)}$.

PROOF. Consider any such embedding of a 1-2 metric in the previous theorem. It is easy convert it into an embedding where each coordinate of each vertex has absolute value at most 2. If we round each coordinate after $\log \frac{d^{\frac{1}{p}}}{\epsilon}$ bits, the change in any distance is at most ϵ. Since this gives a $(1+\epsilon)$-approximate distance labeling with $O(d \log d)$ bits per label, the claim follows. □

7. CONCLUSIONS

We have shown that some important algorithmic and structural properties of Euclidean spaces are shared by doubling metrics. What other properties of Euclidean spaces are shared, and what properties depend intrinsically on the spatial structure?

Our approximation scheme for the traveling salesman problem has a quasipolynomial running time. On the other hand, the known algorithms for the TSP in d-dimensional Euclidean metrics run in time $O(n \log n) + 2^{\frac{1}{\epsilon}^{O(d)}}$. Moreover as shown be Trevisan [46], this doubly exponential dependence on dimension in inherent, unless NP has subexponential algorithms. Is our worse dependence on dimension necessary, or can the algorithms above be modified to run faster?

8. ACKNOWLEDGEMENTS

I would like to thank Christos Papadimitriou, Satish Rao, James Lee, Robert Krauthgamer and Santosh Vempala for helpful discussions. I would also like to thank Sariel Har-Peled and Piotr Indyk for making available preprints of [25] and [28] respectively.

9. REFERENCES

[1] N. Alon, R. M. Karp, D. Peleg, and D. West. A graph-theoretic game and its application to the k-server problem. *SIAM J. Comput.*, 24(1):78–100, 1995.

[2] S. Arora. Polynomial time approximation schemes for Euclidean TSP and other geometric problems. In *37th Annual Symposium on Foundations of Computer Science*, pages 2–11, Burlington, Vermont, 14–16 Oct. 1996. IEEE.

[3] S. Arora. Polynomial time approximation schemes for Euclidean traveling salesman and other geometric problems. *J. ACM*, 45(5):753–782, 1998.

[4] S. Arora. Approximation schemes for NP-hard geometric optimization problems: a survey. *Mathematical Programming*, 97(1–2):43–69, 2003.

[5] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for Euclidean k-medians and related problems. In *ACM Symposium on Theory of Computing (STOC)*, 1998.

[6] P. Assouad. Plongements lipschitziens dans $\mathbf{R}^n$. *Bull. Soc. Math. France*, 111(4):429–448, 1983.

[7] B. Awerbuch, A. B. Noy, N. Linial, and D. Peleg. Improved routing strategies with succinct tables. *J. Algorithms*, 11(3):307–341, 1990.

[8] B. Awerbuch and D. Peleg. Routing with polynomial communication-space trade-off. *SIAM J. Discret. Math.*, 5(2):151–162, 1992.

[9] Y. Bartal. Probabilistic approximation of metric spaces and its algorithmic applications. In *37th Annual Symposium on Foundations of Computer Science*, pages 184–193. IEEE, 1996.

[10] Y. Bartal. On approximating arbitrary metrices by tree metrics. In *30th Annual ACM Symposium on Theory of Computing*, pages 161–168. ACM, 1998.

[11] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659, 2000.

[12] G. Calinescu, H. Karloff, and Y. Rabani. Approximation algorithms for the 0-extension problem. In *12th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 8–16. SIAM, 2001.

[13] P. B. Callahan and S. R. Kosaraju. A decomposition of multidimensional point sets with applications to k-nearest-neighbors and n-body potential fields. *J. ACM*, 42(1):67–90, 1995.

[14] M. Charikar, C. Chekuri, A. Goel, S. Guha, and S. Plotkin. Approximating a finite metric by a small number of tree metrics. In *Proceedings of the 39th Annual Symposium on Foundations of Computer Science*, page 379. IEEE Computer Society, 1998.

[15] K. L. Clarkson. Nearest neighbor queries in metric spaces. *Discrete Comput. Geom.*, 22(1):63–93, 1999.

[16] L. J. Cowen. Compact routing with minimum stretch. *J. Algorithms*, 38(1):170–183, 2001.

[17] T. Eilam, C. Gavoille, and D. Peleg. Compact routing schemes with low stretch factor (extended abstract). In *Proceedings of the seventeenth annual ACM symposium on Principles of distributed computing*, pages 11–20. ACM Press, 1998.

[18] J. Fakcharoenphol, C. Harrelson, S. Rao, and K. Talwar. An improved approximation algorithm for the 0-extension problem. In *ACM-SIAM Symposium on Discrete Algorithms*. 2003.

[19] J. Fakcharoenphol, S. Rao, and K. Talwar. A tight bound on approximating arbitrary metrics by tree metrics. In *35th Annual ACM Symposium on Theory of Computing*. ACM, 2003.

[20] C. Gavoille, M. Katz, N. A. Katz, C. Paul, and D. Peleg. Approximate distance labeling schemes. In *Proceedings of the 9th Annual European Symposium on Algorithms*, pages 476–487. Springer-Verlag, 2001.

[21] C. Gavoille and D. Peleg. Compact and localized distributed data structures. *Distrib. Comput.*, 16(2-3):111–120, 2003.

[22] C. Gavoille, D. Peleg, S. Perennes, and R. Raz. Distance labeling in graphs. In *ACM-SIAM Symposium on Discrete Algorithms*, pages 210–219, 2001.

[23] J. Gudmundsson, C. Levcopoulos, G. Narasimhan, and M. Smid. Approximate distance oracles for geometric graphs. In *Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pages 828–837. Society for Industrial and Applied Mathematics, 2002.

[24] A. Gupta, R. Krauthgamer, and J. Lee. Bounded geometries, fractals and low-distortion embeddings. In *Foundations of Computer Science*, 2003.

[25] S. Har-Peled and S. Mazumdar. Coresets for k-means and k-median clustering and their applications. In *These Proceedings*, 2004.

[26] K. Hildrum, J. Kubiatowicz, S. Ma, and S. Rao. A note on finding the nearest neighbor in growth-restricted metrics. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 553–4, 2004.

[27] P. Indyk. A small approximately min-wise independent family of hash functions. *Journal of Algorithms*. To appear, also appeared in Proc. 10th Symposium on Discrete Algorithms, 1999.

[28] P. Indyk. Algorithms for dynamic geometric problems over data streams. In *These Proceedings*, 2004.

[29] D. Karger and M. Ruhl. Finding nearest neighbors in growth-restricted metrics. In *34th Annual ACM Symposium on the Theory of Computing*, pages 63–66, 2002.

[30] S. Kolliopoulos and S. Rao. A nearly linear-time approximation scheme for the Euclidean k-median problem. In *ESA: Annual European Symposium on Algorithms*, 1999.

[31] R. Krauthgamer and J. Lee. Navigating nets: simple algirithms for proximity search. In *Proceedings of the 15th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2004.

[32] R. Krauthgamer and J. Lee. Nearest neighbour search in the blackbox model. Manuscript, 2004.

[33] T. J. Laakso. Plane with A_∞-weighted metric not bi-Lipschitz embeddable to $\mathbb{R}^N$. *Bull. London Math. Soc.*, 34(6):667–676, 2002.

[34] U. Lang and C. Plaut. Bilipschitz embeddings of metric spaces into space forms. *Geom. Dedicata*, 87(1-3):285–307, 2001.

[35] N. Linial, E. London, and Y. Rabinovich. The geometry of graphs and some of its algorithmic applications. *Combinatorica*, 15(2):215–245, 1995.

[36] J. S. B. Mitchell. Guillotine subdivisions approximate polygonal subdivisions: A simple polynomial-time approximation scheme for geometric TSP, k-MST, and related problems. *SIAM Journal on Computing*, 28(4):1298–1309, Aug. 1999.

[37] T. S. E. Ng and H. Zhang. Predicting internet network distance with coordinates-based approaches. In *21st Annual Joint Conference of the IEEE Computer and Communications Society (INFOCOM-02)*, pages 170–179, 2002.

[38] D. Peleg. *Distributed computing: a locality-sensitive approach*. Society for Industrial and Applied Mathematics, 2000.

[39] D. Peleg. Proximity preserving labeling schemes. *Journal of Graph Theory*, 33:167–176, 2000.

[40] D. Peleg and E. Upfal. A trade-off between space and efficiency for routing tables. *J. ACM*, 36(3):510–530, 1989.

[41] S. A. Plotkin, D. B. Shmoys, and &#201;va Tardos. Fast approximation algorithms for fractional packing and covering problems. *Math. Oper. Res.*, 20(2):257–301, 1995.

[42] S. B. Rao and W. D. Smith. Approximating geometrical graphs via spanners and banyans. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 540–550. ACM Press, 1998.

[43] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[44] M. Thorup and U. Zwick. Approximate distance oracles. In *Proceedings of the thirty-third annual ACM symposium on Theory of computing*, pages 183–192. ACM Press, 2001.

[45] M. Thorup and U. Zwick. Compact routing schemes. In *Proceedings of the thirteenth annual ACM symposium on Parallel algorithms and architectures*, pages 1–10. ACM Press, 2001.

[46] L. Trevisan. When Hamming meets Euclid: The approximability of geometric TSP and steiner tree. *SICOMP: SIAM Journal on Computing*, 30, 2000.