Sublinear Time Algorithms for Metric Space Problems

PIOTR INDYK *
Computer Science Department
Stanford University
(650)-723-4532
indyk@cs.stanford.edu

June 21, 2000

Abstract

In this paper we give approximation algorithms for the following problems on metric spaces: Furthest Pair, k-median, Minimum Routing Cost Spanning Tree, Multiple Sequence Alignment, Maximum Traveling Salesman Problem, Maximum Spanning Tree and Average Distance. The key property of our algorithms is that their running times is *linear* in the number of points. As the full specification of an n-point metric space is of size $\Theta(n^2)$, the complexity of our algorithms is *sublinear* with respect to the input size. All previous algorithms (exact or approximate) for these problems have running times $\Omega(n^2)$. We believe that our techniques can be applied to get similar bounds for other problems.

1 Introduction

In recent years there has been a dramatic growth of interest in algorithms operating on massive data sets. This poses new challenges for algorithm design, as algorithms which are quite efficient on small inputs (for example, having quadratic running time) can become prohibitively expensive for input sizes of, say, several gigabytes. Fortunately, applications like similarity search, clustering, data visualisation and representation [8] often demand answers which are not necessarily entirely accurate or complete. This makes approximate computation a promising tool for coping with large data sizes.

There was a lot of recent algorithmic research along these lines. This includes research on approximate nearest neighbor [5, 15, 13, 16, 12, 1] which shows that efficient algorithms can be designed even for large high dimensional data sets. In particular, the results of [13, 12] show that almost linear time/space algorithms can be designed for approximate proximity problems, e.g., finding near(est) points, closest pair of points, minimum spanning tree or other variants of hierarchical clustering (cf. [1]). Unfortunately, these algorithms are restricted to specific (usually geometric) metric spaces, making them unsuitable for non-geometric ones like edit distance (used in molecular biology) or other domain-specific measures. It is therefore desirable to consider these problems in general metric spaces.

In this paper we address these issues by providing approximation algorithms for the following metric space problems: k-median, Minimum Routing Cost Spanning Tree (MRCST)/Multiple Sequence Alignment (MSA), Maximum Traveling Salesman Problem (MaxTSP), Maximum Spanning

*Supported by Stanford Graduate Fellowship and NSF Grant IIS-9811904

problem	running time	approximation factor	output	objective
Furthest pair	$O(n)$	$\frac{1}{2}$	points p, q	$\max d(p,q)$
k-median	$\tilde{O}(nk/\delta^2)$	$[(1+\delta)(2+4\alpha), 2\beta]$	points $c_1 \ldots c_k$	$\min \sum_p \min_i d(p, c_i)$
1-median	$O(n/\delta^5)$	$1+\delta$		
MaxTSP	$O(n/\delta)$	$\frac{1}{2}-\delta$	a tour T	$\max \sum_{e \in T} d(e)$
MaxST	$\tilde{O}(n/\delta)$	$\frac{1}{2}-\delta$	a tree T	$\max \sum_{e \in T} d(e)$
MRCST	$O(n/\delta)$	$2+\delta$	a tree T	$\min \sum_{p,q} d_T(p,q)$
AvgD	$O(n/\delta^{7/2})$	$1+\delta$	$\frac{1}{\binom{\lvert X \rvert}{2}} \sum_{p,q} d(p,q)$	

Table 1: The result table. The notation $[a, b]$ means the algorithm returns bk median with cost a times the optimal k-median cost. Our algorithm assumes existence of a $\tilde{O}(n^2)$-time algorithm with $[\alpha, \beta]$ approximation guarantees (where, as usual, the notation $\tilde{O}()$ omits factors logarithmic in n and other variables). In particular, Jain and Vazirani [14] give a $(6, 1)$-approximation algorithm, while Charikar and Guha [2] give a $(1+\gamma, 1+2/\gamma)$-approximation.

Tree (MaxST) and Average Distance (AvgD) (see the definitions in Section 2). The exact running times of the algorithms are given in Table 1. The crucial property of our algorithms is that they run in time *linear*[1] in n. As the size of the metric space is $\binom{n}{2}$, their complexity is in fact *sublinear* with respect to the input size. We also provide lower bounds for sublinear time randomized algorithms for several problems (see page 9). Notice that for MaxST, MaxTSP and Furthest Pair the upper and lower bounds match.

Our techniques (described in more detail at the end of this section) are quite general and their applicability does not seem restricted to the problems considered here. Therefore, we expect that similar results can be obtain for other problems on metric spaces. We also note that our algorithms are easy to implement and have small constants in the running time. In fact our k-median algorithm is very similar to the Scatter/Gather clustering procedure by Cutting et al [4]. To the best of our knowledge we provide the first formal analysis of that algorithm (or in fact of any sampling-based clustering algorithm).

In the remainder of this section, we first motivate the problems considered in this paper. Then we discuss the relation of our work to several notions developed recently in the theory of algorithms, in particular to property testing and spot checking. Finally, we give an overview of our techiques.

The problems. The k-median problem is of interest in many areas such as facility location, information retrieval and data mining. In the latter two areas the value of k is usually much smaller than n and therefore our algorithm offers a significant improvement in running time over previous $\Omega(n^2)$-time algorithms. The next two problems (MaxTSP and MaxST) are maximization versions of classical optimization problems; unlike the minimization problems (see page 9) they can be approximated up to a small constant factor. They are included mainly because of theoretical interest. The next problem (MRCST) has applications to network design [22] and molecular biology. The latter application follows from its connection to Multiple Sequence Alignment (MSA), where the goal is to find a common alignment of n genetic sequences. Specifically, Gusfield [9] showed that any tree within cost at most c times the sum of all distances between the sequences can be used to find an alignment within factor c away from optimal. The alignment computation step is linear in n and thus our tree computation procedure speeds up Gusfield's algorithm as well. Finally, the last algorithm (for the Average Distance problem) is a useful tool for gathering statistics over

[1]If a constant probability of success is required; otherwise the running time is multiplied by $\log 1/p$ where $1-p$ is the required probability of success

the metric space. For example, we can apply it to estimate the gap between the actual solution to MSA and its lower bound.

Property testing and spot checking. In the last two decades there has been significant work on estimating the complexity of checking *graph* properties. In particular, Rivest and Vuillemin [21] showed that the problem of deciding of *any* non-trivial monotone graph property requires inspection of at least $\Omega(n^2)$ edges; the same bound is conjectured to hold for randomized algorithms as well. Very few non-trivial graph properties have subquadratic complexity.

The notion of approximate property testing (introduced by Goldreich et al [7]) is a relaxation of the above definition. Specifically, it is assumed that the input graph either satisfies a property P or is within distance at least ϵn^2 from any graph satisfying P; the distance between two graphs is the Hamming distance between their adjacency matrices. Goldreich et al [7] showed many interesting results on complexity of testing approximate properties; in particular they obtained $\text{poly}(1/\epsilon)$ and $\exp(1/\epsilon)$ bounds on the complexity of colorability, clique, cut and bisection problems. Their work was later extended by Ergun et al [6], who introduced the notion of *spot-checking*. They applied it to a variety of problems on graphs, sets and algebra obtaining algorithms with running times varying from $O(\log n \text{poly}(1/\epsilon))$ to $O(\sqrt{n}\text{poly}(1/\epsilon))$, where n is the input/output size and ϵn is the distance from the correct solution.

The main difference between approximate property testing/spot-checking and our approach is that we do not need to relax the original problem w.r.t the "closeness" to the correct solution. Rather than that, we relax the quality of the output, which is a more standard way of defining approximate versions of problems.[2] On the other hand, our techniques seem to be limited to problems over metric spaces only.

Our techniques. All of our algorithms employ random sampling. There are two basic properties of a random sample of a metric space points which we use. The first property is that large distances "cannot hide" in a metric space (more specifically, if there is any pair of vertices of distance Δ, then there are at least n other pairs of distance at least $\Delta/2$ (by triangle inequality). Thus we can spot one such a pair by random sampling or inspection of a neighborhood of one vertex (this for example gives an immediate solution to the Furthest Pair problem). Another property we use is that a randomly sampled edge e from a set S has expected length $d(e)$ equal to the average weight of edges in S. This can be used for finding approximation of S or its cost, provided that we can prove additional properties of the expectation or variance of $w(e)$.

2 Preliminaries

Assume we are given a metric space (X, d), such that $|X| = n$. The problems are then defined as follows.

Definition 1 (k-median): *Find k* medians $c_1 \dots c_k \in X$ *which minimize the value of*

$$\sum_{p \in X} \min_{i=1\dots k} d(p, c_i).$$

The problem is MAX SNP-hard. Lin and Vitter [19, 20] gave a bicriterion $[1+\epsilon, 2(1+1/\epsilon)]$-approximation algorithm for any $\epsilon > 0$, i.e. the algorithm outputs $2(1+1/\epsilon)k$ medians with cost at most $(1+\epsilon)$ times the minimum k-median cost. Their algorithm uses linear programming and its running time is a large degree polynomial in n. Charikar et al [3] gave a $[1, O(1)]$-approximation

[2] In some cases these two approaches coincide; for example it was shown in [7] that by using their techniques one can obtain a $(1+\epsilon)$-approximation of MAX-CUT in dense graphs in time linear in the number of vertices.

algorithm for this problem, also via linear programming. Of particular importance to our paper are two recently developed algorithms with $\tilde{O}(n^2)$-time: the $[1, 6]$-approximation by Jain and Vazirani [14], and the $[1+\gamma, 1+2/\gamma]$-approximation by Charikar and Guha [2].

We also mention that a related *k-center* problem (where the goal is to minimize the *maximum* cluster radius) can be solved in (sublinear) $\tilde{O}(kn)$ time [11, 10].

Definition 2 (Maximum Spanning Tree - MaxST): *Find a spanning tree T of X such that the sum of its edge lengths is maximized.*

This problem can be solved exactly in $O(n^2)$-time by minimum spanning tree techniques.

Definition 3 (Maximum Traveling Salesman Problem - MaxTSP): *Find a simple cycle of length n such that the sum of its edge lengths is maximized.*

The problem is MAX SNP-hard. Kosaraju et al [18] gave a $\frac{5}{7}$-approximation algorithm for this problem.

Definition 4 (Minimum Routing Cost Spanning Tree - MRCST): *Find a spanning tree T such that the metric d_T induced by T minimizes the value of*

$$\sum_{p,q \in X} d_T(p,q).$$

The problem is NP-hard. Wong [23] showed how to construct (in $O(n^2)$-time) a tree with cost at most twice the sum of pairwise distances of the *original* metric, thus also at most twice of the minimum RCST cost. Wu et al [22] gave a $(1+\epsilon)$-approximation algorithm running in time $n^{O(1/\epsilon)}$.

Definition 5 (Average Distance - AvgD): *Compute* $\frac{1}{\binom{|X|}{2}} \sum_{p<q \in X} d(p,q)$.

The problem can be solved trivially in $O(n^2)$-time.

3 1-median

In this section we present an $O(n/\delta^5)$-time $(1+\delta)$-approximation algorithm for the 1-median problem.

For a given point p let $S(p) = \sum_{q \in X} d(p,q)$. The main part of our algorithm is a $O(1/\delta^5)$-time procedure which given any two points p and q performs an approximate comparison of $S(p)$ and $S(q)$. More specifically, if $S(p) > (1+\delta)S(q)$, then with probability $2/3$ (say) the algorithm will return p; if $S(q) > (1+\delta)S(p)$ then with same probability the algorithm will return q; otherwise the output is arbitrary. Using this probabilistic comparison as a subroutine, we can approximate the smallest $S(q)$ using $O(n)$ comparisons.

The comparison procedure is as follows. Assume that $\delta < \frac{1}{2}$. Let $t = d(p,q)$, let $r = 4t/\delta + t$ and let B be the set of points within distance r from p. The algorithm starts from estimating the value of $\gamma = \frac{|B|}{n}$. More specifically, it chooses uniformly at random a set S of $s = O(1/\delta)$ points from X and computes $\gamma' = \frac{|S \cap B|}{|S|}$. Then it checks if $\gamma' > \delta/5$; if so, it concludes that $\gamma > \delta/6$, otherwise, it concludes that $\gamma < \delta/4$ (the constants are chosen mainly for clarity of exposition). It is easy to see that with large constant probability the conclusions are correct. In the following, we show that in both cases we can quickly approximately compare $S(p)$ and $S(q)$. In the first case (if $\gamma \geq \delta/6$) then we can easily sample points from B. Moreover, the distances $d(u,p)$ and $d(u,q)$ for $u \in B$ are bounded from above; also we show that the larger of $S(p)$ and $S(q)$ can be bounded from below. Therefore we can estimate the sum of the distances within B by random sampling (the distances outside of B are again similar for p and q). In the second case, most points $u \in X$ are so far away from both p and q that the difference between $d(p,u)$ and $d(q,u)$ is relatively small, which implies that $S(p)$ and $S(q)$ are close to each other and any choice is correct.

Consider first the case when $\gamma < \delta/4$. In this case we show that the difference between $S(p)$ and $S(q)$ is negligible (i.e. smaller than $\delta \cdot \min(S(p), S(q))$), so any choice made by the algorithm is correct. The argument is as follows. For $v \in X$ define $S_1(v) = \sum_{u \in B} d(v,u)$ and $S_2(v) = \sum_{u \notin B} d(v,u)$. Observe that $S(p) = S_1(p) + S_2(p)$ and $S(q) = S_1(q) + S_2(q)$. Notice that due to the fact that for any $u \in X$ the triangle inequality implies $|d(p,u) - d(q,u)| \leq d(p,q) = t$, we have

$$\Delta = |S(p) - S(q)| \leq n \cdot t \leq \frac{\delta}{4} rn.$$

On the other hand $S(p) \geq S_2(p) \geq (1-\gamma)n(r-t) \geq (1-\frac{\delta}{4})nr$ and the same bound holds for $S(q)$. Thus one can verify that indeed $\Delta \leq \delta \min(S(p), S(q))$.

Consider now the case when $\gamma > \delta/6$. In this case we first choose a random sample R of l points from B; notice that by choosing $O(l/\delta)$ random elements of X and rejecting those which do not belong to B, we can obtain such a sample in $O(l/\delta)$ time with constant probability. Then we compute $S'(p) = \sum_{u \in R} d(p,u)$ and $S'(q) = \sum_{u \in R} d(q,u)$ and choose p if $S'(p) > S'(q)$ or q in the opposite case.

To prove correctness of the above procedure, we first observe that if (say) $S(p) > (1+\delta)S(q)$, then $S_1(p) > (1+\delta)S_1(q)$, as otherwise we would have

$$S(p) = S_1(p) + S_2(p) \leq (1+\delta)S_1(q) + (1+\delta/4)S_2(q) \leq (1+\delta)S(q).$$

Thus if we show that $S'(p)$ estimates $\frac{l}{|B|}S_1(p)$ (and $S'(q)$ estimates $\frac{l}{|B|}S_1(q)$) with an additive error at most

$$\frac{\delta}{3}\frac{l}{|B|}\max(S_1(p), S_1(q))$$

we are done. Since the expected value of $S'(p)$ is equal to $\frac{l}{|B|}S_1(p)$ and a similar fact holds for q, we just need to bound the variance of $S'(p)$ and $S'(q)$ with respect to their expectations. To this end we observe that both variances $D^2[S'(p)]$ and $D^2[S'(q)]$ are bounded by lr^2. On the other hand, we know that for any point $u \in X$ we have $d(p,u) + d(u,q) \geq d(p,q) = t$ and thus

$$\max(S_1(p), S_1(q)) \geq \frac{S_1(p) + S_1(q)}{2} \geq |B|t/2$$

and therefore $\max(E[S'(p)], E[S'(q)]) \geq lt/2$. Recall that we assume $S_1(p) > S_1(q)$. By Chebyshev inequality

$$\Pr[|S'(p) - E[S'(p)]| \geq a \cdot \sqrt{l}r] \leq \frac{1}{a^2}.$$

If we choose $l > a^2 4(4/\delta+1)^2 \cdot 9/\delta^2$ then $\sqrt{l}\frac{\delta}{3}\frac{t}{2} \geq ar$ and the deviation of $S'(p)$ from $E[S'(p)]$ can be bounded by

$$a \cdot \sqrt{l}r \leq \frac{\delta}{3}\frac{t}{2}l \leq \frac{\delta}{3}E[S'(p)] = \frac{\delta}{3}S_1(p)\frac{l}{|B|}$$

which was to be shown. The same error bound can be obtained for $S'(q)$ as well. This completes the proof of correctness for the second case and the whole comparison procedure.

Using the probabilistic comparator as a subroutine, we can easily get an $O(n\text{poly}(\log n, 1/\delta)$-time algorithm for $(1+\delta)$-approximate 1-median in the following way. Construct a binary tournament tree over the points in X, where each internal node performs a comparison of values of the function $S()$ of its children and selects the smaller one. Set the comparison quality parameter δ' to $\delta/\log n$ and perform each comparison $O(\log n)$ times, thus achieving a high probability of correctness of all comparisons. Since the smallest element $S(p)$ is compared at most $\log n$ times, the value of the selected element $S(q)$ is bounded by $(1+\delta')^{\log n} \le 1 + O(\delta)$.

We can achieve a better running time by applying the randomized tournament technique by Kleinberg [15]. The details are very similar, thus we omit the description here.

4 k-median

In this section we present an $O(n)$-time approximation algorithm for the k-median problem. Our procedure uses a black-box $[\alpha, \beta]$-approximation $\tilde{O}(n^2)$-time algorithm, which we refer to as BB. Let $s = a\sqrt{kn\log k}$ for $a > 1$ determined later. The algorithm is as follows.

1. Choose a set S of s points sampled without replacement from X
2. Run the BB k-median algorithm on S, let $C' = c'_1 \ldots c'_{\beta k}$ denote the output
3. Assign each point $p \in X$ to a point in C' within smallest distance from p; let $d(p, C')$ denote that distance
4. Select the set M containing the points p with $m = b\frac{kn}{s}\log k$ largest values of $d(p, C')$, for b determined later
5. Run the BB algorithm on M, let C'' denote the output
6. Output C' and C''

In the following we will use $a = \Theta(1/\delta\sqrt{\log 1/\delta})$ and $b = \Theta(1/\delta^2 \log 1/\delta)$. It is easy to verify that the running time of the algorithm is $\tilde{O}(kn/\delta^2)$. It remains to prove its correctness.

Theorem 1 *For any constant $\delta > 0$ the above algorithm computes a $[(1+\delta)3(2+\alpha), 2\beta]$-approximate solution for the k-median problem with probability $\Omega(\delta)$.*

By running the above procedure $O(1/\delta)$ times and taking the solution with the smallest cost, we can reduce the error probability to any constant.

Proof: The 2β factor is clear, thus we focus on proving the first bound. Let $c_1 \ldots c_k$ denote the optimal medians and let D denote the optimal cost. Let C_i denote the set of points in X closer to c_i than to any other c_j (for simplicity assume there are no ties). Also, let $C'_i = S \cap C_i$. Let $t = b\frac{n}{s}\log k$. Let $L = \{i : |C_i| \ge t\}$, $l = \sum_{i \in L}|C_i|$ and $l' = \sum_{i\in L}|C'_i|$. Clearly $l \ge n - kt$.

Claim 1

$$\Pr[\sum_{i \in L}\sum_{p \in C'_i} d(c_i, p) \ge \frac{s}{n}(1+\epsilon)\sum_i \sum_{p \in C_i} d(c_i, p)] \le \frac{1}{1+\epsilon}$$

Proof: Consider any sampled element p. With a probability of $\frac{l}{n}$ this element belongs to C_i for $i \in L$. Conditioned on this event, the expected value of $d(p, C)$ is $\frac{1}{l}\sum_{i\in L}\sum_{p\in C_i} d(c_i, p)$. Thus the expected value of $\sum_{i\in L}\sum_{p\in C_i'} d(c_i, p)$ is $s\frac{1}{l}\frac{l}{n}\sum_i\sum_{p\in C_i} d(c_i, p)$. The Claim follows from Markov inequality. □

Claim 2 *For any $\gamma > 0$ we have*

$$\Pr[\textit{for every } i \in L, \frac{|C_i|}{|C_i'|} \leq \frac{n}{s}(1+\gamma)] \geq 1 - 2ke^{\frac{-st}{3n\gamma^2}}$$

Proof: Follows from Chernoff bound. □

Assume that both assertions of the above Claims hold (which is true with probability $\Theta(\epsilon)$ for a proper choice of a and b). We show that the centers C' provide a good solution for all points from $\cup_{i\in L} C_i$. Fix any such C_i and consider any function $q_i : C_i \to C_i'$ such that any element from C_i' has at most $\frac{|C_i|}{|C_i'|}$ elements assigned to it. For any $q \in C_i'$ let $c'(q)$ denote the median c_i' closest to q. By triangle inequality

$$d(p, C') \leq d(p, c_i) + d(c_i, q_i(p)) + d(q_i(p), c'(q_i(p))).$$

Thus

$$\begin{aligned}
\sum_{i\in L}\sum_{p\in C_i} d(p, C') &\leq \sum_{i\in L}[\sum_{p\in C_i} d(p, c_i) \\
&\quad + \frac{|C_i|}{|C_i'|}\sum_{q\in C_i'}(d(c_i, q) + d(q, c'(q)))] \\
&\leq D + \frac{n}{s}(1+\gamma)(1+2\alpha)\sum_{i\in L}\sum_{p\in C_i'} d(c_i, p) \\
&\leq D + \frac{n}{s}(1+\gamma)(1+2\alpha)(1+\epsilon)\frac{s}{n}D \\
&= (1+\gamma)(1+\epsilon)(2+2\alpha)D \\
&= (1+\delta)(2+2\alpha)D
\end{aligned}$$

As $l = |\cup_{i\in L} C_i| \geq n - kt$, we know that the cost of the points in $X - M$ with respect to medians C' does not exceed $(1+\delta)(2+2\alpha)$ as well. Thus it is sufficient to bound the cost of clustering of M. To this end notice, that there exists a clustering of M with a cost at most $2D$ (just replace each c_i by its closest neighbor in M). Thus the algorithm will find a clustering of M with cost $2\alpha D$ and the total cost does not exceed $(1+\delta)(2+4\alpha)D$. □

5 Maximum Spanning Tree

In this section we present two approximation algorithms for MaxST, both running in $O(n)$ time. The first one outputs a tree of cost at least $\frac{1}{4}$ times the optimal, while the second one improves the factor to $\frac{1}{2}$.

Let T denote the tree with maximum cost $C(T)$ and let Δ be the diameter of the graph. The first algorithm follows from the following Lemma.

Lemma 1 *A tree T of cost at least $\frac{n}{4}\Delta$ can be found in $O(n)$ time.*

Proof: Let $a, b \in X$ be points such that $d(a,b) = \Delta$. Consider arbitrary point q. As $\Delta = d(a,b) \leq d(a,q) + d(q,b)$, we can find a vertex u such that $d(q,u) \geq \frac{\Delta}{2}$. Consider now two stars: S_q centered at q and S_u centered at u. We can lower bound the sum of their costs by

$$\sum_p d(q,p) + d(u,p) \geq \sum_p d(q,u) \geq n\frac{\Delta}{2}$$

Thus one of S_q and S_u has cost at least $\frac{\Delta}{4}n$. □

The second algorithm uses similar idea, but the pair (q,u) is found by random sampling rather than using the above algorithm. Specifically, we prove that the longest among the $O(n)$ sampled edges has length close to $\frac{C(T)}{n-1}$. The bound then follows from the above argument.

In order to analyze the sampling algorithm, we need the following Lemma (which can be thought of as a reverse version of Markov inequality).

Lemma 2 *Let X be a random variable with values from the set $[0, B]$ such that $E[X] \geq \frac{B}{a}$ for some $a > 1$. Then for any $\alpha > 0$*

$$\Pr[X \geq (1-\alpha)E[X]] \geq \frac{\alpha}{a}.$$

Proof: Similar to the proof of Markov inequality. □

We can now proceed with the proof of correctness of the sampling procedure. Let $e_1 \dots e_{cn}$ denote cn edges sampled at random; let $X = \max_i d(e_i)$. As for any $\epsilon > 0$ and $c = O(\log 1/\epsilon)$ we have the probability of $1-\epsilon$ of hitting a (random) edge from T, we have $E[X] \geq (1-\epsilon)\frac{C(T)}{n-1}$. Therefore (by Lemma 2) we conclude that with a probability of $\Omega(\alpha)$ we have $X \geq (1-\alpha-\epsilon)\frac{C(T)}{n-1}$. Thus we have proven the following theorem.

Theorem 2 *For any $\delta > 0$ there is a $\frac{1-\delta}{2}$-approximation algorithm for MaxST running in $O(n\frac{\log 1/\delta}{\delta})$ time.*

6 Maximum TSP

Let T denote the tour with maximum cost $C(T)$ and let T' be tour chosen uniformly at random from the space of all $(n-1)!$ tours.

Lemma 3 $E[C(T')] \geq \frac{C(T)}{2}$.

Proof: We can write $E[C(T')]$ as $\sum_p \sum_{q \neq p} \frac{d(p,q)}{n-1}$. Fix p and let $T(p)$ denote the successor of p in the tour T. By triangle inequality we know that for any q we have $d(p,T(p)) \geq d(p,q) + d(T(p),q)$ which implies

$$nd(p,T(p)) \geq \sum_q d(p,q) + d(T(p),q).$$

Therefore

$$\begin{aligned}
C(T) &= \sum_p d(p,T(p)) \\
&\leq \sum_p \frac{1}{n}\sum_q d(p,q) + d(T(p),q) \\
&= \frac{2}{n}\sum_p \sum_q d(p,q) \\
&\leq 2E[C(T')]
\end{aligned}$$

$\square$

Thus by Lemmas 2 and 3 we have the following theorem.

Theorem 3 *For any $\delta > 0$ there is a $\frac{1-\delta}{2}$-approximation algorithm for MaxTSP running in $O(\frac{n}{\delta})$ time.*

7 Minimum Routing Cost Spanning Tree

In this section we describe the approximation algorithm for the MRCST problem. The algorithm proceeds by choosing $q \in X$ at random and reporting a star S_q rooted at that vertex. Below we show that the probability the cost $C(S_q)$ exceeds $C = 2\sum_{p,q} d(p,q)(1+\delta)$ is at most $1 - \frac{1}{1+\delta}$, thus by repeating this procedure $O(1/\delta)$ times and taking best star we obtain a constant probability of success.

The probability bound is obtained as follows. Consider the expectation $E[C(S_q)]$. We know that

$$\begin{aligned} E[C(S_q)] &\leq \frac{1}{n}\sum_q C(S_q) \\ &= \frac{1}{n}\sum_q \sum_{p,r} d(p,q) + d(q,r) \\ &= 2\sum_{p,r} d(p,r) \end{aligned}$$

The probability bound follows by Markov inequality.

8 Average Distance

In this section we give an approximation algorithm for the Average Distance problem. The algorithm finds the value of the sum of all the distances (call it A) with multiplicative error $(1+\delta)$ in time $O(n/\delta^{7/2})$. The actual algorithm is simple - we sample a set S (of cardinality $s = an$) of edges by including each edge with probability $\frac{s}{m}$, compute the sum of their length and multiply by $\frac{m}{s}$, where $m = \binom{n}{2}$. The running time of the algorithm is clearly $O(s)$ with high probability. Below we prove that for $s = O(n/\delta^{7/2})$ the result is a good approximation of A. Note that the assumption of (X,d) being a metric is crucial for this result: otherwise one could assign arbitrarily large weight (say w) to one edge, which would not be included in S with high probability and thus the estimation of A would be incorrect. The metric space assumption allows us to avoid this problem, as we know that if $d(p,q) > w$ then for each $a \in X$ either $d(p,x)$ or $d(q,x)$ is greater than $w/2$, thus with a good probability we hit one of these edges.

Let Δ be the diameter of the metric and assume that the minimum interpoint distance is 1. Let $c = 1+\epsilon$, for some $0 < \epsilon < \delta$. Split the interval $[1\ldots\Delta]$ of possible distances into intervals $I_i = [c^i, c^{i+1})$, for $i \geq 0$. Let n_i be the number of distances falling into interval I_i and let s_i be the number of distances from S falling into I_i. Define $\tilde{A} = \sum_i c^i n_i$, $A' = \sum_{e\in S} d(e)$ and $\tilde{A}' = \frac{m}{s}\sum c^i s_i$. Clearly, $A = \tilde{A}(1\pm\epsilon)$ and $A' = \tilde{A}'(1\pm\epsilon)$, thus it is sufficient to show that $\tilde{A}'$ well approximates $\tilde{A}$. To this end notice that $E[\tilde{A}'] = \tilde{A}$, thus it is sufficient to bound the variance of $\tilde{A}'$ and use Chebyshev inequality. The variance $D^2[\tilde{A}']$ can be bounded by

$$\frac{m^2}{s^2}\sum_i c^{2i} n_i \frac{s}{m} = \frac{m}{s}\sum c^{2i} n_i.$$

Thus by Chebyshev inequality we get

$$\begin{aligned} P &= \Pr[|\tilde{A}' - E[\tilde{A}'] \geq \epsilon \cdot E[\tilde{A}]] \\ &\leq \frac{1}{\frac{\epsilon^2 E^2[\tilde{A}']}{D^2[\tilde{A}']}} \\ &= \frac{D^2[\tilde{A}']}{\epsilon^2 E^2[\tilde{A}']} \end{aligned}$$

As $E^2[\tilde{A}'] \geq \sum c^{2i} n_i^2$, it is sufficient to bound from the above the ratio

$$F = \frac{\sum c^{2i} n_i}{\sum c^{2i} n_i^2}.$$

In order to do this we make the following observation (this is the only place where we use the fact that d is a metric). Let $a, b \in X$ be a pair of points such that $d(a,b) = \Delta$. Then for any $p \in X$ we know that $d(a,p) + d(p,b) \geq \Delta$, thus at least one of $d(a,p)$ and $d(p,b)$ is greater than $\Delta/2$. Let k be such that $c^k = \Delta$. By pigeonhole principle there exists $0 \leq j \leq \log_c 2$ such that $n_{k-j} \geq n/\log_c 2$. This enables us to bound F as follows. Let $P = \{i : n_i \geq t\} - \{k-j\}$ for $t = \alpha n$, where α is chosen later. We can write F as $\frac{N_1+N_2}{M_1+M_2}$, where $M_1 = \sum_{i \in P} c^{2i} n_i^2$, $M_2 = \sum_{i \notin P} c^{2i} n_i^2$, $N_1 = \sum_{i \in P} c^{2i} n_i$ and $N_2 = \sum_{i \notin P} c^{2i} n_i$.

Now we observe that $\frac{N_1}{M_1} \leq \frac{1}{t}$ (from the definition) and

$$N_2 \leq t \sum c^{2i} \leq t \frac{c^{2(k+1)}}{c^2 - 1} \leq \frac{\Delta^2 (1+\epsilon)^2}{\epsilon} t$$

while

$$M_2 \geq \left(\frac{\Delta}{2} \frac{n}{\log_c^2 2}\right)^2$$

thus

$$\frac{N_2}{M_2} \leq \frac{1}{n} \frac{4 \log_c^2 2 \alpha (1+\epsilon)^2}{\epsilon}.$$

Therefore

$$F \leq \max\left(\frac{N_1}{M_1}, \frac{N_2}{M_2}\right) = \frac{1}{n} \max\left(\frac{4 \log_c^2 2 (1+\epsilon)^2}{\epsilon} \alpha, \frac{1}{\alpha}\right)$$

and by setting $\alpha = \Theta(\epsilon^{3/2})$ we obtain that $F = O(\frac{1}{\epsilon^{3/2}} \frac{1}{n})$. Thus

$$P = O\left(\frac{1}{\epsilon^2} \frac{m}{an} \frac{1}{\epsilon^{3/2}} \frac{1}{n}\right) = O\left(\frac{1}{a \epsilon^{7/2}}\right)$$

and by setting $\epsilon = \Theta(\delta)$ and $a = O(\frac{1}{\delta^{7/2}})$ we prove the approximation bound. In this way we proved the following Theorem.

Theorem 4 *For any $\delta > 0$, there is a $(1+\delta)$-approximation algorithm for the Average Distance problem running in $O(\frac{n}{\delta^{7/2}})$ time.*

problem	approx. factor	not achievable in time
Closest Pair	any	$o(n^2)$
Furthest Pair	$> \frac{1}{2}$	$o(n^2)$
MaxST/MaxTSP	$> \frac{1}{2}$	$o(n^2)$
MinST/MinTSP	$O(1)$	$o(n^2)$

Table 2: Lower bounds for metric space problems

9 Lower bounds

In this section we investigate limitations of sublinear time algorithms in metric spaces. Our results are depicted in the Table 9; they hold for randomized algorithms. When combined with our bounds, one can observe interesting phenomenon: the minimization problems are difficult to approximate, while the maximization problems are approximable to within a small constant factor. Intuitively, this is due to the fact that small distances can be easily "hidden" in the metric space, while the triangle inequality prevents large distances from "hiding".

Proof: The proofs are as follows

Closest pair : Set all distances to 1 except for one edge chosen at random, which is set to $\epsilon > 0$

Furthest pair : Set all distances to 1 except for one edge chosen at random, which is set to 2

MaxST/MaxTSP : Set all distances to 1 except for edges on a random path of length $n-1$, which are set to 2. The optimal cost of both MaxST and MaxTSP is $(2-\epsilon)n$, but finding δn edges set to 2 for any fixed $\delta > 0$ requires $\Omega(n^2)$ time.

MinST/MinTSP : For any $B = O(1)$ choose a random path of length $n-1$, set all edges on that path to 1 and consider the metric obtained by taking the shortest paths metric induced by the unit edges and limiting the maximum distance to some B. One can observe that the optimum solution has cost $n+B$. On the other hand, finding δn edges with cost $< B$ for any fixed $\delta > 0$ requires $\Omega(n^2/B)$ time.

□

Acknowledgments: The author would like to thank Ashish Goel, Howard Karloff and Sudipto Guha for helpful comments.

References

[1] A. Borodin, R. Ostrovsky, Y. Rabani, "Subquadratic Approximation Algorithms For Clustering Problems in High Dimensional Spaces", STOC'99.

[2] M. Charikar, S. Guha, "Improved Combinatorial Algorithms for the Facility Location and k-Median Problems", FOCS'99.

[3] M. Charikar, S. Guha, E. Tardos, D. Shmoys, "A Constant Factor Approximation for the k-Median Problem", STOC'99.

[4] D.R. Cutting, D.R. Karger, J.O. Pedersen and J.W. Tukey, "Scatter/Gather: A Cluster-based Approach to Browsing Large Document Collections", SIGIR'92.

[5] E. Cohen, D. Lewis, "Approximating matrix multiplication for pattern recognition tasks", SODA'97, pp. 682-691.

[6] F.Ergun, S.Kannan, S.R.Kumar, R. Rubinfeld and M. Viswanathan, "Spot-Checkers", STOC'98, pp. 259-268.

[7] O. Goldreich, S. Goldwasser and D. Ron, "Property testing and its connection to Learning and Approximation", FOCS'96, pp. 339-348.

[8] W. B. Frakes, R. Baeza-Yates, *Information Retrieval - Data Structures and Algorithms*, Prentice Hall, New Jersey, 1992.

[9] D. Gusfield, "Efficient methods for multiple sequence alignments with guaranteed error bounds", *Bulletin of Mathematical Biology*, 55 (1993), pp. 141-154.

[10] T. Gonzales, "Clustering to Minimize the Maximum Inter-Cluster Distance", *Theoretical Computer Science* 38 (1985), pp. 293-306.

[11] D. Hochbaum, D. Shmoys, "A unified approach to approximate algorithms for bottleneck problems", *Journal of the ACM*, 33(1986), pp. 533-550.

[12] P. Indyk, "On Approximate Nearest Neighbors in Non-Euclidean Spaces", FOCS'98, pp. 148-154.

[13] P. Indyk, R. Motwani, "Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality", STOC'98, pp. 604-613.

[14] K. Jain and V. Vazirani, "Primal-Dual Approximation Algorithms for Metric Facility Location and k-Median Problems", FOCS'99.

[15] J. Kleinberg, "Two Algorithms for Nearest Neighbor Search in High Dimensions", STOC'97, pp. 599-608.

[16] E. Kushilevitz, R. Ostrovsky, and Y. Rabani, " Efficient search for approximate nearest neighbor in high dimensional spaces", STOC'98, pp. 614-623.

[17] M. R. Korupolu, C. G. Plaxton, R. Rajamaran, "Analysis of a Local Search Heuristic for Facility Location Problems", SODA'98, pp. 1-10.

[18] S.R. Kosaraju, J.K. Park, C. Stein, "Long Tours and Short Superstrings", FOCS'94.

[19] J. Lin, J.S. Vitter, "ϵ-Approximations with Minimum Packing Constraint Violation", STOC'92, pp. 771-782.

[20] J. Lin, J.S. Vitter, "Approximation Algorithms for Geometric Median Problems", IPL 44 (1992), pp. 245-249.

[21] R.L. Rivest, J. Vuillemin, "On recognizing graph properties from adjacency matrices", *Theoretical Computer Science* 3 (1976), pp. 371-384.

[22] B. Y. Wu, G. Lancia, V. Bafna, K Chao, R. Ravi, C. Y. Tang, "A Polynomial Time Approximation Scheme for Minimum Routing Cost Spanning Trees", SODA'98, pp. 21-32.

[23] R. Wong, "Worst-case Analysis of Network Design Problem Heuristics", *SIAM J. Alg. Discr. Meth.* 1 (1980), pp. 51-63.