Facility Dispersion and Remote Subgraphs

Barun Chandra[1]* and Magnús M. Halldórsson[2]*

[1] Department of Computer Science, Rice University, Houston, Texas
[2] Science Institute, University of Iceland, Reykjavík, Iceland.

Abstract

Dispersion problems involve arranging a set of points as far away from each other as possible. They have numerous applications in the location of facilities and in management decision science. We present several algorithms and hardness results for dispersion problems using different natural measures of remoteness, some of which have been studied previously in the literature and others which we introduce; in particular, we give the first algorithm with a non-trivial performance guarantee for the problem of locating a set of points such that the sum of their distances to their nearest neighbor in the set is maximized.

1 Introduction

As the proud and aggressive owner of the McWoofer burger chain, you are given the opportunity to build p new franchises to be located from any of n available locations. After ensuring that the available slots are all attractive in terms of cost, visibility, etc., what would your criteria be for locating the franchises *relative to each other*?

Locating two identical burger joints next to each other would not increase the number of customers, and thus halve the amount of business that either of them could do if apart. Non-competitiveness is a concern here, which can be alleviated by properly *dispersing* the facilities.

The franchise location example is one of many problems where we seek a subset of points that are, in some sense, as *remote* from each other as possible. Dispersion has found applications in diverse areas: locating undesirable or interfering facilities; aiding decision analysis with multiple objectives; marketing a set of products with different attributes; providing good starting solutions for "grand-tour" TSP heuristics. Dispersion is also of combinatorial interest, as a measure of remote subgraphs.

Which measure of remoteness should be applied? The proper measure is very much a question of the problem under study, and several of the applications we consider give rise to quite different notions of remoteness. In this paper, we present the first provably good approximation algorithms for dispersion problems under several measures of remoteness.

*Work of both authors performed in large part at JAIST-Hokuriku, Japan.

Applications. Location theory is a branch of management science/operations research that deals with the optimal location of facilities. Most of that work deals with desirable facilities, where nearness to users or each other is preferable. More recently, some papers have considered the opposite objective of placing the facilities far from each other.

Strategic facilities that are to be protected from simultaneous enemy attacks is one example suggested by Moon and Chaudhry [9]. This could involve oil tanks [9], missile silos, or ammunition dumps [4], which should be kept separated from each other to minimize the damage of a limited attack. Limiting the range and possible spread of fire or accidents at hazardous installations is also helped by proper spacing [8].

Non-competition is another motivation for dispersal, as in the case of the burger chain example. This may apply to other types of franchises such as gasoline stations, or to the location of radio transmitters with the objective of minimizing interference. Dispersal has also been seen to be desirable in order to obtain a more effective and/or fair coverage of a region. In fact, White [12] cites some example of government regulations to that effect, including firehouses and ambulance stations in New York City.

Yet another dispersal issue in facility location involves undesirable interaction between all facilities that grows inversely with distance [4]. This may apply to dormitories at a university, or chairs during an examination.

All of the above applications suggest a metric sensitive to the largely two-dimensional nature of our world. This is not the case, however, for the problems outside the location area.

White [12] considers dispersion problems motivated by *multiple objective analysis* in decision theory. Given a potential set of *actions* for a decision maker, we are to find a fixed-size subset of these that are as dispersed as possible, for further consideration by the decision makers. White lists several studies that have used dispersal as a filter of the possible choices, involving, for example, oil drilling, media selection, and forestry management

Dispersion also has applications in product development. The marketing of new but related products is helped by diversity [2]. From dimensions including price, quality, shape, packaging, etc., a set of products can be produced which are likely to gain greater market coverage if they are easily distinguished rather than all very similar.

Dispersion Formulations. A considerable body of work has appeared on facility dispersion problems in the management science and operations research literature [9, 2, 3, 4, 12, 8]. Most previous work has focused on either easily solvable tree networks, or empirical studies of heuristics. Only recently have some of these heuristics been analyzed analytically.

We suggest a simple formalism which lets us describe different dispersal problems in a uniform way and with a more "standardized" terminology. The input is an integer p and a network $G = (V, V \times V)$ with a distance function d on the edges satisfying the triangular inequality $d(u, v) \leq d(u, z) + d(z, v)$. The output

is a set P of p vertices. The objective is a function of the subgraph induced by P, and is given by the sum of a certain set of edges within that subgraph, this edge set being chosen to be the one of minimum weight among all edge subsets satisfying a graph property Π (Π depends on the particular dispersal problem under consideration). In general, for a property Π of graphs, the objective function for Remote-Π is the weight of the minimum-weight subgraph satisfying property Π within the induced subgraph on P. The goal of the algorithm is to pick these p vertices so as to maximize the objective function.

For instance, in the Remote-tree problem, the objective function is a sum of the edge weights of a minimum-weight spanning tree over the vertex set P. The goal is to pick a subset of p vertices so as to maximize the minimum-weight spanning tree on these vertices.

We list some problems under different graph properties; most of these have been studied previously, and some are introduced in this paper.

Remote-edge	$\min_{v,u \in P} d(u,v)$
Remote-clique	$\sum_{v,u \in P} d(u,v)$
Remote-star	$\min_{v \in P} \sum_{u \in P} d(u,v)$
Remote-pseudoforest	$\sum_{v \in P} \min_{u \in P} d(u,v)$
Remote-tree	$wt(MST(P))$
Remote-cycle	$\min_T wt(T)$, where T is a TSP tour on P
Remote-matching	$\min_M wt(M)$, where M is a perfect matching on P

A *pseudo-forest* is the undirected equivalent of a directed graph where each vertex is of out-degree one and each component contains as many edges as vertices.

Observe the adversarial nature of these problems. The "algorithm" produces a vertex set P, and implicitly a graph $G[P]$. The "adversary" produces a set of edges on $G[P]$ satisfying property Π. The value of the solution is the sum of these edges.

Related work. The names used in the literature are quite different and varied. Remote-edge is known as *p-Dispersion* [2, 8] and *Max-Min Facility Dispersion* [10]; Remote-clique as *Maxisum Dispersion* [8] and *Max-Avg Facility Dispersion* [10]; Remote-star as *MaxMinSum dispersion* [4]; Remote-pseudoforest as *p-Defense* [9] and *MaxSumMin dispersion* [4].

For Remote-edge, Tamir [11], White [12, 13] and Ravi, Rosenkrantz and Tayi [10] independently showed that a simple "furthest-point greedy" algorithm is 2-approximate. This greedy algorithm, henceforth called GREEDY, works by successively selecting the next vertex so as to maximize the distance to the set of already selected vertices, till p vertices have been selected. [10] also showed that obtaining an approximation strictly less than 2 was NP-hard.

For Remote-clique, Ravi et al. gave a (different) greedy algorithm that they showed came within a factor of 4, while Hassin, Rubinstein and Tamir [6] have recently given elegant proofs of two 2-approximate algorithms. This problem has also been studied for non-metric graphs under the name Heavy Subgraph Problem

by Kortsarz and Peleg [7], and they presented a sequence of algorithms that converge with a performance ratio of $O(n^{3.865})$.

No analytic bounds have been previously given for either Remote-star or Remote-pseudoforest problems. Moon and Chaudhry [9] suggested the star problem. Erkut and Neuman [4] gave branch-and-bound algorithms that solves all four of these problems.

Remote-tree and Remote-cycle were considered by Halldórsson, Iwano, Katoh, and Tokuyama [5], under the names *Remote-MST* and *Remote-TSP*, respectively. They showed that GREEDY approximates both problems within a factor of 4, while obtaining a ratio less than 2 is NP-hard. They proposed Remote-matching as an open problem.

All of the problems listed above can be seen to be NP-hard by a reduction from the maximum clique problem. The same reduction also establishes that Remote-edge cannot be approximated within a constant smaller than 2 [10]. Further, when the weights are not constrained to be metric, the problem is as hard to approximate as Max Clique, which implies that $n^{1/6}$-approximation is NP-hard [1]. Reductions from MaxClique also yield the same hardness for the pseudoforest problem [5]. On the other hand, no hardness results are known for Remote-clique and Remote-star.

Motivation. In various applications the utility of an individual facility may be directly related to its (locally measured) remoteness from the rest of the facilities. In this case, the measure of the global remoteness is the sum of the utilities of the individual points.

One example would be the average distance measure (or clique problem), in which the utility is the average distance from the other points. Note, however, that this measure is large for very clustered instances, as long as the clusters are far from each other. In many, if not most, cases, a more logical measure of utility would be the *minimum* distance to the remaining point set, i.e. the nearest neighbor distance. This gives rise to the Remote-pseudoforest problem.

Another meaning for a subgraph to be "remote" would be that its "nearness" measure would be high. One common nearness measure is that of a *center* : the smallest total distance from all the vertices to a single center vertex. This gives rise to the Remote-star problem (or alternatively Remote 1-Median).

Overview of paper. We review the much-studied GREEDY algorithm in section 2, and apply it to various remote problems. These involve partitions into independent segments, with the objective being the sum or maximum of the objectives on the independent segments. We also present numerous limitation results on the power of the algorithm.

In section 3 we present $O(\log p)$-approximate algorithms for Remote-matching and Remote-pseudoforest , and matching lower bounds for these algorithms.

In section 4 we use a recent result of Hassin, Rubinstein and Tamir [6] to give good approximation for Remote-star. We end with a summary and open problems.

1.1 Notation

For a vertex set $X \subseteq V$, let $\Pi(X)$ ($\pi(X)$) denote the maximum (minimum) weight set of edges in the induced subgraph $G[X]$ that form a graph satisfying property Π, respectively. In particular, we consider $\mathsf{star}(X)$ (min-weight spanning star), $\mathsf{pf}(X)$ (min-weight pseudoforest), $\mathsf{tree}(X)$ (min-weight spanning tree), and $\mathsf{MAT}(X)$ and $\mathsf{mat}(X)$ (max- and min-weight matching).

For a set of edges E', let $wt(E')$ denote the sum of the weights of the edges of E'. We shall also overload E' to stand for $wt(E')$ when used in expressions.

Let HEU be the vertex set selected by the algorithm in question, and let OPT be any other set of same cardinality, presumably standing for the solution that yields an optimal value. A set of p vertices is called a *p-set*.

The input is assumed to be a complete graph, with the weight of an edge (v, u), or the *distance* between v and u, denoted by $d(v, u)$. For a set of vertices X and a point v, the distance $d(v, X)$ is the shortest distance from v to some point in X, or $\min_{u \in X} d(v, u)$.

2 GREEDY algorithm

In this section we study the ubiquitous GREEDY algorithm, and obtain numerous positive and negative results. GREEDY is simple, efficient, arguably the most natural algorithm for these problems, and has been shown to be provably good for many of these problems. In addition, it is *online* (i.e. independent of p), allowing for the incremental construction of facilities that is essential in practice. As such, it warrants special attention, not only in the form of positive results but negative results as well.

2.1 The GREEDY algorithm and Anti-covers

A set X of points is said to be an *anti-cover* iff each point outside X is closer to X than the smallest distance between pairs of points in X. Namely,

$$d(v, X) \leq d(x, X - \{x\}), \quad \text{for any } v \in V,\ x \in X.$$

The direct way of producing an anti-cover is via the *GREEDY* algorithm. It first selects an arbitrary vertex and then iteratively selects a vertex of maximum distance from the previously selected points. We let $Y = \{y_1, y_2, \ldots, y_p\}$ denote this set of points found by GREEDY. Let Y_i be the prefix set $\{y_1, y_2, \ldots, y_i\}$, for $1 \leq i \leq p$. Let $r_i = d(y_{i+1}, Y_i)$ denote the distance of the $i+1$-th point to the previously selected points.

Observe that every prefix Y_i of the greedy solution is an anti-cover. Thus, for each i, $1 \leq i \leq p-1$:

$$d(v, Y_i) \leq r_i, \text{ for each } v \in V, \text{ and} \tag{1}$$

$$d(x, y) \geq r_i, \text{ for each } x, y \in Y_{i+1}. \tag{2}$$

Clearly, whenever the anti-cover property is needed, the greedy solution will suffice. However, if the additional properties of the greedy solutions are not needed, it is quite plausible that an alternative, possibly parallel, algorithm can be formulated.

GREEDY has been previously applied with success on the edge problem [11, 12, 13], and the tree, cycle and Steiner-tree problems [5]. It has also been shown to perform badly for a host of problems. We shall add a number of results in the remainder of this section.

2.2 Limitations of GREEDY

This subsection presents lower bounds on the performance of GREEDY on various remoteness problems. First, a general result.

Proposition 2.1 *The performance ratio of GREEDY on* any *remote problem is at least 2.*

Proof. Consider an instance with two types of points: x-points of distance 1 apart and y-points of distance 2 apart, with points of different type of distance 1 apart.

GREEDY may start with some x-point, from which all points are of distance 1, and then continue choosing x-points. That is, the induced subgraph selected is complete with unit-weight edges. An optimal solution will contain only y-points, inducing a complete graph with all edge weights 2. Whatever measure used, it will be twice as large in the latter subgraph as the one chosen by GREEDY. ∎

We can generalize a result of [5] that shows that there is no upper bound on the performance ratio of GREEDY on a host of problems. Their proof applied only to graph properties where the graph was not connected but each vertex was of non-zero degree.

We are given a network with a a given partition $(S, V-S)$. Let H be an instance of the given graph property Π. The following definition counts the necessary number of cut edges of any Π-graph H that has s vertices on one side of the cut.

Definition 2.2

$$Cross(\Pi, s) = \min_{\substack{P \subseteq V \\ |S \cap P| = s}} \min_{H \in \Pi} |\{(v,w) \in H \ : \ v \in S \wedge w \in V - S\}|$$

Proposition 2.3 *Let* $d_{min} = \min_s Cross(\Pi, s)$ *and* $d_{max} = \max_s Cross(\Pi, s)$. *The performance ratio of GREEDY on* Remote-Π-structure *is at least arbitrarily close to* d_{max}/d_{min} *when* $d_{min} > 0$, *and unbounded when* $d_{min} = 0$ *and* $d_{max} > 0$.

Proof. Let s_{min} (s_{max}) be the value of s that minimizes (maximizes) $Cross(\Pi, s)$. Without loss of generality, $s_{min}, s_{max} \leq p/2$.

Construct an instance, consisting of a block of vertices S with $p - s_{min}$ vertices and a block S' of p vertices. Vertices within S are distance ϵ apart,

vertices within S' $\epsilon/2$ apart, and vertices in different partitions 1 apart, where ϵ is arbitrarily small.

GREEDY first chooses all vertices in S followed by s_{min} vertices from S'. The cost of this pointset is the sum of $Cross(\Pi, s_{min}) = d_{min}$ crossing edges of weight 1 along with a number of ϵ weights. On the other hand, the optimal solution chooses s_{max} vertices in S and the rest in S', for a cost of at least d_{max}. ∎

This shows that the performance ratio of GREEDY on the following problems is unbounded: matching, pseudoforest, degree-bounded subgraph. Also, the performance ratio is at least $\Omega(p)$ on clique, and star. We also obtain the the following lower bound for high-degree properties.

Corollary 2.4 *Suppose* Π *is satisfied only for structures with minimum degree at least* k*, for some positive integer* k*. Then, the performance ratio of GREEDY for* Remote-Π *is at least* $k/2$*.*

Some examples of Π are minimum-degree-k, clique-number-k, and k-edge-connected.

2.3 Multiple trees, tours and Steiner trees

We apply GREEDY to remote problems involving several trees or cycles. The k spanning trees problem is a generalization of the Remote-tree problem, where the adversary partitions the p vertices into k sets so that the sum of the k spanning trees is minimized; the k Steiner trees and k TSP tours problems are similarly defined. We can generalize the analysis of [5].

Proposition 2.5 *Anti-covers attains a ratio of 4 for remote problems of* k spanning trees*, and a ratio of* $\min(5, 1 + 2p/(p-k))$ *for* k Steiner trees *and* k TSP tours*.*

Proof.

We consider the k spanning tree problem. The analysis for k Steiner trees and k TSP tours is similar and is omitted.

Consider any k-partitioning of the greedy points. Let GR_j be the heuristic value on partition j. We construct a solution for OPT as follows. Divide its points among the partitions according to nearness. Thus, each optimal point is within distance r_p to some greedy point in its partition. We may assume without loss of generality that each partition contains at least one optimal point; to any empty partition we may assign an arbitrary single optimal point which will not contribute to the value of the solution.

The optimal and greedy points combined form a Steiner tree. To obtain the k-MST value, we multiply by the Steiner ratio $2 - 2/q_j$, where q_j is the number of optimal points in partition j, with the assumption that $\forall j\, q_j \geq 1$ (which also implies $\forall j\, q_j \leq p - k$). Thus, $OPT \leq \sum_j (GR_j + q_j \cdot r_p) \cdot (2 - 2/q_j)$
$= \sum_j GR_j(2 - 2/q_j) + r_p(2q_j - 2) \leq GR(2 - 2/p) + 2r_p(p - k)$. Note that the

heuristic solution contains $p-k$ edges, and so $GR \geq (p-k)r_p$. Hence, a ratio of $4-2/p$ follows.

∎

It is easy to construct instances to show that these bounds are tight.

3 PREFIX Algorithm

We consider the Remote-pseudoforest and Remote-matching problems, and introduce a new algorithm that approximates these problems within a logarithmic factor. As we have seen, GREEDY does not guarantee any ratio for these problems.

We first consider the problem where we want to select p vertices so as to maximize the minimum weight pseudoforest (pf). A pseudoforest is a collection of directed edges so that the outdegree of each vertex is at least one, and hence pf is the sum of the nearest neighbor distances. More formally, $wt(\mathsf{pf}(W))$ is defined to be $\sum_{x \in W} d(x, W - \{x\})$.

A related concept is that of an *edge-cover*. A set of edges covers the vertices if each vertex is incident on some edge in the set. A pseudoforest is also an edge-cover, while it can be produced from an edge-cover on the same vertex set by counting each edge at most twice. Thus, the values of these problems differ by a factor of at most two.

3.1 Upper Bounds

We present an algorithm for selecting p vertices for Remote-pseudoforest; the same algorithm (i.e. choosing the same set of vertices) works well for Remote-matching as well.

We take a two step approach to the problem. In the first step we select some number ($\leq p$) of vertices that induce a large pseudoforest. This is done by considering the sequence of vertices selected by GREEDY, and choosing some prefix of this sequence according to a simple optimality criteria. In the second step, we choose the remaining vertices in such a way that the weight of pseudoforest already accumulated does not decrease too much. This is done by ensuring that the additional vertices selected don't destroy too many of the vertices chosen in the first step by being too near.

For simplicity, we assume that $p \leq n/3$, where n is the total number of vertices. It is easy to see that the algorithm can be easily modified when this is not the case, as long as p is less than some constant fraction of n.

The PREFIX Algorithm :

Step 1 : Run the GREEDY algorithm, obtaining a set $Y = \{y_1, \ldots, y_p\}$. Recall that $r_i = \min_{j=1}^{i}\{d(y_{i+1}, y_j)\}$, $1 \leq i \leq p-1$. Let $q \in \{1, 2, \ldots, p-1\}$ be the value which maximizes $q \cdot r_q$. Let Y_{q+1} be a prefix subsequence of Y of length $q+1$.

Step 2 : Let S_i be the set of vertices of distance at most $r_q/2$ from y_i, $i = 1, \ldots, q+1$. Points of distance exactly $r_q/2$ from more than one y_i are assigned arbitrarily to one sphere. The S_i are disjoint spheres centered at y_i.

Let $z = \lceil (q+1)/3 \rceil$. Let $\{S_{i_1}, S_{i_2}, \ldots, S_{i_z}\}$ be the z sparsest spheres and let *Good* be the set of their centers $\{y_{i_1}, y_{i_2}, \ldots, y_{i_z}\}$. Let *Rest* be any set of $p - z$ vertices from $V - \cup_j S_{i_j}$.

Let $PRE = Good \cup Rest$.

Our main result is a tight bound on PREFIX.

Theorem 3.1 *The performance ratio of PREFIX is* $\theta(\log p)$ *for* Remote-pseudoforest *and* Remote-matching.

We first prove the upper bound for pf.

Proof. First we verify that we can actually find the set *Rest* of additional vertices. The spheres contain at most $n/(q+1)$ vertices on average, so the sparsest z of them contain at most $\lceil (q+1)/3 \rceil n/(q+1) \leq 2n/3$ vertices, and at least $n/3$ can be chosen from outside the spheres as desired.

We propose that

$$\mathsf{pf}(PRE) \geq \mathsf{tree}(Y_p)/(6 \log p). \tag{3}$$

For any center $y_i \in Good$, and node z outside of S_i, $d(y_i, z) \geq r_q/2$. Hence,

$$\mathsf{pf}(PRE) \geq \sum_{x \in Good} d(x, PRE - \{x\}) \geq \sum_{x \in Good} r_q/2 \geq \frac{q+1}{3}\frac{r_q}{2} > \frac{q r_q}{6}. \tag{4}$$

Consider the spanning tree T' on Y_p which contains an edge from y_{i+1} to $Y_i = \{y_1, \ldots, y_i\}$ of weight r_i, for $i = 1, \ldots p-1$. Recall that by the choice of q, $r_i \leq \frac{q r_q}{i}$. Hence,

$$\mathsf{tree}(Y_p) \leq wt(T') = \sum_{i=1}^{p-1} r_i \leq \sum_{i=1}^{p-1} \frac{q r_q}{i} \leq q r_q \log p. \tag{5}$$

Equation 3 now follows from Equations 4 and 5.

We next show that

$$\mathsf{tree}(Y_p) \geq \mathsf{pf}(OPT)/8. \tag{6}$$

The following problem was considered in [5]: Find a set of p points F_p such that $wt(\mathsf{tree}(F_p))$ is maximized. It was shown [5, Theorem 3.1] that $\mathsf{tree}(Y_p) \geq \mathsf{tree}(F_p)/4$, and by definition $\mathsf{tree}(F_p) \geq \mathsf{tree}(OPT)$. Observe that $\mathsf{tree}(X) \geq (p-1)/p \cdot \mathsf{pf}(X) \geq \mathsf{pf}(X)/2$, for any pointset X. From these previous equations we get (6). The desired upper bound of $48 \log p$ on the approximation ratio $\mathsf{pf}(OPT)/\mathsf{pf}(HEU)$ follows from Equations 6 and 3. ∎

We now show the same upper bound for Remote-matching from selecting the same set PRE of vertices. We assume that p is even.

Observe that for any vertex set X, $\mathsf{tree}(X) \geq \mathsf{mat}(X)$. (It is well known that $\mathsf{tree}(X) \geq \mathsf{cycle}(X)/2$ and since a Hamilton cycle consists of two matchings, $\mathsf{cycle}(X)/2 \geq \mathsf{mat}(X)$.) Also, $\mathsf{mat}(X) \geq \mathsf{pf}(X)/2$, since doubling the edges of a matching yields a pseudoforest. Thus,

$$\mathsf{mat}(PRE) \geq \mathsf{pf}(PRE)/2 \geq \frac{\mathsf{tree}(OPT)}{48 \log p} \geq \frac{\mathsf{mat}(OPT)}{48 \log p}.$$

Same bounds hold for Remote Cycle-Cover, where a cycle-cover is a minimum weight subgraph that induces a collection of cycles that contain each vertex in P.

3.2 Lower Bounds

The performance analysis is tight within a constant factor.

Theorem 3.2 *The performance ratio of PREFIX for* Remote-pseudoforest *and* Remote-matching *is* $\Omega(\log n)$.

We give the construction for pseudoforest; the one for matching is similar.

Proof. We construct a sequence of graphs G_p on $O(p^{3/2})$ vertices for which the ratio attained by PREFIX is $\log p/20 = \Omega(\log n)$.

For simplicity, we assume that $p = 1 + 4 + \cdots + 4^t$, for integer t. The vertex set of G_p is partitioned into $t+1$ levels, and each level i is partitioned into 4^i blocks. Each block contains 2^t vertices, each labeled with a distinct binary string of t bits. The distance between two vertices in the same block at level i is $1/4^{i+j}$, where j is the index of the first character where labels of the vertices differ. The distance between two vertices in different blocks, either at the same level i or different levels i, i', $i < i'$, is $1/4^i$.

It is easy to verify that the triangle inequality is satisfied for the edge weights of this graph. The theorem follows from the following two lemmas. ∎

Lemma 3.3 $\mathsf{pf}(OPT) \geq t + 1$.

Proof. Choose one vertex from each block. There are 4^i vertices at level i, $i = 0, 1, \ldots, t$, and each vertex at level i is at a distance 4^{-i} to its nearest neighbor in this set. ∎

Lemma 3.4 $\mathsf{pf}(PRE) \leq 10$.

Proof. Consider first the contribution of the greedy prefix Y_q. Let a be such that $r_q = 1/4^a$. Then, Y_q contains at most one vertex at level higher than a, since after that vertex is selected, other such vertices are at distance less than r_q. Y_q contains vertices from each block of level at most $a-1$. In fact, for each block at level j, it contains at least one vertex for each value of the first

$a-1-j$ bits of the vertex label (as otherwise there would be a vertex of distance $1/4^{j+(a-1-j)} = 1/4^{a-1}$ from other selected vertices). Thus, the nearest neighbor distance of each vertex is at most $1/4^{a-1}$. On the other hand, Y_q contains at most one vertex for each value of the first $a-j$ bits. Thus, the total number q of selected vertices is at most

$$1 + \sum_{j=0}^{a} 4^j \cdot 2^{a-j} = 1 + \sum_{j=0}^{a} 2^{a+j} \leq 2 \cdot 4^a.$$

Thus, the total weight of the greedy prefix is at most $(1/4^{a-1}) \cdot 2 \cdot 4^a = 8$.

In order to bound from above the contribution of *Rest*, it suffices to supply a particular break into spheres and a choice of vertices from the dense spheres. For each vertex outside the sparse spheres, there is another vertex there of distance at most $1/4^t$. Suppose in our selection of *Rest* we ensure that for each chosen vertex, such a close neighbor be also present. Then, the contribution of rest is at most $(p-q)4^{-t} \leq 4/3$. ∎

We can also show (proof omitted in this version) that, for Remote-pseudoforest, no analysis of any algorithm which uses the idea of comparing the pf to the tree (as we do in the analysis of PREFIX) can hope to do any better; namely, we can construct graphs for which a large (logarithmic) gap exists between the weight of the tree of the whole graph and the pf of any subset of vertices.

4 MATCHING Algorithm

We consider the MATCHING algorithm of Hassin, Rubinstein and Tamir [6] and apply it to the Remote-Star problem. Recall that we seek a set of p points P that maximizes: $\min_{v \in P} \sum_{w \in P} d(v,w)$.

A *maximum-weight p-matching* is a maximum-weight set of $\lfloor p/2 \rfloor$ independent edges. It can be found efficiently via ordinary matching computation by appropriately padding the input graph [6].

The MATCHING algorithm of [6] used for the Remote-Clique is:

> Select the points of a maximum weight p-matching and add an arbitrary vertex if p is odd.

We consider the same algorithm for Remote-star i.e. the vertices for the Remote-star problem are the vertices selected by MATCHING.

Theorem 4.1 *The performance ratio of MATCHING for* Remote-star *is at most* 2.

Proof. Let HEU be the vertex set found by MATCHING. From the triangular inequality, we can see that $\mathsf{star}(HEU) \geq \mathsf{MAT}(HEU)$. The optimality of the p-matching tells us that $\mathsf{MAT}(HEU) \geq \mathsf{MAT}(OPT)$. And, the following lemma establishes that $\mathsf{MAT}(OPT) \geq \frac{\lfloor p/2 \rfloor}{p-1}\mathsf{star}(OPT)$. Thus, $\mathsf{star}(OPT)/\mathsf{star}(HEU)$ is always at most 2, and at most $2 - 2/p$ when p is even. ∎

It is easy to construct an example of an $(1,2)$-weighted graph where this ratio is attained.

Lemma 4.2 *For any pointset X on p vertices,*

$$\frac{\mathsf{star}(X)}{p-1} \leq \frac{MAT(X)}{\lfloor p/2 \rfloor}.$$

Proof. There are p spanning stars of X and each edge is contained in exactly two of them. The average cost of a star is $(2/p) \cdot wt(X)$ and some star is of weight at most that. Similarly, each edge appears in exactly $\lfloor p/2 \rfloor / \binom{p}{2}$ fraction of the different maximal matchings in X, and thus some matching is of weight at least the average cost of $\lfloor p/2 \rfloor / \binom{p}{2} wt(X)$. ∎

This lemma can be generalized to relate any minimum and any maximum structures for which each instance has the same number of edges (here, $p-1$ and $\lfloor p/2 \rfloor$, respectively).

5 Discussion

We have presented a framework for studying dispersion problems, given approximation algorithm for several natural measures of remoteness, and shown several hardness results as well as limitations on the ubiquitous greedy approach.

We have considered a number of extensions of the formalism presented here. These include bottleneck problems – where the objective function is a maximum, rather than the sum, of the edge weights of a structure; Steiner problems – where the objective function may include vertices outside the selected set P; and min-max problems – where we ask for a set of points that minimizes some maximization structure.

Many threads are left open for further study. Are there constant-factor approximation algorithms for the remote pseudoforest and matching problems, or can we prove super-constant hardness results? Can we give an exhaustive classification of the approximability of a large class of remote problems, and/or can we succinctly describe those problems for which GREEDY will do well? And finally, a further study on the applied aspects from the management science viewpoint would be desirable.

We conclude with a conjecture about the complexity of (exact solutions of) remote subgraph problems: tha the remoteness operator pushes the problem exactly one level higher in the polynomial-time hierarchy.

Conjecture 5.1 *Let Π be a coNP-hard property. Then the problem of deciding whether there exists a subset S of the input of size p that forms a valid instance where Π holds is Σ_2^p-hard.*

References

[1] M. Bellare and M. Sudan. Improved non-approximability results. *STOC 1994.*

[2] E. Erkut. The discrete p-dispersion problem. *Europ. J. Oper. Res*, 46:48–60, 1990.

[3] E. Erkut and S. Neuman. Analytical models for locating undesirable facilities. *Europ. J. Oper. Res*, 40:275–291, 1989.

[4] E. Erkut and S. Neuman. Comparison of four models for dispersing facilities. *INFOR*, 29:68–85, 1990.

[5] M. M. Halldórsson, K. Iwano, N. Katoh, and T. Tokuyama. Finding subsets maximizing minimum structures. *SODA 1995.*

[6] R. Hassin, S. Rubinstein, and A. Tamir. Notes on dispersion problems. Dec. 1994.

[7] G. Kortsarz and D. Peleg. On choosing a dense subgraph. *FOCS 1993.*

[8] M. J. Kuby. Programming models for facility dispersion: The p-dispersion and maxisum dispersion problems. *Geog. Anal.*, 19(4):315–329, Oct. 1987.

[9] I. D. Moon and S. S. Chaudhry. An analysis of network location problems with distance constraints. *Mgmt. Science*, 30(3):290–307, Mar. 1984.

[10] S. S. Ravi, D. J. Rosenkrantz, and G. K. Tayi. Heuristic and special case algorithms for dispersion problems. *Op. Res*, 42(2):299–310, 1994.

[11] A. Tamir. Obnoxious facility location on graphs. *SIAM J. Disc. Math.*, 4(4):550–567, Nov. 1991.

[12] D. J. White. The maximal dispersion problem and the "first point outside the neighborhood" heuristic. *Computers Ops. Res.*, 18(1):43–50, 1991.

[13] D. J. White. The maximal dispersion problem. *J. Applic. Math. in Bus. and Ind.*, 1992.