Approximation Algorithms for Dispersion Problems

Barun Chandra
Department of Computer Science
University of New Haven
New Haven, Connecticut
barun@charger.newhaven.edu

Magnús M. Halldórsson
Science Institute
University of Iceland
Reykjavik, Iceland
mmh@hi.is

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 suggest a simple formalism which lets us describe different dispersal problems in a uniform way. 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 nontrivial 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.

Index Terms: Facility Location, Dispersion Algorithms, Approximation Algorithms.

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 at 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 would thus halve the amount of business that either of them could do if apart. Noncompetitiveness 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.

In this paper, we unify these and the other dispersion problems in the literature by a novel formalization, where each dispersion problem P corresponds to a certain class of graphs Π. This by itself suggests various interesting new dispersion problems. We then present the first provably good approximation algorithms for dispersion problems under several of these measures of remoteness.

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 [11]. This could involve oil tanks [11], 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 [10].

Noncompetition is another motivation for dispersal, as in 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 found desirable to obtain an effective and/or fair coverage of a region. White [15] 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.

The above applications suggest a metric sensitive to the largely two-dimensional nature of our world. However, this need not be the case for various problems outside the area of facility location.

White [15] 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 to filter the possible choices, e.g. 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 parameters including price, quality, shape, packaging etc., a set of products can be produced, which are likely to gain greater market coverage if easily distinguishable.

Dispersion Formulations. A considerable body of work has appeared on facility dispersion problems in the management science and operations research literature [11, 2, 3, 4, 15, 10]. 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 [13, 15, 16, 12, 6, 1]

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 the problem 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 in Table 1 some problems under different graph properties; most of these have been studied previously, and some are introduced in this paper. For a set of edges E', $wt(E')$ denotes the sum of the weights of the edges of E'. For a point set P, $P = P_1|P_2|\cdots|P_k$ denotes a partition of P into sets P_1 through P_k.

Remote subgraph problems	
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 $T = tsp(T)$ is a TSP tour on P
Remote k-trees	$\min_{P=P_1\|\cdots\|P_k} \sum_{i=1}^{k} wt(mst(P_i))$
Remote k-cycles	$\min_{P=P_1\|\cdots\|P_k} \sum_{i=1}^{k} wt(tsp(P_i))$
Remote-matching	$\min_M wt(M)$, where M is a perfect matching on P
Remote-bipartition	$\min_B wt(B)$, where B is a bipartition of P [1]
Bottleneck problems	
Destruction Radius	$\min_{S \subseteq P, \|S\|=k} \max_{x \in P} d(x,S)$,
Destruction Steiner Radius	$\min_{S \subseteq V, \|S\|=k} \max_{x \in P} d(x,S)$,
Destruction Diameter	$\min_{P=P_1\|\cdots\|P_k} \max_{x,y \in P_i,\, i=1,2,\ldots,k} d(x,y)$.

Table 1: Dispersion problems considered

A *pseudo-forest* is the undirected equivalent of a directed graph where each vertex has 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 the induced subgraph $G[P]$. The "adversary" produces a set of edges on $G[P]$ satisfying property Π. The problems we look at are max-min problems, i.e. the algorithm tries to pick P such that the smallest set of edges satisfying Π on P is as large as possible. The value of the solution is the sum of these edges.

Which measure? 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 have considered give rise to quite different notions of remoteness.

In various applications the utility of an individual facility is 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 is 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 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 interpretation 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.

In the end, the appropriate measure is highly context-sensitive. The intended objective function is likely to involve more factors than are specified in the combinatorially pure problem specifications, even more so than in many other problem domains. To a large extent, computational facility dispersion is meant as an aid to decision-making, instead of as a solution provider. It is therefore valuable to try to understand better the impact of modifying the measure on the near-optimal computability of the problem. By introducing and examining a wider range of natural objective measures, unifying them into a consistent whole, and analyzing new and old practical algorithms on these measures, we hope to obtain a deeper understanding of the computational issues involved in dispersion.

Terrorism defense/Bottleneck problems. One of the motivations for dispersion problems is defense against accidents or attacks. The adversary attacks with k 'explosives' of a given destruction radius. Our objective is to select p sites that are dispersed so as to maximize the difficulty (i.e. force the adversary to use large explosives) of these p sites being destroyed by the adversary's explosives.

We can formally define this problem as follows. We are given a graph G, and positive integers p, k, $k \leq p-1$. We are to find a set $P \subseteq V$ of p vertices that maximizes $\min_{S \subseteq P, |S|=k} \max_{x \in P} d(x, S)$

Several similar types of optimization problems arise, depending on which parameter is to be maximized. These radius problems do not fit in the framework we defined earlier of the weight of a Π-subgraph. However, they can be viewed as the *bottleneck* versions of such properties, i.e. the value is the maximum weight edge of a subgraph satisfying the property. For example, the problem discussed above is the bottleneck version of Remote-star.

Related work. The names used in the literature are quite different and varied. Remote-edge is known as *p-Dispersion* [2, 10] and *Max-Min Facility Dispersion* [12]; Remote-clique as *Maxisum Dispersion* [10] and *Max-Avg Facility Dispersion* [12]; Remote-star as *MaxMinSum dispersion* [4]; Remote-pseudoforest as *p-Defense* [11] and *MaxSumMin dispersion* [4].

Problem	*u.b.*	*l.b.*
Remote-edge	2 [13, 15, 12]	2 [12]
Remote-clique	2 [7]	- [12]
Remote-tree	4 [6]	2 [6]
Remote-cycle	3 [6]	2 [6]
Remote k-trees	4	2
Remote k-cycles	5	2
Remote-pseudoforest	$O(\log n)$	2 [6]
Remote-matching	$O(\log n)$	2 [6]
Remote-star	2	-
Remote-bipartition	3	-
Destruction Radius	4	-
Destruction Steiner Radius	2	2
Destruction Diameter	2	2

Table 2: New and old upper and lower bounds on the approximability of dispersion problems considered in this paper.

We list in Table 2 the known upper and lower bounds on approximating the various dispersion problems. Where no citation occurs, the result is in the current paper. A lower bound of, e.g. 2, means that is it NP-hard to obtain an approximation ratio better than 2. A dash denotes that the problem is NP-hard, but no nontrivial approximation lower bound is known.

For Remote-edge, Tamir [13], White [15, 16] and Ravi, Rosenkrantz and Tayi [12] (see also [14]) 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. It was shown in [12] that obtaining an approximation strictly less than 2 was NP-hard. Baur and Fekete [1] have recently given a 3/2-approximation algorithm for a geometric case where weights correspond to distances between points within a rectilinear polygon, and showed this problem to be hard to approximate within a factor of less than 14/13.

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 [7] gave elegant proofs of two 2-approximate algorithms. This problem has also been studied for nonmetric graphs under the name Dense Subgraph Problem by Kortsarz and Peleg [9], with the current best ratio known being $O(n^{\delta})$, for some constant $\delta < 1/3$ [5].

No analytic bounds have been previously given for either Remote-star or Remote-pseudoforest problems. Moon and Chaudhry [11] suggested the star problem. Erkut and Neuman [4] gave a branch-and-bound algorithm that solves all four of these problems.

Remote-tree and Remote-cycle were considered by Halldórsson, Iwano, Katoh, and Tokuyama [6], under the names *Remote-MST* and *Remote-TSP*, respectively. They showed that GREEDY approximates these problems within a factor of 4 and 3, repectively. They also showed that obtaining a ratio less than 2 for these problems is NP-hard, and that holds also for Remote-pseudoforest and Remote-matching. 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 [12]. Further, when the weights are not constrained to be metric, the problem is as hard to approximate as Max Clique, which implies that $n^{1-\epsilon}$-approximation is hard, for any $\epsilon > 0$ [8]. Reductions from MaxClique also yield the same hardness for the pseudoforest problem [6]. On the other hand, no hardness results are known for Remote-clique and Remote-star.

Overview of paper. We introduce the notation in Section 2 and describe the concept of an anticover as computed by a GREEDY algorithm.

In Section 3, we show that GREEDY attains good approximation on a host of remote problem that involve partitions into independent clusters, with the objective being the sum or maximum of the objectives on the independent clusters. In section 4 we use the MATCHING algorithm of [7] to approximate Remote-star, and Remote-bipartition.

We introduce an algorithm, PREFIX, in Section 5. It combines the practicality of the GREEDY algorithm with the means to avoid falling into the traps that sharply limit the performance of the GREEDY algorithm. We use it to obtain a $\theta(\log p)$ performance ratio for the Remote-matching and Remote-pseudoforest problems, for the first non-trivial approximations for these problems.

In Section 6 we show that GREEDY yields good approximations for the bottleneck/radius problems for terrorism defense. We match these upper bounds with similar approximation hardness results. Then, in Section 7, we prove NP-hardness of all remoteness problems, for a nontrivial graph property Π. We also present negative results on the power of the GREEDY algorithm for a general class of problems. We end with a summary and open problems.

2 Preliminaries

2.1 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 forms 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 also overload E' to stand for $wt(E')$ when used in expressions.

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 X of vertices and a point v, the distance $d(v, X)$ is the shortest distance from v to some point in X, or $\min_{u \in X} d(v, u)$.

A *p-set* refers to a set of p vertices. Let OPT denote the p-set that yields the optimal value. Let $P = P_1|P_2|\cdots|P_k$ denote that the set P is partitioned into sets P_i, i.e. $\cup_{i=1}^{k} P_i = P$ and $P_i \cap P_j = \emptyset$, for any $i \neq j$. Throughout the paper we assume that the triangle inequality holds.

2.2 Anticovers and the Greedy algorithm

A set X of points is said to be an *anticover* iff each point outside X is at least as close to X as the smallest distance between any pair of points in X, i.e.

$$\max_{v \in V - X} d(v, X) \leq \min_{x \in X} d(x, X \setminus \{x\}).$$

The direct way of producing an anticover 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 also an anticover. Thus, for each i, $1 \leq i \leq p - 1$,

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

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

Greedy is simple, efficient, arguably the most natural algorithm for many facilities dispersal problems, and has been shown to be provably good for many of the 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.

Greedy has been previously applied with success on the edge problem [13, 15, 16], and the tree, cycle, and Steiner-tree problems [6]. We show (Sections 3 and 6) that Greedy performs well on problems involving multiple spanning trees or tours, and on the terrorism defense problems. On the other hand, we show (Section 7.1) that Greedy performs poorly on a large class of problems which include matching, pseudoforest, clique, and star. [2]

[2]@@@MH4: Reordered, to match the order in the paper

3 Tree and Cycle Problems

In this section, we apply GREEDY to remote problems involving several spanning trees or tours. 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. More formally, the objective value on a given point set P is given by

$$k\text{-trees}(P) = \min_{P=P_1|\cdots|P_k} \sum_{i=1}^{k} wt(mst(P_i)).$$

Thus, the Remote k-Trees problem is to find a set P of p vertices such that k-trees(P) is maximized. Similarly, in the k-Steiner trees problem the adversary partitions the p vertices into k sets so that the sum of the k Steiner trees is minimized, and in the k-cycles problem the adversary partitions the p vertices into k sets so that the sum of the k TSP tours is minimized. We can generalize the analysis of [6] for the case $k = 1$.

Proposition 3.1 *Anticovers yield a ratio of* $4 - 2/(p - k + 1)$ *for remote problems of* k-trees*, and a ratio of* $\min(5, 1 + 2p/(p - k))$ *for* k-Steiner trees *and* k-cycles.

Proof. Focus first on the k spanning tree problem. Let GR be the greedy point set and OPT be the point set of the optimal solution. Just as the adversary is allowed to partition the greedy point set, we can partition the optimal solution knowing that the cost of the corresponding spanning forest upper bounds the optimal cost.

Let $GR_1, GR_2, \ldots, GR_k$ be a partition of GR which minimizes the sum of the spanning trees. We form a partition of OPT into nonempty sets $Q_1, \ldots, Q_k$ such that for any Q_i with two or more vertices, $v \in Q_i$ implies that the nearest neighbor of v in GR is in GR_i. That no Q_i is empty is ensured as follows: if no vertex in OPT happens to be closer to GR_i than to any other GR_j, then we arbitrarily take a vertex from some $Q_j, |Q_j| \geq 2$ and put it in Q_i. Let q_i be the cardinality of Q_i. The partitioning ensures that for each i, $q_i \geq 1$, and hence that $q_i \leq p-k+1$ (since each of the other $k-1$ classes contain a vertex). Also, each vertex in a Q_i with $q_i \geq 2$ is of distance at most r_p from P_i, by the anticover property.

For each class Q_i with $q_i \geq 2$, consider the minimum spanning tree of the point set $Q_i \cup GR_i$. This forms a Steiner tree of Q_i. The cost of this tree is at most

$$mst(Q_i \cup GR_i) \leq mst(GR_i) + q_i \cdot r_p.$$

We can bound the cost of the spanning tree of Q_i by applying the Steiner ratio, obtaining

$$mst(Q_i) \leq mst(Q_i \cup GR_i) \cdot (2 - 2/q_i) \leq mst(GR_i) \cdot (2 - q_i) + 2r_p(q_i - 1).$$

Recall that $GR = \sum_{i=1}^{k} mst(GR_i)$. Summing up over all values of i gives

$$OPT \leq \sum_{i=1}^{k} mst(Q_i) \leq GR(2 - 2/(p - k + 1)) + 2r_p(p - k).$$

Since GR contains $p - k$ edges and the distance between any pair of points is at least r_p,

$$GR \geq (p - k)r_p.$$

Hence

$$\rho = \frac{OPT}{GR} \leq 4 - 2/(p - k + 1).$$

We now turn our attention to Steiner trees. We know that since pairwise distances in GR are at least r_p, the cost of k Steiner trees in GR is bounded by

$$GR \geq p' r_p/2,$$

where p' is the number of greedy points in partitions with at least 2 greedy points, $p' \geq p-k+1$. The Steiner forest of $GR_i \cup Q_i$, $i = 1, \ldots, k$, yields that

$$OPT \leq GR + pr_p, \tag{3}$$

for a ratio of $1 + 2p/(p-k+1) = 3 + 2(k-1)/(p-k)$. Recall that $q_i \geq 1$, for each i. Observe that for large values of k, the number of i with $q_i = 1$ is at least $2k - p$. Those points do not contribute to the cost of the solution. Thus,

$$OPT \leq GR + \min(2p - 2k, p) r_p,$$

for a ratio of at most 5.

For k-cycles we obtain the same ratio as for k-Steiner trees. Namely, by similar arguments we see that $GR \geq (p-k+1)r_p$, and by taking an Euler tour of each Steiner tree, the optimal k-tours cost is at most twice the cost of the k Steiner trees. $\square$

The upper bound of 4 for k-trees is tight via the matching lower bound for $k = 1$ of [6]. The constructions of [6] for 1-cycle and 1-Steiner trees have the property if OPT has $p-k+1$ points and GR has p points, for $t \geq p/2$, we get a matching bound for (3) of

$$OPT \geq GR + (p-k+1)r_p - o(1).$$

If we thus force GREEDY to assign singletons to $k-1$ of the partitions, while OPT picks points that are mutually $2r_p$ apart in each, we obtain a matching lower bound for k-cycles. For 1-Steiner trees the best lower bound of [6] was 2.46, thus our lower bound for $k = p/2$ is similarly $4.46 - o(1)$.

4 Star and Bipartition Problems

Hassin, Rubinstein and Tamir [7] gave the following algorithm MATCHING:

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

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 [7]. They used it to obtain a 2-approximation of Remote-Clique. In this section, we apply this algorithm to the Remote-Star and the Remote-bipartition problems.

Recall that in the Remote-Star problem we seek a set of p points P that maximizes $\min_{v \in P} \sum_{w \in P} d(v,w)$. Let HEU be the vertex set found by MATCHING. We first prove a useful lemma.

Lemma 4.1 *Let X be a set of p vertices. Let Π_1 and Π_2 be properties that always have the same number of edges on X, e_1 and e_2, respectively. Then,*

$$\frac{\pi_1(X)}{e_1} \leq \frac{wt(X)}{\binom{p}{2}} \leq \frac{\Pi_2(X)}{e_2}.$$

Proof. Consider any property Π that always has e edges on p points. E.g. star and tree have $p-1$ edges, tour and pseudoforest have p edges, and matching has $\lfloor p/2 \rfloor$ edges. Let $wt(X)$ denote the sum of the weights of edges with both endpoints in X. Since any permutation of the vertices is possible, each edge appears in equally many Π-structures. In fact, each edge appears in a $e/\binom{p}{2}$ fraction of all Π-structures on X. Thus, the average cost of a Π-structure on X is $wt(X)e/\binom{p}{2}$. A minimum Π_1 structure is therefore of cost at most $wt(X) \cdot e_1/\binom{p}{2}$, while a maximum Π_2-structure is of cost at least $wt(X)e_2/\binom{p}{2}$. □

Theorem 4.2 *The performance ratio of* MATCHING *for* Remote-star *is* 2.

Proof. Let HEU be the vertex set found by MATCHING, and recall that $\mathsf{MAT}(X)$ represent the maximum-weight matching on point set X. From the triangular inequality, observe that

$$\mathsf{star}(HEU) \geq \mathsf{MAT}(HEU).$$

By the definition of the algorithm, $\mathsf{MAT}(HEU) \geq \mathsf{MAT}(OPT)$, and by Lemma 4.1,

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

Thus, the performance ratio $\mathsf{star}(OPT)/\mathsf{star}(HEU)$ is always at most 2, and is less when p is even.

We construct a graph for which this ratio approaches 2. The vertices of the graph are $v_1, v_2, \ldots, v_n$. $p = n-1$ is even, all distances between the vertices are 2 except the distance from v_1 to each of $v_2, v_3, \ldots, v_{n-1}$ is 1, $d(v_1, v_n) = 2$. The optimum choice is to pick the p vertices $v_2, v_3, \ldots, v_n$, so $\mathsf{star}(OPT) = 2(p-1)$. MATCHING can pick the vertices $v_1, v_3, v_4, \ldots, v_n$, so $\mathsf{star}(HEU) = p - 2 + 2 = p$. Hence the ratio $\mathsf{star}(OPT)/\mathsf{star}(HEU) = 2 - \frac{2}{p}$. [3] □

We can also apply the MATCHING algorithm to the problem where Π is a bipartition of $G[X]$, i.e. the minimum-weight cut into two sets of size $p/2$. Let $\mathsf{bp}(X)$ denote a minimum-weight bipartition of $G[X]$.

Theorem 4.3 *The performance ratio of* MATCHING *for* Remote-bipartition *is at most* 3.

Proof. A bipartition is a union of $p/2$ matchings. Thus, in particular for OPT,

$$\mathsf{bp}(OPT) \leq \frac{p}{2} \mathsf{MAT}(OPT).$$

By definition, $\mathsf{MAT}(OPT) \leq \mathsf{MAT}(HEU)$. It remains to be shown that $\mathsf{bp}(HEU) \geq p/6 \cdot \mathsf{MAT}(HEU)$.

Let (L, R) be a bipartition of HEU of minimum cost, and let M be the edges of a maximum weight (perfect) matching on HEU. For simplicity, we assume that p is even, so that $|L| = |R| = |M| = p/2$. Let M_{LL} be the edges in M with both endpoints in L, M_{RR} those with both endpoints in R, and M_{LR} those with endpoints in both L and R. Let P_1 be the set of vertices induced by $M_{LL} \cup M_{RR}$ and P_2 be the set of vertices induced by M_{LR}. Let B be the set of edges crossing (L, R), and partition them into B_{11}, of edges with both endpoints in P_1, B_{22} with both endpoints in P_2, and B_{12} with endpoints in both P_1 and P_2.

[3]@@@MH4: Placed construction inside the proof.

By the triangle inequality,

$$\sum_{uv \in M_{LL}} \sum_{x \in R} [w(u,x) + w(x,v)] \geq \sum_{uv \in M_{LL}} \sum_{x \in R} w(u,v) = |R|\, w(M_{LL}). \tag{4}$$

The LHS counts the edges of B_{11}, as well as those edges of B_{12} with one endpoint in M_{LL}. A similar bound follows for $w(M_{RR})$. Combined,

$$w(B_{12}) + 2\, w(B_{11}) \geq |R|(w(M_{LL}) + w(M_{RR})). \tag{5}$$

Also, by the triangle inequality,

$$\sum_{uv \in M_{LR}} \sum_{x \in L} \sum_{y \in R} [w(u,x) + w(x,y) + w(y,v)] \geq \sum_{uv \in M_{LR}} \sum_{x \in L} \sum_{y \in R} w(u,v) = |L|\,|R| w(M_{LR}). \tag{6}$$

The middle edge of the LHS above counts all crossing edges $|M_{LR}|$ times. The first and the last edge of the LHS together counts the endpoints of edges in P_1 $|R|$ times, and thus count edges in B_{12} $|R|$ times, and edges in B_{22} $2|R|$ times. Thus,

$$|M_{LR}| \cdot \mathsf{bp}(HEU) + |R| w(B_{12}) + 2|R| w(B_{22}) \geq |R|^2 w(M_{LR}). \tag{7}$$

Adding (7) and $|R|$ times (5), we obtain

$$(|M_{LR}| + 2|R|)\mathsf{bp}(HEU) \geq |R|^2 w(M).$$

Thus, we have

$$\mathsf{bp}(HEU) \geq \frac{|R|}{3} \mathsf{MAT}(HEU) = \frac{p}{6} \mathsf{MAT}(HEU),$$

as desired. □

It is an open question whether the bound of 3 from Theorem 4.3 is tight.

5 Pseudoforest and Matching Problems

In this section, we introduce an algorithm PREFIX that approximates the Remote-pseudoforest and Remote-matching problems within a logarithmic factor. As we shall see in Section 7.1, the GREEDY algorithm alone cannot guarantee any ratio for these problems that is independent of the weights.

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 one, and hence pf is the sum of the nearest neighbor distances. More formally, $wt(\mathsf{pf}(W))$ is defined to be $\sum_{x \in W} d(x, W - \{x\})$. Each component of a pseudoforest is a graph with equally many vertices as edges, sometimes called a *cactus*.

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.

5.1 Upper Bounds

We present an algorithm for selecting p vertices for Remote-pseudoforest; the same algorithm (i.e. 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 criterion. In the second step, we choose the remaining vertices so as to avoid overly reducing the weight of the pseudoforest. This is done by ensuring that the additional vertices selected be close to only few of the vertices chosen in the first step.

For simplicity, we assume that $p \leq n/2$, where n is the total number of vertices. It is easy to see that the algorithm can be modified when this is not the case. The ratio attained stays the same within a constant factor as long as p is less than some constant fraction of n. The problem changes character if $n - p$ is small, which we do not attempt to address here.

The Prefix Algorithm :

Step 1 : Run the Greedy algorithm, obtaining a set $Y = \{y_1, \ldots, y_p\}$. Let $q \in \{1, 2, \ldots, p-1\}$ be the value which maximizes $q \cdot r_q$. Let Y_{q+1} be the 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, $i = 1, \ldots, q+1$. The S_i are disjoint *spheres* centered at y_i. Points of distance exactly $r_q/2$ from more than one y_i are assigned arbitrarily to one sphere.

Let $z = \lfloor (q+1)/2 \rfloor$. 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 $V - \cup_{j=1}^{z} S_{i_j}$.

Output $PRE = Good \cup Rest$.

Our main result is a tight bound on Prefix. Let H_t be the harmonic number $\sum_{i=1}^{t} 1/i \leq 1 + \ln n$.

Theorem 5.1 *The performance ratio of* Prefix *is* $O(\log p)$ *for* Remote-pseudoforest.

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 $\lfloor (q+1)/2 \rfloor n/(q+1) \leq n/2$ vertices. Hence, at least $n/2 \geq p$ vertices can be chosen from outside the spheres as desired.

We propose that

$$\mathsf{pf}(PRE) \geq \mathsf{tree}(Y_p)/(4H_p). \tag{8}$$

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

$$\mathsf{pf}(PRE) \geq \sum_{x \in Good} d(x, PRE - \{x\}) \geq \sum_{x \in Good} r_q/2 \geq \lfloor \frac{q+1}{2} \rfloor \frac{r_q}{2} \geq \frac{q r_q}{4}. \tag{9}$$

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,

$$\mathsf{tree}(Y_p) \leq wt(T') = \sum_{i=1}^{p-1} r_i \leq \sum_{i=1}^{p-1} \frac{q r_q}{i} = q r_q H_{p-1}. \tag{10}$$

Equation 8 now follows from Equations 9 and 10.

We next show that

$$\mathsf{tree}(Y_p) \geq \mathsf{pf}(OPT)/8. \tag{11}$$

The Remote-tree problem was considered in [6]: Find a set of p points F_p such that $\mathsf{tree}(F_p)$ is maximized. It was shown [6, Theorem 3.1] that $\mathsf{tree}(Y_p) \geq \mathsf{tree}(F_p)/4$. 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 point set X. From these inequalities we get (11).

The desired upper bound of $32H_p = O(\log p)$ on the approximation ratio $\mathsf{pf}(OPT)/\mathsf{pf}(PRE)$ follows from (11) and (8). □

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

Theorem 5.2 *The performance ratio of* Prefix *is* $O(\log p)$ *for* Remote-matching.

Proof. 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,

$$\mathsf{mat}(PRE) \geq \mathsf{pf}(PRE)/2 \geq \frac{\mathsf{tree}(OPT)}{32H_p} \geq \frac{\mathsf{mat}(OPT)}{32H_p}.$$

□

5.2 Lower Bounds

The performance analysis is tight within a constant factor.

Theorem 5.3 *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 $\log p/20 = \Omega(\log n)$.

Let t be such that $p \leq 1+4+\cdots+4^t = (4^{t+1}-1)/3$. Let n be $2^t(4^{t+1}-1)/3$. For simplicity, we assume that $p = 1+4+\cdots+4^t$, for integer t. The vertex set of G_p is partitioned into levels $0, 1, \ldots, t$, 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', $i \leq i'$, is $1/4^i$.

We first verify that the triangle inequality is satisfied for the edge weights of this graph. We consider the different cases:

- One vertex a is in a different block from the other two, and the level i of a's block is at most that of the other two. Then, $d(a,b) = d(a,c) = 1/4^i \geq d(b,c)$, satisfying the triangle inequality.

- Two vertices b, c are in the same block at level i, while a is in a block at level i', $i < i'$. Then, $d(a, b) = d(a, c) = 1/4^i \geq d(b, c)$, satisfying the triangle inequality.

- All three vertices a, b, c are in the same block at level i. Assume, without loss of generality that $d(b, c) = \min\{d(a, b), d(b, c), d(a, c)\} = 1/4^{i+j_1}$. Let j_2, $j_2 < j_1$, be the first bit where the label of a differs from the labels of b and c. Then, $d(a, c) = d(a, b) = 1/4^{i+j_2} \geq d(b, c)$, and the triangle inequality holds.

The theorem now follows from the following two lemmas. □

Lemma 5.4 $\mathsf{pf}(OPT) \geq t + 1$.

Proof. Consider the solution formed by choosing one vertex from each block. There are 4^i vertices chosen from 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. Hence, the cost of this solution, and therefore of OPT also, is at least $t + 1$. □

Lemma 5.5 $\mathsf{pf}(PRE) \leq 10$.

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, all 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 show one particular choice of vertices from outside the sparsest spheres which will make *Rest* small. Two vertices in the same block whose labels differ only in the last bit are called *buddies*. The distance between buddies is at most $1/4^t$. It can be easily seen that there are enough buddies outside the sparse spheres so that *Rest* can be formed entirely with buddies. Then, the contribution of *Rest* is at most $(p - q)4^{-t} \leq 4/3$. □

We can also show more generally that any performance analysis that is based on comparing the pf to the tree can at best result in a logarithmic ratio. Namely, we 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.

Theorem 5.6 *For infinitely many n, there exist graphs G_n such that*

$$\frac{\mathsf{tree}(G_n)}{\max_{P \subset V_n} \mathsf{pf}(P)} \geq \Omega(\log n).$$

Let $n = 2^t$, $t \geq 2$. We construct a family of graphs $G_n = (V_n, E_n)$ as follows. Each vertex has a label of the form $[e_1, e_2, \ldots, e_t]$, where $e_j \in \{0, 1\}$. The distance between distinct vertices $e = [e_1, e_2, \ldots, e_t]$ and $f = [f_1, f_2, \ldots, f_t]$, is $1/2^i$, where i be the smallest index such that $e_i \neq f_i$. Vertices are grouped into *metavertices*; a metavertex at level i, $[e_1, e_2, \ldots, e_i*]$ contains all vertices of the type $[e_1, e_2, \ldots, e_i, x_{i+1}, x_{i+2}, \ldots, x_t]$, $x_j \in \{0, 1\}$. A metavertex at level t consists of just a single vertex while the metavertex at level 0 contains all the vertices. The metavertices $[e_1, e_2, \ldots, e_{i-1}, 0*]$ and $[e_1, e_2, \ldots, e_{i-1}, 1*]$ are called a pair at level i. It is easy to verify that the triangle inequality is satisfied.

The theorem follows from the following two lemmas.

Lemma 5.7 $\mathsf{tree}(V) \geq t/2$.

Proof. Consider the spanning tree formed by connecting the 2^{t-1} pairs at level $t-1$ by edges of length $1/2^t$, 2^{t-2} pairs at level $t-2$ by edges of length $1/2^{t-1}$, ..., 2^i pairs at level i by edges of length $1/2^{i+1}$, ..., 2^1 pairs at level 1 by edges of length $1/2^2$, and a single edge of length $1/2$.

To verify that this is a minimum spanning tree, consider the cut $(S_i, V - S_i)$, where S_i is a metavertex at level i, and observe that the sole edge in the tree crossing the cut is a lightest edge across the cut. It is easily verified that the weight of this tree is $t/2$. $\square$

Lemma 5.8 $\max_{V' \subset V_n} \mathsf{pf}(V') \leq 1$.

Proof. Fix $V' \subset V_n$. The *value* of a metavertex is the sum $\sum_u d(u, V' - \{u\})$, where u ranges over all the vertices from V' in the metavertex. We say that a metavertex *contains* a vertex if the vertex belongs to the intersection of the metavertex and V'.
Claim: If a metavertex at level $i \leq t-1$ contains at least two vertices, its value is at most $1/2^i$.

Proof. By downward induction on i. The base case $i = t-1$ holds since a metavertex has 2 vertices of distance 2^i. For the inductive step, assume that a metavertex $[e_1, e_2, \ldots, e_{i-1}*]$ at level $i-1$ contains at least two vertices. If either one of the metavertices $[e_1, e_2, \ldots, e_{i-1}, 0*]$ or $[e_1, e_2, \ldots, e_{i-1}, 1*]$ contains no vertex, we are done by the inductive hypothesis. Otherwise, if one of them contains only a single vertex, its value is $1/2^i$ (the distance to other metavertex), and if it contains two or more vertices, then by the inductive hypothesis its value is at most $1/2^i$. Hence, the sum of the values of the pair, which equals the value of the metavertex at level $i-1$, is at most $1/2^{i-1}$. $\square$

The lemma follows by applying the claim to the metavertex at level 0 containing all the vertices of V'. $\square$

6 Bottleneck Problems

In this section, we examine the problem of maximizing the destruction radius and other related bottleneck problems. Our objective is to select p sites so as to maximize the difficulty of an adversary attack with k 'explosives' causing a complete destruction. Several optimization problems arise, depending on which parameter is to be maximized. We look at three such problems. In this section, we analyze the performance of GREEDY on these problems. In subsection 6.1 we give hardness results.

1. (Radius version) Any facility is a potential explosives site, i.e. a site for the placement of explosives. The objective is to force the adversary to use large explosives i.e. we pick the facility sites so as to maximize the destruction radius of the adversary's explosives.

2. (Steiner-radius version) Any vertex is a potential site for the placement of explosives. Thus, the explosion sites may be "Steiner points", in that they do not belong to the set of facility sites.

3. (Diameter version) The objective is the maximum distance between pairs of points in the same partition.

These radius problems do not fit in the framework we defined earlier of the weight of a Π-subgraph. However, they can be viewed as the *bottleneck* versions of such properties, i.e. the value is the maximum weight edge of a subgraph satisfying the property. Thus, the Radius version is the bottleneck problem of Remote-star, and Diameter version the bottleneck problem of a Remote k-Cliques problems.

We can formally define these problems as follows. We are given a graph G, and positive integers p, k, $k \leq p-1$. We are to find a set $P \subseteq V$ of p vertices that maximizes the following objective function.

Radius: $\min_{S \subseteq P, |S|=k} \max_{x \in P} d(x, S)$,

Steiner Radius: $\min_{S \subseteq V, |S|=k} \max_{x \in P} d(x, S)$,

Diameter: $\min_{P=P_1|\cdots|P_k} \max_{x,y \in P_i, i=1,2,\ldots,k} d(x, y)$.

For a point set X, let Radius(X), SteinerRadius(X) and Diameter(X) denote the values of the three variant problems on X. Observe that these values differ by a factor of at most 2.

$$\text{SteinerRadius}(X) \leq \text{Radius}(X) \leq \text{Diameter}(X) \leq 2\,\text{SteinerRadius}(X).$$

Let GR denote the value of the greedy solution.

Claim 6.1 *No matter how the greedy points are split into k parts, there exist two points which lie in the same parts and which are at a distance at least r_k apart.*

Proof. Consider the first $(k+1)$ greedy points. By the pigeonhole principle, some two will fall in the same part. The distance between those two is at least r_k. □

Theorem 6.2 *The performance ratio of* Greedy *is 2 for the Steiner-Radius and Diameter versions, but 4 for the Radius version.*

Proof. Claim 6.1 implies that GR is at least r_k for the Diameter case, and at least $r_k/2$ for the Radius and Steiner-Radius cases. Since each point of the optimal solution is within distance r_k from some point in GR, OPT is at most r_k for the Steiner-Radius case: this follows from the fact that we can use the vertices from GR as Steiner vertices. Also, if we partition the vertices of OPT according to the nearest vertex in GR, vertices in the same part are at most $2r_k$ apart. Thus, OPT is at most $2r_k$ for the Radius and Diameter cases. Hence the upper bounds claimed.

We show that these bounds are tight for the case $k = 1$ and $p = 4$, with other cases an easy variation. Consider the shortest-path distance graph of the unweighted 10-vertex graph in Fig. 1. Greedy may select $y_1, \ldots, y_4$ in sequence, resulting in diameter 2 and radius 1, while the optimal solution consists of $x_1, \ldots, x_4$, with diameter 4, Steiner-radius 2, and radius 4. □

Notice that in the process, we have given a constructive solution of the adversarial problem of selecting the explosion sites against the optimal solution. For instance, picking the first k greedy points as the explosion sites in the Steiner-Radius version results in a destruction radius that is at most twice the best possible of an optimal selection of p sites.

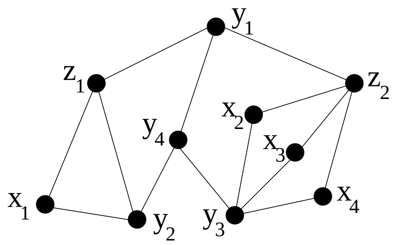

Figure 1: Hard instance for GREEDY for radius problems.

Also, note that using the explosion sites of the greedy solution as the sites of the optimal solution yields a ratio of 3 for the Steiner-Radius problem. This is because the distance of any optimal point to some greedy point is at most r_k, and from that greedy point to a greedy explosion site is at most GR. This ratio is tight for a basic anticover, as seen by the three points on a line: $0, 1$, and 3.

6.1 Hardness of bottleneck problems

The destruction radius problems are bottleneck problems.

- Remote-1-Radius is the bottleneck problem of Remote-star. More generally, Remote-k-Radius problem is the bottleneck problem of the Remote-Π problem that asks if there is a p-set P such that the size of the minimum dominating set of $G[P]$ is greater than k.
- Remote-k-Diameter is the bottleneck problem of the problem that asks if there is a p-set P such that $G[P]$ is $k+1$-chromatic.
- Remote-k-SteinerRadius is the bottleneck problem of the following one-way domination problem: Is there a p-set P such that no set of k vertices in G dominates P.

Given these characterizations, we can clarify the complexity of these problems.

Proposition 6.3 Remote-k-Diameter *is polynomial solvable for $k = 1$ or 2. For $k \geq 3$, it is co-NP hard, and hard to approximate within factor less than 2.*

Proof. The hardness follows from Theorem 7.4 and the fact that deciding 4-chromaticity is co-NP-hard.

Notice that any p-set that contains the furthest pairs of points is an optimal solution for $k = 1$. For $k = 2$, the non-k-colorable subgraphs are precisely the odd cycles. Let w_0 be the largest w such that T_w contains an odd cycle of length at most p. It suffices to choose any p-set that contains an odd cycle in T_{w_0} as an optimal solution. In particular, we can find the smallest odd cycle in T_{w_0} by running breadth-first search (for at most $p/2$ levels) starting from each of the n nodes. Adding any nodes to the vertices forming the cycle then yields an optimal solution. □

Proposition 6.4 Remote k-Steiner-Radius *is NP-hard, for any $k \geq 1$.*

Proof. We show this for $k = 1$, with other values an easy extension.

Let H be a $(1,2)$-graph of an unweighted graph G. Suppose a p-set P has a 1-Steiner radius 2 in H. Then, $\forall v \in V, \exists w \in P, d(v,w) = 2$. That is, $\forall v \in V, \exists w \in P, (v,w) \in E(G)$. Hence, P

is a *total dominating set* of G, i.e. every vertex in G (including those in P) has a neighbor in P. On the other hand, if P has a 1-Steiner radius 1, then there exists a vertex $v \in V$ of distance 1 from each vertex in P. Then, v is adjacent to each vertex of P, or nonadjacent in G to each vertex of P. Thus P does not totally dominate G. Hence, the 1-Steiner radius of H is at least 2 iff G contains a totally dominating set of size at most p.

The existence of a total dominating set is NP-hard, for arbitrary p. Hence, so is Remote 1-Steiner Radius. Similarly, obtaining an approximation less than 2 is also hard, as well as obtaining an approximation within any factor on nonmetric graphs. □

Proposition 6.5 Remote k-Radius *is NP-hard, for $k \geq 2$, but polynomial solvable for $k = 1$.*

Proof. The case $k = 1$ is equivalent to finding a p-set P in the threshold graph such that each vertex in $G[P]$ is of degree less than $p-1$ (or, alternatively, each vertex in the complement graph $\overline{G[P]}$ is of positive degree). Such a set can easily be found by a greedy accumulation.

Consider now the case $k = 2$. Given a $(1,2)$-graph H of an unweighted graph G, the question if Remote 2-Radius on H equals 2 is equivalent to asking if there exists a p-set P satisfying the following property: For any potential center-pair x, y, there is a vertex z of distance 2 from both x and y. This is equivalent to the following problem on G:

Pairwise-distance-2 Set
Given: Graph $G' = (V', E')$, integer p.
Question: Is there a set $P \subseteq V'$ of at least p vertices such that for any $x, y \in P$, there is a $z \in P$ such that $(x, z) \in E'$ and $(y, z) \in E'$.

We show the Pairwise-distance-2 set problem to be NP-hard by a reduction from Clique. Given a graph $G = (V, E)$ as input to the k-Clique decision problem, form the graph $G' = (V', E')$ as follows. The vertex set consists of C copies of V, for $C = n^2$, called *node-vertices* and an *edge-vertex* for each edge of G. The node-vertices are adjacent to the edge-vertices corresponding to the edges to which they are incident in G. The node-vertices internally form an independent set, while the edge-vertices form a clique. Formally,

$$\begin{aligned} V' &= \{v_i^j : i = 1, \ldots, |V|, j = 1, \ldots, C\} \cup \{v_{xy} : (x, y) \in E\} \\ E' &= \{v_x^j v_{xy} : (x, y) \in E, j = 1, \ldots, C\} \cup \{v_{xy} v_{zw} : (x, y), (z, w) \in E\} \end{aligned}$$

Let S be a feasible pairwise-distance-2 set. First observe that for any pair $v_x^i, v_y^j \in S$, the only 2-path from v_x^i to v_y^j is through an edge-vertex to which both vertices are incident, and this happens only if x and y are adjacent in G. It follows that

$$|S| \leq n^2 \cdot \omega(G) + |E(G)|,$$

where $\omega(G)$ is the clique number of G.

In the other direction, let X be a clique of G, and consider the set $S = \{v_x^i, v_y^j, v_{xy} : v_x, v_y \in X, i, j = 1, \ldots, C\}$. Then, it is easily verified that S is a pairwise-distance-2 set. Thus,

$$OPT(G') \geq n^2 \cdot \omega(G) + \binom{\omega(G)}{2}.$$

Combined we see that $\lfloor OPT(G')/n^2 \rfloor = \omega(G)$. Thus, even obtaining an n^ϵ-approximate solution to the pairwise-distance-2 problem is NP-hard, for some $\epsilon > 0$, given the hardness of approximating $\omega(G)$. □

7 General Hardness Results

We give in this section hardness results that apply to general classes of dispersion problems.

For previously studied problems, NP-hardness has been established: edge [13, 15, 12], clique [12], tree, cycle, pseudoforest, matching [6]. Further, these reduction showed that for all of the problems listed above except for clique, 2-approximation is NP-hard for graphs with weights 1 and 2, and $n^{1-\epsilon}$-approximation is hard for nonmetric networks, for every $\epsilon > 0$.

We argue here that all nontrivial remoteness problems are NP-hard. A property of graphs yields a trivial remoteness problem if it holds for graphs with no edges or fails for all graphs (or for all but a finite number of graphs).

A $(1,2)$-*graph* H of an unweighted graph G is a complete graph on the same vertex set, with edge weights 1 and 2 such that uv has weight 2 in H iff uv is an edge in G.

Proposition 7.1 *The decision problem for* Remote-Π *is either trivial or NP-hard.*

Proof. The proof is by reduction from Clique, as in previous papers [12, 6]. Given an unweighted graph G as an input to Clique, form the corresponding $(1,2)$-graph H. Then, there exists a p-subgraph in H where the weight of any pair is 2 iff there exists is a p-clique in G. Let l be the number of edges in the minimum size structure satisfying Π. If there is a p-clique in G, then there is a subgraph in H where every structure is of weight at least $2l$. If there is no p-clique in G, then every subgraph in H contains an edge of weight 1, and thus every p-subgraph contains a valid structure of weight at most $2l-1$. □

The decision problem for Remote-Π is to determine, given a graph G and integer t, if there exists a subgraph P with p vertices such that $\pi(G[P]) \geq t$, where $G[P]$ is the subgraph induced by P. Thus, if $p = n$, it reduces to the complement of the Π-problem. For some instances it may be harder than the search problem, which only outputs a set but does not say anything regarding its value.

Observation 7.2 *Let* Π *be a NP-hard property, i.e. it is NP-hard to decide whether a given graph* G *has the property* Π*. Then, the* Remote-Π *decision problem is co-NP-hard, and thus hard for both NP and co-NP.*

For an upper bound, we can only argue that the Remote-Π problem is at most one level higher in the polynomial-time hierarchy than the Π-decision problem itself. We conjecture that for any NP-complete graph property, the corresponding remote problem is Σ_2^p-complete.

Conjecture 7.3 *Let* Π *be a NP-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.*

We may also consider *bottleneck* problems. The Remote-Π-bottleneck decision problem is to determine, given a graph G and a real value w, whether there exists a set P of p vertices such that any Π-structure on $G[P]$ contains an edge of weight at least w. As an example, Remote-edge is a bottleneck problem for Remote-clique.

These problems are best observed in terms of related unweighted graphs. Given a weighted graph G and a weight w, the unweighted *threshold* graph T_w of G has the same vertex set and contains an unweighted edge for each edge of weight at least w in G. Note that if we take an unweighted graph G, form its $(1,2)$-graph H and take the threshold graph T_2 of H, we obtain the original graph G.

Observe that the Π-bottleneck value on P equals the smallest value w such that Π holds on $G[P]$. Then, we have that Remote-Π-bottleneck on G is at least w_0 iff w_0 is the smallest value w such that Π holds on the threshold graph T_w of $G[P]$ for some P. To find w_0 we can apply binary search, adding a $\log n$ factor to the time complexity. Hence, optimization is polynomial reducible to the decision problem on the threshold graphs.

Theorem 7.4 *Let Π be a co-NP-hard property. Then, the* Remote-Π-bottleneck *problem is co-NP-hard. In fact, it is hard to approximate it within a factor of less than 2.*

Proof. Given a graph G input to the co-NP-hard Π-decision problem, form its $(1,2)$-graph H. Observe that the Π-bottleneck value on H with $p = |V(G)|$ equals 2 iff G satisfies Π. Thus, distinguishing whether the value of the bottleneck problem is 1 or 2 is co-NP-hard. □

The approximation hardness results carry over to the related partition problems, where the objective function is measured within a fixed number of parts of the subgraph.

Proposition 7.5 *Let Π be a property where vertices have degree at least one (in a nontrivial Π-graph). Let* Remote-k-Π *be the remote problem where the objective function is the sum of the weights of the edges of k disjoint Π-subgraphs. Then* Remote-k-Π *(on $p-k+1$ points) is at least as hard to approximate as* Remote-Π *(on p points). This also holds for the bottleneck problem, where the objective is the maximum weight of an edge in any of the k Π-subgraphs.*

Proof. Given a hard instance G for Remote-Π, form a network G' by adding $k-1$ vertices to G and make the weight of their incident edges be (effectively) infinity.

Consider the Remote-k-Π solution P' consisting of the $k-1$ new nodes along with an optimal Remote-Π set P on $p-k+1$ nodes. Then, if the adversary assigns any two of the new nodes in the same part, the value of the objective function would be infinite; thus, the only reasonable partitioning is to assign the $k-1$ new nodes to separate parts, with P in the last part. The new nodes now do not contribute to the objective function, implying that

$$OPT_{k\text{-}\Pi}(G') \geq OPT_{\Pi}(G),$$

where OPT_X denotes the value of the optimal solution for problem X.

On the other hand, no matter what set P' is chosen by a Remote-k-Π algorithm, the adversary can always partition it so that all but one part contain only single vertices that do not contribute to the objective value, with the last part containing $p-k+1$ nodes from G. Thus,

$$ALG_{k\text{-}\Pi}(G') \leq ALG_{\Pi}(G),$$

that is, the set produced on G' gives at least as good solution on G (when appropriately restricted to vertices of G). Hence, approximating Remote-k-Π is no easier than Remote-Π. This argument holds equally for the corresponding bottleneck problems. □

Some examples of Π for which the above proposition applies are tree, pseudoforest, cycle, and clique.

7.1 Limitations of Greedy

Here we give lower bounds on the performance of Greedy on some general classes of remoteness problems. It is generally possible to argue similar bounds even if Greedy tries all n possible starting points, such as was done in [6] for the Remote-tree problem. This requires some added technical complications that may be problem specific, thus we do not pursue that here.

Proposition 7.6 *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 in the one chosen by Greedy. □

We can show that for a host of problems the performance ratio of Greedy either grows linearly with p, or there is no upper bound on it that is independent of the edge weights.

Definition 7.7 *The following function counts the number of edges in any p-vertex Π-structure that must cross an $(s, p-s)$-cut.*

$$Cross_p(\Pi, s) = \min_{\substack{H \in \Pi \\ V(H)=\{v_1,\ldots,v_p\}}} |\{v_i v_j \in E(H) \ : \ 1 \le i \le s, s+1 \le j \le p\}|$$

Let us consider the value of $Cross(\Pi, s)$ for some of the structures considered here, for $1 \le s \le p-s$. We can see that $Cross_p(\mathsf{pf}, s) = 0$, for any s, and $Cross_p(\mathsf{mat}, s) = 0$ when s is even, since one can form these structures without crossing a particular cut of the subgraph. Also, $Cross_p(\mathsf{tree}, s) = Cross_p(\mathsf{cycle}, s) = 1$, $Cross_p(\mathsf{star}, s) = s$, and $Cross_p(\mathsf{clique}, s) = s \cdot (p-s)$.

Intuitively, we construct an instance with a "cutting cleavage", i.e. consisting of two clusters that are far apart. The distance between the two clusters overwhelms the intracluster distances, so that the only measure that matters is the number of edges in the minimum Π-structure on the selected point set that cross the cleavage. *Cross* counts this for different ways of splitting the p vertices among the two clusters. By adjusting the edge weights, Greedy can be made to pick the number of points from one cluster that yields the fewest number of forced crossing edges, while OPT picks the number that maximizes the number of forced crossing edges.

Proposition 7.8 *Let $d_p = \min_{1 \le s \le p-1} Cross_p(\Pi, s)$ and $D_p = \max_s Cross_p(\Pi, s)$. The performance ratio of* Greedy *on* Remote-Π *is at least arbitrarily close to D_p/d_p when $d_p > 0$, and unbounded when $d_p = 0$ and $D_p > 0$. This holds even for the 1-dimensional case, where edge weights correspond to distances on the real line.*

Proof. Let t_p (T_p) be the value of s that minimizes (maximizes) $Cross_p(\Pi, s)$. Without loss of generality, $t_p, T_p \le p/2$.

Let ϵ be a small number and let $\epsilon' = \epsilon/n$. For any n such that $n \ge 2p - t_p$, we construct an instance on n vertices, consisting of two clusters, Q with $p - t_p$ vertices, and Q' with the remaining $n - p + t_p \ge p$ vertices. Vertices in Q correspond to the points $-\epsilon i$ on the real line, $i = 1, \ldots, p - t_p$, while vertices in Q' correspond to the points $1 + i/\epsilon'$, $i = 1, \ldots, n - p + t_p$. Observe that all distances between pairs of points in Q' are less than all distances within Q.

GREEDY first chooses a vertex from each of Q and Q', followed by the rest of Q, and finally selects $t_p - 1$ vertices from Q'. Since by definition there exists a Π-structure with only d_p edges crossing a $(t_p, p - t_p)$-cut, the cost of that structure is the sum of $Cross_p(\Pi, t_p) = d_p$ crossing edges of weight 1, along with at most $\binom{p}{2}$ edges of weight ϵ or ϵ'. On the other hand, the optimal solution chooses T_p vertices in Q and the rest in Q'. Every Π-structure has at least D_p edges crossing a $(T_p, p - T_p)$-cut, hence the minimum cost Π-structure of OPT is of cost at least D_p. Since ϵ can be arbitrarily close to 0, the ratio between the cost of the optimal solution to that of the greedy solution can be arbitrarily close to D_p/d_p. □

This shows that the performance ratio of GREEDY on e.g. Remote-matching and Remote-pseudoforest is unbounded. Also, the performance ratio is at least $\Omega(p)$ on Remote-clique, and Remote-star.

For an explicit example where GREEDY fails for Remote-pseudoforest, consider the following: $V = \{a_1, a_2, b_1, b_2, \ldots, b_{n-2}\}$, $d(a_i, b_j) = 1$, $d(a_1, a_2) = 2\epsilon$, $d(b_i, b_j) = \epsilon$, $p = 4$. An optimal selection of the vertices is any one of the a_i's and any three of the b_i's giving $\mathsf{pf} > 1$, while GREEDY selects a_1, a_2 and two of the b_i's for $\mathsf{pf} = 6\epsilon$.

8 Discussion

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

We have also considered a number of extensions of previously studied problems. These include bottleneck problems – where the objective function is a maximum, rather than the sum, of the edge weights of a structure and Steiner problems – where the objective function may include vertices outside the selected set P.

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? Is there a parallel algorithm for obtaining an anticover? And finally, a further study on the applied aspects from the management science viewpoint would be desirable.

References

[1] C. Baur and S. Fekete. Approximation of geometric dispersion problems. *Algorithmica*, to appear.

[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] U. Feige, G. Kortsarz, and D. Peleg. The dense k-subgraph problem. *J. Algorithms*, to appear.

[6] M. M. Halldórsson, K. Iwano, N. Katoh, and T. Tokuyama. Finding subsets maximizing minimum structures. *SIAM J. Disc. Math.*, 12(3):342–359, 1999.

[7] R. Hassin, S. Rubinstein, and A. Tamir. Approximation algorithms for maximum dispersion. *OR Letters*, 21:133–137, 1997.

[8] J. Håstad. Clique is hard to approximate within $n^{1-\epsilon}$. *Acta Mathematica*, 182:105–142, 1999.

[9] G. Kortsarz and D. Peleg. On choosing a dense subgraph. In *Proc. 34th IEEE Symp. on Found. of Comp. Sci.*, pages 692–701, Nov. 1993.

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

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

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

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

[14] A. Tamir. Comments on the paper 'Heuristic and special case algorithms for dispersion problem' by S.S. Ravi, D.J. Rosenkrantz and G.K. Tayi. *Operations Research*, 46:157–158, 1998.

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

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

9 Significant changes made

1. As suggested by Referee 1, the paper has been substantially reorganized to emphasize problems instead of algorithms.
2. As suggested by Referee 1, the section on Terrorism Defense/Bottleneck problems in the Introduction has been significantly expanded.
3. We have briefly addressed the issue, raised by Referee 1, of considering lower bounds of the generalized version of Greedy i.e. if different starting points are considered.
4. As suggested by both Referees, we have added a table of known results, as well as expanding the list of dispersion problems in Section 1.
5. Figure 1 has been corrected, addressing a point made by Referee 1.
6. As suggested by Referee 2, an example for the lower bound of the MATCHING algorithm for the Remote-Star problem has been added.
7. Nearly all the minor comments of both Referees have been addressed.