Minimizing Movement

Erik D. Demaine* MohammadTaghi Hajiaghayi*†‡ Hamid Mahini§¶
Amin S. Sayedi-Roshkhar§ Shayan Oveisgharan§ Morteza Zadimoghaddam§

Abstract

We give approximation algorithms and inapproximability results for a class of movement problems. In general, these problems involve planning the coordinated motion of a large collection of objects (representing anything from a robot swarm or firefighter team to map labels or network messages) to achieve a global property of the network while minimizing the maximum or average movement. In particular, we consider the goals of achieving connectivity (undirected and directed), achieving connectivity between a given pair of vertices, achieving independence (a dispersion problem), and achieving a perfect matching (with applications to multicasting). This general family of movement problems encompass an intriguing range of graph and geometric algorithms, with several real-world applications and a surprising range of approximability. In some cases, we obtain tight approximation and inapproximability results using direct techniques (without use of PCP), assuming just that $\mathrm{P} \neq \mathrm{NP}$.

1 Introduction

Consider a group of firefighters surrounding a forest fire. Each firefighter is equipped with a reliable but short-range radio (walkie-talkie) as well as limited connectivity to a satellite (or other central location) for triangulating and sharing the approximate positions of firefighters. To form an effective communication network (for voice or data traffic), the firefighters' radios must form a connected graph. This scenario naturally leads to the following problem: given the current locations of the firefighters, find the minimum distance (time) required for each firefighter to move to reach a configuration that induces a connected radio network. More precisely, we wish to minimize the maximum movement of the firefighters such that, in their final positions, any two firefighters can talk to each other in the reliable radio network, possibly using multiple hops.

Of course, this playful description of the problem is rhetorical: in reality, the objects are not firefighters but are, say, autonomous robots with limited wireless connectivity and limited mobility in the field because of energy and resource constraints, so they wish to minimize the use of these resources to form a reliable radio network. See, e.g., [CHP$^+$04a, CHP$^+$04b, BDHR05] for descriptions of such practical scenarios.

The problem described above is one example of a natural broader family of problems, called *movement problems*, which we study systematically in this paper. In particular, the firefighting problem can be abstracted into a problem called ConMax: minimize maximum movement to reach connectivity. This basic connectivity problem has many variations. For example, ConSum asks to minimize the total movement, which may be useful for reducing average power consumption; while ConNum asks to minimize the number of firefighters (robots) that have to move. In DirConMax, and analogously DirConSum and DirConNum, the radio connectivity is not necessarily symmetric and forms a directed network, e.g., because different radios have different power levels, and the goal is to ensure that everyone can receive messages from a fixed root (the captain). In PathMax, PathSum, and PathNum, the goal is to re-arrange the objects to connect two specified locations.[1]

Many more variations arise from changing the desired property of the final configuration. In general, for a specified property P of configurations of objects, the goal of a movement problem is to minimize the (maximum or total/average) movement in a motion that ends with a configuration satisfying property P. The objects can be represented either as points or equivalently bodies that can only be translated, say in the plane, or as pebbles placed on the vertices of a graph that can move along edges. Many problems in this family arise naturally in the context of robotics, particularly in organizing the behavior of swarms of robots (see, e.g., [HAB$^+$03, LaV06, RW95, SPS03]).

A simple version of the movement problem is *collocation*, where the goal is to move all objects to a common lo-

*MIT Computer Science and Artificial Intelligence Laboratory, {edemaine,hajiagha}@mit.edu. Supported in part by NSF under grant number ITR ANI-0205445.

†Department of Computer Science, Carnegie Mellon University.

‡Supported in part by Institute for Theoretical Physics and Mathematics (IPM) under grant number CS1384-6-01.

§Department of Computer Engineering, Sharif University of Technology, {mahini,sayedi,oveisgharan,zadimoghaddam}@ce.sharif.edu

¶Supported in part by Institute for Theoretical Physics and Mathematics (IPM) under grant number CS1384-2-01.

[1]For example, the firefighters might want to chain together their water hoses from a fire hydrant to the fire.

cation. In this case, we obtain two classic problems: when minimizing the maximum movement, we have the 1-center problem; when minimizing the total movement, we have the 1-median problem. These problems have well-known polynomial-time exact solutions.

Another interesting version of the movement problem is *dispersion*, where the goal is to distribute the objects in order to guarantee a minimum pairwise separation between the objects. In the context of a radio network, this goal is equivalent to guaranteeing that the radio network forms an empty graph or an independent set. Thus, we refer to this problem as IndMax, IndSum, or IndNum according to the objective function. This problem effectively asks to spread out the objects (e.g., robots) while keeping them as close as possible to their original locations. The problem also has applications to map labeling [DMM$^+$97, JBQZ04, SW01, JQQ$^+$03], where the goal is to find placements of labels as close as possible to the specified features of the map such that the labels do not overlap each other (so their centers are sufficiently separated).

Another version of the movement problem that arises in the context of broadcasting or multicasting is to move the objects into nearby pairs so that these pairs can exchange information. More precisely, in MatchMax, MatchSum, and MatchNum, the goal is to minimize the movement of the objects to a position having a perfect matching of the objects such that each matched pair can communicate (i.e., the objects are within distance 1 of each other). This problem is essentially a mobile version of the pseudo-matching problem (also known as path-matching) considered in the context of broadcasting and multicasting in cut-through routed networks [CFKR98, Coh98, GHMM02]. The MatchMax problem is also closely related to one "round" of the freeze-tag problem [ABF$^+$02, ABG03, SABM04] in which a swarm of mobile robots must collectively "wake up", starting from a single awake robot, and moving awake robots next to sleeping robots to awaken them.

Several of the problems considered in this paper can be viewed as considering the extent to which we exploit the mobility of existing resources to achieve desired global properties of the network such as connectivity. Related to this endeavor is work that considers how to augment networks (consisting of nonmobile sensors) by adding additional resources to achieve such global properties; see, e.g., [BDHR05, CHP$^+$04a, CHP$^+$04b]. In fact, we can view the class of movement problems as strictly more general than these augmentation problems, by imagining additional mobile resources initially "at infinity" and the goal is to minimize the total movement of these resources (and therefore minimize the number of resources moved).

1.1 Motion Problems and Model.

Before we describe our specific results, we formally define the model and the movement problems we consider.

The three general families of problems we consider are *minimum maximum movement to property* P, *minimum total movement to property* P, and *minimum number of movements to property* P. In all cases, we are given an (undirected or directed) graph $G = (V, E)$ with $|V| = n$ vertices, m *pebbles*, and a property P on "configurations". A *configuration* is a function assigning each pebble to a vertex of V; more than one pebble can be on a single vertex. We say that each such assigned vertex is *occupied* by a pebble. We are given an *initial configuration* for the pebbles. A *motion* assigns a path $\pi(p)$ in the graph G for each pebble p, starting at the vertex specified by the initial configuration and ending at some *target vertex*, also called the *target position*. (Thus, pebbles can move only along edges.) The length $|\pi(p)|$ of the path is the *movement* of p. The *maximum movement* of a motion is the maximum length of any path; the *total movement* is the total length of all paths; and the *number of movements* is the number of paths of nonzero length. The target vertices of pebbles define the *target configuration* of the motion. The goal is to find a motion that minimizes one of these three measures subject to the target configuration satisfying property P.

This graph-theoretic formulation of the movement problems also captures the geometric setting. For example, the *Euclidean plane* is defined by an infinite graph whose vertices correspond to points $p = (p_x, p_y)$, and edges connect two distinct vertices p and q whose Euclidean distance $d(p, q) = \sqrt{(p_x - q_x)^2 + (p_y - q_y)^2}$ is less than 1. This definition models mobile nodes with unit communication radius. Because the graph is infinite, there is no notion of "n", so we define $n = m$, the number of pebbles.

We define the following properties P of interest and their associated problems of minimizing maximum movement. In most cases, we state a property P on graphs, implicitly referring to the subgraph of G induced by the vertices occupied by pebbles in the configuration.

1. Minimum maximum movement to connectivity (ConMax): P is connectivity.
2. Minimum maximum movement to connectivity in directed graphs (DirConMax): P is directed connectivity from every vertex to some root vertex.
3. Minimum maximum movement to s-t connectivity (PathMax): P is having a path between two certain vertices s and t.
4. Minimum maximum movement to independence (IndMax): P is that no two pebbles occupy the same or adjacent vertices.
5. Minimum maximum movement to perfect matchability (MatchMax): P is the property that there is a perfect matching in the graph on pebbles in which two pebbles p and q are adjacent precisely if their distance $d_G(p, q)$ in G is at most 1.

	Max	**Sum**	**Num**
Con	$O(\sqrt{m/\mathrm{OPT}})$ (§2.1)	$\tilde{O}(\min\{n,m\})$ (§5.2) $\Omega(n^{1-\varepsilon})$ (§5.1)	$O(m^{\varepsilon})$ $\Omega(\log n)$
Path	$O(\sqrt{m/\mathrm{OPT}})$ (§2.2)	$O(n)$ (§5.3)	polynomial
DirCon	εm (§2.3) $\Omega(n^{1-\varepsilon})$ (§2.4)	open	$O(m^{\varepsilon})$ $\Omega(\log^2 n)$
Ind	$1+\frac{1}{\sqrt{3}}$ additive in $\mathbb{R}^2$ (§3)	open	PTAS in $\mathbb{R}^2$
Match	polynomial (§4)	polynomial (§5.4)	polynomial

Table 1: A summary of our results.

Analogously, we define the problems of minimizing total movement (ConSum, DirConSum, IndSum, PathSum, and MatchSum) and minimizing the number of movements (ConNum, DirConNum, IndNum, PathNum, and MatchNum) to achieve the same properties. To our knowledge, none of these problems have been considered before in an algorithmic setting.

1.2 Our Results. We prove several approximation and inapproximability results for the problems listed above, in many cases obtaining tight bounds (assuming just P $\neq$ NP). The various movement problems show a surprising range of difficulty, not consistent with the nonmovement (standard) version of each problem. For example, testing connectivity of a graph is trivial, but DirConMax and ConSum are $\Omega(n^{1-\varepsilon})$-inapproximable, while the best approximation so far for ConMax is $O(\sqrt{m/\mathrm{OPT}})$ in a graph or in the Euclidean plane, and we give evidence that even the geometric scenario is difficult. On the other hand, we give an additive $O(1)$-approximation for IndMax in the Euclidean plane, even though the nonmovement version (independent set) is very hard for graphs and not known to be solvable exactly in the Euclidean plane. Yet some movement problems such as MatchMax turn out to have polynomial-time solutions. Our hardness results are particularly strong, yet they do not use techniques such as PCP and thus avoid any higher-level complexity assumptions, making them of independent interest.

We focus primarily on the maximum-movement problems, proving various approximability and inapproximability results in Sections 2, 3, and 4. We then consider the total-movement versions of the problems in Section 5. The number-of-movements versions tend to be less interesting because there is little correlation between a pebble's initial and final position, so we omit the details from this extended abstract. Table 1 summarizes all of our results.

In the interest of space, several proofs are deferred to the appendices.

2 Minimum Maximum Movement to Connectivity

We begin with the problem of ConMax, a well-motivated problem as described in the introduction. To provide some intuition about the problem, Figure 1 gives an example of a challenging instance. Here there is a "global" solution using maximum movement of 1, but any "local" solution (such as all pebbles approaching a common location) requires maximum movement of $\Omega(n)$.

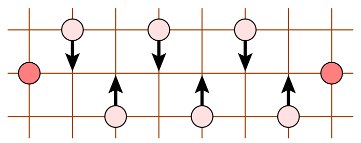
Figure 1: Optimally moving the pebbles (drawn as disks) into a connected configuration requires a global solution (drawn with arrows).

It is also not hard to see that the problem is NP-complete in general, even to approximate better than a factor of 2. We can reduce from Hamiltonian Path as follows. Given a graph $G = (V, E)$, we subdivide each edge in E into a path of three edges, and attach a new leaf vertex to each vertex in V. We place two pebbles on each vertex in V and we place one pebble on each added leaf. Any solution to this instance of ConMax of maximum movement 1 can move the pebble on each leaf to its neighboring vertex in V, and must move the two pebbles on each vertex in V toward neighboring vertices to induce a connected subgraph. Such a solution corresponds to a connected maximum-degree-2 subgraph in G that visits every vertex in V, i.e., a Hamiltonian path in G.

2.1 $O(\sqrt{m/\mathrm{OPT}})$-Approximation for ConMax. In this section we develop an $O(\sqrt{m/\mathrm{OPT}})$-approximation algorithm for ConMax, where m is the number of pebbles. (Note that m can be much smaller than n.) In particular, this algorithm is an $O(\sqrt{n})$-approximation algorithm if the initial configuration places at most one pebble on each vertex. We can also show how to convert this approximation algorithm, or indeed any approximation algorithm for ConMax, to work in the Euclidean plane at a small extra cost (a multiplicative $1 + \varepsilon$ factor in the approximation ratio).

THEOREM 2.1. *There is an $O(\sqrt{m/\mathrm{OPT}})$-approximation algorithm (and thus also an $O(\sqrt{m})$-approximation algorithm) for ConMax.*

Given a subset S of vertices in a graph G, the *dth power induced on S*, denoted by $G^d[S]$, has vertex set S and has an edge (u, v) between two vertices $u, v \in S$ if and only if there is a path in G between u and v with at most d edges.

LEMMA 2.1. *Consider an instance of ConMax problem, consisting of a graph G and an initial configuration of m pebbles, with an optimal solution of maximum movement* OPT. *For any integer k between 0 and $m/2$, there is a subset S of vertices of G satisfying the following properties:*

1. *Every vertex in S is occupied by a pebble in the initial configuration.*

2. *The shortest-path distance between any two distinct vertices in S is greater than $2k + 4\mathrm{OPT}$.*

3. *The $(2k + 6\mathrm{OPT} + 1)$th power of G induced on S is connected.*

4. *Every vertex v in S has at least $2k$ pebbles whose shortest-path distance to v is at most $k + 2\mathrm{OPT}$.*

5. *For every vertex w occupied by a pebble in the initial configuration, there is a vertex u in S whose shortest-path distance to w is at most $3k + 8\mathrm{OPT} + 1$.*

Proof: We compute S via a greedy algorithm. Initially S is the empty set, which satisfies Properties 1–4. In each step, if there is a vertex whose addition to S would still satisfy Properties 1–4, we add the vertex to S.

First we prove that the greedy algorithm computes a nonempty set S, i.e., at the first step, there is a vertex we can add it to S. Let T be a spanning tree of the (connected) graph induced by the target configuration in the optimal solution OPT. Define a *center* c of T to be a vertex of T that minimizes the maximum distance from c to any vertex of T. We claim that c is within distance k of at least $2k$ target positions of pebbles, and thus the initial position u of any pebble whose final position is c is within distance $k + 2\mathrm{OPT}$ of at least $2k$ initial positions of pebbles, and therefore $S = \{u\}$ satisfies Properties 1–4. The proof of this claim divides into two cases. In Case 1, every vertex of T is within distance k of c, and thus the target positions of all m pebbles are within distance k of c, proving the claim. In Case 2, there is at least one vertex at distance exactly $k + 1$ from c. In this case, we claim by induction on k that there are at least $2k$ vertices of T within distance k of c. Note that we remain in Case 2 even when considering smaller values of k. In the base case, $k = 0$ and the claim is vacuous. In the general case $k > 0$, by induction, there are at least $2k - 2$ vertices of T within distance $k - 1$ of c. Because we are in Case 2, there is at least one vertex v at distance exactly k from c, and at least one vertex w at distance exactly $k + 1$ from c. If there are at least two vertices at distance exactly k from c, then we have the claim. If vertex v is the only vertex at distance exactly k from c, then we argue that c cannot be a center. Moving c one step toward v decreases the distance from c to v, w, and any vertices of T with distance at least k, in particular decreasing w's distance of $k + 1$, while the distance from c to all other vertices (which have distance at most $k - 1$) increases by at most 1 and so the distance remains at most k. Therefore, this move decreases the maximum distance from c to any vertex of T, contracting centrality of c.

Now consider the maximal set S output by the greedy algorithm, and suppose for contradiction that some vertex w is not within distance $3k + 8\mathrm{OPT} + 1$ of its nearest vertex s in S. (If there is more than one such vertex s, we choose one arbitrarily.) For any vertex v, let ND_v denote the distance between v and its nearest vertex in S. Thus, $ND_w > 3k + 8\mathrm{OPT} + 1$. Let w' and s' denote the target positions for some pebble initially on vertex w and for some pebble initially on vertex s, respectively, in the optimal solution OPT. Refer to Figure 2. Let $P = \langle w' =$

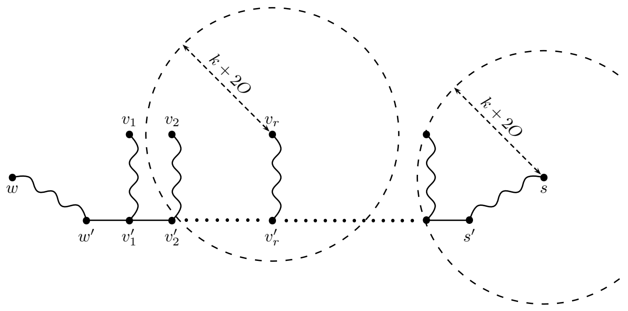

Figure 2: Path between w and s.

$v'_0, v'_1, v'_2, \ldots, v'_j = s'\rangle$ be a path between w' and s' in the (connected) graph induced by the target positions in OPT. Let v_i denote the initial position of some pebble whose target position is v'_i in OPT, making choices so that $v_0 = w$ and $v_j = s$. Thus, the distance between v_i and v'_i is at most OPT. By the triangle inequality, $ND_{v_i} \leq ND_{v_{i+1}} + 2\mathrm{OPT} + 1$. Because $ND_{v_0} = ND_w > 3k + 8\mathrm{OPT} + 1$, because $ND_{v_j} = ND_s = 0$, and because ND_{v_i} decreases by at most $2\mathrm{OPT} + 1$ each time we increment i, there is an index r such that $2k + 4\mathrm{OPT} < ND_{v_r} \leq 2k + 6\mathrm{OPT} + 1$.

We claim that we can add v_r to S while satisfying Properties 1–4, contradicting the maximal choice of S. By our choice of v_r, we satisfy Properties 1–3. By the triangle inequality, the distance between w and v_r is at least $ND_w - ND_{v_r} > (3k + 8\mathrm{OPT} + 1) - (2k + 6\mathrm{OPT} + 1) = k + 2\mathrm{OPT}$. Thus, the distance between w' and v'_r, namely r, is at least $k + 2\mathrm{OPT} - 2\mathrm{OPT} = k$. Similarly, by the triangle inequality, the distance between v_r and s is at least $ND_{v_r} - ND_s > 2k + 4\mathrm{OPT}$. Thus, the distance between v'_r and s', namely $j - r$, is at least $2k + 4\mathrm{OPT} - 2\mathrm{OPT} = 2k + 2\mathrm{OPT} > k$. Therefore, $v'_{r-k}, v'_{r-k+1}, \ldots, v'_r, \ldots, v'_{r+k-1}, v'_{r+k}$ are $2k + 2$ vertices along the path P. The corresponding vertices $v_{r-k}, v_{r-k+1}, \ldots, v_r, \ldots, v_{r+k-1}, v_{r+k}$ are occupied by pebbles and have distance at most $k + 2\mathrm{OPT}$ from v_r. Hence, we satisfy Property 4. $\square$

LEMMA 2.2. *Given an instance of ConMax problem with m pebbles and with an optimal solution of maximum movement* OPT, *for any integer k between* 0 *and* $m/2$, *there is a polynomial-time algorithm to find a motion with maximum movement at most* $5k + 14\mathrm{OPT} + 2 + (6\mathrm{OPT} + 1)m/(2k)$.

Proof: The algorithm proceeds as follows.

1. Find a subset S of vertices of G with the properties of Lemma 2.1.

2. Move each pebble to its nearest vertex in S.

3. Let H be the $(2k + 6\mathrm{OPT} + 1)$th power of G induced on S. By Property 3, H is connected, so let T be a spanning tree of H, and root it at an arbitrary vertex.

4. For each vertex v in S other than the root, move all but one of the pebbles on v to occupy some of the vertices

on the path in G corresponding to the edge between v and its parent in the tree T.

5. Let T' be the tree in G obtained by combining the paths corresponding to the edges of T. For every vertex of T' that is unoccupied by a pebble, move all pebbles one step in T' toward that vertex.

By Property 5, Step 2 moves each pebble at most $3k + 8\mathrm{OPT} + 1$ steps. By Properties 2 and 4 of Lemma 2.1, for every vertex s in S, there are at least $2k$ pebbles that are closer to s than to any other vertex in S. Thus, after Step 2 of the algorithm, every vertex s in S is occupied by at least $2k + 1$ pebbles. By Property 3, Step 4 moves each pebble at most $2k + 6\mathrm{OPT} + 1$ steps. After Step 4, at most $6\mathrm{OPT} + 1$ vertices of each path corresponding to an edge of T lack a pebble. Thus the tree T' in G has at most $|S|(6\mathrm{OPT} + 1)$ vertices that lack a pebble. Each iteration of the loop in Step 5 removes at least one of these vertices at a cost of 1. Thus Step 5 moves each pebble by at most $|S|(6\mathrm{OPT} + 1)$ steps. But $|S|$ is at most $m/(2k)$, because we assign at least $2k$ pebbles to each vertex in S and the total number of pebbles is m. Therefore the total cost is $(3k+8\mathrm{OPT}+1)+(2k+6\mathrm{OPT}+1)+(6\mathrm{OPT}+1)m/(2k) = 5k+14\mathrm{OPT}+2+(6\mathrm{OPT}+1)m/(2k)$, proving the lemma. □

To prove Theorem 2.1, we first check whether OPT is zero, i.e., whether the pebbles already induce a connected graph. Otherwise, we apply Lemma 2.2 with $k = \sqrt{m/x}$ where x is a guessed value of OPT. With $k = \sqrt{m/\mathrm{OPT}}$ (or with the best guess of x), we obtain an approximation ratio of $O(\sqrt{m/\mathrm{OPT}})$.

Finally, it is worth mentioning that ConMax can be solved exactly on special classes of graphs. The following solution for the case of trees interestingly uses bipartite matching as its main tool, not the usual dynamic programming on trees.

THEOREM 2.2. *Given a tree T and a configuration of k pebbles on T, ConMax can be solved in polynomial time.*

2.2 $O(\sqrt{m/\mathrm{OPT}})$-Approximation for PathMax. Our techniques can be extended to obtain the same approximation factor for connectivity between just two fixed vertices s and t. The previous approach does not apply directly to this problem because not all of the pebbles need to be involved in the solution; we can select an arbitrary subset of pebbles to use for our path.

THEOREM 2.3. *There is an $O(\sqrt{m/\mathrm{OPT}})$-approximation algorithm (and thus also an $O(\sqrt{m})$-approximation algorithm) for PathMax.*

It is easy to prove that PathMax is NP-hard via a reduction from Hamiltonian Path.[2]

2.3 εm-Approximation for DirConMax. Next we consider the directed version of ConMax, DirConMax, where we obtain nearly tight results: an εm-approximation and $m^{1-\varepsilon}$ inapproximability assuming P $\neq$ NP. Our approximability result is based on another extension of our techniques from ConMax.

THEOREM 2.4. *For any integer constant k, there is an $n^{k+O(1)}$ algorithm that, given an instance of DirConMax with m pebbles and root r, finds a motion with maximum movement at most $\mathrm{OPT} + 2\lfloor m/k \rfloor$. (In particular, this algorithm is a $2\lfloor m/k \rfloor$-approximation.)*

2.4 $\Omega(n^{1-\varepsilon})$ Inapproximability for DirConMax. Next we prove that the εn-approximation algorithm is essentially tight, assuming only that P $\neq$ NP without the use of PCP-type arguments:

THEOREM 2.5. *For every constant ε, $0 < \varepsilon < 1$, it is NP-hard to approximate DirConMax within an $n^{1-\varepsilon}$ factor.*

Proof: We prove that, if DirConMax can be approximated within $n^{1-\varepsilon}$, then set cover can be solved in polynomial time. Let $S = (E, C, k)$ be an instance of set cover, where $E = \{e_1, e_2, \ldots, e_m\}$ is the universe of elements, $C = \{c_1, c_2, \ldots, c_s\}$ is the set of subsets of E, and k is an integer. Without loss of generality, assume that $m \geq s$; otherwise, we can place $s - m$ dummy elements $e_{m+1}, \ldots, e_s$ in every subset in C.

We convert the set-cover instance S into a graph G as follows; refer to Figure 3. Let L be $m^{2/\varepsilon}$. We start with a root vertex r in G and then, for every subset $c_i \in C$, we add a vertex v_i to G. Then, for each v_i, we add a directed path P_i of length $L+1$ from v_i to r. Label the vertices along path P_i from source to destination as $v_i, u_{i,1}, u_{i,2}, \ldots, u_{i,L}, r$. For each j, $1 \leq j \leq L$, we add a vertex s_j and we add an edge from s_j to $u_{i,j}$ for each i, $1 \leq i \leq s$. We also add a vertex s_0 and we add an edge from s_0 to v_i for each i, $1 \leq i \leq s$. For every element $e_i \in E$, we add two vertices w_i and w'_i to G, and we add an edge from w'_i to w_i. Finally, we add an edge from w_i to v_j precisely when $e_i \in c_j$. This graph has $n = 1 + (s + 1)(L + 1) + 2m = O(m^{2/\varepsilon+1})$ vertices. In our instance of DirConMax, we place one pebble on each w'_i

[2]Duplicate the vertices of a graph G into $n = |V(G)|$ levels, for a total of n^2 vertices, and adding edges between every pair of adjacent levels corresponding to edges of G, for a total of $2|E(G)|(n-1)$ edges. For each vertex v of G, add a path of length n from each copy of v to a common new vertex $\hat{v}$, at which we place a single pebble. Finally, connect the source s to every vertex in the first level, and connect the sink t to every vertex in the last level; and attach to each of s and t a path of length n, the end of which has a single pebble. G has a Hamiltonian path if and only if we can move each pebble to an instance of its corresponding vertex with a maximum movement of n, and construct a path from s to t.

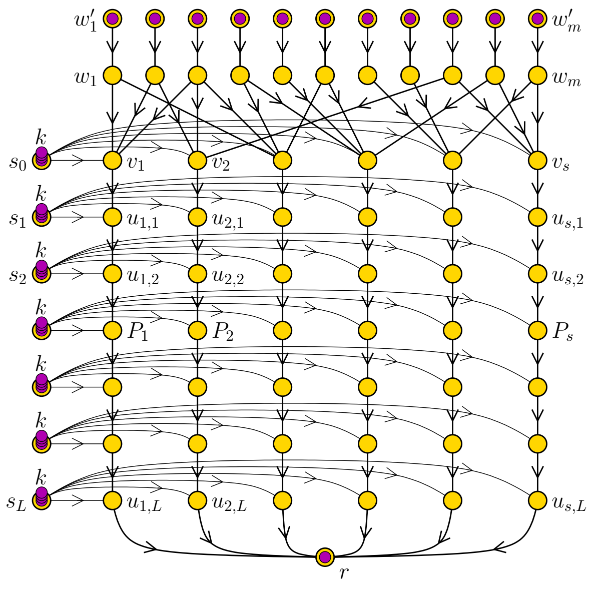

Figure 3: Reduction in Theorem 2.5.

and one pebble on r. We also place k pebbles on each s_i, $0 \le i \le L$.

If S has a set cover $C' = \{c_{p_1}, c_{p_2}, \ldots, c_{p_{k'}}\}$ of size $k' \le k$, then we can connect the pebbles in G using a maximum movement of 1. Namely, we move k' pebbles from each s_i, $1 \le i \le L$, to $u_{p_1,i}, u_{p_2,i}, \ldots, u_{p_{k'},i}$. Then we move the pebble from w'_j to w_j for each j, $1 \le j \le m$, and we move k' pebbles from s_0 to $v_{p_1}, v_{p_2}, \ldots, v_{p_{k'}}$.

Now we prove that, if S has no set cover of size at most k, then the maximum movement of any solution to this instance of DirConMax is at least $m^{2/\varepsilon-1}$. Consider a solution with maximum movement less than $m^{2/\varepsilon-1}$ and let L' be $m^{2/\varepsilon-1}$. Because the pebble at r can never move, the final positions of the pebbles must form a directed tree T rooted at r. We call a path P_i *semicompleted* if $u_{i,L'}$ is in T. Let $P' = \{P_{i_1}, P_{i_2}, \ldots, P_{i_{k'}}\}$ be the set of semicompleted P_i's. We assert that the set $C'' = \{c_{i_1}, c_{i_2}, \ldots, c_{i_{k'}}\}$ is a set cover of size k' for the instance S. Let f_i be the final position of the pebble starting on w'_i. This vertex f_i cannot be $u_{i',j}$ for any i' and j with $1 \le i' \le s$ and $L' \le j \le L$. So the directed path from f_i to the root r must visit some vertex $u_{i_j,L'}$ along a semicompleted path P_{i_j} for some j, $1 \le j \le k'$. Thus, $e_i \in c_{i_j}$, and this property holds for all i, $1 \le i \le m$, so C'' is indeed a set cover of size k' for S. Now we prove that $k' \le k$, contradicting that S has no such set cover. For each j, $1 \le j \le k'$, we need at least $L - L'$ pebbles to occupy the vertices $u_{i_j,j'}$, $L' < j' \le L$. The total number of pebbles that can have a final position of $u_{i,j}$, where $1 \le i \le s$ and $L' < j \le L$, is less than kL. Thus $k'(L - L') < kL$, i.e., $1 - 1/m < k/k'$. Because $k' \le s \le m$, $1 - 1/k' \le 1 - 1/m$, and therefore $1 - 1/k' < k/k'$, i.e., $k' \le k$.

On the other hand, if there is a set cover of size at most k, then there is a solution with maximum movement 1. Thus any solution to DirConMax with maximum movement at least L' has an approximation ratio at least $L' = m^{2/\varepsilon-1}$, which is asymptotically larger than $m^{2/\varepsilon+1-2-\varepsilon} = (m^{2/\varepsilon+1})^{1-\varepsilon} = \Theta(n^{1-\varepsilon})$. Therefore, we can decide whether there is a set cover of size at most k by testing whether an $O(n^{1-\varepsilon})$-approximation algorithm for ConSum produces a solution of maximum movement less than $m^{2/\varepsilon-1}$. □

3 Minimum Maximum Movement to Independence

It is NP-hard to decide whether IndMax even has a valid solution: an instance has a solution precisely if the graph has an independent set of size m, the number of pebbles. Thus, to obtain any approximability result, we must restrict our attention to special family of graphs.

In this section we focus on a particularly useful case of the *Euclidean plane*. This scenario has applications in the fields of map labeling and sensor networks, as described in the introduction. Recall that in this case we define $n = m$. We use the notation d for a more general notion of Euclidean distance: for a point p and a finite set Q of points, $d(p, Q)$ denotes the minimum distance $\min_{q \in Q} d(p, q)$.

THEOREM 3.1. *There is a polynomial-time algorithm solving IndMax in the Euclidean plane using maximum movement at most the optimal plus* $1 + \frac{1}{\sqrt{3}}$.

The heart of our approximation algorithm is the triangular lattice, illustrated in Figure 4, in which every two distinct vertices have distance at least 1. Thus, these vertices induce an independent set of the plane. The vertex set is given by

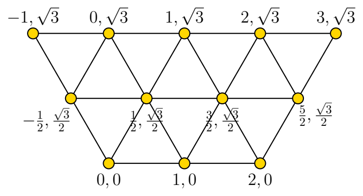
$-1, \sqrt{3}$
$0, \sqrt{3}$
$1, \sqrt{3}$
$2, \sqrt{3}$
$3, \sqrt{3}$
$-\frac{1}{2}, \frac{\sqrt{3}}{2}$
$\frac{1}{2}, \frac{\sqrt{3}}{2}$
$\frac{3}{2}, \frac{\sqrt{3}}{2}$
$\frac{5}{2}, \frac{\sqrt{3}}{2}$
$0, 0$
$1, 0$
$2, 0$

Figure 4: Decomposition of the plane into equilateral triangles.

$$A = \left\{(i, j\sqrt{3}), (i + \tfrac{1}{2}, j\sqrt{3} + \tfrac{\sqrt{3}}{2}) \;\middle|\; i, j \in \mathbb{Z}\right\}.$$

Let C denote the decomposition of the plane into equilateral triangles with side length 1 induced by this lattice.

For a finite set R of points, we define two additional concepts. Let $\mathrm{Neighbor}(R)$ denote the set of points in A whose distance to R is at most $1 + \frac{1}{\sqrt{3}}$. Let $\mathrm{Circle}(R)$ denote the union of disks centered at points in R with radius $\frac{1}{2}$. In particular, if every two distinct points in R have distance at least 1, then $\mathrm{Circle}(R)$ has area $|R| \cdot \frac{\pi}{4}$.

LEMMA 3.1. *The optimal solution has maximum movement at most* $2n - 2$.

Proof: Suppose for contradiction that there is a pebble x with initial position p and with target position q in the optimal solution, yet $d(p, q) > 2n - 2$. We define n points $r_0, r_1, \ldots, r_{n-1}$ on the line segment from p to q according to $d(p, r_i) = 2i$. The distance between any two of these points r_i and r_j, $i \neq j$, is at least 2, so any point can have distance less than 1 with at most one of these points $r_0, r_1, \ldots, r_{n-1}$. By the Pigeonhole Principle, there is at least one point r_i that is not within distance 1 of the target position of any pebble other than x. Thus we can change the target position of pebble x to r_i and obtain a valid solution in which the movement of x is at most $2n - 2$. By induction, we can reduce the movement of every pebble to at most $2n - 2$, giving us a solution with maximum movement at most $2n - 2$, contradicting optimality of the original solution.[3] □

LEMMA 3.2. *The number of points in A within distance at most $2n - 2$ from an arbitrary point p in the plane is at most a polynomial function of n.*

Proof: Consider the square S centered at p and with side length $4n - 4$. All points of A within distance $2n - 2$ from p are in this square. Consider a decomposition of S into a grid of subsquares of side length $\frac{1}{2}$. Because the distance between each pair of points in such a subsquare is at most $1/\sqrt{2} < 1$, at most one point of A can be in each subsquare. Thus the number of subsquares is an upper bound on the number of points of A in S, which is an upper bound on the number of points of A within distance $2n - 2$ from p. The number of subsquares is $(4n - 4)^2/(\frac{1}{2})^2 = O(n^2)$. □

LEMMA 3.3. *Let C_A and C_B be two disks of radius $1/\sqrt{3}$ centered at points A and B, respectively. Let $d = d(A, B)$ be the distance between A and B. The area of intersection of the two disks C_A and C_B is $\frac{2}{3}\arccos(d\sqrt{3}/2) - d\sqrt{\frac{1}{3} - \frac{1}{4}d^2}$.*

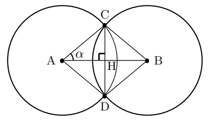

Figure 5: Intersection of the circles C_A and C_B.

Proof: In Figure 5, we have $\overline{AC} = \overline{BC} = \overline{AD} = \overline{BD} = 1/\sqrt{3}$, $\overline{AB} = d$, $\angle BAC = \alpha$. Thus, $\cos\alpha = \overline{AH}/\overline{AC} = (d/2)/(1/\sqrt{3}) = d\sqrt{3}/2$, so $\alpha = \arccos(d\sqrt{3}/2)$. Hence, the area of the pie wedge of C_A given by the angle $\angle DAC$ is $\frac{2\alpha}{2\pi} \cdot \frac{\pi}{3} = \frac{1}{3}\alpha = \frac{1}{3}\arccos(d\sqrt{3}/2)$. By symmetry, the area of the pie wedge of C_B given by the angle $\angle CBD$ is the same. These pie wedges overlap at precisely the intersection of C_A and C_B. Their union is the quadrangle $ABCD$. Thus, the desired area of intersection is the sum of the areas of the pie wedges, $\frac{2}{3}\arccos(d\sqrt{3}/2)$, minus the area of the quadrangle $ABCD$. Now $\overline{CH} = \sqrt{\overline{AC}^2 - (\overline{AB}/2)^2} = \sqrt{\frac{1}{3} - \frac{1}{4}d^2}$, so the area of the quadrangle $ABCD$ is $\frac{1}{2}\overline{AB} \cdot \overline{CD} = \overline{AB} \cdot \overline{CH} = d\sqrt{\frac{1}{3} - \frac{1}{4}d^2}$. Therefore, the desired area of intersection of C_A and C_B is $\frac{2}{3}\arccos(d\sqrt{3}/2) - d\sqrt{\frac{1}{3} - \frac{1}{4}d^2}$ as desired. □

Now we are ready to prove the main theorem.

Proof of Theorem 3.1: For $i \in \{1, 2, \ldots, n\}$, let p_i and q_i be the initial and target position of pebble i in the optimal solution OPT. Because OPT is a solution to IndMax, we have $d(q_i, q_j) \geq 1$ for all distinct $i, j \in \{1, 2, \ldots, n\}$. Furthermore, the optimal solution minimizes $\text{OPT} = \max_{1 \leq i \leq n} d(p_i, q_i)$ subject to this constraint. First we prove that there is a polynomial-time algorithm to move every pebble to a point of A such that no two pebbles move to the same point, and subject to minimizing the maximum movement M. Then we prove that we can move the pebbles from their target positions in OPT to points of A so that no two pebbles move to the same point and each pebble moves at most $1 + \frac{1}{\sqrt{3}}$. Thus, our approximate solution of maximum movement M satisfies $M \leq \text{OPT} + 1 + \frac{1}{\sqrt{3}}$.

The algorithm constructs a complete weighted bipartite graph $H = (X, Y, E)$. For $i \in \{1, 2, \ldots, n\}$, we place a vertex x_i in X representing pebble i. By Lemma 3.1, $\text{OPT} \leq 2n - 2$. By the second part of the proof, the optimal movement M to points of A satisfies $M \leq \text{OPT} + 1 + \frac{1}{\sqrt{3}} \leq 2n - 2 + 1 + \frac{1}{\sqrt{3}}$. Thus, in M, no pebble moves more than $2n$. For each point p of A within distance $2n - 2$ from the set $\{p_1, p_2, \ldots, p_n\}$ of initial positions, we place a vertex in Y_p. By Lemma 3.2, the number of these points is polynomial in n, so the graph H has polynomial size. For each $x \in X$ and $y \in Y$, we set the weight $w(x, y) = d(x, y)$. The algorithm finds a perfect matching in H of minimum maximum weight. For each edge (x_i, y_p) in the matching, we move the ith pebble to point p of A. In this way, we move the pebbles to points of A such that no two pebbles move to the same point using the minimum maximum movement.

Now we reach the heart of the proof: we prove that we can transform OPT by moving each target position by at most $1 + \frac{1}{\sqrt{3}}$ such that every new target position is a point of A and no two target positions are the same. We prove that there is a perfect matching from the set $Q = \{q_1, q_2, \ldots, q_n\}$ of target positions in OPT to the points of A such that the distance between matched points is at most $1 + \frac{1}{\sqrt{3}}$. By Hall's Theorem, it suffices to show that, for each subset $R \subseteq Q$, $|R| \leq |\text{Neighbor}(R)|$.

Consider a subset $R = \{r_1, r_2, \ldots, r_m\} \subseteq Q$, and the region $W = \text{Circle}(R)$. Because the distance between every two points in R is at least 1, $\text{Circle}(R)$ has area $|R| \cdot \frac{\pi}{4}$. Consider the set $\text{Neighbor}(R) \subseteq A$, and the

[3] This argument can be improved to obtain a bound of $O(\sqrt{n})$ on the maximum motion, but it does not affect our main result.

region $V = \operatorname{Circle}(\operatorname{Neighbor}(R))$. Again V has area $|\operatorname{Neighbor}(R)| \cdot \frac{\pi}{4}$. We prove that the area of $\operatorname{Circle}(R)$ is at most the area of $\operatorname{Circle}(\operatorname{Neighbor}(R))$, which implies $|R| \leq |\operatorname{Neighbor}(R)|$, completing the proof.

Consider a disk of radius $\frac{1}{\sqrt{3}}$ centered at each point of R. Define the region S to consist of the equilateral triangles of the decomposition C that intersect at least one of these disks. The vertices of the triangles in S are the points of $\operatorname{Neighbor}(R)$, because these vertices are the points of A within distance $1 + \frac{1}{\sqrt{3}}$ from the points of R.

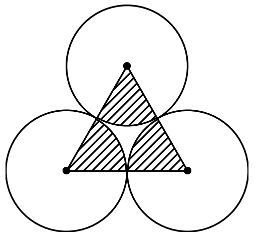
Figure 6:

Next we prove that $\operatorname{Area}(\operatorname{Circle}(\operatorname{Neighbor}(R))) \geq \operatorname{Area}(S) \cdot \frac{\pi}{2\sqrt{3}}$. For each triangle T in S, there are only three circles of $\operatorname{Circle}(\operatorname{Neighbor}(R))$ that intersect with it, those whose centers are placed on the vertices of T;. see Figure 6. These circles have area $\frac{\pi}{8}$ in common with T. Therefore, the ratio of this common area to the area of T is $\frac{\pi}{8}/\frac{\sqrt{3}}{4} = \frac{\pi}{2\sqrt{3}}$. Because this ratio is the same for every triangle in S, so is the ratio $\operatorname{Area}(\operatorname{Circle}(\operatorname{Neighbor}(R)) \cap S)/\operatorname{Area}(S) = \frac{\pi}{2\sqrt{3}}$. Therefore, $\operatorname{Area}(\operatorname{Circle}(\operatorname{Neighbor}(R)))/\operatorname{Area}(S) \geq \frac{\pi}{2\sqrt{3}}$ or $\operatorname{Area}(\operatorname{Circle}(\operatorname{Neighbor}(R))) \geq \operatorname{Area}(S) \cdot \frac{\pi}{2\sqrt{3}}$

Next we prove that $\operatorname{Area}(S) \cdot \frac{\pi}{2\sqrt{3}} \geq \operatorname{Area}(\operatorname{Circle}(R))$, which would prove the theorem. For each point r in R, we assign a region $\operatorname{Region}(r)$ contained in S of area at least $\frac{\sqrt{3}}{2}$ such that every two regions $\operatorname{Region}(r)$ and $\operatorname{Region}(s)$, $r \neq s$, are disjoint. Because these regions pack a subset of S, we obtain $\operatorname{Area}(S) \geq |R| \cdot \frac{\sqrt{3}}{2}$. Therefore, $\operatorname{Area}(S) \cdot \frac{\pi}{2\sqrt{3}} \geq |R| \cdot \frac{\sqrt{3}}{2} \cdot \frac{\pi}{2\sqrt{3}} = |R| \cdot \frac{\pi}{4} = \operatorname{Area}(\operatorname{Circle}(R))$.

It remains to assign to each point r of R a region $\operatorname{Region}(r)$. We do so according to the following algorithm:

1. For each point r in R, initially set $\operatorname{Region}(r)$ to the disk $\operatorname{Region}_0(r)$ of radius $\frac{1}{\sqrt{3}}$ centered at r. (Thus, $\operatorname{Region}_0(r)$ has area $\frac{\pi}{3}$.)

2. For two arbitrary points r and s in R, if $\operatorname{Region}_0(r)$ intersects $\operatorname{Region}_0(s)$, omit half of their intersection from $\operatorname{Region}(r)$ and omit the other half from $\operatorname{Region}(s)$ according to the perpendicular bisector of r and s, as shown in the Figure 7.

Obviously, the resulting regions are pairwise disjoint and each region is contained in S.

We prove that the sum of the areas omitted from $\operatorname{Region}(r)$ is at most $\frac{\pi}{3} - \frac{\sqrt{3}}{2}$, for each point r in R; thus, $\operatorname{Region}(r)$ keeps an area of at least $\frac{\sqrt{3}}{2}$ as desired. Let $e(r,s) = \frac{1}{2}\operatorname{Area}(\operatorname{Region}_0(r) \cap \operatorname{Region}_0(s))$ be the area of $\operatorname{Region}(r)$ omitted because of $\operatorname{Region}(s)$. (This definition

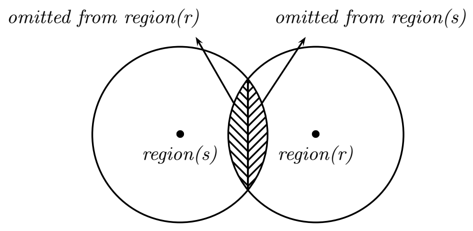

Figure 7: The omitted region from the disks.

actually overestimates the omitted area if multiple overlapping regions are omitted.) Thus, our goal is to prove that $\sum_{s \neq r \in R} e(r,s) \leq \frac{\pi}{3} - \frac{\sqrt{3}}{2}$. For a fixed point r in R, consider the points s of R that have a nonzero value $e(r,s)$. Sort these points according to the angle of the ray from r to s with respect to the x axis, resulting in a sequence $s_1, s_2, \ldots, s_l$.

We prove that $e(r,s_i) + e(r,s_{i+1}) \leq \left(\frac{1}{3} - \frac{\sqrt{3}}{2\pi}\right) \angle s_i r s_{i+1}$. Let $a = d(r,s_i)$, $b = d(r,s_{i+1})$, and $c = d(s_i,s_{i+1})$. Because r, s_i, and s_{i+1} are points of R, we have $a,b,c \geq 1$, and in particular, $c^2 \geq 1$. Because $e(r,s_i), e(r,s_{i+1}) \neq 0$, we have $a,b \leq \frac{2}{\sqrt{3}}$. By the Law of Cosines, we have $c^2 = a^2 + b^2 - 2ab\cos(\angle s_i r s_{i+1})$, and thus $\cos(\angle s_i r s_{i+1}) \leq \frac{a^2+b^2-1}{2ab}$, so $\angle s_i r s_{i+1} \geq \arccos\left(\frac{a^2+b^2-1}{2ab}\right)$. By Lemma 3.3, we have

$$\begin{aligned} e(r,s_i) &= \tfrac{1}{2}\left(\tfrac{2}{3}\arccos(a\sqrt{3}/2) - a\sqrt{\tfrac{1}{3} - \tfrac{1}{4}a^2}\right) \\ \text{and } e(r,s_{i+1}) &= \tfrac{1}{2}\left(\tfrac{2}{3}\arccos(b\sqrt{3}/2) - b\sqrt{\tfrac{1}{3} - \tfrac{1}{4}b^2}\right). \end{aligned}$$

One can check algebraically that

$$\tfrac{1}{2}\left(\tfrac{2}{3}\arccos(a\sqrt{3}/2) - a\sqrt{\tfrac{1}{3} - \tfrac{1}{4}a^2} + \tfrac{2}{3}\arccos(b\sqrt{3}/2)\right.$$
$$\left. - b\sqrt{\tfrac{1}{3} - \tfrac{1}{4}b^2}\right) \leq \left(\tfrac{1}{3} - \tfrac{\sqrt{3}}{2\pi}\right)\arccos\left(\tfrac{a^2+b^2-1}{2ab}\right).$$

Thus, $e(r,s_i) + e(r,s_{i+1}) \leq \left(\frac{1}{3} - \frac{\sqrt{3}}{2\pi}\right) \angle s_i r s_{i+1}$.

By summing the previous inequality, we obtain

$$\begin{aligned} e(r,s_1) + e(r,s_2) &\leq \left(\tfrac{1}{3} - \tfrac{\sqrt{3}}{2\pi}\right) \angle s_1 r s_2 \\ e(r,s_2) + e(r,s_3) &\leq \left(\tfrac{1}{3} - \tfrac{\sqrt{3}}{2\pi}\right) \angle s_2 r s_3 \\ \vdots \quad & \quad \vdots \\ e(r,s_l) + e(r,s_1) &\leq \left(\tfrac{1}{3} - \tfrac{\sqrt{3}}{2\pi}\right) \angle s_l r s_1 \\ \Rightarrow \quad 2\sum_{i=1}^{l} e(r,s_i) &\leq \left(\tfrac{1}{3} - \tfrac{\sqrt{3}}{2\pi}\right) 2\pi \end{aligned}$$

Therefore, $\sum_{i=1}^{l} e(r,s_i) \leq \left(\frac{\Pi}{3} - \frac{\sqrt{3}}{2}\right)$ as desired.

In summary, for each $R \subseteq Q$, we have $|R| \leq |\operatorname{Neighbor}(R)|$, so there is a perfect matching from Q to $\operatorname{Neighbor}(Q)$; thus, we can move each pebble to a unique point in A such that the maximum movement is at most $1 + \frac{1}{\sqrt{3}}$. $\square$

4 Minimum Maximum Movement to Perfect Matchability

In contrast to the difficult problems of ConMax and IndMax, we show that minimizing movement does not make perfect matching much harder: there is a polynomial-time algorithm for MatchMax.

LEMMA 4.1. *If two pebbles p and q are within distance 1 in the target configuration, then $|\pi(p)|+|\pi(q)| \geq d_G(p,q)-1$, and thus $\max\{|\pi(p)|,|\pi(q)|\} \geq \left\lceil \frac{d_G(p,q)-1}{2} \right\rceil$.*

Proof: Each step in the motion path of p or q may decrease $d_G(p,q)$ by at most 1. Therefore the sum of the movements of p and q must be at least their original distance $d_G(p,q)$ minus their target distance of 0 or 1. □

THEOREM 4.1. *There is a polynomial-time algorithm solving MatchMax.*

Proof: We assume that the number of pebbles in each connected component of G is even; otherwise, no solution exists. We can also consider each connected component separately, so we assume without loss of generality that G is connected. Let $p_1, p_2, \ldots, p_{2n}$ denote the pebbles.

Define the weighted complete undirected graph H as follows. For each pebble p_i we make a vertex v_i in graph H. For each edge $e = \{v_i, v_j\}$ in H, we set its weight $w(e)$ to $\lceil \frac{d_G(p_i,p_j)-1}{2} \rceil$. Define the *maximum weight* $w(M) = \max_{e \in M} w(e)$ of a perfect matching M of H to be the maximum weight of its edges.

Our algorithm computes a perfect matching M in H of minimum maximum weight $w(M)$ (in polynomial time), and converts this matching into a motion as follows. For each edge $\{v_i, v_j\}$ in the matching M, we move p_i by $\left\lceil \frac{d_G(p_i,p_j)-1}{2} \right\rceil$ steps toward p_j along a shortest path from p_i to p_j in G, and we move p_j by $\left\lceil \frac{d_G(p_i,p_j)-1}{2} \right\rceil$ steps toward p_i along the same shortest path. (Note that $\left\lceil \frac{d_G(p_i,p_j)-1}{2} \right\rceil \geq 0$.) Thus, after the motion, p_i and p_j are at distance at most 1 in G. The maximum movement in this motion is the maximum weight of such a matched edge $\{v_i, v_j\}$, which is precisely $w(M)$.

Now we argue that no solution to MatchMax has maximum movement less than $w(M)$. By definition of MatchMax, any solution induces a perfect matching M' in the graph H (i.e., on the pebbles) with the property that, in the target configuration, every two matched pebbles have distance at most 1 in G. For every edge $e = \{v_i, v_j\}$ in this matching M', by Lemma 4.1, we have that $\max\{|\pi(p_i)|,|\pi(p_j)|\} \geq \left\lceil \frac{d_G(p,q)-1}{2} \right\rceil = w(e)$. Therefore, the maximum movement in the solution must be at least $\max_{e \in M'} w(e) = w(M')$. But M was chosen to minimize this lower bound $w(M)$, so every solution must have maximum movement at least $w(M)$, proving optimality of our strategy of maximum movement $w(M)$. □

5 Minimum Total Movement

In this section, we consider the variations of the movement problems in which the goal is to minimize total movement instead of maximum movement. For both ConSum and MatchSum, we obtain tight results.

5.1 Connectivity: $\Omega(n^{1-\varepsilon})$ Inapproximability.

THEOREM 5.1. *For every constant ε, $0 < \varepsilon < 1$, it is NP-hard to approximate ConSum within an $n^{1-\varepsilon}$ factor.*

5.2 Connectivity: $\tilde{O}(\min\{n, m\})$ Approximation.

Note that $O(nm)$-approximation is trivial for ConSum, where n is the number of vertices and m is the number of pebbles. If the pebbles already induce a connected graph, then there is nothing to do. Otherwise, the optimal solution has total motion at least 1, and we can move all m pebbles to any particular vertex using at most $m(n-1)$ total movement.

THEOREM 5.2. *There is an $O(\min\{n \log n, m\})$-approximation algorithm for ConSum.*

5.3 Path Connectivity: $O(n)$ Approximation.

THEOREM 5.3. *There is an $O(n)$-approximation algorithm for PathSum.*

It is also easy to prove that PathSum is NP-hard via a reduction from Hamiltonian Path.[4]

5.4 Perfect Matchability.

Like MatchMax, the MatchSum variation can also be solved in polynomial time:

THEOREM 5.4. *There is a polynomial-time algorithm solving MatchSum.*

6 Conclusion

This paper makes a systematic study of movement problems which, despite connections to several practical problems, have not been studied before in theoretical computer science. Among the problems we consider, we highlight one open problem of primary concern: the approximability of ConMax and PathMax. For directed graphs, we proved essentially tight approximability and inapproximability results for DirConMax, of roughly $\tilde{\Theta}(n)$. However, for undirected graphs, we obtained an $O(\sqrt{m/\mathrm{OPT}}) = O(\sqrt{m})$ approximation for ConMax and PathMax. Can these approximations be improved, or are there matching inapproximability

[4] Apply the same construction as Footnote 2. G has a Hamiltonian path if and only if PathSum has a solution of total movement $n(n+2)$.

results? Figure 1 shows a difficult example which might be extended to prove inapproximability for both problems.

It would also be interesting to consider more problems in the practical scenario of the Euclidean plane, either for improved approximation ratios compared to general graphs or for problems that cannot be solved on general graphs. In particular, in the latter category, we obtained an additive $O(1)$-approximation for IndMax, but even the existence of a multiplicative $O(1)$-approximation for IndMax remains open. (Specifically, when $\mathrm{OPT} = o(1)$, we lack good approximations.)

Several other movement problems fit into our general framework. One variation changes the notion of the graph induced by a set of pebbles to include edges between pebbles within a given fixed distance d. This variation models the situation in which pebbles can communicate within a fixed distance d, but they still move one unit at a time (so we cannot simply take the dth power of the graph). Another variation, the *facility-location movement problem*, introduces two types of pebbles—clients and servers—and the target property is that every client is collocated with some server. If only the clients are permitted to move, this problem is trivial: each client moves to its nearest server. If both clients and servers can move, this solution is a 2-approximation to the maximum-movement version, but can we do better? What about the other versions of the problem, e.g., total movement?

References

[ABF$^+$02] Esther M. Arkin, Michael A. Bender, Sándor P. Fekete, Joseph S. B. Mitchell, and Martin Skutella. The freeze-tag problem: how to wake up a swarm of robots. In *Proceedings of the 13th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 568–577, San Francisco, California, 2002.

[ABG03] Esther M. Arkin, Michael A. Bender, and Dongdong Ge. Improved approximation algorithms for the freeze-tag problem. In *Proceedings of the 15th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 295–303, San Diego, California, USA, 2003.

[BDHR05] Jonathan L. Bredin, Erik D. Demaine, Mohammad-Taghi Hajiaghayi, and Daniela Rus. Deploying sensor networks with guaranteed capacity and fault tolerance. In *Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC 2005)*, pages 309–319, Urbana-Champaign, Illinois, May 25–28 2005.

[CCC$^+$99] Moses Charikar, Chandra Chekuri, To-Yat Cheung, Zuo Dai, Ashish Goel, Sudipto Guha, and Ming Li. Approximation algorithms for directed steiner problems. *Journal of Algorithms*, 33(1):73–91, 1999.

[CFKR98] Johanne Cohen, Pierre Fraigniaud, Jean-Claude König, and André Raspaud. Optimized broadcasting and multicasting protocols in cut-through routed networks. *IEEE Transactions on Parallel and Distributed Systems*, 9(8):788–802, 1998.

[CHP$^+$04a] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme. Autonomous deployment of a sensor network using an unmanned aerial vehicle. In *Proceedings of the 2004 International Conference on Robotics and Automation*, New Orleans, USA, 2004.

[CHP$^+$04b] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme. Deployment and connectivity repair of a sensor net with a flying robot. In *Proceedings of the 9th International Symposium on Experimental Robotics*, Singapore, 2004.

[Coh98] Johanne Cohen. Broadcasting, multicasting and gossiping in trees under the all-port line model. In *Proceedings of the 10th Annual ACM Symposium on Parallel Algorithms and Architectures*, pages 164–171, Puerto Vallarta, Mexico, 1998.

[DMM$^+$97] Srinivas Doddi, Madhav V. Marathe, Andy Mirzaian, Bernard M. E. Moret, and Binhai Zhu. Map labeling and its generalizations. In *Proceedings of the 8th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 148–157, New Orleans, LA, 1997.

[GHMM02] M. Ghodsi, M.T. Hajiaghayi, M. Mahdian, and V. S. Mirrokni. Length-constrained path-matchings in graphs. *Networks*, 39(4):210–215, 2002.

[GK99] Sudipto Guha and Samir Khuller. Improved methods for approximating node weighted steiner trees and connected dominating sets. *Information and Computation*, 150(1):57–74, 1999.

[HAB$^+$03] T.-R. Hsiang, E. M. Arkin, M. A. Bender, S. P. Fekete, and Joseph S. B. Mitchell. Algorithms for rapidly dispersing robot swarms in unknown environments. In J.-D. Boissonnat, J. Burdick, K. Goldberg, and S. Hutchinson, editors, *Algorithmic Foundations of Robotics V*, volume 7 of *Springer Tracts in Advanced Robotics*, pages 77–94. Springer-Verlag, 2003.

[JBQZ04] Minghui Jiang, Sergey Bereg, Zhongping Qin, and Binhai Zhu. New bounds on map labeling with circular labels. In *Proceedings of the 15th International Symposium on Algorithms and Computation*, volume 3341 of *Lecture Notes in Computer Science*, pages 606–617, Hong Kong, China, December 2004.

[JQQ$^+$03] Minghui Jiang, Jianbo Qian, Zhongping Qin, Binhai Zhu, and Robert Cimikowski. A simple factor-3 approximation for labeling points with circles. *Information Processing Letters*, 87(2):101–105, 2003.

[KR95] Philip N. Klein and R. Ravi. A nearly best-possible approximation algorithm for node-weighted steiner trees. *Journal of Algorithms*, 19(1):104–115, 1995.

[LaV06] S. M. LaValle. *Planning Algorithms*. Cambridge University Press, 2006. http://msl.cs.uiuc.edu/planning/.

[RW95] John H. Reif and Hongyan Wang. Social potential fields: a distributed behavioral control for autonomous robots. In *Proceedings of the Workshop on Algorithmic Foundations of Robotics*, pages 331–345, 1995.

[SABM04] Marcelo O. Sztainberg, Esther M. Arkin, Michael A. Bender, and Joseph S. B. Mitchell. Theoretical and experimental analysis of heuristics for the “freeze-tag” robot awakening problem. *IEEE Transactions on Robotics and Automation*, 20(4):691–701, 2004.

[SPS03] Alan C. Schultz, Lynne E. Parker, and Frank E. Schneider, editors. *Multi-Robot Systems: From Swarms to Intelligent Automata*. Springer, 2003. Proceedings from the 2003 International Workshop on Multi-Robot Systems.

[SW01] Tycho Strijk and Alexander Wolff. Labeling points with circles. *International Journal of Computational Geometry & Applications*, 11(2):181–195, 2001.

[Wes96] Douglas B. West. *Introduction to Graph Theory*. Prentice Hall Inc., Upper Saddle River, NJ, 1996.