Building a Communication Bridge with Mobile Hubs

Onur Tekdas, *Student Member, IEEE,*Yokesh Kumar, *Student Member, IEEE,* Volkan Isler, *Senior Member, IEEE,* Ravi Janardan, *Senior Member, IEEE.*

***Abstract*—Mobile robots can be used as mobile hubs to provide communication services on-demand. This capability is especially valuable in disaster response scenarios where there is no communication infrastructure. In such scenarios, mobile hubs can provide a communication infrastructure in a dynamic fashion.**

In this paper, we study the problem of building a communication bridge between a source s and a destination t with mobile robots. Given a set of robots $\mathcal{P}$ and their initial locations, our goal is to find a subset S of robots and their final locations such that the robots in S create a communication bridge between s and t in their final locations. We introduce a new optimization problem for building communication bridges. The objective is to minimize the number of hubs (i.e. $|S|$) while simultaneously minimizing the robots' motion. The two mobility measures studied in this paper are: (i) maximum travel distance and (ii) total travel distance of the robots. For a geometric version of the problem where the robots must move onto the line segment $[s, t]$, we present polynomial time algorithms which use the minimum number of hubs while remaining within a constant factor of a given motion measure.

***Note to Practitioners* – Mobile robotic hubs can provide connectivity service in applications such as disaster response where the underlying communication infrastructure is broken. In such applications, often a communication bridge between two sites (e.g. a command center and a specific site) must be established. In such scenarios, robots can autonomously deploy themselves and create a communication bridge.**

In this work, we study the efficient use of mobile robots to create a communication bridge between a source s and a destination t. This yields a challenging resource allocation problem in which mobility and communication constraints must be addressed simultaneously. Specifically we study the following problem: Given s and t, and the initial locations of the mobile hubs, find their final locations so as to minimize the number of hubs used in the bridge *and* (either maximum or total) distance traveled by the hubs. We present efficient, provably correct approximation algorithms for a special version of this optimization problem in which the hubs are required to move onto the line segment $[s, t]$. From a practical perspective, this special case is important because on the open plane it minimizes the overall number of hubs on the bridge. Thus, our algorithms can be used in scenarios such as robots operating in open spaces (land, water, or air). From a theoretical perspective, it provides an important first step toward the solution of the general problem where the hubs can be placed at arbitrary locations.

***Index Terms*—data mules, robotic routers, sensor networks**

Corresponding author is Onur Tekdas. The authors are with the Department of Computer Science and Engineering at the University of Minnesota, 4-192 Keller Hall, 200 Union St SE, Minneapolis, MN 55455, USA. Emails: {tekdas,kumaryo,isler,janardan}@cs.umn.edu. An earlier version of this paper appeared in the proceedings of the 5th International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS 2009) [1].

I. Introduction

The task of building a communication bridge connecting two locations arises frequently. For example, when fighting forest fires, a high capacity connection between the command center and a temporary base may be needed. When there is no underlying communication infrastructure (which is typically the case in emergency response scenarios), mobile entities with communication capabilities can be used to build a communication bridge. In particular, with recent advances in robotics, using mobile robots for this purpose is becoming feasible.

In this work, we address the problem of building a communication bridge in an efficient fashion. Imagine that we are given a source s and a destination t (the two locations that need to be connected), and initial locations of n robots (or *mobile hubs*). The goal is to pick a small subset of these robots and determine their final locations, so that when the robots arrive at their final locations, there is a path between s and t in the underlying communication graph. In this case, we say that a *communication bridge* between s and t has been established. Throughout the paper, we assume that two entities can communicate if and only if they are within a given communication radius r. See also Figure 1. Further, we study the problem in the open plane without obstacles. Even though we focus on simple communication and environment settings, we believe that the problem is important for two reasons. First, as we show in the paper, finding solutions with global performance guarantees is a hard problem even for this basic version. Our results provide a starting point for the study of more sophisticated versions of the problem. Second, there are practical scenarios such as robots operating in open spaces (which can be on the land, on the water surface, or in the air at a fixed height) in which our setup is applicable.

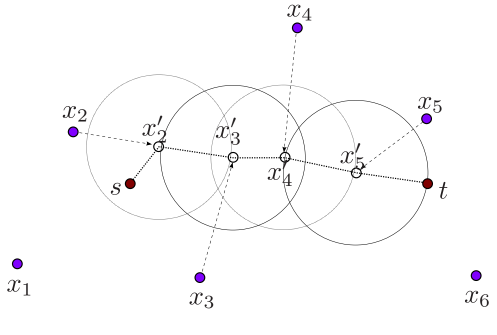

Fig. 1. Initial locations of the robots are x_i, $i = 1, \dots, 6$; s and t cannot communicate. By moving $x_2 \to x_2'$, $x_3 \to x_3'$, $x_4 \to x_4'$ and $x_5 \to x_5'$ a communication bridge with four hubs connecting s and t is established. The circles around the nodes illustrate the communication radius.

We focus on two measures of efficiency. The first one is the distance traveled by the robots to establish the communication bridge. Relevant objectives are minimizing the maximum or the total Euclidean (L_2) distance traveled. This measure is important when the robots have limited battery power. The maximum distance traveled also determines how quickly the bridge can be established. The second measure is the number of robots required to establish the communication bridge. This is an important parameter because if we use a small number of robots for the given task, then the remaining robots can be used for other tasks. In addition, a communication bridge with a small number of hubs is desirable in order to minimize the latency of the network.

Our results and techniques. We believe that the two metrics mentioned above are equally important. Therefore, we study the resulting bi-criteria optimization problem. Specifically, we present algorithms to minimize the number of hubs in the communication bridge for a given maximum (or total) travel distance in L_2 metric.

The general problem where the environment is represented with an arbitrary graph is NP-hard and, in fact, cannot be approximated efficiently [2]. Hence, in this paper, we focus on a geometric version where the underlying environment is the Euclidean plane, and the chosen robots are required to move onto the straight line segment $[s,t]$ to form a communication bridge. This special case is important from a practical standpoint because moving the robots onto this line segment yields the minimum number of hubs in the communication bridge, as compared to any other curve joining s and t. Another motivation for this model is low power, inexpensive infra-red communication which is becoming a popular choice for small robots: In an extreme case, if each robot is equipped with only two IR receivers/transmitters such that the pairs are placed 180 degrees apart, a straight line communication is necessary to establish a communication bridge between s and t. From a theoretical perspective, these problems turn out to be quite challenging. One of the major sources of difficulty is the lack of an "ordering property" in the optimal solution (We make the ordering property explicit in Section II-A.) As an example, consider the version where we are given a maximum travel distance for each robot. Suppose robot a (resp. robot b) can reach points inside the line segment $[l_a, r_a]$ (resp. $[l_b, r_b]$). It is possible to build instances where r_a is to the left of r_b but in the optimal solution robot a moves to the right of robot b (Figure 2).

For the maximum distance version (*MaxDist*), we overcome this hurdle by relaxing the distance requirement: if the optimal algorithm can build a communication bridge with at most k hubs by moving each robot at most distance d, we present an approximation algorithm which builds a communication bridge with k hubs by moving each robot at most distance $\sqrt{2}d$ (Section II-A). The key result enabling the algorithm is the presence of an ordering property for the relaxed version. For the sum version (*SumDist*), we show that there is an ordering property but for the L_1 metric (sum of absolute values of coordinate differences). We present an algorithm which exploits this ordering property and returns the optimal solution for the L_1 metric. This in turn yields a $\sqrt{2}$-approximation algorithm for the L_2 case (Section II-B).

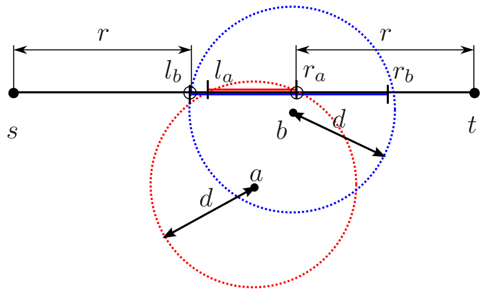

Fig. 2. Robot a (resp. robot b) can reach points inside the line segment $[l_a, r_a]$ (resp. segment $[l_b, r_b]$). Although a is to the left of b, a must move to the right of b, to r_a, and b must move to the left of a, to l_b, to establish a communication bridge. The final locations of robots are shown by unfilled circles.

The algorithms we present are dynamic programming solutions which exploit the ordering property. However, even with the ordering property, the dynamic programming solutions are not straight-forward. This is mainly because the final locations of the robots must be chosen from the continuous set of points on the line segment $[s,t]$: There are instances in which robots must be placed precisely to achieve the optimal solution, and slightly perturbing the optimal solution (to a finite set of points) breaks connectivity. Therefore, our algorithms avoid an apriori discretization of the line segment.

Finally, we present an interesting property regarding the number of hubs. Let $L > r$ be the distance between s and t. Clearly, at least $n^* = \lceil L/r \rceil - 1$ hubs are required to connect s and t. However, building a bridge with n^* hubs may not be feasible due to the motion constraint. We show that any minimal solution which satisfies the motion constraint uses at most $2n^*$ hubs (Section III). This means that by removing constraints on distance we gain a factor of at most 2 in the number of hubs.

A. Related Work

In the robotics literature, the interactions between robots and a static sensor network have been studied for network repair [3], connectivity [4] and data-collection problems [5]. From a systems perspective, researchers have proposed architectures that exploit controlled mobility [6], [7], [8], [9]. A recent review on the state of the art in exploiting sink mobility can be found in [10]. However, there are very few results which establish bounds regarding the number of necessary robots to accomplish a communication related task.

In [2], Demaine et al. studied the problem of moving pebbles along the edges of a graph (with n vertices) so as to achieve various connectivity objectives while minimizing the number of moves. In particular, they sketch an $O(n)$-approximation algorithm for the problem of creating a path of pebbles between two given vertices with minimum sum distance. They also show that minimizing the total or maximum distance is NP-hard, and that the maximum distance case cannot be approximated within a factor $\Omega(n^{1-\epsilon})$. Since connectivity and mobility are coupled in their model, their results do not directly apply to the problems studied here.

In this paper, we present the first results for the problem of building a communication bridge while minimizing the number of hubs and the distance traveled by them for a given communication radius.

II. Building a Bridge with the Minimum Number of Hubs

In this section, we study the problem of building a communication bridge between s and t while optimizing the number of hubs and the movement of the robots. We present solutions to two bi-criteria optimization problems: In the first problem (*MaxDist*), we seek a solution with the minimum number of hubs subject to the constraint that each robot moves at most a given distance d. In the second problem (*SumDist*), the constraint is that the total movement must not exceed B. In this paper, we present algorithms for given d or B. To find the minimum value of d (resp. B), one can perform a binary search on d (resp. B resp.).

A couple of remarks: When the distance between s and t is less than r, i.e. $|st| \leq r$, there is no need for any intermediate robots. Hence, we consider the case where $|st| > r$. Also, in order to achieve a bridge between s and t, it is both necessary and sufficient that the distance along $[s,t]$ between every consecutive pair in the communication bridge is at most r. Therefore, $|st| \leq (n+1)r$ holds. Hence, we assume that the number of robots n is at least $\lceil |st|/r \rceil - 1$.

A. MaxDist: Minimizing Maximum Distance

In *MaxDist*, we are given points s and t and a set, $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, of point-robots in the plane and a maximum traveling distance d. Any two members of $\mathcal{P} \cup \{s,t\}$ can communicate with one another if they are within (Euclidean) distance r of each other. Let $u_i = (x_i, y_i)$ be the initial position of p_i. We wish to select a subset $S \subseteq \mathcal{P}$ and compute a final position $v_i = (x'_i, y'_i)$ on the line segment $[s,t]$ for each $p_i \in S$ such that (i) s and t are connected via point-to-point communication links where points are selected from the final locations of robots in S and link lengths are not greater than the communication distance r, (ii) the distance traveled by each robot p_i is not greater than d (i.e. $\forall_{p_i \in S} |u_i v_i| \leq d$), and (iii) the total number of hubs in the communication bridge (i.e. $|S|$) is minimized.

Let L be the line passing through s and t. We place a coordinate frame where the x-axis is aligned with L, s is at 0 (i.e. $x_s = 0$) and t is at location $x_t > 0$. Without loss of generality, we define *right* as the positive direction of this frame. The final location of robots $p_i \in S$ can be determined as $v_i = (x'_i, 0)$ in this new coordinate frame. Hence, we can use x'_i to denote the final location $v_i = (x'_i, 0)$. Also note that the projection of the initial location $u_i = (x_i, y_i)$ on to L is simply x_i.

We start by pruning the set $\mathcal{P}$ and removing robots which are more than distance d away from L (i.e. if $|y_i| > d$ then p_i is removed). Moreover, we can remove the robots p_i such that $x_i < -d$ or $x_i > x_t + d$. This is because these robots cannot reach the line segment $[s,t]$. Let us call the new set which consist of robots satisfying the above constraints as $\mathcal{P}'$.

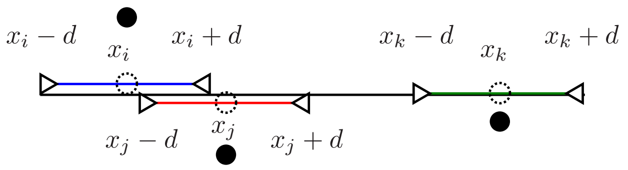

Fig. 3. Let x_i be the projection of the initial location of robot p_i. We relax the final location of p_i to $l_i : [x_i - d, x_i + d]$ which is shown as the left-most line segment.

Clearly, removing these robots does not change the feasibility of the problem.

For each robot $p_i \in \mathcal{P}'$, we compute a line segment $l_i : [x_i - d, x_i + d]$ (Figure 3). We will pick the final location of p_i from this line segment. Note that this is a relaxation because the robot may have to move more than distance d. But the deviation is bounded as it is stated in the following proposition [1]:

Proposition 1: For any final location $x'_i \in [x_i - d, x_i + d]$ where $p_i \in \mathcal{P}'$, the distance traveled is not greater than $\sqrt{2}d$, i.e. $|u_i v_i| \leq \sqrt{2}d$.

The number of hubs required for the relaxed version is not more that the number of hubs required for the original problem:

Proposition 2: Let k^* and k be the number of hubs used in an optimal solution to the original problem and an optimal solution to the relaxed problem, respectively. Then, $k \leq k^*$.

The relaxed version of the problem satisfies a simple ordering property which allows us to design an efficient algorithm. As mentioned previously (Figure 2) the original problem may not have the ordering property. We now explain the ordering property satisfied in the relaxed version.

Consider a placement of robots on L where the final location of each robot p_i is chosen from the line segment $[x_i - d, x_i + d]$. We order the robots according to x_i values in non-decreasing order. We say that the placement is *well-ordered* if for any two robots p_i and p_j such that $x_i \leq x_j$ we have $x'_i \leq x'_j$.

Lemma 3 (Ordering Property): There exists a well-ordered optimal solution for the relaxed problem.

Proof: In an optimal placement, let us call (p_i, p_j) an *unordered consecutive pair* if two robots p_i and p_j which are consecutive in the final bridge, are placed at respective locations x'_i and x'_j with $x_i \leq x_j$ but $x'_i > x'_j$. We claim that there is an optimal solution with zero unordered consecutive pairs. Consider an optimal solution which has the minimum number of unordered pairs. Suppose that this number is non-zero. Let p_i and p_j be two robots forming a consecutive unordered pair (if an unordered pair exists, so does a consecutive one). We show that the final locations of these two robots can be swapped, reducing the number of unordered pairs by one. This contradicts with the minimality of the number of unordered pairs.

First, from the relaxed segment assumption (i) $x'_i \leq x_i + d$ and (ii) $x_j - d \leq x'_j$ holds. Since this is an unordered pair, we have: (iii) $x_i \leq x_j$ and (iv) $x'_i > x'_j$. From (i)-(iv) we

[1]The proofs of these two propositions are straightforward and will be omitted in the final version. We included them in the Appendix for verification purposes.

have: $x_i - d \leq x_j - d \leq x'_j < x'_i \leq x_i + d$. Observe that $x_i - d \leq x'_j < x_i + d$ holds, hence we can move p_i to x'_j which is in its feasible region.

Similarly, we find that $x_j - d \leq x'_j < x'_i \leq x_i + d \leq x_j + d$. Hence, x'_i is in the feasible region of p_j which makes it possible to move p_j to x'_i.

Finally, we can conclude that we can swap the final locations of p_i and p_j and decrease the number of unordered pairs by one while p_i and p_j remain in their respective feasible regions. Moreover, since p_i and p_j are consecutive, swapping does not introduce additional unordered pairs. This contradicts the fact that the solution has the minimum number of unordered pairs. ■

The ordering property allows us to use dynamic programming to compute an optimal solution.

Before presenting the algorithm, we define the *reach* of a solution $S = \{p_1, p_2, \ldots, p_m\}$. Without loss of generality, let us assume that S is sorted in increasing order. If there is a communication bridge between s and p_m, then we have a reachable region from 0 to $x'_m + r$ where we can place a robot connected to s. As we assume that reach starts from 0, we can define the *reach* of S with a single parameter, i.e. $reach(S) = x'_m + r$.

Let $OPT(k, i)$ be the maximum *reach* which uses k robots from the set $\{p_1, p_2, \ldots, p_i\}$ to form a connected set with s where $\forall_{1 \leq j \leq i}\, p_j \in \mathcal{P}'$. To simplify the notation, we define the function $conn(k, i)$. This function returns true if and only if $[x_i - d, x_i + d]$ intersects with the *reach* of $OPT(k, i-1)$. In other words, this function tests whether a robot x_i can extend the *reach* $OPT(k, i-1)$ by moving inside its feasible region l_i and extend the *reach* of s. This condition is satisfied if the following holds: $OPT(k, i-1) \geq x_i - d$ and $x_i + d \geq 0$.

We now present the dynamic programming algorithm.

$$OPT(0, i) = r \tag{1}$$

$$OPT(1, i) = \begin{cases} \min(x_i + d, r) + r & if\ conn(0, i) \\ 0 & \text{o/w} \end{cases} \tag{2}$$

$$OPT(k, i) = 0 \quad if\ i < k \tag{3}$$

$$OPT(k, i) = \begin{cases} \min(x_i + d, & \\ OPT(k-1, i-1)) + r & if\ conn(k-1, i) \\ OPT(k, i-1) & \text{o/w} \end{cases} \tag{4}$$

The first two equations constitute the base cases. When we do not use any robots (i.e. $k = 0$) then the *reach* is r which is the reachability region of s (first equation). The second equation sets the initial values for $OPT(1, i)$. If feasible region l_i intersects with $[0, r]$ then we put the robot p_i at $\min(x_i + d, r)$ and set the *reach* of $OPT(1, i)$ as $\min(x_i + d, r) + r$. Otherwise since p_i cannot be connected to s we put a 0 value. Since $OPT(k, i)$ uses k robots from the set $\{p_1, p_2, \ldots, p_i\}$ the cardinality of this set cannot be less than k. This condition is addressed by Equation 3.

In the last equation, we compute all remaining entries $OPT(k, i)$. We know that the optimal solution chooses one of the $j \leq i$ as the k^{th} hub. We consider two cases: (1) the last hub is p_i: we look up the optimal solution with $k-1$ hubs which are selected from the set $\{p_1, p_2, \ldots, p_{i-1}\}$. If $[x_i - d, x_i + d]$ intersects with $OPT(k-1, i-1)$ then the optimum solution will put p_i to the rightmost possible location which is $x'_i = \min(x_i + d, OPT(k-1, i-1))$ and we set the *reach* $OPT(k, i) = x'_i + r$. (2) The last hub is not p_i: Then the k^{th} hub should be selected from set $\{p_1, p_2, \ldots, p_{i-1}\}$ whose maximum value is calculated by $OPT(k, i-1)$ in the previous iterations. If the first case suffices, we pick it since it extends *reach* more than the second case (due to the ordering property) otherwise we pick the second case and set it to $OPT(k, i)$.

Using the above formula, we calculate the dynamic programming table where both k and i vary between 0 and m where $m \leq n$ is the cardinality of pruned set $\mathcal{P}'$. From this table we find the minimum k such that $OPT(k, m) \geq x_t$. This yields the optimal solution to the relaxed problem. By Proposition 1, our solution gives a $\sqrt{2}$ approximation on the maximum distance traveled by using at most the same number of hubs used in the optimal solution (due to Proposition 2).

The running time of our algorithm is $O(n^2)$. This is because the size of the table is $O(n^2)$ and for each entry we take the maximum of two values (Equation 4).

Theorem 4: If there exists a solution to $MaxDist$ that uses k hubs such that each robot moves at most distance d, then we can compute a solution where we use at most k hubs and each hub moves at most $\sqrt{2}d$ in $O(n^2)$ time.

B. SumDist: Minimizing the Total Distance

In *SumDist*, we are given points s and t and a set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ of mobile hubs, as well as a budget B on the total distance traveled. Let $u_i = (x_i, y_i)$ be the initial position of p_i on the plane. We wish to select a subset $S \subseteq \mathcal{P}$ and compute a final position $v_i = (x'_i, y'_i)$ on the line segment $[s, t]$ for each $p_i \in S$ such that (i) s and t are connected via point-to-point communication links, (ii) the total L_2 (Euclidean) distance traveled is not greater than B (i.e. $\sum_{p_i \in S} |u_i v_i| \leq B$), and (iii) the total number of hubs in the communication bridge (i.e. $|S|$) is minimized.

Similar to *MaxDist*, we place a coordinate frame where the x-axis is aligned with L (the line passing through s and t), s is at $x = 0$ and t is at $x_t > 0$. The *reach* of a solution is defined as before.

Unfortunately, there exist instances where the ordering property does not hold in the L_2 metric. However, it turns out that when the underlying distance metric is L_1, there is an optimal solution which satisfies an ordering property, which in turn enables a dynamic programming based solution. We say that a placement is *well-ordered* if for any two robots p_i and p_j such that $x_i \leq x_j$ we have $x'_i \leq x'_j$.

Lemma 5: If the distance metric is L_1, then there exists a *well-ordered optimal solution.*

Proof: Let us assume that OPT_1^* is an optimal solution which includes the least number of unordered pairs. Let p_i and p_j be consecutive hubs used in OPT_1^* such that $x_i \leq x_j$ but $x'_i > x'_j$. We will show that swapping p_i and p_j's final locations does not increase the budget, i.e. if $b = |x_i - x'_i| + |x_j - x'_j|$ and $b' = |x_i - x'_j| + |x_j - x'_i|$ then $b \geq b'$ holds. On the other hand, the number of unordered pairs decreases by one.

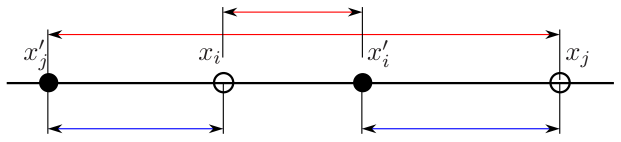

Fig. 4. Figure shows the case: $x'_j \leq x_i < x'_i \leq x_j$. Upper line segments show the total cost for the initial solution and lower line segments show the costs after swapping. When we swap the final locations of robots, we decrease the total cost while satisfying the ordering property.

This contradicts the minimality of the number of unordered pairs. Note that, since we only swap the final locations of the hubs, the connectivity is preserved. Further, swapping does not change the total budget used in the y direction. Therefore, the overall budget does not increase as well.

Assume that we fix the locations of x_i and x_j: we have three "bins" ($x \leq x_i$, $x_i < x \leq x_j$ and $x_j < x$) for possible locations of x'_i and x'_j. The following set of equations correspond to all 6 possible cases. In each case, the claim above holds. In Figure 4, the second statement in the first line is illustrated.

$$
\begin{aligned}
x'_j < x'_i \leq x_i \leq x_j &\Rightarrow b = b' \\
x'_j \leq x_i < x'_i \leq x_j &\Rightarrow b > b' \\
x'_j \leq x_i \leq x_j < x'_i &\Rightarrow b \geq b' \\
x_i \leq x'_j < x'_i \leq x_j &\Rightarrow b > b' \\
x_i \leq x'_j \leq x_j < x'_i &\Rightarrow b \geq b' \\
x_i \leq x_j \leq x'_j < x'_i &\Rightarrow b = b'
\end{aligned}
$$

■

We now solve *SumDist* optimally for the L_1 metric (up to an arbitrarily small additive cost). We start by building a table $T(k, i, B)$ which stores the maximum *reach* using k hubs subject to: (i) the i^{th} robot is the k^{th} hub, and (ii) the budget for the first k robots is at most B. The entries are computed as follows:

$$T(0, i, B) = r \ \ \forall_i \tag{5}$$

$$T(k, i, B) = 0 \ \ \forall_{k>i} \tag{6}$$

$$T(k, i, 0) = \begin{cases} x_i + r & \text{if a } k \text{ hub } \textit{bridge} \text{ exists initially} \\ 0 & \text{o/w} \end{cases} \tag{7}$$

$$T(k+1, i, B) = \max_{k \leq j < i} \max_{b' \in C(x_i)} \min(T(k, j, B-b), x_i + b') + r \tag{8}$$

$$T(k, i, B+\varepsilon) = \max_{k \leq j < i} \max_{b' \in C(x_i)} \min(T(k, j, B+\varepsilon-b), x_i + b') + r \tag{9}$$

where B is discretized by ε, $b' = b - y_i$ and $C(x_i)$ is a set of possible values for b'. We will discuss ε and $C(x_i)$ shortly. The first two equations are the base cases. If initially the robots create a communication bridge between s and p_i with k hubs, then Equation 7 sets the *reach* $T(k, i, 0)$ to $x_i + r$. This can be checked by building a graph G whose vertices are $P' \cup \{s, t\}$ where P' is the set of hubs that are initially on $[s, t]$. There is an edge between two vertices if the distance between them is at most r. If G has a path between s and p_i of length at most k, then a communication bridge from s to p_i can be formed with budget 0.

Here, we discuss only how to extend the first dimension of the dynamic programming formulation (Equation 8). The argument for the other dimension (Equation 9) is similar.

To calculate $T(k+1, i, B)$, we consider the optimal *reach* with k hubs when using the p_j as the k^{th} hub for all $j < i$ (due to the ordering property we do not need to consider the locations of earlier hubs in the optimal solution). Let $R = T(k, j, B-b)$ be the maximum *reach* achievable by using k robots with p_j as the last hub and a total budget of $B - b$. The final location of p_j in this optimal *reach* is $R - r$. We need to compute the *reach* for $k+1$ hubs where p_i is the last hub and p_i travels at most b units. For this, we consider all possibilities for R.

Note that the distance of the initial location $u_i = (x_i, y_i)$ to L is y_i. Hence, $b \geq y_i$ must hold for p_i to act as a hub. Let $b' = b - y_i$, then, $[x_i - b', x_i + b']$ is the region that robot p_i can be placed on the line L with a budget of b'. Due to the ordering property, p_i must be placed to the right of p_j. Therefore, its location is after $R - r$ and before R (otherwise p_j and p_i cannot communicate). In other words, valid locations for p_i are given by the intersection of $[x_i - b', x_i + b']$ and $[R - r, R]$, and this set should be non-empty.

We now compute the set of valid budgets b for robot p_i. Since the robot has to travel y_i for the vertical component, the remaining budget for the horizontal component is $b' = b - y_i$. Let $C(x_i)$ be the set of possible values for b'. This set is computed as follows:

$$C(x_i) = \{b' | b' \leq B - y_i \wedge Z(x_i, R)\} \tag{10}$$

$$Z(x_i, R) = \begin{cases} R - r - x_i \leq b' \leq R - x_i & \text{if } x_i \leq R - r \\ 0 \leq b' \leq R - x_i & \text{if } R - r < x_i \leq R \\ b' = x_i - R & \text{o/w} \end{cases} \tag{11}$$

For a budget b' to be valid, we must have $b \leq B$. This gives the first condition for b': $b' \leq B - y_i$. We use the function Z to constrain b' as a function of x_i and the current *reach* R. We consider the three cases based on the location of x_i with respect to the location of the last robot (x'_j) and the *reach* $R = x'_j + r$. See Figure 5.

Case 1 ($x_i \leq x'_j$):, In this case, we must have $b' \geq R - r - x_i$, (otherwise p_i cannot extend the current *reach*) and $b' \leq R - x_i$ (if p_i moves further to the right, p_j and p_i can't communicate).

Case 2 ($x'_j < x_i \leq R$): Similar to case 1, b' should not be greater than $R - x_i$. The lower bound is obtained by the non-negativity of b'.

Case 3: When x_i is to the right of the current *reach* R, there is only one value robot p_i should move: the rightmost reachable point.

The new *reach* after placing robot p_i to $min(R, x_i + b)$ is $min(R, x_i + b) + r$. In order to compute $T(k+1, i, B)$, among all possible $j < i$ and all possible budgets $b' \in C(x_i)$, we find

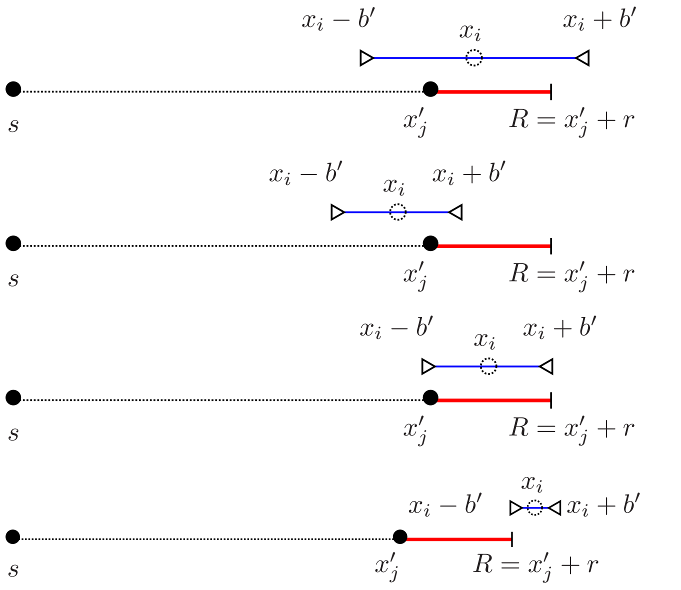

Fig. 5. Let x'_j be the last hub location at the *reach* and p_i be the robot considered at the current iteration. **Top Figure:** When b' is too large both end points of feasible region is out of the the region $[x'_j, R]$, hence b' is redundant in this example. **Next Three Figures:** The three cases considered in Equation 11 are illustrated.

the optimal *reach*. Since the size of the set $C(x_i)$ is bounded by r/ε, each entry can be calculated in $O(nr/\varepsilon)$ time.

We now show how this result yields an approximation algorithm for L_2. Let OPT_1^* and OPT_2^* be optimal solutions for L_1 and L_2 metrics, respectively. The following lemma bounds the deviation between OPT_1^* and OPT_2^*.

Lemma 6: Let OPT_2^* be the optimal solution for the L_2 metric with a given budget B. Suppose OPT_2^* can connect s and t using k hubs. There exists optimal solution OPT_1^* for the L_1 metric which can connect s and t by using k hubs and a budget of $\sqrt{2}B$.

Proof: Let (x_i, y_i) be the initial location of a robot used in OPT_2^* and x'_i be the final location. The L_1 and L_2 distances are $|x_i - x'_i| + y_i$ and $\sqrt{|x_i - x'_i|^2 + y_i^2}$, respectively. Without loss of generality, we scale the distances by $1/y_i$ so that the L_1 and L_2 distances become $a+1$ and $\sqrt{a^2+1}$, respectively where $a = |x_i - x'_i|/y_i$. From elementary calculus, it is easy to show that: $f(a) = \frac{a+1}{\sqrt{a^2+1}} \leq \sqrt{2}$. ■

To obtain the optimal L_1 solution for budget B, we solve $T(k, i, B)$ for all possible k, i, B (where B is discretized with ε intervals). Due to the discretization, the total budget used here can be at most $k_1^*\varepsilon$ than the budget used by OPT_1^* where k_1^* is the number of hubs used by OPT_1^*. In other words, our dynamic programming algorithm can find a solution with k_1^* hubs by using at most $B_1 + k_1^*\varepsilon$ budget where B_1 is the used budget with L_1 metric. This means that B', the total budget used by our solution will be bounded by $B_1 + n\varepsilon$. Consequently, the total budget used by our algorithm will be at most $\sqrt{2}B + n\varepsilon$ where B is the given budget in L_2 metric. We can choose ε to achieve an arbitrarily small additive error.

We now establish the running time of the algorithm. The size of the table is $O(\frac{n^2 B}{\varepsilon})$ and as we discussed earlier each entry can be calculated in $O(nr/\varepsilon)$ time. Hence, the time complexity of our algorithm is $O(\frac{n^3 Br}{\varepsilon^2})$.

Theorem 7: If there exists a solution to *SumDist* that uses k hubs such that the total movement of robots is B in the L_2 (Euclidean) metric, then we can compute a solution where we use at most k hubs and the total movement of robots is at most $\sqrt{2}B + n\varepsilon$ in $O(\frac{n^3 Br}{\varepsilon^2})$ time, where ε is the discretization constant.

III. Bounds on Number of Hubs

Let $OPT(d)$ be the number of hubs in an optimal solution to *MaxDist* with distance constraint d. How does this constraint affect the number of hubs on the bridge? In other words, if $\mathit{OPT}(\infty) = \lceil |st|/r \rceil - 1$ is the number of hubs required in the unrestricted version, how far is $OPT(d)$ from $\mathit{OPT}(\infty)$? In this section, we show that $OPT(d)/\mathit{OPT}(\infty) \leq 2$.

Assume that $m - 1 < |st|/r \leq m$, for some integer $m > 1$. (The case $m = 1$ is uninteresting, as s and t are then within distance r, hence connected.)

Partition $[s, t]$ into m equal-length intervals, labeled from s to t as $I_1, I_2, \ldots, I_m$. Each interval has length greater than $(1 - 1/m)r$ and at most r. Consider any solution for $\mathit{OPT}(d)$. This solution connects s and t with the fewest number of hubs. In such a solution, we can have at most two hubs inside any I_j, $2 \leq j \leq m-1$. To see this, note that if there were three or more hubs in I_j, then all but the two extreme ones in I_j could be removed without losing connectivity (since the length of I_j is at most r), thereby obtaining a solution for $\mathit{OPT}(d)$ that has fewer hubs than the original optimal solution—a contradiction. Along similar lines note that I_1 and I_m can each contain at most one hub; if there was more than one hub in I_1 (resp. I_m), then all the ones except the one farthest from s (resp. t) can be removed without losing connectivity.

It follows that for any optimal solution, we have $\mathit{OPT}(d) \leq 2(m-2) + 2 = 2(m-1)$. Also, $OPT(\infty) = \lceil |st|/r \rceil - 1 = m - 1$. Hence, we have the following.

Lemma 8: $OPT(d)/\mathit{OPT}(\infty) \leq 2$.

Using similar arguments, it can be shown that the same bound applies for *SumDist*.[2]

Next, we show that the bound in Lemma 8 is tight: We claim that, for any finite d, there is an instance of *MaxDist* with the optimal solution $\mathit{OPT}(d)$ for which $OPT(d)/OPT(\infty) = 2$.

Let $|st|/r = m > 1$; thus, each interval $I_1, I_2, \ldots, I_m$ defined above has length r. Let ε be a real number in the (open) interval $(0, \frac{r}{m-1})$. Consider a set $V = \{v_1, v_2, \ldots, v_{2(m-1)}\}$ of points on $[s, t]$, defined as follows: for $j = 2, 4, \ldots, 2(m-1)$, $v_j = \frac{j}{2}\varepsilon + \frac{j}{2}r$, and for $j = 1, 3, \ldots, 2m-3$, $v_j = \frac{j+1}{2}\varepsilon + \frac{j-1}{2}r$. See Figure 6.

The set V satisfies the following (easily-verifiable) properties: (i) $v_1 \neq s \in I_1$ and $v_{2(m-1)} \neq t \in I_m$; (ii) successive points in $V \cup \{s, t\}$ are within distance r; and (iii) at least one pair of successive points in $V' \cup \{s, t\}$ is not within distance r for any $V' \subset V$.

Let $\mathcal{P}$ be a set of $n \geq 2(m-1)$ robots $\{p_1, p_2, \ldots, p_n\}$ and choose their initial positions in $\mathbb{R}^2$ as follows: for $j = 1, 2, \ldots, 2(m-1)$, place p_j at initial position $u_j = (v_j, d)$.

[2]Reviewers can find the proof in the Appendix section which will be removed in the final revision.

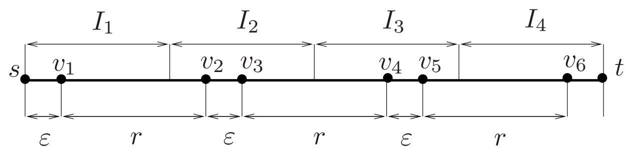

Fig. 6. Selection of points $v_1, v_2, \ldots, v_{2(m-1)}$ on $[s,t]$, with $m = 4$.

Place any remaining hubs in $\mathcal{P}$ at some distance greater than d from $[s,t]$.

Observe that only $p_1, p_2, \ldots, p_{2(m-1)}$ can move onto $[s,t]$ and, moreover, each such p_j can move only to the location v_j. By properties (ii) and (iii) above, it follows that $p_1, p_2, \ldots, p_{2(m-1)}$ are necessary and sufficient to establish a communication bridge between s and t. Therefore, $OPT(d) = 2(m-1)$ and the claim follows.

IV. Conclusion

In this paper, we introduced the problem of building a communication bridge between two points s and t while minimizing the number of hubs on the bridge and satisfying a maximum (or total) distance constraint for the robots. For both versions we presented constant factor approximation algorithms for the geometric version where the robots must move onto $[s,t]$.

There are many interesting directions for future work. It is not clear whether the $\sqrt{2}$ approximation factor for the geometric version can be improved. The general version in which the final locations of hubs can be anywhere on the plane seems difficult. Solving the version where there are multiple source and destination pairs seems to be even harder.

Acknowledgment

This work is supported in part by NSF IIS-0917676, NSF CCF-0907658, NSF CNS-0936710, NCF INT-0422775 and NCF CCF-0514950. Volkan Isler thanks Jie Gao and Sanjeev Khanna for useful discussions.

References

[1] O. Tekdas, Y. Kumar, V. Isler, and R. Janardan, "Building a communication bridge with mobile hubs," in *5th International Workshop on Algorithmic Aspects of Wireless Sensor Networks (ALGOSENSORS)*, 2009, pp. 179–190.
[2] E. D. Demaine, M. Hajiaghayi, H. Mahini, A. S. Sayedi-Roshkhar, S. Oveisgharan, and M. Zadimoghaddam, "Minimizing movement," in *SODA '07*, 2007, pp. 258–267.
[3] P. Corke, S. Hrabar, R. Peterson, D. Rus, S. Saripalli, and G. Sukhatme, "Autonomous deployment and repair of a sensor network using an unmanned aerial vehicle," in *ICRA*, vol. 4, 2004, pp. 3602–3608.
[4] N. Atay and B. Bayazit, "Mobile wireless sensor network connectivity repair with k-redundancy," *Algorithmic Foundation of Robotics VIII*, pp. 35–49.
[5] O. Tekdas, J. Lim, A. Terzis, and V. Isler, "Using mobile robots to harvest data from sensor fields," *IEEE Wireless Communications*, 2009.
[6] A. Kansal, A. A. Somasundara, D. D. Jea, M. B. Srivastava, and D. Estrin, "Intelligent fluid infrastructure for embedded networks," in *MobiSys '04*. ACM, 2004, pp. 111–124.
[7] I. Chatzigiannakis, A. Kinalis, and S. Nikoletseas, "Efficient data propagation strategies in wireless sensor networks using a single mobile sink," *Comput. Commun.*, vol. 31, no. 5, pp. 896–914, 2008.
[8] W. Wang, V. Srinivasan, and K.-C. Chua, "Using mobile relays to prolong the lifetime of wireless sensor networks," in *MobiCom '05*. ACM, 2005, pp. 270–283.
[9] M. M. B. Tariq, M. Ammar, and E. Zegura, "Message ferry route design for sparse ad hoc networks with mobile nodes," in *MobiHoc '06*, 2006, pp. 37–48.
[10] J. Ma, C. Chen, and J. P. Salomaa, "mWSN for large scale mobile sensing," *J. Signal Process. Syst.*, vol. 51, no. 2, pp. 195–206, 2008.

Appendix

Proof: **[Proposition 1]**

The maximum distance traveled for p_i is $\sqrt{d^2 + y_i^2}$ when the movement is relaxed to l_i. Since $y_i \leq d$ holds, the claim follows, i.e., $\sqrt{d^2 + y_i^2} \leq \sqrt{2}d$. ■

Proof: **[Proposition 2]**

An optimal solution to the original problem cannot place a robot p_i outside of $l_i : [x_i - d, x_i + d]$. Because otherwise the distance traveled in x-direction exceeds the distance constraint d. Hence an optimal solution to the original problem is also a solution for the relaxed case, and k cannot exceed k^*. ■

Proof: **[Lemma 8 for sum case]**

Let $m = \lceil |st|/r \rceil$, we partition $[s,t]$ into m equal-length intervals, $I_1, I_2, \ldots, I_m$. For each I_j, $(1-1/m)r < |I_j| \leq r$ holds where $|I_j|$ is the length of the interval. Let *OPT*(B) be the solution which uses the minimum number of hubs. In *OPT*(B) we can have at most two hubs inside any I_j, $2 \leq j \leq m-1$. As claimed in the max case, if there were three or more hubs in I_j, then all but the two extreme ones in I_j could be removed without losing connectivity. This is a contradiction with the minimality assumption of the solution. Similarly I_1 and I_m can each contain at most one sensor.

It follows that for any optimal solution, we have *OPT*$(B) \leq 2(m-2)+2 = 2(m-1)$. Also, $OPT(\infty) = \lceil |st|/r \rceil - 1 = m-1$. Hence, we have: $OPT(d)/$*OPT*$(\infty) \leq 2$. ■