Sensor Networks Deployment Using Flip-Based Sensors

Sriram Chellappan, Xiaole Bai, Bin Ma and Dong Xuan

***Abstract*— In this paper, we study the issue of mobility based sensor networks deployment. The distinguishing feature of our work is that the sensors in our model have limited mobilities. More specifically, the mobility in the sensors we consider is restricted to a flip, where the distance of the flip is bounded. Given an initial deployment of sensors in a field, our problem is to determine a movement plan for the sensors in order to maximize the sensor network coverage, and minimize the number of flips. We propose a minimum-cost maximum-flow based solution to this problem. We prove that our solution optimizes both the coverage and the number of flips. We also study the sensitivity of coverage and the number of flips to flip distance under different initial deployment distributions of sensors. We observe that increased flip distance achieves better coverage, and reduces the number of flips required per unit increase in coverage. However, such improvements are constrained by initial deployment distributions of sensors, due to the limitations on sensor mobility.**

I. Introduction

Sensor networks deployment has received significant attention in the recent past [1], [2], [3], [4], [5], [6], [7]. An important goal during deployment is to ensure that the sensors in the network meet critical network objectives including coverage, load balancing etc. In this realm, a class of work has recently appeared where mobility of sensors is leveraged for deployment [1], [2], [3]. Typically in such works, the sensors detect lack of desired deployment objectives. The sensors then estimate locations to which they should move, and make the resulting movement. While the above works are quite novel in their approaches, the mobility of the sensors in their models is unlimited. Specifically, if a sensor chooses to move to a desired location, it can do so without any restriction.

In practice however, it is quite likely that the mobility of sensors is limited. Towards this extent, a class of Intelligent Mobile Land Mine Units (IMLM) [8] to be deployed across battlefields have been developed by DARPA. The units are expected to detect breaches, and move in order to repair them. The mobility of the IMLM units is limited. Specifically, the mobility in the units is restricted to only a *hopping* mechanism. Each IMLM unit carries onboard fuel tanks and a spark initiation system to propel the hop. The hop distance is limited, depending on the amount of fuel and the propeller dynamics. Such a model typically trades-off mobility with energy consumption, cost etc. In fact, in many applications, the latter goals outweigh the necessity for advanced mobilities, making such mobility models quite practical in the future.

In this paper, we study sensor networks deployment using sensors with limited mobilities. In our model, sensors can *flip* (or hop) only once to a new location, and the flip distance is bounded. The initial deployment in the sensor network may have holes in the network that are not covered by any sensor. In this framework, our problem is to determine optimal movement plan of the flip-based sensors to maximize the coverage in the network (or minimize the number of holes), and simultaneously minimize the total number of flips. We propose a minimum-cost maximum-flow based solution to our deployment problem. Our approach is to construct a graph (called *virtual graph*) based on the initial deployment and mobility model, and determine the minimum cost maximum flow in the virtual graph. The resultant flow in the virtual graph is appropriately translated as physical flip sequences by sensors in the network. We prove that our solution maximizes coverage, while it simultaneously minimizes the total number of sensor flips required. We also study the sensitivity of coverage and the number of flips to flip distance under different initial deployment distributions of sensors. We observe that increased flip distance achieves better coverage, and reduces the number of flips required per unit increase in coverage. However, such improvements are constrained by initial deployment distributions of sensors, due to the limitations on sensor mobility.

The rest of our work is organized as follows. We present important related work in Section II. In Section III, we formally define our flip-based mobility model, and the problem definition. We then present our solution, optimality properties, and some discussions in Section IV. In Section V, we present results of our performance evaluations and discussions on our results. Finally, we conclude our paper with some final remarks in Section VI.

II. Related Work

Deploying sensor networks has received significant attention in the recent past. The first deployment strategy is randomly deploying sensors in the field [9], [10]. In this class some recent work like [11] have appeared, where the authors choose to deploy sensors in groups. The distribution pattern of group deployment is exploited for localization purposes. The second class of deployments follows incremental strategies, where sensors are deployed iteratively after making some measurements on the quality of previous partial deployments [12],

Sriram Chellappan, Xiaole Bai and Dong Xuan are with The Department of Computer Science and Engineering, The Ohio-State University, Columbus, OH 43210, U.S.A. E-mail: {chellapp, baixia, xuan}@cse.ohio-state.edu. Bin Ma is with The Department of Computer Science, University of Western Ontario, London, ON N6A5B7, Canada. Email: bma@csd.uwo.ca.

This work was partially supported by NSF under grant No. ACI-0329155. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

[13], [14], [15]. However, the key shortcoming is that such approaches need to be conducted by an external mobile robot or a human being and as such are restrictive, especially in large-scale or hostile zone deployments.

More recently, mobile sensors have been considered for deployment [1], [2], [3]. The key objective in [1] and [3] is to detect holes in the network and to ensure that they are covered by at least one sensor. Sensors locally detect holes, estimate their new positions, and move towards the new positions, to cover detected holes. In the approach proposed by Wang, Cao and La Porta, in [3] the detection of holes is based on each sensor constructing Voronoi diagrams. The authors then propose three protocols, namely Vector-based algorithm, Voronoi-based algorithm and Minimax algorithm to for sensor movements to repair holes. In [1], Howard, Mataric and Sukhatme propose the idea of constructing potential fields for sensor movements. The fields are constructed such that each node is repelled by both obstacles and by other nodes, thereby forcing the network to spread itself throughout the environment. Another related work in deployment using mobile sensors is Wu and Yang's work [2]. In [2], the sensor network is divided into clusters. The objective is to ensure that the number of sensors per cluster is uniform. The movement algorithm is to scan the clusters in two stages (row-wise and column-wise) and determining the new sensor locations (or clusters). Another feature in the above works is that their algorithms attempt to also minimize the overall sensor movement, during deployment, since mobility is an energy consuming operation.

In general our work in this paper shares similar objectives with the above works on mobility based deployment. We study the issue of repairing holes in the network. However, the distinguishing feature of our work from the above is the sensor mobility model we use in this paper. Existing approaches discussed above [1], [2], [3] consider sensors with unlimited mobility. In our model sensors can *flip* only once to a new location, and the flip distance is bounded. We believe that such a model is realistic and practical in the future. DARPA has already conducted research on a class of Intelligent Mobile Land Mine Units (IMLM) [8] that are similar to the flip-based sensors we model in this paper. Briefly, the mobility system in [8] is based on a hopping mechanism that is actuated by a single-cylinder combustion process. Each IMLM unit in the field carries onboard fuel tanks and a spark initiation system. For each hop, the fuel is metered into the combustion chamber and ignited to propel the IMLM unit into the air. A steering system enables proper orientation during the hop. There can also be other technologies that can assist in such mobilities, like spring actuation, external launchers etc. However, the fact is that such models for sensor mobilities are quite realistic and is what we study in this paper.

III. Mobility Model and Problem Definition

A. The Flip-based Sensor Mobility model

In this paper, we model sensor mobilities as a *flip*, where a sensor flips when triggered by an appropriate signal (external or internal signal). Such a movement can be realized in practice by propellers that are powered by fuels [8], coiled springs that unwinds for flipping, external agents launching sensors after being deployed in the field etc. In such an event, the sensor physically moves from its current location by flipping (or jumping) to the new location.

In our model, sensors can *flip* only once to a new location. This could be due to fuel being constrained, or the spring being completely uncoiled on a flip, or the external agent launching the sensor. The flip distance (denoted by F) is bounded. The flip distance depends on the quantity of fuel available, or the degree of spring coil, or launching distance. The sensor can flip in a desired angle. Mechanisms in [8] can be used for orientation prior to a flip. We denote d as the basic unit of flip distance. In our paper, we first study the case, where the flip distance $F = d$. We then extend this to study cases where the flip distance $F > d$ (but is still bounded).

B. Problem Definition

We address a deployment problem in this paper. The sensor network we study is a rectangular field. It is divided into 2-dimensional regions, where each region is a square of size R. The initial deployment of sensors in the field may not cover all regions. In this context, our problem statement is; Given a sensor network of size D, a desired region size R, an initial deployment of N flip-based sensors that can flip once to a distance F, our goal is to determine an optimal movement plan for the sensors, in order to maximize the number of regions that is covered by at least 1 sensor, while simultaneously minimizing the total number of flips required. The input to our problem is the initial deployment (number of sensors per region) in the sensor network, and the mobility model of sensors. The output is the detailed movement plan of the sensors across the regions (which sensors should move, and where) that can achieve our desired objectives.

The region size R is contingent on the application. It is determined by the system deployer based on sensor coverage, connectivity, sensing/ transmission ranges and application demands. In this paper, we first assume that the desired region size R is an integral multiple of the basic unit of flip distance, i.e., $R = i * d$, where i is an integer ≥ 1. We discuss the general case of R subsequently. We assume that each sensor can know which region it resides it. To do so, the methods proposed in [16] can be applied, where location of sensors is determined by using sensors themselves as landmarks. In our model, a sensor can flip to its left, right, top and bottom regions only.

We illustrate our problem further with an example. Figure 1 (a) shows an instance of initial deployment. The shaded circles denote sensors, and the numbers denote the id of the corresponding region. The neighbors of any region are its immediate left, right, top and bottom regions. For instance in Figure 1, the neighbors of region 6 are regions 2, 5, 7 and 10. The initial deployment may have holes that have no sensors in them. For instance, in Figure 1 (a) after the initial deployment, regions 1, 6, 11, 12, 16 are not covered by any sensor and are thus holes. This is the problem we address in this paper.

The above problem is not easy to solve. For instance, consider Figure 1 (a). For ease of elucidation, let the desired

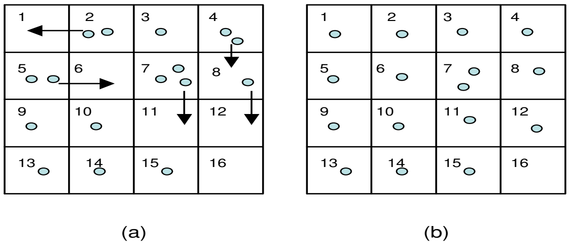

Fig. 1. A snapshot of the sensor network and a movement plan to maximize coverage (a), and the resulting deployment (b)

region size $R = d$. Let the flip distance $F = d$. One intuitive approach towards maximizing coverage is to let sensors from source regions (more than 1 sensor) to flip to hole regions (no sensors) in their neighborhood, using local information around them. In Figure 1 (a), region 7 has 3 sensors in it, while region 11, a neighbor of 7 is empty. Similarly, region 8 has a sensor while its neighbor, region 12 is empty. If we allow neighbors to obtain local neighbor information, then intuitively a sensor from region 7 will attempt to cover regions 11 and 16. This intuition is because region 7 (with extra sensors) is nearest to holes 11 and 16. Similarly, region 4 will try to cover region 12. The resulting sequence of flips, and the corresponding deployment are shown in Figures 1 (a) and (b) respectively.

With this movement plan, region 16 is still uncovered. This is because, while region 7 has extra sensors, there are no mobile sensors in regions 11 and 12. Also, region 15 cannot provide a sensor, without making itself (or some other region) a hole as shown in Figure 1. This means that all paths to region 16 are *blocked* in this movement plan, preventing region 16 from being covered. However, there exists an optimal plan that can cover all regions in this case as shown in Figure 2. For optimal deployment, the path of movements to cover region 16 starts from region 5. In fact, for optimal coverage, this plan also requires the minimum number of flips (10 flips). The key challenges we have to overcome to solve our problem are 1) the trade-offs in simultaneously attempting to optimize both coverage and number of flips and 2) the constraints arising from limited mobility, due to which a sensor from a far away region may need to flip towards a far away hole, and a *chain* of sensors may need to progressively move towards the particular hole for covering it. Determining such a movement plan for optimizing both coverage and number of flips is not trivial.

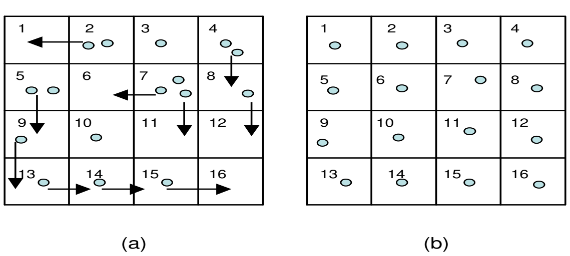

Fig. 2. A snapshot of the sensor network and the optimal movement plan (a), and the resulting deployment (b)

IV. Our Solution

A. Design Rationale

In this paper, we propose a solution, where the information on the number of sensors per region is collected across all regions, and a movement plan for the sensors is determined prior to their flip. In our solution, a centralized node (a Base-station) collects information about the number of sensors in the regions. We propose a minimum-cost maximum-flow based solution that is executed by the Base-station using the region information. The output of our solution is a movement plan (which sensors should move and where) for the sensors. The Base-station will then forward the movement plan to corresponding sensors. We prove subsequently that execution of this plan, will result in maximizing coverage, and minimizing the number of flips. For collecting region information, sensors can exchange information on number of sensors in each region among themselves, or sensors can send the region information directly to the Base-station. Another solution that does not require a centralized node is to let individual sensors collect region information, and execute our solution independently to determine the movement plan. We discuss the latter solution in Section IV-E.

Let us denote regions with at least one mobile sensor as *sources*, and regions without any sensor as *holes*. Source Regions can provide sensors (like region 5 initially in Figure 2), or be on a path from another source to a hole (like regions 9, 13, 14 and 15 initially). Holes can only accept sensors (regions 1, 11 and 16). If we are able to maintain state information for each region on, i) whether it has a mobile sensor, ii) establish possible paths among multiple sources regions and between multiple source regions and multiple holes and iii) clearly constrain the paths based on the desired objective and limited mobilities available, then our problem can be translated to determining how to maximize the flow of sensors from source regions to holes, without violating the path constraints between them.

If we identify regions (sources or holes) using vertices, and incorporate neighbor relationships in the sensor network as edges (with constrained capacities) between the vertices, then from a graph-theoretic perspective, our problem is a version of the multi-commodity maximum flow problem, where the problem is to maximize flows from multiple sources to multiple sinks, while ensuring that the capacity constraints on the edges in the graph are not violated. While obtaining the optimal plan to maximize coverage, we also want to minimize the number of flips. That is, if we associate a cost with each flip, we wish to minimize the overall cost of flips while still maximizing coverage. This problem is then a version of the minimum-cost multi-commodity maximum-flow problem, where the objective is to find paths that minimize the overall cost while still maximizing the flow. Our solution is to model the sensor network as an appropriate graph structure following the objectives discussed above, determine the minimum cost maximum flow in the graph and translate it back as flip sequences in the sensor network. For the rest of the paper, if the context is clear, we will address our solution as minimum-cost maximum-flow solution.

We have so far pointed out the core features of our solutions. A major task is how to incorporate the sensor network and its initial deployment as a directed graph so that we can map the minimum-cost maximum-flow problem directly to our problem. We call this graph as a *virtual graph*. The Base-station will execute the minimum-cost maximum-flow algorithm on the virtual graph. The resulting maximum flow along the edges in the virtual graph is then appropriately translated as flip (or movement) plan in the sensor network. The plan indicates which sensor should move and where.

B. Constructing the virtual graph from the initial deployment

To construct the virtual graph, we need the initial deployment (with N sensors), the granularity of desired coverage (region size R), flip distance (F) and the number of sensors per region i (n_i). We denote the number of regions in the network as Q. Let $G_S(V_S, E_S)$ denote an undirected graph representing the sensor network. Each $i \in V_S$ denotes one region in the sensor network and each $e \in E_S$ represents the neighboring relationship between the regions in the sensor network. G_S purely represents the initial network structure (and does not reflect whether regions are sources or holes), and as such is undirected. The virtual graph (denoted by $G_V(V_V, E_V)$) is constructed from G_S.

The key task in constructing the virtual graph is to first determine its vertices (the set V_V) commensurate with the status of each region as a source or hole. Then, we have to establish the edges (the set E_V) in the virtual graph between the vertices. This includes direction of edges, their capacities and costs. To do so, we have to carefully incorporate the direction of edges (based on sources or holes), edge capacities and costs (based on F). The final objective is to ensure that execution of the minimum-cost maximum-flow algorithm on the virtual graph can be translated into an optimal movement (flip) plan for the sensors in the network. We first discuss construction of the virtual graph *in a simple,* but representative case. We then discuss extensions of the simple case subsequently.

We introduce important additional notations before describing the construction. We first introduce the notion of time t. The minimum-cost maximum-flow algorithm on a graph executes iteratively. At each iteration, flows along edges are updated. The term t denotes the current iteration in the algorithm execution. For ease of usage, we consider iterations using the notion of time. Let us denote $L(i,t)$ as the total number of sensors in region i at time t. We denote the total number of mobile sensors in a region i at time t as $M(i,t)$. Thus, $L(i,0) = M(i,0) = n_i$ for all i. Let us denote $C(p,q)$ as the capacity of the edge between vertices p and q in the virtual graph (G_V). Note here that G_V is directed.

Construction when $F = d$ and $R = d$: In this case, the flip distance F is equal to basic unit d. The desired region size $R = d$. In the virtual graph, each region (of size R) is represented by 3 vertices. For each region, whose id is i, we have a vertex for it in G_V called base vertex, denoted as v_i^b. For each region, we need to keep track of the number of sensors from other regions that have flipped to it, and the number of sensors that have flipped from this regions to other regions. The former task is accomplished by introducing an in vertex, and the latter is accomplished by adding an *out* vertex to each region. For each vertex i, its *in* vertex in the virtual graph is denoted as v_i^{in} and its *out* vertex is denoted as v_i^{out}.

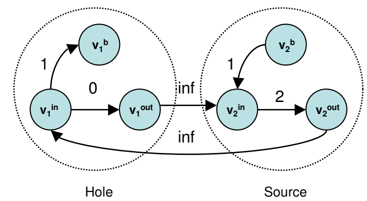

Fig. 3. The Virtual Graph with only regions 1 and 2 in it

Having established the vertices, we now discuss how edges (and their capacities) are added between the vertices in the G_V. For each region i that has ≥ 1 sensors, it is a source region. We are thus interested in how to optimally *push* sensors from such regions. In the virtual graph, an edge is added from the corresponding v_i^b to v_i^{in} with edge capacity equal to $n_i - 1$. The interpretation of this is that when attempting to determine the flow from the base vertex (v_i^b), at least 1 sensor will remain in the corresponding region i. Then an edge with capacity n_i is added from the same v_i^{in} to v_i^{out}. This ensures that it is possible for up to n_i mobile sensors in this region to flip to its neighbors. Recall the example in Figure 1 (a). Region 2 is a source. The virtual graph construction corresponding to this region is shown in Figure 3, where there is an edge with capacity $n_2 - 1 = 1$ from vertex v_2^b to v_2^{in}, and an edge of capacity $n_i = 2$ from v_2^{in} to v_2^{out}. Other source regions are treated similarly in the virtual graph.

For each region i that has 0 sensors it is considered a hole. We are interested in how to optimally *absorb* sensors in such regions. For holes, an edge is added from the corresponding v_i^{in} to base vertex v_i^b with edge capacity equal to 1. This is to allow a maximum of 1 sensor into the base vertex v_i^b of hole region i. If a sensor flips to this hole, the hole is then covered, and no other sensor needs to flip to this region. Then an edge with capacity 0 is added from the same v_i^{in} to v_i^{out}. This is because a sensor that moves into a hole will be not able to flip further [1]. Recall again from the example in Figure 1 (a). Region 1 is a hole. In Figure 3, there is an edge with capacity 1 from vertex v_1^{in} vertex to v_1^b, and edge of capacity 0 from v_i^{in} to the v_1^{out}. Other holes are treated similarly in the virtual graph. Based on the above discussions we now have,

$\forall v_i^{in}$ and $v_i^{out} \in V_V$,

$$C\ (v_i^{in}, v_i^{out}) = M(i,t). \quad (1)$$

$\forall v_i^b$ and $v_i^{in} \in V_V \mid M(i,t) \geq 1$ and $L(i,t) \geq 2$,

$$C\ (v_i^b, v_i^{in}) = L(i,t) - 1. \quad (2)$$

$\forall v_i^{in}$ and $v_i^b \in V_V \mid L(i,t) = 0$,

$$C\ (v_i^{in}, v_i^b) = 1. \quad (3)$$

[1] In practice an edge with capacity 0 need not be *specifically* added. We do so to retain the symmetricity in the virtual graph construction.

The final step is to incorporate the neighbor relation that holds in the original deployment field into the virtual graph. Recall that the paths between the regions are determined by the flip distance (F). In the virtual graph, an edge of infinite capacity (denoted by inf) is added from v_i^{out} to v_j^{in}, and another edge of infinite capacity is added from v_j^{out} to v_i^{in} if regions i and j are neighbors (infinity is denoted as *inf* in the virtual graph in Figure 3). This is to allow any number of flips between neighbor sensors, if there are mobile sensors. For regions 1 and 2 that are neighbors, in Figure 3, edges with infinite capacity are added from the v_1^{out} to v_2^{in}, and from v_2^{out} to v_1^{in}. Formally, for all regions i and j that are neighbors in the sensor network, we have

$$C\ (v_i^{out}, v_j^{in}) = C\ (v_j^{out}, v_i^{in}) = inf. \tag{4}$$

Having discussed the capacity among edges, we now incorporate costs for each flow in G_V. From equation (4), we can see that the flips between regions (say i and j) in the sensor network is translated in G_V by an edge from v_i^{out} to v_j^{in}, and from v_j^{out} to v_i^{in}. Each flip between neighboring regions in the sensor network incurs a cost of 1. In order to capture the number of flips between these regions, we add a cost value to these corresponding edges in G_V, with cost value equal to 1. Let us denote $Cost(i, j)$ as the cost for a flip between vertices i and j. Formally, for all regions i and j that are neighbors in the sensor network, we have

$$Cost\ (v_i^{out}, v_j^{in}) = Cost\ (v_j^{out}, v_i^{in}) = 1. \tag{5}$$

The cost for all other edges in G_V is 0. An instance of the original deployment and the corresponding virtual graph at the start are shown in Figures 4 (a) and (b) respectively. In Figure 4 (a) the numbers denote id of the corresponding region [2].

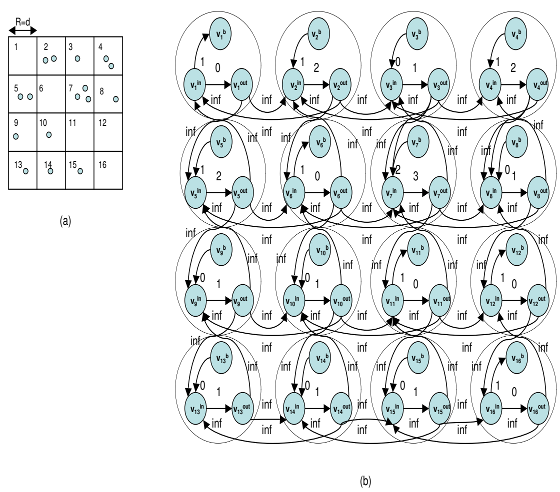

Fig. 4. The initial sensor network deployment (a) and the corresponding virtual graph at the start (b) in Case $R = d$

Construction when $F = d$ and region size $R > d$: In this case, the flip distance F is equal to basic unit d. The desired region size $R > d$ (sensors with larger sensing ranges). We first discuss the case where the region size is an integral multiple of the basic unit of flip distance (i.e., $R = x * d$, where x is an integer) [3]. For instance, if $x = 2$, then this requirement translates to maximizing number of regions (of size $2d$) with at least 1 sensor. This is shown in Figure 5 (a), where the region ($R = 2d$) is the area contained within dark borders. We call each area within the shaded lines as sub-regions. Each sub-region has a size d. There are thus 4 regions and 16 sub-regions in Figure 5. The id of the regions is the number in bold at the center of the corresponding region. For ease of understanding, we still keep the id of the sub-regions in Figure 5. To explain the construction process better, we say that a region i represents its sub-regions. For instance in Figure 5, region 1 is a representative of sub-regions 1, 2, 5 and 6.

The principle of constructing G_V is unchanged. In G_V now, each region (of size R), and whose id is i, is denoted as v_i^b. For each region (of size R), we are interested in how many sensors from other sub-regions have flipped to it. Despite covering multiple sub-regions, we are interested in coverage of the region in itself (and not the sub-regions). Thus, we still need only one in vertex (v_i^{in}) for each region. However, each region has multiple sub-regions, and sensors in them can be pushed (if they are mobile) or absorbed (if neighbor sub-regions have mobile sensors). In Figure 5 (b), the region size is $2d$, and there are four out vertices for each region.

We now discuss how edges are added between vertices in G_V. For each region i that has ≥ 1 sensors, it is a source region, and an edge is added from v_i^b to v_i^{in} with edge capacity equal $\sum_{j=1}^{x^2} n_j - 1$ in the virtual graph. For example in Figure 5, region 1 is a source, and there is an edge with capacity $\sum_{j=1}^{4} n_j - 1 = 3$ from vertex v_1^b to v_1^{in}. The interpretation of this edge capacity is still the same as in the previous case. While determining the flow, we want to ensure at least one sensor remains in this region. Then edges with capacity n_k are added from this v_i^{in} to each $v_i^{out\ k}$ as shown in Figure 5, where k denotes the id of sub-region represented by the corresponding out vertex. For each hole j, we add an edge with capacity 1 from v_j^{in} vertex to v_j^b, an edge of 0 capacity from v_j^{in} to each $v_j^{out\ k}$ in the virtual graph. Finally, to incorporate the neighbor relationship between regions, an edge of infinite capacity is added from each $v_i^{out\ m}$ to v_j^{in}, and an edge of infinite capacity is added from each $v_j^{out\ k}$ to v_i^{in} if regions i and j are neighbors *and* sub-region m is a neighbor of region j and sub-region k is a neighbor of region i as shown in Figure 5. We formally do not define capacities ($C(i, j)$) and costs ($Cost(i, j)$) between edges in this case, since they can be easily obtained following from equations (1) to (5).

Construction when $F > d$: We now discuss the case, where the flip distance is larger than the basic unit d. In this model, sensors can still flip only once, but the distance of flips is an integral multiple of the basic unit d ($i \times d$, where i is $1, 2, 3 \ldots n$). That is, the sensor can flip only once, but its flip distance can be one of d, $2d$, $3d$, $\ldots$,nd. We denote the flip distance F in this case as $n \times d$. In this case, sensors can flip

[2] We do not show the cost values between neighboring regions in the virtual graphs.

[3] Extensions to handle any arbitrary region size can be found in [17]

beyond their immediate neighboring regions. To handle this case, edges of infinite capacity are added from *out* vertices to all *reachable in* vertices from it. For instance in Figure 4, if $F = 2d$, then the reachable in vertices for v_1^{out} are not only v_2^{in}, v_5^{in}, but also v_3^{in} and v_9^{in}. Other vertices, edges and their capacities follow the construction principles discussed in the preceding cases. Once again, the capacities ($C(i,j)$) and costs ($Cost(i,j)$) between edges in this case can be obtained following from equations (1) to (5).

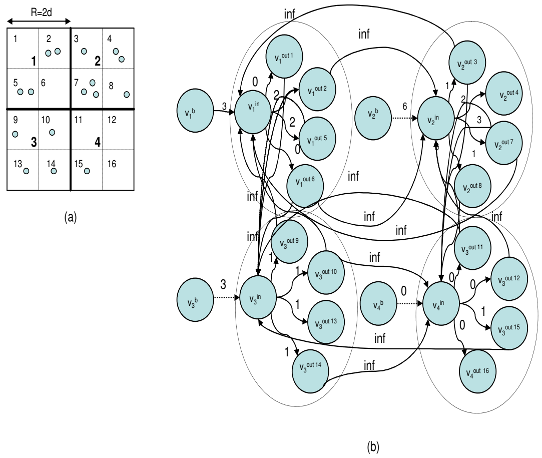

Fig. 5. The initial sensor network deployment (a) and the corresponding virtual graph at the start (b) in Case $R > d$

C. Determining the optimal movement plan from the virtual graph

Having constructed G_V, the Base-station determines minimum-cost maximum-flow between Source vertices and Hole vertices. The source vertices are the base vertices in G_V corresponding to source regions in the sensor network. The Hole vertices are the base vertices in G_V corresponding to holes in the sensor network. We determine the minimum cost maximum flow between the source vertices and hole vertices as follows. We first determine the value of the maximum flow in G_V from all Source vertices to Hole vertices using the Edmonds-Karp algorithm [18]. We then use the implementation in [19] to get the minimum cost flow. The implementation uses the maximum flow value computed above. The implementation in [19] is that of the successive approximation cost scaling algorithm [20] to determine minimum cost flow. It works by starting to find an approximate solution and then iteratively improving the current solution. For more details, readers can refer to [19] and [20]. After this, we determine the set of flows between individual source-hole pairs in G_V that results in minimized cost, with maximum flow.

Let W^V denote the flow plan (a set of flows) returned after executing the minimum-cost maximum-flow algorithm on G_V, where the amount of each flow is 1. Each flow $w_{i,j}^V \in W^V$ is a flow from v_i^b to v_j^b in G_V, and is of the form $\langle v_i^b, v_i^{in}, v_i^{out}, v_k^{in}, v_k^{out}, v_l^{in}, v_l^{out} \ldots, v_n^{in}, v_n^{out}, v_j^{in}, v_j^b \rangle$, which denotes that the path of the flow is from v_i^b to v_i^{in}, from v_i^{in} to v_i^{out} …from v_j^{in} to v_j^b. Recall from the construction of G_V from G_S, that the neighboring relation between regions i and j in G_S is translated in G_V as edges from v_i^{in} to v_j^{out}, and from v_j^{in} to v_i^{out}. Thus, for the flow plan W^V, we can map it to a corresponding flip plan W^S (set of flip sequences) in G_S. Each $w_{i,j}^S \in W^S$ is a sequence of flips in the sensor network between regions i and j. That is for each $w_{i,j}^V$ ($\in W^V$) of the form $\langle v_i^b, v_i^{in}, v_i^{out}, v_k^{in}, v_k^{out}, v_l^{in}, v_l^{out} \ldots, v_n^{in}, v_n^{out}, v_j^{in}, v_j^b \rangle$, the corresponding $w_{i\ j}^S$ ($\in W^S$) is of the form $\langle r_i, r_k, r_l \ldots r_n, r_j \rangle$, where r_i, r_k, r_l … r_n, r_j correspond to regions i, k, l, … n, j in the sensor network respectively. Physically, this means that one sensor should flip from regions i to k, k to l, … n to j. The sensor flip plan (also called movement plan) W^S is the output of our solution.

D. Properties of our Solution

Before discussing the properties of our solution, we introduce the concept of feasible flow first. We call a flow $w_{i,j}^V$ of the form $\langle v_i^b, v_i^{in}, v_i^{out}, v_k^{in}, v_k^{out}, v_l^{in}, v_l^{out} \ldots, v_n^{in}, v_n^{out}, v_j^{in}, v_j^b \rangle$ as feasible at time t in G_V, if there exists positive edge capacities between v_i^b and v_i^{in}, v_i^{in} and v_i^{out}, v_i^{out} and v_k^{in} $\ldots v_j^{in}$ and v_j^b at time t. We call a flip sequence $w_{i,j}^S$ of the form $\langle r_i, r_k, \ldots, \ r_n, r_j \rangle$ as feasible at time t in G_S, if there is at least one mobile sensor in each of regions $i, k, \ldots$ and n, and source region i has at least 2 sensors at time t. We now have the following Lemma.

Lemma 1: A flow $w_{i,j}^V$ in G_V is feasible iff the corresponding flip sequence $w_{i,j}^S$ is feasible in G_S (For proof, please refer to [17]).

Using Lemma 1, we obtain the following corollary.

Corollary 1: For a feasible flow set W_*^V in G_V, a corresponding feasible flip sequence set W_*^S can be found in G_S and vice versa (For proof, please refer to [17]).

The optimality of our solution in maximizing coverage and number of flips is formally stated in the following Theorem.

Theorem 1: Let W_{opt}^V be the minimum-cost maximum-flow plan in G_V. Its corresponding flip plan W_{opt}^S will maximize coverage and minimize the number of flips (For proof, please refer to [17]).

We now discuss the time complexity of our solution. There are three phases in our solution while determining the optimal movement plan. The first is the virtual graph construction, the second is determining the maximum flow, and the third is determining the minimum-cost flow. The time complexity is dominated by determining the maximum flow and determining the minimum cost flow. The resulting time complexity using our implementations is $O(max\ (|V||E|^2, |V|^2|E|log|V|))$, where $|E| = O((\lceil \frac{F}{d} \rceil)(\lceil \frac{R}{d} \rceil)(\lceil \frac{D}{R} \rceil^2))$ and $|V| = O((\lceil \frac{D}{R} \rceil^2)(\lceil \frac{R}{d} \rceil)^2)$.

E. Discussions

Executing our Solutions: Our solution requires information on the number of sensors in each region in the network. In the above, we proposed to let a Base station collect this information, and determine a movement plan. We now discuss distributed approaches for our problem, inspired in part by

Open Shortest Path First (OSPF) algorithm in determining shortest paths in a network. In this approach, sensors in the network share the information about the number of sensors in regions and execute our solution. In the extreme case, each sensor executes our solution independently, and moves accordingly after intra-region synchronization. Alternatively, a set of regions can form an area and the region information in each area is exchanged among all areas. Our solution can be executed independently in each area to determine the movement plan. This will help reduce the overall computational complexity as computation is done across areas than in individual regions. This approach in principle is similar to the OSPF approach, where the link-state information is first obtained, and the routing algorithm is executed independently by routers. An alternate distributed approach following from above is to let each area to obtain region information *only* in their area. Each area can then execute our solution only with this information (without exchanging information with other areas). While this approach cannot guarantee global optima, it can reduce the computational and messaging complexity.

Network Partitions: In some situations, sensors in one part of the network may not be able to communicate with sensors in another part during information exchange, forming a partition. In the approach proposed by Wu and Wang [2] to repair partitions, empty holes are filled by placing a *seed* from a non-empty region to a hole. The algorithms to place seeds are tuned to meet load balancing objectives. We can apply the algorithms in [2] to handle partitions in our case. However, we are still constrained by the mobility in the sensors. Addressing this issue part of our future work.

Multiple Flips and Arbitrary Flip Directions: Our solution can be extended to handle multiple flips, and arbitrary flip directions. In the virtual graph, we now add edges from a source region to all *reachable* regions. The reachable regions change now to reflect multiple flips, and arbitrary flip directions respectively.

V. Performance Analysis

In this section, we study the sensitivity of coverage and the number of flips to flip distance under different initial deployment scenarios and coverage requirements.

A. Metrics and Evaluation Environment

Let the total number of regions in the network be Q. We denote Q_o as the number of regions with at least 1 sensor after the movement plan given by our solution is executed, and Q_i as the number of regions with at least one sensor at initial deployment. We define coverage improvement (CI) as the improvement in coverage as a result of our solution compared when compared to the initial deployment. We then have,

$$CI = Q_o - Q_i. \tag{6}$$

We define the Flip demand as the number of flips required per unit increase in coverage. Denoting J as the optimal number of flips as determined by our solution , we define Flip Demand FD as,

$$FD = \frac{J}{Q_o - Q_i}. \tag{7}$$

Our default network is a field of size 150×150 units. The default region size $R = 10$ units. We vary the flip distance F from 10 units to 40 units. The number of sensors deployed is equal to the number of regions. All data reported here were collected across 10 iterations, and averaged. Our implementations of the maximum flow algorithm is the Edmonds-Karp algorithm [18], and minimum cost flow algorithm is the one in [19]. We conducted our experiments using MATLAB. We used a topology generator for 2D-Normal distribution. The X and Y co-ordinates are independent of each other ($\sigma_x = \sigma_y$). We use $\sigma = \frac{1}{\sigma_x^2}$ to denote the degree of concentration of deployment in the center of the network field. Thus, larger values for σ implies more concentrated deployment in the center of the field. When $\sigma = 0$, the deployment is uniform.

B. Our Performance Results

Figure 6 shows how the flip distance F impacts coverage improvement (CI), under different region sizes (R). Here we set $\sigma = 1$. In order to study the sensitivity of CI to region size fairly, the number of regions for different regions sizes should be the same. In Figure 6, we conduct our experiments on two different field sizes to study sensitivity of R. When $R = 10$, the field size is the default one (150×150). When $R = 20$, the field size was set as (300×300). Thus, the number of regions in both cases is $15 \times 15 = 225$. We observe that when flip distance (F) increases, CI is consistently better irrespective of R. Increases in flip distances, enable our solution to exploit more choices while striving for optimal coverage. Consequently, CI increases. The second observation we make from Figure 6 is that as R increases, CI decreases. This is because, when R is small, neighboring regions are closer to each other (irrespective of the number of regions). When R is less, and for the same flip distance of sensors, it is likely to find sensors from nearby regions that can flip to fill holes. However, when R is large, the sensors that can flip from one region to another have to be relatively close to the borders of the region. Thus, the number of sensors that can be found can find to flip are less. Naturally CI (which captures improvement) decreases when R is large. Thus, performance improvement due to increases in flip distance is constrained by region size.

Figure 7 shows how the flip distance F impacts CI under different distributions in initial deployment. Here we set our default field size (150×150) and region size ($R = 10$). We change σ from 0 (uniform distribution) to 4 (highly concentrated at the center of the field). The first observation we make here is that increases in flip distance (F) increases CI. However, the degree of increase in CI is impacted by σ. When $\sigma = 0$ (uniform), CI is almost the same for all values of F. This is because, in our simulations, close to full coverage is achieved when $\sigma = 0$. Since the initial deployment is the same for all cases, the degree of improvement is the same.

We now study the trade-off between F and σ. Figure 7 shows the gradual shift of F dominating over σ. In Figure 7,

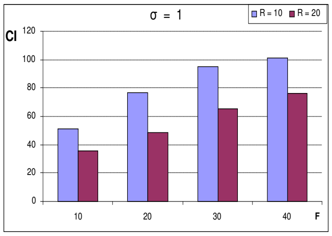

Fig. 6. Sensitivity of CI to F when R varies

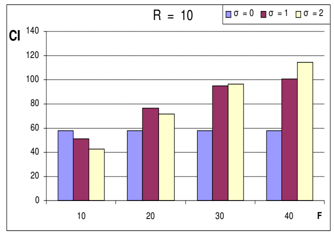

Fig. 7. Sensitivity of CI to F when σ varies

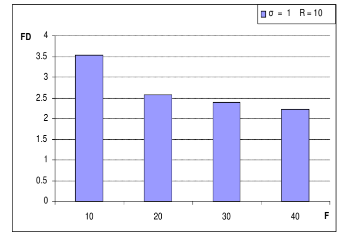

Fig. 8. Sensitivity of FD to F

when $F = 10$, σ has a dominating effect compared to F. We can see that as σ increases (bias increases), CI deceases. The increase in bias cannot be compensated using sensors with flip distance of only 10 units. However, this trend appears different when F increases. When F increases, our solution can exploit more choices. Thus F dominates when it increases. However, the degree of domination still depends on σ. This can be seen when $F > 10$ in Figure 7. When $F = 20$, $\sigma = 1$ performs better that $\sigma = 0$. Here, the increase in bias can be compensated better when $F = 20$ (than when $F = 10$). Thus CI increases. However, increasing σ beyond this point makes the bias dominate and consequently CI decreases when $F = 20$ and $\sigma > 1$. When $F > 20$, the increase in flip distance consistently dominates the increase in bias (although the degree of domination is different), showing that performance improvement due to increases in flip distance is constrained by initial deployment distribution.

Figure 8 shows how flip distances impacts FD in a 150×150 network with $R = 10$ and $\sigma = 0$. Since FD is a ratio of number of flips per unit increase in coverage, we cannot capture its essence when we compare FD in cases where the coverage is different. Hence, we illustrate one case ($\sigma = 0$ and $R = 10$), where the final deployment covers all regions. Since the initial distribution is the same, CI is the same. The comparison becomes more meaningful. From Figure 8, we can see as F increases, FD decreases for the same deployment and region size. This is because, when F is small, in order to achieve optimality, there may be multiple flips from sensors farther away from a hole (although the number of flips required for optimality is still minimum). As F increases, it is likely that far away sensors can flip to this hole, consequently minimizing the required number of flips.

VI. Final Remarks

In this paper, we proposed a minimum-cost maximum-flow based solution to optimize sensor network deployment using flip-based sensors. We then studied the sensitivity of performance to flip distance, under different initial deployment scenarios. In this paper, we considered flips in increments of a basic unit (d). Our current work is on relaxing this in order to handle continuous mobility, although the overall movement distance (F) is still limited.

References

[1] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in *Proceedings of International Symposium on Distributed Autonomous Robotics Systems (DARS)*, Fukupka, Japan, June 2002.

[2] J. Wu and S. Wang, "Smart: A scan-based movement-assisted deployment method in wireless sensor networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Miami, March 2005.

[3] G. Wang, G. Cao, and T. La Porta, "Movement-assisted sensor deployment," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Hong Kong, March 2004.

[4] G. Wang, G. Cao, T. La Porta, and W. Zhang, "Sensor relocation in mobile networks," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Miami, March 2005.

[5] W. Li, P. Wan, Y. Wang, and C. Yi, "Fault tolerant deployment and topology control in wireless networks," in *Proceedings of ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, Annapolis, June 2003.

[6] Y. Zou and K. Chakrabarty, "Uncertainty-aware sensor deployment algorithms for surveillance applications," in *Proceedings of IEEE Global Communications Conference (GLOBECOM)*, San Francisco, CA, December 2003.

[7] C. Wang, L. Xiao, and J. Rong, "Sensor localization in an obstructed environment," in *Proceedings of IEEE/ ACM Intl Conference on Distributed Computing in Sensor Systems (DCOSS)*, Marina del Rey, June 2005.

[8] "http://www.darpa.mil/ato/programs/shm/index.html," .

[9] S. Kumar, T. H. Lai, , and J. Balogh, "On k-coverage in a mostly sleeping sensor network," in *Proceedings of ACM International Conference on Mobile Computing and Networking (MOBICOM)*, Philadelphia, September 2004.

[10] S. Shakkottai, R. Srikant, , and N. B. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, San francisco, April 2003.

[11] W. Du, L. Fang, and P. Ning, "Lad: Localization anomaly detection for wireless sensor networks,," in *Proceedings of IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, Rhodes Island, April 2005.

[12] V. Isler, K. Daniilidis, and S. Kannan, "Sampling based sensor-network deployment," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sendai, Japan, September 2004.

[13] N. Bulusu, J. Heidemann, and D. Estrin, "Adaptive beacon placement," in *Proceedings of IEEE International Conference on Distributed Computing Systems (ICDCS)*, Phoenix, AZ, April 2001.

[14] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. Saluja, "Sensor deployment strategy for target detection," in *Proceedings of ACM international conference on Wireless Sensor Networks and Applications (WSNA)*, Atlanta, September 2002.

[15] A. Howard, M. J. Mataric, and G. S. Sukhatme, "An incremental self-deployment algorithm for mobile sensor networks," in *Autonomous Robots, Special Issue on Intelligent Embedded Systems*, September 2002.

[16] A. Howard, M. J. Mataric, and G. S. Sukhatme, "Relaxation on a mesh: a formation for generalized localization," in *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Maui, November 2001.

[17] S. Chellappan, X. Bai, B. Ma, and D. Xuan, "Mobility limited flip-based sensor networks deployment," Tech. Rep., Dept. of Computer Science and Engineering, The Ohio-State University, August 2005.

[18] T. Cormen, C. Leiserson, R. Rivest, and C. Stein, "Introduction to algorithms," in *MIT Press*, 2001.

[19] A. V. Goldberg, "An efficient implementation of a scaling minimum-cost flow algorithm," in *J. Algorithms 22*, 1997.

[20] A. V. Goldberg and R. Tarjan, "Solving minimum-cost flow problems by successive approximation," in *Proceedings of ACM Symposium on Theory of Computing (STOC)*, New York, May 1987.