Mobility Limited Flip-based Sensor Networks Deployment

Sriram Chellappan, Xiaole Bai, Bin Ma and Dong Xuan

Abstract

An important phase of sensor networks operation is deployment of sensors in the field of interest. Critical goals during sensor networks deployment include coverage, connectivity, load balancing etc. A class of work has recently appeared, where mobility in sensors is leveraged to meet deployment objectives. In this paper, we study deployment of sensor networks using mobile sensors. 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. We call such sensors as *flip-based* sensors. Given an initial deployment of flip-based 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.

Index Terms

Sensor Networks Deployment, Limited Mobility, *Flip*-based Sensors.

I. Introduction

Sensor networks deployment in an important phase of sensor network operation. A host of works has appeared in this realm in the recent past [1], [2], [3], [4], [5], [6], [7]. One of the important goals of sensor

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.

networks deployment is to ensure that the sensors in the network meet critical network objectives that may include coverage, connectivity, load balancing etc. When a number of sensors are to be deployed, it is not practical to manually position sensors in desired locations. In many situations, the sensors are deployed from a remote site (like from an airplane) that makes it very hard to control deployment.

To address this issue, a class of work has recently appeared where mobility of sensors is taken advantage of to achieve desired deployment [1], [2], [4], [8]. Typically in such works, the sensors detect lack of desired deployment objectives. The sensors then estimate new locations, and move to the resulting locations. 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 limitation in the movement distance.

In practice however, it is quite likely that the mobility of sensors is constrained. Towards this extent, a class of Intelligent Mobile Land Mine Units (IMLM) [9] to be deployed across battlefields have been developed by DARPA. The IMLM units are expected to detect breaches, and move in order to repair them. However the mobility of the IMLM units is constrained. Briefly, the mobility system in [9] 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/ propeller system. For each hop, the fuel is metered into the combustion chamber and ignited to propel the IMLM unit into the air. The hop distance is limited, depending on the amount of fuel and the propeller dynamics. The units include a righting system to properly orient itself after landing, and a steering system that provides directional control for each hop. Other technologies can also assist in such mobilities, like sensors enabled with spring actuation, external agents launching sensors after being deployed in the field etc. Such a model typically trades-off mobility with energy consumption and cost. 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. We call such sensors as *flip-based* sensors. 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 an optimal movement plan for 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.

The above problem is not easy to solve. The first challenge is to resolve inherent conflicts that may be present in optimizing more than one objective at the same time. In our problem, these objectives are maximizing coverage and simultaneously minimizing the number of flips. Any step that attempts to achieve one objective may in fact compromise the other. For instance, a flip to repair a nearby hole, may reduce the total number of flips, but it may be at the cost of compromising coverage maximization and vice versa. Such conflicts have to be resolved. The second challenge arises due to limited mobilities. When sensors are limitedly mobile, not only can they not make independent decisions to move, but they are also not independent during their motion. What we argue is that; it is not always the case that only *one* sensor needs to move to repair a hole. Due to limited mobility, a set of sensors may need to move for repairing a particular hole. A flip may need to start from a far away point, and a set of sensors may need to sequentially flip (like a chain) to cover the particular hole. This challenge does not arise if sensors are unlimitedly mobile.

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 a movement plan (flip sequences) for the sensors in the network. We prove that our solution maximizes coverage, while it simultaneously minimizes the total number of sensor flips required. We also propose multiple approaches that sensors can adopt to execute our solution in practice. We then perform simulations to 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.

Deployment using mobile sensors has been addressed before. However, sensors with unlimited mobilities may not always be feasible. As argued before, it is very likely that practical considerations limit the mobility in sensors. The significance of our work is that we define a practical flip-based mobility constrained model for sensors. Using such flip-based sensors, we address an important deployment problem, namely coverage maximization while simultaneously minimizing the number of flips required, which is a critical deployment objective in many applications.

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

II. Mobility Model and Problem Definition

A. The Flip-based Sensor Mobility model

In this paper, we model sensor mobilities as a *flip*. That is, the motion of the sensor is in the form of a flip (or jump) from its current location to a new one 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 [9], coiled springs that unwinds for flipping, external agents launching sensors after being deployed in the field etc.

In our model, sensors can flip only once to a new location. This could be due to propeller dynamics, or the spring unable to recoil after a flip, or the external agent launching the sensor. The distance to which a sensor can flip is limited. The sensor can flip in a desired angle. Mechanisms in [9] can be used for orientation prior to a flip. The limitation in the mobility of flip-based sensors comes from the bound on the maximum distance they can move, which again depends on the available fuel quantity, degree of spring coil etc. We study two models of flip-based mobility in this context. The first one is a fixed distance mobility model, while the second is a variable distance mobility model.

We denote the maximum distance a sensor can flip to as F. In the first model, the distance to which a sensor can flip is fixed and is equal to F. We extend the above model further. Although the number of flips is still one, in many cases, depending on the triggering signals, fuel can be metered variably, or the spring can unwind only partially, or the external agent can variably adjust the flip distance during its launching. Thus in the second model, sensors can flip to distances between 0 and F. We denote d as the basic unit of distance flipped. We assume that F is an integral multiple of the basic unit d. Thus in the second model, sensors can flip once to distances d, $2d$, $3d$, $\ldots nd$ from its current location, where $nd = F$. To clearly differentiate the above two models, we introduce the notation C to denote *choice* for flip distance. When $C = 1$, we denote the first model, where the sensor has only one fixed choice for distance of flip (the maximum distance F). When $C = n$, we denote the second model, where the sensor has n choices for the distance of flip (between d and maximum distance F). For the rest of the paper, unless otherwise clearly specified, the term flip distance denotes the maximum flip distance F.

B. Problem Definition

We address a deployment problem in this paper. The sensor network we study is a rectangular field. It is partitioned 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. Such regions that are not covered by any sensor are *holes*. 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 maximum 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 one 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 sensing/ transmission ranges of sensors, 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., $R = m * d$, where m is an integer (≥ 1). We discuss the general case of R subsequently in Section III-F. We assume that each sensor can know which region it resides it. To do so, the methods proposed in [10] can be applied, where location of sensors is determined by using sensors themselves as landmarks. In our model, the regions to which a sensor can flip, are those in its left, right, top and bottom directions. However, those regions need not be just the neighbors. They in fact depend on the flip distance F. After discussing the above case, the general case, where a sensor can flip to any arbitrary direction is discussed subsequently in Section III-F.

The significance of our problem is that we maximize coverage in the sensor network, while simultaneously minimize the number of flips using only limited mobility sensors. We believe that our flip-based model is representative and practical. Using such sensors, we minimize the number of holes present after an initial deployment. In many applications presence of holes seriously impairs functionality from the perspective of coverage, connectivity etc. For instance, in intruder tracking applications, the presence of holes can in fact disrupt the entire network operation. The importance of minimizing the number of flips is in preserving the mobility of sensors as much as possible. Preserving the mobility will be beneficial for a variety of reasons at later stages of sensor network operation. This is especially important, when the sensors themselves are limited in mobility.

We illustrate our problem further with an example. Figure 1 (a) shows an instance of initial deployment in the sensor network. The shaded circles denote sensors, and the numbers denote the id of the corresponding region in the network. 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.

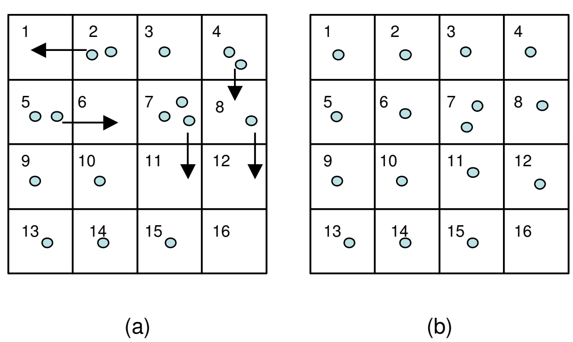

Fig. 1. A snapshot of the sensor network and a movement plan to maximize coverage (a), and the resulting deployment (b)

The above problem is not easy to solve. For instance, consider Figure 1 (a). For ease of elucidation, let the desired 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 one 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, the fact that there are no mobile sensors in regions 11, 12 and 15 mean that all paths to region 16 are *blocked*, 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 coverage, 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).

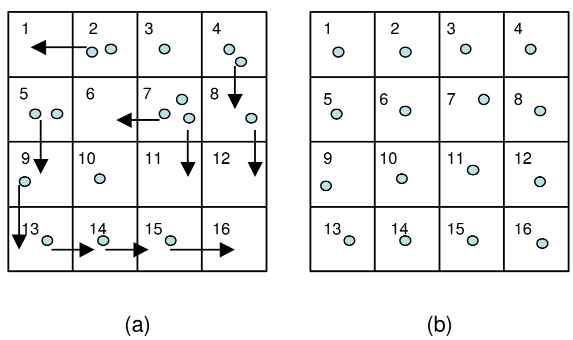

Fig. 2. A snapshot of the sensor network and the optimal movement plan (a), and the resulting deployment (b)

C. Challenges

There are two key challenges to our problem. The first challenge is due to our objective of simultaneously optimizing coverage and the number of flips. Consider the movement plan in Figure 1 (a). Region 7 that has 3 sensors in it, wishes to cover region 16 with a sensor. The intuition as pointed out earlier is that region 7 is the closest one that has extra sensors to cover 16. But this plan, that attempted to minimize the number of flips cannot cover region 16. There is thus a conflict that may be present in maximizing coverage, while trying to minimize the number of flips using local information. For optimum coverage, the path to cover region 16 starts from region 5 (in Figure 2). This path is longer than the one from region 7, but this in fact is the path that can cover all regions.

The second challenge arises due to limited mobility in the sensors. Due to limited mobility, if a sensor in one region wishes to cover a sensor in another region, then there must be one or more mobile sensors in all regions that are in the path from the source to the destination. If there is no mobile sensor in some intermediate region in the path, no sensor can flip beyond that region, resulting in the path being blocked. In Figure 1 (b), although region 7 still has an extra mobile sensor, all paths from region 7 to region 16 are blocked, as there is no mobile sensor in regions 11, 12 and 15, each of which are on path(s) from region 7 to 16. Therefore, region 16 cannot be covered. The challenge is in determining the best paths from sources to holes, such that mobile sensors exist in all intermediate regions. For optimal coverage, such a path may traverse long number of intermediate regions as shown in Figure 2 (a), where the path from region 5 to 16 traverses 4 intermediate regions. Determining such a path cannot be done if regions just posses information about their local neighborhood. They have to share local region information with sensors in other regions. A related challenge due to limited mobility is that, once a sensor flips, its movement cannot be *undone*. It is therefore preferable for sensors to make a movement plan (which sensors should move, and where) prior to their flip.

III. 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 III-E.

We now discuss how to translate our problem into a minimum-cost maximum-flow problem. Let us call regions with at least one mobile sensor as a *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 objectives and limited mobilities, then our problem can be translated as determining how to maximize the flow of sensors from source regions to holes, while minimizing the cost of the flow, without violating the path constraints between sources and holes.

If we identify regions (sources or holes) using vertices, and incorporate path 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 call our solution as minimum-cost maximum-flow solution.

We have so far pointed out the core features of our solution. 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 flow along the edges in the virtual graph is then appropriately translated as a movement plan (flip sequences) for sensors in the network. The plan indicates which sensor should move and where.

In Section III-B, we first introduce the virtual graph, the principles in its construction, and some important notations. We then discuss in detail how to construct the virtual graph for a simple, yet representative basic case. We then discuss how to construct the virtual graph for general case. In Section III-C, we explain how to translate the results from our minimum-cost maximum-flow solution into a movement plan (flip sequences) for sensors, in order to optimize coverage and the number of flips in the sensor network. We discuss important properties of our solution (including optimality proofs and time-complexity) in Section III-D. We discuss alternate approaches to execute our solution in Section III-E, and provide some discussions in Section III-F.

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 $G_S(V_S, E_S)$ be an undirected graph representing the sensor network. Each vertex in $\in V_S$ represents one region in the sensor network and each edge $\in E_S$ represents the path 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 how edges should be added, their direction, their capacities and costs. For any region i in the sensor network, we denote its *reachable* regions as those to which a sensor from region i can flip to. Obviously, the reachable regions depend on the flip distance F. In the virtual graph, edges are added between such reachable regions. The direction of edges between vertices are based on whether the corresponding regions are sources or holes in the sensor network. The capacities of the edges depends on the number of sensors in the regions, while the cost is used to quantify the number of sensor flips between regions in the sensor network. 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 basic 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 at time $t = 0$. 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.

1) Constructing the virtual graph for a basic case: In this case, the region size R is equal to the basic unit of flip distance d. To explain the virtual graph construction process clear, we first describe it for the case where the flip distance $F = d$, and $C = 1$. That is, the flip distance is the basic unit d and the sensor has only one choice for flip distance. We discuss the case where $F > d$ and $C = 1$, and $F > d$ and $C = n$ (multiple choices) subsequently. The case where $R > d$ is discussed in Section III-B.2.

a) Construction when $F = d$ and $C = 1$: 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}.

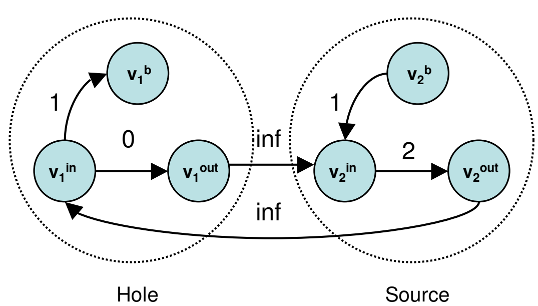

Fig. 3. The Virtual Graph with only regions 1 and 2 in it

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 mobile 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 one 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 from it. 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 one 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 $v_i^{out} \in V_V$,

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

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

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

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

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

[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 *reachable* relationship that holds in the original deployment field into the virtual graph. Recall that for any region i in the sensor network, its *reachable* regions as those regions to which a sensor from region i can flip to, which is determined by the flip distance (F). We have to incorporate this in the virtual graph. To do so, 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 reachable from each other. This is to allow any number of flips between reachable regions, if there are mobile sensors in them. In Figure 3, regions 1 and 2 are reachable from each other since $R = d$ and $F = d$. Thus, 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 reachable from each other 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. If a flip occurs from some region i to some region j in the sensor network, then a penalty of one is incurred. We want to minimize the overall penalty (or cost of flips), while simultaneously maximizing the number of regions with at least one sensor. From equation (4), we can see that the flips between reachable 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}. 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 reachable from each other in the sensor network, we thus have

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

The $Cost$ for other edges in G_V is 0. This is because, from the view of the sensor network, the edges apart from those between *out* and *in* vertices across regions, are internal to a region. They cannot be counted towards sensor flips (which only occurs across regions). Instances of such edges are those from v_1^{in} to v_1^b, from v_1^{in} to v_1^{out} in Figure 3. Consequently, the $Cost$ of such edges in G_V is 0. An instance of 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 the id of the corresponding region. We do not show the $Cost$ values in the virtual graph in Figure 4 (b).

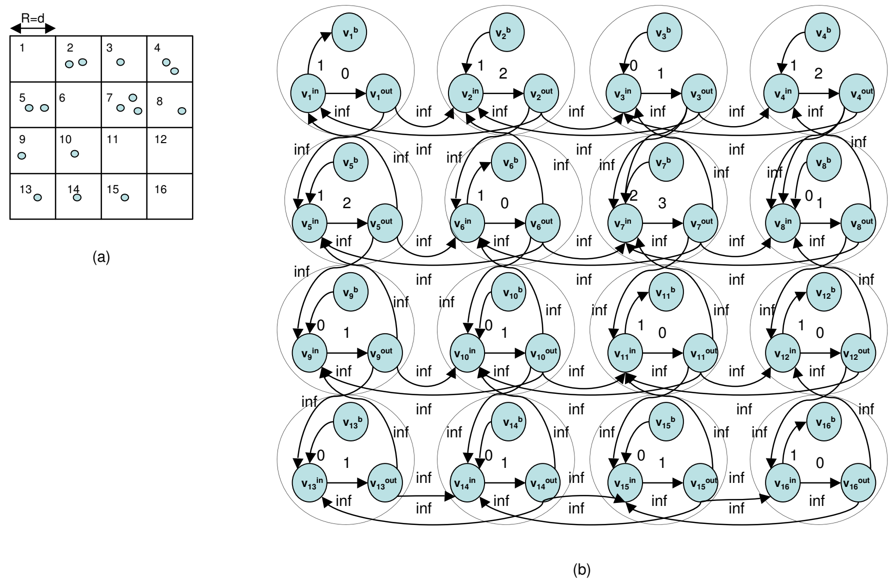

Fig. 4. The initial sensor network deployment (a) and the corresponding virtual graph at the start (b) in Case $R = d$

b) Construction when $F > d$ and $C = 1$: In the above, we discussed the virtual graph construction for the case where $R = d$, $F = d$ and $C = 1$. We now discuss the case where for $R = d$, the flip distance $F > d$ and $C = 1$. The construction methodology is still the same; only the *reachability* relationship changes. Since $R = d$, $F > d$ and $c = 1$ here, immediate neighboring regions are not reachable anymore from each other like in the above case, where $F = d$. When $F > d$, regions beyond immediate neighboring regions become reachable (depending on F). We have to incorporate the new reachability relationship in the virtual graph. To do so, edges of infinite capacity are added from v_i^{out} to v_j^{in}, and from v_j^{out} to v_i^{in} if regions i and j have are reachable from each other. For example, if $F = 2d$ and $C = 1$ in Figure 4, then regions 5 and 2 are not reachable from region 1. The new reachable regions for region 1 are regions 3 and 9. Thus the edges between v_1^{out} to v_2^{in} and v_5^{in}, from v_2^{out} to v_1^{in}, and from v_5^{out} to v_1^{in} are removed, and edges of infinite capacity are added from v_1^{out} to v_3^{in} and v_9^{in}, from v_3^{out} to v_1^{in}, and from v_9^{out} to v_1^{in} to reflect the new reachability relationship. Other edges in the virtual graph are treated similarly. The capacities ($C(i, j)$) and costs ($Cost(i, j)$) between edges in this case can be obtained following from equations (1) to (5).

c) Construction when $F > d$ and $C = n$: We now discuss the virtual graph construction for the case where for $R = d$, the flip distance $F > d$ and $C = n$. Here again, only the reachability relationship changes. Recall that if $F > d$ and $C = n$, the distance of flip can be d, $2d$, $3d$, $\ldots nd$, where $F = nd$. Thus, reachable regions for some region, say region i are those regions that are immediate neighbors of region i and beyond (depending on F). We have to incorporate this new reachability relationship in the virtual graph. For example if $F = 2d$ and $C = 2$ in Figure 4, then the reachable regions of region 1 are regions 2, 3, 5 and 9. Thus, apart from edges from v_1^{out} to v_2^{in} and v_5^{in}, from v_2^{out} to v_1^{in}, and from v_5^{out} to v_1^{in}, edges are also added from v_1^{out} to v_3^{in} and v_9^{in}, from v_3^{out} to v_1^{in}, and from v_9^{out} to v_1^{in} to reflect the new reachability relationship. Other edges in the virtual graph are treated similarly. The capacities ($C(i, j)$) and costs ($Cost(i, j)$) between edges in this case can be obtained following from equations (1) to (5). (5).

2) Constructing the virtual graph for larger region sizes: In this case, the region size $R > d$. We first describe the virtual graph construction for this case where the region size is an integral multiple of the basic unit of flip distance, i.e., $R = m * d$, where m is an integer (≥ 1) [2], and the flip distance F is equal to basic unit d, and $C = 1$. For instance, if $m = 2$, then this requirement translates to maximizing number of regions (of size $2d$) with at least one 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.

Our objective in constructing the virtual graph remains unchanged. Thus, the construction of the virtual graph here follows the above cases in principle. In the virtual graph, 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)

[2] We discuss the general case of R in Section III-F

or absorbed (if sensors flip from other sub-regions). Thus the number of *out* vertices per regions is equal to the number of sub-regions. In Figure 5 (b), the region size is $2d$, and the number of sub-regions is four. Thus, there are four out vertices for each region.

Having established the vertices, we now discuss how edges are added between then in the virtual graph. 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 reachability relationship between regions, an edge of infinite capacity is added from $v_i^{out\ m}$ to v_j^{in}, and an edge of infinite capacity is added from $v_j^{out\ p}$ to v_i^{in} if regions i and j are reachable from each other *and* sub-region m is reachable from region j and sub-region p is reachable from region i as shown in Figure 5. The cost, that captures number of flips between regions is 1 between the corresponding edges above. Other edges that are between regions also have cost value 1. The cost of edges that are internal to a region is 0, similar to the preceding case. We can obtain equations for capacities and costs similar to equations (1) to (5) in the preceding case. An instance of original deployment and the corresponding virtual graph at the start are shown in Figures 5 (a) and (b) respectively. We do not show the cost values in the virtual graph in Figure 5 (b).

The extensions to construct the virtual graph when $F > d$ and $C = 1$, and when $F > d$ and $C = n$ for the case where $R > d$ are similar to extensions proposed for $F > d$ and $C = n$, and $F > d$ and $C = 1$ respectively for the case where $R = d$. Due to space limitations, we do not describe the construction for these cases.

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

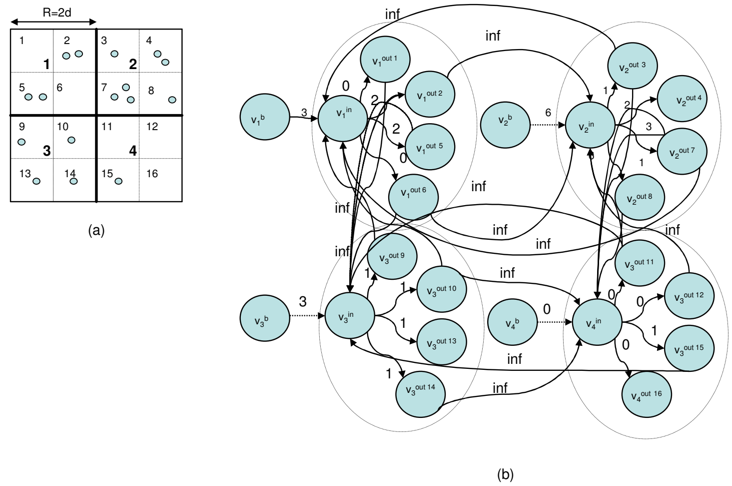

Fig. 5. The initial sensor network deployment (a) and the corresponding virtual graph at the start (b) in Case $R > d$

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 [11]. We then use the implementation in [12] to get the minimum cost flow. The implementation uses the maximum flow value computed above. The implementation in [12] is that of the successive approximation cost scaling algorithm [13] 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 [12] and [13]. 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 $w^V_{i,j} \in W^V$ is a flow from v^b_i to v^b_j in G_V, and is of the form $\langle v^b_i, v^{in}_i, v^{out}_i, v^{in}_k, v^{out}_k, v^{in}_l, v^{out}_l \ldots, v^{in}_n, v^{out}_n, v^{in}_j, v^b_j \rangle$, which denotes that the path of the flow is from v^b_i to v^{in}_i, from v^{in}_i to v^{out}_i …from v^{in}_j to v^b_j. Recall from the construction of G_V from G_S, that the reachability relation between regions i and j in G_S is translated in G_V as edges from v^{in}_i to v^{out}_j, and from v^{in}_j to v^{out}_i. 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 $w^S_{i,j} \in W^S$ is a sequence of flips in the sensor network between regions i and j. That is for each $w^V_{i,j}$ $(\in W^V)$ of the form $\langle v^b_i, v^{in}_i, v^{out}_i, v^{in}_k, v^{out}_k, v^{in}_l, v^{out}_l \ldots, v^{in}_n, v^{out}_n, v^{in}_j, v^b_j \rangle$, the corresponding $w^S_{i\ j}$ $(\in W^S)$ is of the form $\langle r_i, r_k, r_l \ldots r_n, r_j \rangle$, where r_i, r_k, r_l $\ldots r_n$, r_j correspond to regions i, k, l, $\ldots n$, j in the sensor network respectively. Physically, this means that one sensor should flip from regions i to k, k to l, $\ldots n$ to j. The sensor flip plan (also called movement plan) W^S is the output of our solution.

D. Properties of our Solution

Theorem 1 shows that the flip plan obtained by our solution optimizes both coverage and the number of flips.

Theorem 1: Let W^V_{opt} be the minimum-cost maximum-flow plan in G_V. Its corresponding flip plan W^S_{opt} will maximize coverage and minimize the number of flips.

In order to prove Theorem 1, we first introduce the concept of feasible flows and flip sequences, and introduce Lemma 1. We call a flow $w^V_{i,j}$ of the form $\langle v^b_i, v^{in}_i, v^{out}_i, v^{in}_k, v^{out}_k, v^{in}_l, v^{out}_l \ldots, v^{in}_n, v^{out}_n, v^{in}_j, v^b_j \rangle$ as feasible at time t in G_V, if there exists positive edge capacities from vertices v^b_i to v^{in}_i, v^{in}_i to v^{out}_i, v^{out}_i to v^{in}_k $\ldots v^{in}_j$ to v^b_j at time t. We call a flip sequence $w^S_{i,j}$ 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 movable sensor in each of regions $i, k, \ldots$ and n, and source region i has at least 2 sensors at time t.

Lemma 1: A flow $w^V_{i\ j}$ in G_V is feasible iff the corresponding flip sequence $w^S_{i\ j}$ is feasible in G_S.

Proof: We first prove if $w^S_{i,j}$ is feasible, then $w^V_{i,j}$ is feasible. At any time t, if $w^S_{i,j}\langle r_i, r_k, \ldots, \ r_n, r_j \rangle$ is feasible, then there is at least one movable sensor in each of regions $i, k, \ldots \ and\ n$, and source region i has at least 2 sensors. That is, $L(i,t) \geq 2$, and $M(i,t) \geq 1$. Thus, $C(v^b_i, v^{in}_i)$,$C(v^{in}_i, v^{out}_i)$,$C(v^{out}_i, v^{in}_k)$,$C(v^{in}_k, v^{out}_k)$,..., $C(v^{out}_n, v^{in}_j)$, and $C(v^{in}_j, v^b_j)$ are all ≥ 1 (where $C(i,j)$ was defined in equations (1) to (4)). Thus, $w^V_{i,j}$ is feasible.

We now prove if $w^V_{i,j}$ is feasible, then $w^S_{i,j}$ is feasible. At any time t, if $w^V_{i,j} = \langle v^b_i, v^{in}_i, v^{out}_i, v^{in}_k, v^{out}_k,$ $\ldots, v^{out}_n, v^{in}_n, v^{in}_j, v^b_j \rangle$ is feasible, then $C(v^b_i, v^{in}_i)$, $C(v^{in}_i, v^{out}_i)$, $C(v^{out}_i, v^{in}_k)$, $C(v^{in}_k, v^{out}_k)$, ..., $C(v^{out}_n, v^{in}_j)$, and $C(v^{in}_j, v^b_j)$ are all ≥ 1. This implies that for any $r_q \in \langle r_i, r_k, \ldots r_n \rangle$, $M(r_q, t) \geq 1$ and $L(i,t) \geq 2$ and so there is a feasible flip sequence $\langle r_i, r_k, \ldots, r_n, r_j \rangle$. Thus $w^S_{i,j}$ is feasible. ■

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.

Proof: We first prove that for a feasible flow set W^V_* in G_V, a corresponding flip sequence set can be found in G_S. Consider an arbitrary feasible flow w^V_* in W^V_*. By Lemma 1, a corresponding feasible flip sequence w_*^S in G_S can be found for the flow w_*^V in G_V. The set of such flips sequences is W_*^S in G_S.

We now prove that for a feasible flip sequence set W_*^S in G_S, a corresponding flow set can be found in G_V. Consider an arbitrary feasible flip sequence w_*^S in W_*^S. By Lemma 1, a corresponding feasible flow w_*^V in G_V can be found for the flip sequence w_*^S in G_S. This set of such flows is W_*^V in G_V.

■

With Lemma1 and Corollary 1, we now prove Theorem 1.

Proof: We first prove that our solution is optimal in terms of maximizing coverage. We prove this by contradiction. Consider a flip plan W_{opt}^S that corresponds to a flow plan W_{opt}^V determined by executing the minimum-cost maximum-flow algorithm on G_V. Let this flip plan be non-optimal in terms of coverage. This implies that there is a better flip plan, W_x^S that can cover at least one extra region in the sensor network. By Corollary 1, a corresponding flow plan W_x^V can be found in G_V. The amount of flow in this plan is larger than the maximum flow achieved using plan W_{opt}^V. This is a contradiction. Hence W_{opt}^S is the optimal movement (flip) plan for sensors that maximizes coverage.

We now prove that our solution is optimal in terms of minimizing the number of flips. We prove this by contradiction. Consider a flip plan W_{opt}^S that corresponds to a flow plan W_{opt}^V determined by executing the minimum-cost maximum-flow algorithm on G_V. Let this flip plan be non-optimal in terms of the number of flips. This implies that there is a better plan, W_x^S that can that can reduce at least one flip in the sensor network. By Corollary 1, a corresponding flow plan W_x^V can be found in G_V. The number of flips in this plan is less than the number of flips achieved using W_{opt}^V. This is a contradiction. Hence W_{opt}^S is the optimal movement (flip) plan for sensors that minimizes the number of flips. ■

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 the execution of the minimum-cost flow algorithm. The time complexity is dominated by determining the maximum flow and determining the minimum cost flow. The resulting time complexity using our implementations is $O(max\ (|V||E|^2, |V|^2|E|log|V|))$. Here $|V|$ and $|E|$ denote the number of vertices and edges in the virtual graph, and are given by, $|V| = O((\lceil \frac{D}{R} \rceil^2)(\lceil \frac{R}{d} \rceil)^2)$ and $|E| = O((\lceil \frac{F}{d} \rceil)(\lceil \frac{R}{d} \rceil)(\lceil \frac{D}{R} \rceil^2))$.

E. Alternate Approaches to Execute our Solution

Our solution requires information on the number of sensors in each region in the network. In the above, we proposed to let a centralized node (a Base station) to collect this information, execute the algorithm and forward the movement plan to sensors. In the following, we discuss distributed approaches to execute our solution. Our approach is similar to the one used by Open Shortest Path First (OSPF) routing protocol in determining shortest paths in a network.

In this approach, sensors in the network once again share the information about the number of sensors in the regions. In the extreme case, each region (i.e., the sensors in the region) will execute our solution independently. Depending on the output, the sensors move accordingly. 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 sensors. 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.

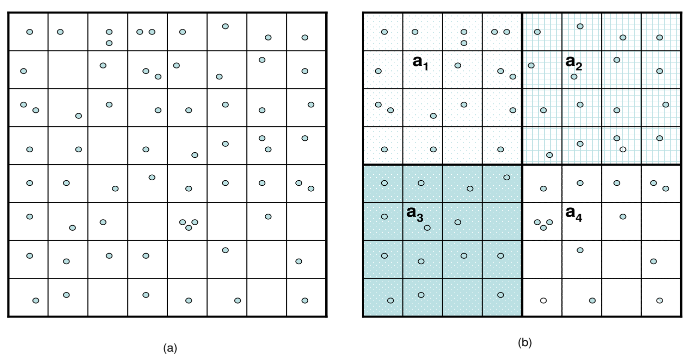

Fig. 6. The initial sensor network deployment (a) and after dividing the network into multiple areas (b)

An alternate distributed approach following from the concept of areas in OSPF is to divide the network into multiple areas. We 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). An illustration of this approach is shown in Figures 6 (a) and (b). In the sensor network in Figure 6 (a), there are $8 \times 8 = 64$ regions. We divide the network into areas of size 4×4. There are thus four areas. The areas, a_1, a_2, a_3, a_4 are shaded and shown within dark borders in Figure 6 (b). The minimum-cost maximum-flow solution we proposed above will be executed by each area independently to determine the movement plan for the particular area. This approach can only achieve local optima within each area and cannot guarantee global optima in the sensor network. Nevertheless, it reduces the overall computational and messaging complexity. We call this approach as *area-based* approach. We study the performance of area-based approach further in our performance evaluations in Section IV.

F. Discussions

We provide discussions on two issues in this sub-section. We first discuss how to construct the virtual graph for any general value of region size R. We then discuss the issue of network partitions and means to repair them.

a). Extensions to construct the virtual graph for any Region size R: In the above, we discussed the construction of the virtual graph when R was an integral multiple of d. In the following, we discuss the construction for any general value of R.

Without loss of generality, let the region size $R = sd + xd$, where, s is an integer (≥ 0) and x is a real number ($1.0 \geq x \geq 0$). In the cases we have discussed before, x took the values 1.0 ($R = (s+1)d$) or 0 ($R = sd$), where our solution is optimal. Let us now consider the case when ($1.0 > x > 0$). For two adjacent regions (say regions i and j), we cannot determine whether a sensor in region i can flip to region j in case $1.0 > x > 0$. To circumvent this problem, we leverage the concept of sub-regions. For each region of size R, we add a certain number of sub-regions each with the same size that meets the following condition; the size of each sub-region should be a factor of flip distance d, and should be a factor of region size R. In this situation, we can correctly determine if a sensor in a particular sub-region can or cannot flip to another region. This can be done by traversing an integral number of sub-regions depending on d and the size of sub-regions created. For example, let $d = 1$ and $R = 1.4$. This means $R = 1.4d$. We create 7 sub regions each of size 0.2. For a sensor in each sub-region, we now have to traverse 5 sub-regions to determine newly reachable regions. Note that our above solution is optimal if x is a terminating decimal. If x is non-terminating (e.g. $x = \frac{1}{3}$, $\frac{2}{3}$ etc.), then it is not possible to create sub-regions meeting the above condition. In such cases, we can choose an approximate sub-region size, such that the number of sub-regions is an integral multiple of R. The smaller the size of the sub-region, smaller is the error from optimality in this case.

b). Network Partitions: In some situations it is likely that the network is partitioned. Thus, sensors in one part of the network may not be able to communicate with sensors in another part during information exchange (in the absence of one hop communication to Base-station). In such cases, we have to repair such partitions, while still being constrained by mobility. In the approach proposed by Wu and Wang [2], 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 repair partitions in our case. However, we are still constrained by the mobility in the sensors. Addressing the issue of repairing network partitions using flip-based sensors is a part of our on-going work.

c). Arbitrary Flip Directions: The solution we proposed above can be extended to handle arbitrary directions for sensor flips apart from left, right, top and bottom directions. The *reachability* relationship between regions change under arbitrary flip directions. In the virtual graph, we have to add edges from each region to all newly *reachable* regions from it, corresponding to arbitrary directions of sensor flips.

IV. Performance Analysis

In the preceding discussions, we proved the optimality of our solution in terms of coverage and the number of flips. We now study the sensitivity of coverage and the number of flips to flip distance under different choices (for flip distance), initial deployment scenarios and coverage requirements. We also study performance when our solution is executed using the area-based approach discussed in Section III-E.

A. Performance Metrics and Evaluation Environment

Let the total number of regions in the network be Q. We denote Q_i as the number of regions with at least one sensor at initial deployment, and denote Q_o as the number of regions with at least one sensor after the movement plan determined by our solution is executed. We define Coverage Improvement (CI) as the improvement in coverage as a result of our solution 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 region increase in coverage. Denoting

J as the optimal number of flips as determined by our solution, we define Flip Demand FD as,

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

We conduct our simulations on two network sizes, 300×300 units and 150×150 units. The region sizes are $R = 10$ and $R = 20$ units. The basic unit of flip distance $d = 10$ units. We vary the flip distance F from 10 units to 40 units. The choices are $C = 1$ and $C = n$. Recall that if F is say, 40 and $C = 1$, the flip distance for the sensors is fixed as 40 units. When $C = n$, we can have flip distances between 0 and F in discrete steps of the basic unit of flip distance d (= 10 units). Thus if $F = 40$ ($4d$), we have $C = 4$, and in this case, sensors can flip to distances 10, 20, 30, 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 [11], and minimum cost flow algorithm is the one in [12].

We conduct our simulations using MATLAB. We use a topology generator for 2D-Normal distribution. The X and Y co-ordinates are independent of each other (i.e., $\sigma_x = \sigma_y$). 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*

We first study the sensitivity of Coverage Improvement CI and Flip Demand FD under different flip distances (F) and choices (C). Specifically, we want to understand the impacts that larger flip distances and multiple choices have on CI and FD. We then study the sensitivity of CI to F under different region sizes R and different degrees of initial distributions σ, in order to understand how coverage requirements and initial deployment scenarios impact CI. Finally, we study the performance of area-based approach by studying the sensitivity of CI under different area sizes.

1) Sensitivity CI and FD to F under different C: Figures 7 and 8 show the sensitivity of CI to flip distance F under different choices C in two different network sizes, namely 150×150 and 300×300. We set $R = 10$ and $\sigma = 1$ in both cases. The basic unit of flips distance $d = 10$. Thus the number of regions in the networks are $15 \times 15 = 225$ and $30 \times 30 = 900$ respectively. We observe that in both figures, for

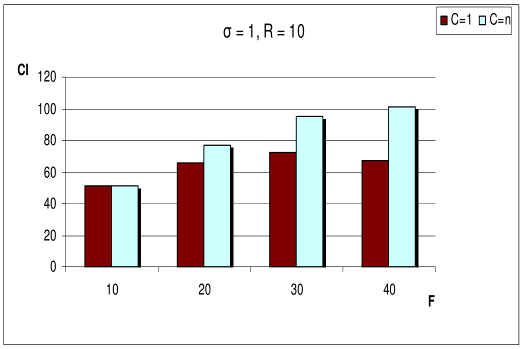

Fig. 7. Sensitivity of CI to F under different C in 150×150 network

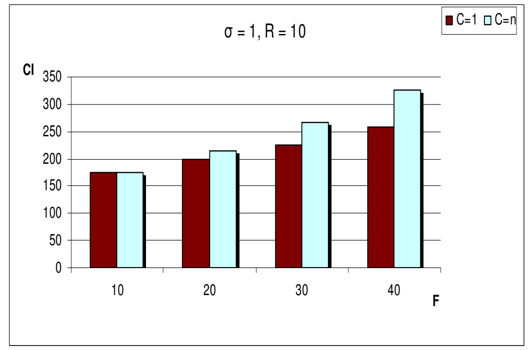

Fig. 8. Sensitivity of CI to F under different C in 300×300 network

a given value of F, $C = n$ has larger CI compared to $C = 1$, except for $F = d = 10$, when both $C = n$ and $C = 1$ have the same performance. This is because, when $F = 10$, we have $n = 1$, and so $C = n$ is the same as $C = 1$. However, when $F > 10$, for a given F there are more choices that our solution can exploit to flip when $C = n$ than when $C = 1$ for optimizing coverage. This naturally increases CI when $C = n$ compared to $C = 1$. The second observation we make is that when $C = n$, as F increases CI increases in both figures. This is also because when $C = n$, an increase in F means more choices to exploit, which increases CI. However, the trend is different when $C = 1$. In general, a large flip distance (even when $C = 1$) may appear to be perform better as more far away holes can be filled if F increases. However, when $C = 1$, such improvements depend on the size of the network. Beyond a certain point (depending on the network size), an increase in F becomes counter-productive. This is due to two reasons. First many sensors near the borders of the sensor network have flips that cannot be exploited when F is too large. Secondly, the chances of sequential flips (i.e., sensor x flipping to sensor y's region, sensor y flipping to sensor z's region and so on) to cover holes are reduced when F is too large. The value of F, where this shift takes place in the 150×150 network in Figure 7 is $F = 40$, where CI decreases. Such a shift is not observed for the 300×300 network in Figure 8, as the network size is quite large.

Figures 9 and 10 show the sensitivity of flip demand FD to flip distance F under different choices C in two different network sizes, namely 150×150 and 300×300. Since FD is a ratio of the number of flips per region increase in coverage, we cannot capture its essence when we compare FD in cases where the resulting coverage is different. To study the sensitivity of FD fairly, for both network sizes 150×150 and 300×300, we set $R = 10$ and $\sigma = 0$. In these cases, the final deployment covers all regions. Since the initial distribution is the same, the coverage improvement is the same. The comparison

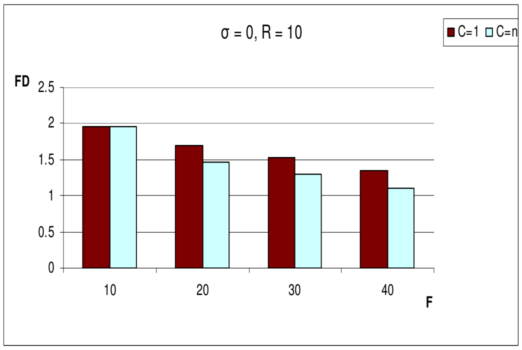

Fig. 9. Sensitivity of FD to F under different C in 150×150 network

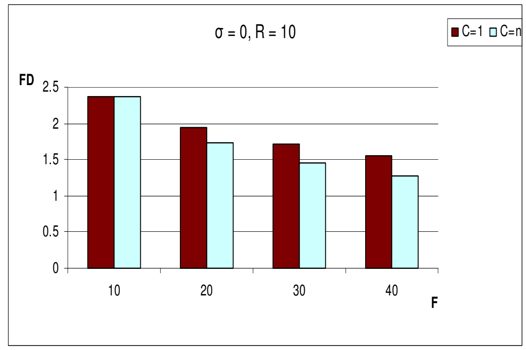

Fig. 10. Sensitivity of FD to F under different C in 300×300 network

becomes more meaningful and fair. From Figures 9 and 10, we see that for a given value of F, $C = n$ has a lower FD than $C = 1$, except for $F = d = 10$, when both $C = n$ and $C = 1$ have the same performance. This observation is consistent with our earlier observations on CI. More choices imply less number of flips per region increase in coverage. Thus FD is smaller when $C = n$ compared to $C = 1$. The second observation we make is that as F increases, FD decreases irrespective of C in both figures. 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 such holes directly, consequently minimizing the required number of flips.

2) Sensitivity CI to F under different R and σ: Figure 11 shows how the flip distance F impacts coverage improvement (CI), under different region sizes (R). Here $\sigma = 1$ and $C = n$. 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 11, we conduct our simulations on two different network sizes 300×300 and 150×150, with corresponding region sizes as $R = 20$ and $R = 10$ respectively. 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 ($C = n$) while striving for optimal coverage. Consequently, CI increases. The second observation we make from Figure 11 is that as R increases, CI decreases. The reason for this pattern is because, when R is small, neighboring regions are closer to each other (in terms of distance between the centers of the regions). When R is small, and for the same F, our solution is more likely to find sensors from other regions that can flip to fill holes. However, when R is large for the same F, the sensors that can flip from one region to another have

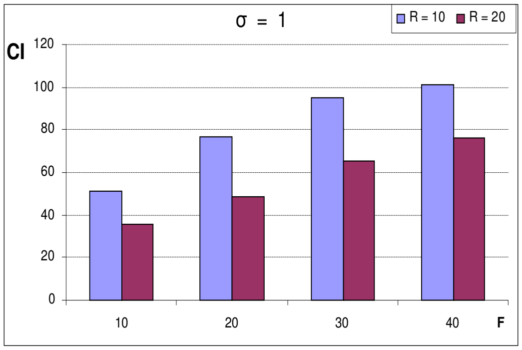

Fig. 11. Sensitivity of CI to F under different R

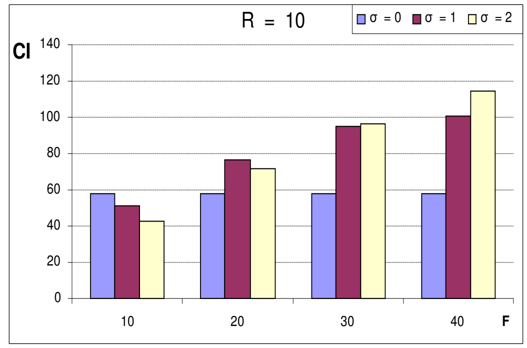

Fig. 12. Sensitivity of CI to F under different σ

to be relatively close to the borders of the regions. Thus, the number of sensors that can be found 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 the desired region size.

Figure 12 shows how the flip distance F impacts CI under different distributions in initial deployment. The network size is 150×150, $R = 10$, and $C = n$. We vary 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 increasing F and σ in terms of which parameter has a more dominating effect on CI. In Figure 12, 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, we have a situation where our solution can exploit more choices ($C = n$) while striving for optimality. Thus F can dominate when it increases. However, the degree of domination still depends on σ. This trend can be seen when $F > 10$ in Figure 12. When $F = 20$, $\sigma = 1$ performs better that $\sigma = 0$. This is because, the increase in bias can be compensated better when $F = 20$ (than was the case, 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

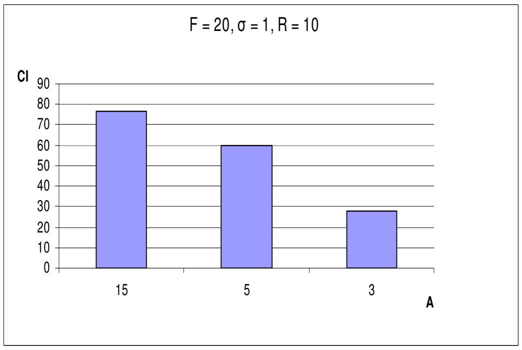

Fig. 13. Sensitivity of CI to A in 150×150 network

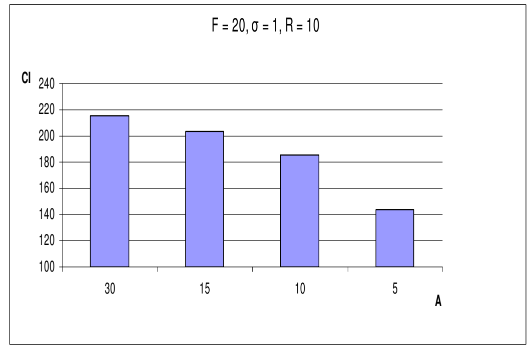

Fig. 14. Sensitivity of CI to A in 300×300 network

performance improvement due to increases in flip distance is limited by initial deployment distribution.

3) Sensitivity of CI to area size: Recall from Section III-E that an alternate approach to execute our solution is to divide the entire network into smaller areas and let each area execute our solution independently to determine the movement plan for the sensors in the areas. We study this under two network sizes: 150×150 and 300×300. We set $R = 10$, $F = 20$, $C = n$ and $\sigma = 1$ for both cases. Thus, the number of regions are $15 \times 15 = 225$ and $30 \times 30 = 900$ respectively. We divide the networks into multiple areas. We denote the area size A, as the number of regions along one dimension in each area. In both networks, for optimal coverage, the area size should be 15 and 30 respectively. For the 150×150 network, we study CI under three area sizes namely, 15, 5 and 3. For the 300×300 network, we study CI under four area sizes namely, 30, 15, 10 and 5. From figures 13 and 14 we see the sensitivity of CI, when the network is divided into smaller areas and each area executes our solution independently. In both figures, we can see that CI is sensitive to A. As A decreases, CI decreases. This is because, when individual areas execute our solution independently, the areas can only achieve coverage that is only locally optimum and not globally optimum.

V. Related Work

Deploying sensor networks has received significant attention in the recent past. Such works can be classified into three types. The first strategy is randomly deploying sensors in the field [14], [15], [16], [17], [18]. In this class some recent work like [19] [20] have appeared, where the authors choose to deploy sensors randomly in groups. The distribution pattern of group deployment is exploited for localization purposes and secure key management. More recently, Zhang and Hou in [7] discuss the relationship between coverage and connectivity in large scale sensor networks. Specifically, in the case when node density at deployment is sufficiently high, a set of optimality conditions are derived under which only a subset of working nodes are chosen for complete coverage. The second class of deployments follows incremental strategies for deploying sensors. Typically, sensors are deployed iteratively after making some measurements on the quality of previous partial deployments [3], [5], [6], [21]. The goals include localization, coverage, tracking efficiency etc. The major shortcoming in incremental deployment 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, a class of work have appeared that uses mobile sensors for deployment [1], [2], [4], [8]. The key objective in [1] and [4] is to detect holes in the network and to ensure that they are covered by at least one sensor. Sensors in the field make local measurements to detect holes. Then the sensors estimate their new positions and move towards the new positions, with the objective of covering the detected holes. In the approach proposed by Wang, Cao and La Porta, in [4] the detection of holes is based on constructing Voronoi diagrams. Each sensor constructs its own Voronoi polygon, which enables it to detect a hole. The authors then propose three protocols, namely Vector-based algorithm, Voronoi-based algorithm and Minimax algorithm to maximize coverage using mobile sensors. In [1], Howard, Mataric and Sukhatme propose the idea of constructing potential fields to maximize coverage. 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 uniformity is for load balancing. The algorithms proposed to address this problem is based on efficiently scanning the clusters in two stages (row-wise and column-wise) and determining the new sensor locations (or clusters). Sensors then move to the new clusters to achieve uniform deployment. In [8], Butler and Rus argue for deploying more than necessary number of mobile sensors a field to handle uncertainties. The authors design algorithms that uses mobile sensors to handle uncertainties in the field of interest like sudden fires, animal herds etc. In such cases, sensors move towards the event to collect more accurate data nearer to the event.

Our work in this paper does share some objectives with the above works on mobility based deployment. The deployment objectives in the above works include coverage maximization, load balancing, handling uncertainties in events etc. Our objective is coverage maximization by repairing holes in the sensor network. A related objective in the above works is also minimizing overall movement distance of sensors, which is also one of our objectives in this paper. However, the distinguishing feature of our work from the above is the sensor mobility model. Existing work discussed above [1], [2], [4], [8] consider sensors with unconstrained mobility. In our model, sensor mobility itself is a hard constraint. Sensors are capable of only *flip*-based mobility with limited flip distance, which poses additional challenges. We believe that our flip-based mobility model is realistic and practical in the future. DARPA has already conducted research on a class of Intelligent Mobile Land Mine Units (IMLM) [9], which we discussed in Section I. Such models typically trade-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, and is what we studied in this paper.

VI. Final Remarks

In this paper, we studied sensor network deployment using flip-based sensors. We proposed a minimum-cost maximum-flow based solution to maximize the coverage in the network and minimize the number of flips. We proved that our solution is optimal in terms of both maximizing coverage and minimizing the number of flips. We also proposed multiple approaches to execute our solution in practice. We conducted detailed performance evaluations to study sensitivity of coverage and the number of flips to flip distance, under different initial deployment scenarios (region sizes and sensor node distributions). We observed that while increased flip-distances achieves better coverage improvement, and reduces the number of flips required per region increase in coverage, such improvements are constrained by initial deployment distributions of sensors, due to the limitations on sensor mobility.

Sensor networks deployment using limited mobility sensors is new. We believe that our work in this paper is an important step in this regard. Our current work focuses on relaxing the assumption on discrete flip distances. In this paper, we considered sensors that can flip to distances in increments of a basic unit (d). We are working on continuous mobility of sensors, although the overall movement distance (F) is still limited.

References

[1] Andrew Howard, Maja J Mataric, and Gaurav S Sukhatme, "Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem," in *Proceedings of DARS*, Fukupka, Japan, June 2002.

[2] Jie Wu and Shuhui Wang, "Smart: A scan-based movement-assisted depployment method in wireless sensor networks," in *Proceedings of IEEE INFOCOM*, Miami, March 2005.

[3] T. Clouqueur, V. Phipatanasuphorn, P. Ramanathan, and K. Saluja, "Sensor deployment strategy for target detection," in *Proceedings of ACM WSNA*, September 2002.

[4] Guili Wang, Guohong Cao, and Tom La Porta, "Movement-assisted sensor deployment," in *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

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

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

[7] Honghai Zhang and Jennifer C. Hou, "Maintaining coverage and connectivity in large sensor networks," in *The Wireless Ad Hoc and Sensor Networks: An International Journal*, 2005.

[8] Zack Butler and Daniela Rus, "Event-based motion control for mobile sensor networks," in *IEEE Pervasive Computing, 2(4), 34-43*, Oct-Dec 2003.

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

[10] Andrew Howard, Maja J Mataric, and Gaurav S Sukhatme, "Relaxation on a mesh: a formation for generalized localization," in *Proceedings of IEEE/RSJ*, 2001.

[11] Thomas Cormen, Charles Leiserson, Ronald Rivest, and Clifford Stein, "Introduction to algorithms," in *MIT Press*, 2001.

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

[13] A. V. Goldberg and R. Tarjan, "Solving minimum-cost flow problems by successive approximation," in *Proceedings of the nineteenth annual ACM conference on Theory of computing*, New York, 1987.

[14] S. Kumar, T. H. Lai, and J. Balogh, "On k-coverage in a mostly sleeping sensor network," in *Proceedings of ACM Mobicom*, 2004.

[15] S. Shakkottai, R. Srikant, and N. B. Shroff, "Unreliable sensor grids: Coverage, connectivity and diameter," in *Proceedings of IEEE INFOCOM*, San francisco, April 2003.

[16] S. Slijepcevic and M. Potkonjak, "Power efficient organization of wireless sensor networks," in *Proceedings of IEEE ICC*, 2001.

[17] Yi Zou and Krishnendu Chakrabarty, "Uncertainty-aware sensor deployment algorithms for surveillance applications," in *Proceedings of IEEE Globecom*, San Francisco, CA, December 2003.

[18] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *HICSS,*, Jan 2000.

[19] Wenliang Du, Lei Fang, and Peng Ning, "Lad: Localization anomaly detection for wireless sensor networks,," in *Proceedings of IPDPS*, April 2005.

[20] W. Du, J. Deng, Y. Han, S. Chen, and P. Varshney, "A key management scheme for wireless sensor networks using deployment knowledge," in *Proceedings of IEEE INFOCOM*, Hong Kong, March 2004.

[21] N. Bulusu, J. Heidemann, and D. Estrin, "Adaptive beacon placement," in *Proceedings of IEEE ICDCS*, Phoenix, AZ, April 2001.