Coverage in Hybrid Mobile Sensor Networks

Wei Wang, *Student Member*, *IEEE*, Vikram Srinivasan, *Member*, *IEEE*, and Kee-Chaing Chua, *Member*, *IEEE*

Abstract—This paper considers the coverage problem for hybrid networks which comprise both static and mobile sensors. The mobile sensors in our network only have limited mobility, i.e., they can move only once over a short distance. In random static sensor networks, sensor density should increase as $O(\log L + k \log\log L)$ to provide k-coverage in a network with a size of L. As an alternative, an all-mobile network can provide k-coverage with a constant density of $O(k)$, independent of network size L. We show that the maximum distance for mobile sensors is $O(\frac{1}{\sqrt{k}}\log^{3/4}(kL))$. We then propose a hybrid network structure, comprising static sensors and a small fraction of $O(\frac{1}{\sqrt{k}})$ of mobile sensors. For this network structure, we prove that k-coverage is also achievable with a constant sensor density of $O(k)$. Furthermore, for this hybrid structure, we prove that the maximum distance which any mobile sensor has to move is bounded as $O(\log^{3/4} L)$. We then propose a distributed relocation algorithm, where each mobile sensor only requires local information in order to optimally relocate itself. We verify our analysis via extensive numerical evaluations and show an implementation of the mobility algorithm on real mobile sensor platforms.

Index Terms—Sensor networks, mobility, coverage.

1 Introduction

Wireless sensor networks (WSNs) are networks formed by a large number of simple and low cost sensors. Sensors are self-organized to perform certain tasks such as environment monitoring, target tracking, or infrastructure surveillance. An important research problem in WSNs is the coverage problem, which studies how well the field is monitored by sensors [2]. In coverage problems, the sensing region of a single sensor is often abstracted as a disk with radius r centered at it. The field is said to be k-covered when every point in the field is within the sensing region of at least k sensors [3], [4].

A critical aspect that determines the quality of coverage is network deployment. Due to a variety of factors such as the scale of the network and inaccessibility of the terrain, optimal deterministic deployment of the network is often infeasible. A common scenario envisioned for deployment is that of randomly scattering sensor devices over the field of interest. Although this eases the task of network deployment, it makes the task of guaranteeing coverage much harder.

In this paper, we define a metric, *overprovisioning factor*, which indicates the efficiency of a network deployment strategy. For a given network deployment strategy, if a sensor density of λ is required to guarantee k-coverage, then we say that the deployment strategy has an *overprovisioning factor* $\eta = \frac{\lambda}{k}$. Consider a random deployment strategy with static sensors of sensing range $r = \frac{1}{\sqrt{\pi}}$ over a square region of area L. Then to guarantee k-coverage, we need sensor density $\lambda = \log L + (k+2)\log\log L + c(L)$ with $c(L) \to +\infty$ when $L \to +\infty$ [4], [5], [6]. Since $c(L)$ can grow slower than $O(\log\log L)$, the overprovisioning factor is $\eta = \Theta(\frac{\log L}{k} + \log\log L)$. Compared to a deterministic deployment, which has $\eta = \Theta(1)$, the random static deployment has an unbounded overprovisioning factor as network size L grows. Loosely speaking, for the random deployment, many areas in the field will have far more than k sensors covering them, while a few critical regions will have around k sensors covering them. Consequently, the random deployment strategy has a high overprovisioning factor and low efficiency for large networks.

As an alternative, mobility can be used to improve network coverage efficiency [7], [8]. Mobile sensors can relocate themselves to heal coverage holes in the network so that the randomness in sensor deployment can be compensated. Clearly, the overprovisioning factor for a network with all-mobile sensors can be $\Theta(1)$ since the sensors have the flexibility of relocating themselves to the optimal locations. Unfortunately, this extra degree of freedom does not come cheap. First, mobile sensors are far more expensive than static sensors. Second, mobility consumes more energy than communication or sensing. However, most research studies in mobile sensor networks do not consider the cost of movement for mobile sensors. If a mobile sensor is required to move over long distances, then its entire energy supply may be depleted in locomotion. Moreover, the redeployment process may take considerable time in large networks since the speed of mobiles is limited.

In this paper, we first consider the coverage problem for an all-mobile sensor network, where the mobile sensors only have limited mobility. Specifically, we consider the case where each mobile can only move once, over a short distance which is predetermined by the hardware limitations [9].

- *W. Wang and K.-C. Chua are with the Computer Networks and Distributed Systems Laboratory, Department of Electrical and Computer Engineering, National University of Singapore, Blk E4A #05-06, 3 Engineering Drive 3, Singapore 117576.*
  E-mail: {wang.wei, eleckc}@nus.edu.sg, wangwei.ww@gmail.com.
- *V. Srinivasan is with Bell Labs Research, Salarpuria Ascent, 3rd Floor, No. 77, Jyothi Nivas College Road, Koramangala Industrial Layout, Ward No. 68, Bangalore 560095, India.*
  E-mail: vikramsr@alcatel-lucent.com.

Manuscript received 21 Sept. 2007; accepted 9 Apr. 2008; published online 4 Apr. 2008.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2007-09-0289. Digital Object Identifier no. 10.1109/TMC.2008.68.

1536-1233/08/$25.00 © 2008 IEEE Published by the IEEE CS, CASS, ComSoc, IES, & SPS

Unlike conventional mobile robots, these mobile sensors can use simple mobility mechanisms such as propeller systems powered by fuels [10]. Using the well-known result in minimax grid matching [11], we show that if the mobile sensors are uniformly deployed, the maximum distance that any mobile sensor has to move is $O(\frac{1}{\sqrt{k}}\log^{3/4}(kL))$ with high probability[1] (*w.h.p.*) to provide k-coverage. In other words, although the overprovisioning factor is $O(1)$ for an all-mobile sensor network, the maximum distance moved by any sensor scales as $O(\log^{3/4} L)$ for a fixed value of k. We see that the maximum movement distance for an all-mobile network scales slower than the overprovisioning factor for a static sensor network when k is fixed.

Although this result is promising, a mobile sensor is more expensive than a static sensor. This motivates us to investigate the design of a hybrid sensor network structure comprising a large number of static sensors and a small fraction of mobile sensors. The key question is whether, with such a structure, it would still be possible to maintain a constant overprovisioning factor and simultaneously limit the rate at which the maximum movement distance scales. We propose a hybrid network structure which requires a small fraction of $O(\frac{1}{\sqrt{k}})$ of the sensors to be mobile. Our hybrid network structure has an overprovisioning factor which is $O(1)$ for a given k. Moreover, we show that *w.h.p.*, the maximum moving distance for mobiles is $O(\log^{3/4} L)$. Therefore, the moving distance for the mobiles is small compared to the size of the network and only a small number of mobile sensors are required. This implies a significant cost advantage over the static and all-mobile deployment strategies.

The main results of this paper are as follows:

- We show that *w.h.p.* sensor networks of all-mobile sensors can use an overprovisioning factor of $\eta = \frac{\pi}{2}$ and a maximum moving distance of $O(\frac{1}{\sqrt{k}}\log^{3/4}(kL))$ to provide k-coverage over the whole field (Section 3).
- We propose a hybrid network structure which uses a static sensor density of $\lambda = 2\pi k$ and mobile sensor density of $\frac{\lambda}{\sqrt{2\pi k}}$ to provide k-coverage over the field. The maximum moving distance for mobile sensors is $O(\log^{3/4} L)$ *w.h.p.* in our scheme (Section 4).
- We describe a distributed algorithm to find the movement schedule for mobile sensors. Mobile sensors only need to have knowledge of neighbors within distance of $O(\log^{3/4} L)$ in the algorithm. The algorithm has time complexity of $O(L^2)$ and uses $O(L^3 \log^{3/2} L)$ message exchanges (Section 5).
- There is a tradeoff between mobile sensor density and static sensor density. With higher static sensor densities, the mobile density can be reduced exponentially while the moving distance for mobiles still scales with the network size as $O(\log^{3/4} L)$ (Section 6).
- We show that the mobility algorithm is simple enough that it can be implemented in popular sensor platforms. We demonstrate an implementation on the Cricket Motes [12], integrated with the Boe-Bot Robot [13]. The algorithm is robust to packet loss, which often happens in real world (Section 8).

1. In the rest of this paper, the term "with high probability" means that the probability is larger than $1 - L^{-c\log^{1/2} L}$ for some constant c, when $L \to +\infty$.

TABLE 1
Sensor Density and Moving Distance Tradeoff for k-Coverage

Network type	*Static sensor density*	*Mobile density*	*Maximum moving distance*
Static	$O(k \log\log L + \log L)$	0	0
All mobile	0	$O(k)$	$O(\frac{1}{\sqrt{k}} \log^{3/4}(kL))$
Hybrid	$O(k)$	$O(\sqrt{k})$	$O(\log^{3/4} L)$

In summary, Table 1 compares the sensor density and moving distance for different network structures.

2 Related Work

Mobile sensors are widely studied in sensor networks for coverage improvement [7], [8], [14], [15], [16], load balancing [17], or lifetime extension [18], [19], [20]. Most of these approaches do not consider the limitations on the distance that a mobile sensor can move. Mobile sensors in these approaches are assumed as powerful rechargeable devices. In this paper, we study the possibility of using cheap and simple mobile devices to achieve similar performance as complex mobile sensors.

Chellappan et al. introduce flip-based sensors for network coverage improvement in [9]. The flip-based sensors can only move once over a limited distance, therefore the costs of such sensors are quite low. A network flow based algorithm is used in [9] to find the mobility schedule which maximizes network coverage. The problem is further formulated as an optimization problem which minimizes the variance of sensors in different regions in [21]. However, both [9] and [21] do not provide performance bounds for fraction of area covered or maximum moving distance. In this paper, we show that the network can be completely k-covered by using only a small fraction of mobile sensors. We give bounds on the maximum moving distance of mobile sensors. Furthermore, our movement schedule formulation is simpler than in [9]. We also provide a distributed algorithm which achieves the optimal solution, while the algorithm in [21] is a distributed heuristic algorithm, with no guarantees of optimality.

The bound on maximum mobile moving distance in this paper is based on the minimax grid matching result for uniformly distributed points [11]. The minimax grid matching result has been applied to solve load balancing problems on graphs in [22] and emulation problems for sensor networks in [23]. In this paper, we extended the original minimax matching result to different distributions to bound the matching distance between mobiles and vacancies in hybrid sensor networks, where the vacancies are not uniformly distributed.

3 Coverage with Mobile Sensors

3.1 System Model

Consider a square sensing field with side length l and area $L = l \times l$. We assume that there are $N = \lambda L$ static sensors uniformly and independently scattered in the network.

When N is large, the number of static sensors in a region with area of A, which is denoted as n_A, will be Poisson distributed with mean of λA [24]:

$$\mathbb{P}\{n_A = i\} = \frac{(\lambda A)^i e^{-\lambda A}}{i!}. \quad (1)$$

Also, the number of sensors in disjoint areas will be asymptotically independent to each other [24, p. 39]. Thus, our point process can be approximated by a stationary Poisson point process when the network is large enough. In later derivations, we directly use the properties of Poisson point processes, since we study large networks where these assumptions are valid.

We assume that each static sensor can cover a disk with radius $r = \frac{1}{\sqrt{\pi}}$ centered at it. In other words, every sensor can cover a disk with *unit area*. The field is said to be k-covered when every point in it can be covered by at least k sensors. The communication range for sensors is assumed to be larger than $2r$ so that the network will be connected when it is completely covered [25].

We also assume that mobile sensors are uniformly and independently scattered in the network, and the total number of mobiles is $M = \Lambda L$. The mobile sensors have the same coverage range as static sensors. Due to energy and cost considerations, we assume that each mobile sensor only moves once over a limited distance, to heal coverage holes in the network. We assume that the mobiles are provisioned with sufficient energy, so that after relocation, they can sense and communicate for at least the same duration as the static sensors. Finally, our goal is to guarantee that the entire field is k-covered, where k is determined by the network operator prior to deployment.

3.2 Overprovisioning Factor

We define a new metric, which we call the *overprovisioning factor* $\eta = \frac{\lambda+\Lambda}{k}$, i.e., the ratio of sensor (static and mobile) density to the coverage requirement of the network. Clearly, the smaller the value of η, the more efficient is the network deployment in providing k-coverage.

For deterministically deployed networks, the optimal overprovisioning factor is $\Theta(1)$. The upper bound for η can be found by placing sensors on regular grids. For example, placing sensors on square grids with side length of $d_s = \sqrt{2}r$ can provide 1-coverage over the network. If k-coverage is required, we can place k sensors at each grid point. Thus, the overprovisioning factor for this deterministic deployment is $\eta_s = \frac{k}{k(\sqrt{2}r)^2} = \frac{\pi}{2}$. Note that η_s is larger than 1 in this case. This is because there are still some overlapping areas between adjacent sensors in deterministic deployments. For higher efficiency, we can place sensors on equilateral triangular lattices to achieve $\eta_t = \frac{2\sqrt{3}\pi}{9}$, which is the most efficient regular lattice for 1-coverage [26]. It is easy to see that the overprovisioning factor is lower bounded by 1 for any deployment, since the sum of areas of sensing regions of all sensors should be k times larger than the sensing field size. Therefore, the optimal overprovisioning factor for deterministically deployed sensor networks is $\Theta(1)$.

Let us now investigate the overprovisioning factor for randomly deployed static sensor networks with density λ. By the theory of random coverage processes [24, Theorem 3.6], the total expected area which is uncovered is $e^{-\lambda}L$. By choosing a large enough λ, the percentage of uncovered area, which is $e^{-\lambda}$, can be made arbitrarily small. However, the probability that there exists a connected coverage hole larger than unit area approaches one for a network with constant sensor density λ when the network size $L \to \infty$. The reason for this is as follows: Consider the case that a point in the network has no sensors within a distance of $2r$ from it. If such a point exists, the disk with radius $r = \frac{1}{\sqrt{\pi}}$ around it will be uncovered, which is a coverage hole with an area of at least 1. Note that such a point always exists when the network is *not* completely covered with an increased sensing range of $2r$. As shown by the theory of random coverage processes [24, Theorem 3.1], with probability approaching one, a network cannot be completely covered by a constant density of sensors with range of $2r$ when the network sizes go to infinity. Therefore, we see that a constant sensor density of λ cannot guarantee that there are no big holes in the network as the network size grows, even though most areas of the field will be covered.

To achieve k-coverage in a large network, the static sensor density needs to grow with the network size as $\lambda = \log L + (k+2)\log\log L + c(L)$, where $c(L) \to +\infty$ as $L \to +\infty$ [4]. The overprovisioning factor for a randomly deployed static sensor network is

$$\eta_s = \frac{\log L + (k+2)\log\log L + c(L)}{k}, \quad (2)$$

which is $O(\log L)$ for fixed values of k. This shows that the coverage efficiency for random static sensor networks become worse as the network size increases.

3.3 All-Mobile Networks

We now consider coverage in networks when all sensors are mobiles and are randomly deployed. These mobile sensors then reposition themselves so as to provide k-coverage. Clearly, in this case, we should be able to achieve $\eta_m = \Theta(1)$. However, the key question is: what is the maximum distance that each sensor has to move in order to place itself at the optimum location, since movement consumes a significant amount of energy [27]? Most prior research tries to minimize the total distance moved or total number of movements made by all the sensors, e.g., [17]. This is inadequate since energy is not transferable between mobile sensors. Therefore, it is better to limit the maximum moving distance for each mobile by moving several mobiles over a short distance, such as the cascaded movement in [28].

We bound the maximum moving distance for all-mobile networks as follows:

Theorem 1. *Consider an all-mobile sensor network uniformly and independently distributed over a square field with area L. The network can provide k-coverage with an overprovisioning factor of $\eta_m = \frac{\pi}{2}$ and the maximum distance moved by any mobile sensor is $O(\frac{1}{\sqrt{k}}\log^{3/4}(kL))$ w.h.p.*

Proof. In order to provide a tight bound on the moving distance, we use a different placement than the deterministic placement discussed in the previous section, which places k sensors at the same grid point of side length

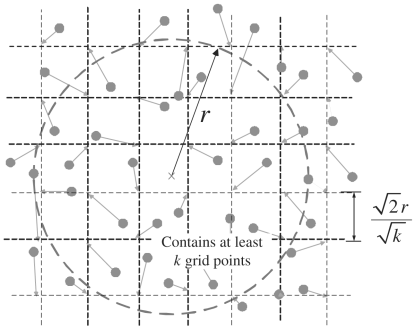


Fig. 1. Matching mobiles to grid points in all-mobile networks.

$d_s = \sqrt{2}r$. Instead, here we divide the sensing field into square grids with side length of $d_a = \frac{\sqrt{2}r}{\sqrt{k}}$, as shown in Fig. 1. It is easy to see that the density of the grid points is $\frac{k}{2r^2} = \eta_m k$.

We first show that the network can be k-covered with one mobile at each of these grid points. Then, we will bound the maximum moving distance for uniformly distributed mobiles to achieve such a regular grid deployment.

Assume that the mobiles have been relocated so that each grid point has exactly one mobile on it. First, consider coverage on interior network areas which have distance more than r to the field boundary. If a point is within distance r to at least k grid points, it is then k-covered. By the lower bounds on lattice points covered by a circle [29], there are at least $W(k)$ lattice points of side length of d_a covered by a circle of radius r centered at an arbitrary point:

$$W(k) \geq \frac{\pi\left(r - \frac{1}{\sqrt{2}}d_a\right)^2}{d_a^2} = k \times \frac{\pi}{2}\left(1 - \frac{1}{\sqrt{k}}\right)^2. \quad (3)$$

Note that $W(k)/k$ is a monotonically increasing function when $k \geq 1$, and we have $W(k) > k$ when $k \geq 25$. It is also easy to verify that the network is at least k-covered when $1 \leq k < 25$. Thus, we can see that if there is one sensor at each grid point, then the network interior is completely k-covered. To cover points near the boundary, we can slightly increase the deployment field to a $(l + 2r) \times (l + 2r)$ square. This only increases the density by a fraction of $O(\frac{r}{l})$, which is negligible when the network size is large.

After mobiles are randomly deployed in the network, we need to relocate mobiles so that each grid point has exactly one mobile. This is essentially a matching problem between mobile sensors and grid points. The maximum moving distance for mobile sensors can be derived from the results of the minimax grid matching problem studied in [11]:

Consider an $l \times l$ square region with square grids of unit side length. If we randomly and independently scatter $L = l^2$ points in the region according to a uniform distribution, then w.h.p., there exists a perfect match between the L random points and the L grid points with maximum distance between any matched pairs of $O(\log^{3/4} L)$.

Note that the total number of grid points is $\frac{k}{2r^2}L$ in our network instead of L. Therefore, the maximum moving distance will be $O(\log^{3/4}(kL))$ times the side length of the grid. Since our grid size is $d_a = \frac{\sqrt{2}r}{\sqrt{k}}$ instead of 1, we get the maximum moving distance bound of $O(\frac{1}{\sqrt{k}}\log^{3/4}(kL))$. □

Theorem 1 shows that it is possible to relocate the mobiles by only a small distance to achieve deterministic sensor placement. The actual relocation algorithm will be discussed in Section 5.

An interesting point in an all-mobile sensor network is that the mobiles compensate the randomness in large networks differently when compared to static approaches. The static approach needs to use higher density, scaled as $O(\log L)$, to compensate for the network size. In mobile sensor networks, the sensor density remains constant while mobiles need to increase their moving distance as $O(\log^{3/4} L)$ as the network size increases.

4 Coverage of Hybrid Networks

The all-mobile network can achieve deterministic sensor deployment by moving sensors over a small distance. However, mobile sensors are much more expensive than static sensors. In order to reduce the network cost, it is preferable to use only a small number of mobiles to improve the network performance. In this section, we study the coverage of hybrid networks in which a large number of static sensors and a small fraction of mobile sensors are deployed. We provide a constructive proof to show that the overprovisioning factor is $O(1)$ and the fraction of mobile sensors required is less than $\frac{1}{\sqrt{2\pi k}}$. We further show that for this particular deployment, the maximum distance that any mobile sensor will have to move is $O(\log^{3/4} L)$ *w.h.p.*

4.1 Density of Mobile Sensors

In this section, we first fix the static sensor density at $\lambda = 2\pi k$. The tradeoff between static sensor density and mobile sensor density will be further discussed in later sections.

We divide the network into square *cells* with equal side length of $d_h = r/\sqrt{2}$. Since the sensing range is r, any sensor in the cell can completely cover the cell. The average number of static sensors in each cell will be $2\pi k d_h^2 = k$.

The network will be k-covered if all cells contain at least k sensors. However, some cells may contain fewer than k static sensors due to the randomness in deployment. If a cell i contains $n_i < k$ static sensors, we say cell i has $v_i = k - n_i$ *vacancies*. According to the Poisson approximation, n_i will be asymptotically independently and identically distributed as

$$\mathbb{P}\{n_i = j\} = \frac{k^j e^{-k}}{j!}. \quad (4)$$

The random variable $v_i = [k - n_i]^+$, where $[x]^+$ means $\max\{x, 0\}$, will be distributed as

$$\mathbb{P}\{v_i = j\} = \begin{cases} \frac{k^{k-j}e^{-k}}{(k-j)!}, & 1 \leq j \leq k, \\ 1 - \sum_{m=0}^{k-1}\frac{k^m e^{-k}}{m!}, & j = 0, \\ 0, & otherwise. \end{cases} \quad (5)$$

The expected number of vacancies in a cell will be

$$\begin{aligned}\mathbb{E}\{v_i\} &= \sum_{j=1}^{k} j\frac{k^{k-j}e^{-k}}{(k-j)!} \\ &= \sum_{l=0}^{k-1}(k-l)\frac{k^l e^{-k}}{l!} \\ &= \sum_{l=1}^{k-1}\left[\frac{k^{l+1}e^{-k}}{l!} - \frac{k^l e^{-k}}{(l-1)!}\right] + ke^{-k} \\ &= \sum_{l=1}^{k}\frac{k^l e^{-k}}{(l-1)!} - \sum_{l=1}^{k-1}\frac{k^l e^{-k}}{(l-1)!} \\ &= \frac{k^k e^{-k}}{(k-1)!} = k\mathbb{P}\{n_i = k\}.\end{aligned} \tag{6}$$

Since v_i are independently and identically distributed random variables, we drop the subscript i in $\mathbb{E}\{v_i\}$ in later derivations. The average number of vacancies per cell will converge to $\mathbb{E}\{v\}$ when the network size is large, by the Law of Large Numbers [30]. In other words, the average number of vacancies per cell will be within a range of $[(1-\epsilon)\mathbb{E}\{v\}, (1+\epsilon)\mathbb{E}\{v\}]$ for arbitrarily small values of ϵ when $L \to \infty$. Therefore, with a mobile density of $\Lambda = \frac{(1+\epsilon)\mathbb{E}\{v\}}{(r/\sqrt{2})^2} = (1+\epsilon)2\pi\mathbb{E}\{v\}$, the number of mobiles is almost surely larger than or equal to the total number of vacancies for large networks. As ϵ can be made arbitrarily small, we just use the asymptotic mobile density of $\Lambda = 2\pi\mathbb{E}\{v\}$ in future derivations.

Using Stirling's approximation, $k! \approx k^k e^{-k}\sqrt{2\pi k}$:

$$\mathbb{E}\{v\} = k\frac{k^k e^{-k}}{k!} \approx \frac{\sqrt{k}}{\sqrt{2\pi}}. \tag{7}$$

Note that the error of Stirling's approximation has the order of $O(e^{1/(12k)})$. Thus, we have $\mathbb{E}\{v\} \to \sqrt{k}/\sqrt{2\pi}$ as $k \to \infty$ (see Fig. 5). Consequently, we have $\Lambda \approx \sqrt{2\pi k}$. As the static sensor density is $2\pi k$, the density ratio of mobile sensors compared to static sensors is $\frac{\Lambda}{\lambda} \approx \frac{1}{\sqrt{2\pi k}}$. As k increases, a smaller fraction of sensors need to be mobile to fill the vacancies. This agrees with the intuition that the Poisson distributed number of static sensors in a cell will be more concentrated around the mean of k as k increases.

The summation of mobile and static sensor density is still $O(k)$ since we have $\lambda = 2\pi k$ and $\Lambda \approx \sqrt{2\pi k}$. More precisely, the overprovisioning factor $\eta_h = \frac{\lambda+\Lambda}{k} \le 2\pi + \sqrt{2\pi}$ for an arbitrarily large network and any integer value of k. The fraction of mobiles needed for different k values are plotted in Fig. 5. For k larger than 15, fewer than 10 percent of the sensors need to be mobile. However, for small k values, the mobile sensor density Λ can be larger than the density of all-mobile networks. For example, when $k = 1$, we can use mobile density of $\frac{\pi}{2}$ to achieve a deterministic square coverage over the field while our solution requires density of $\Lambda = \sqrt{2\pi} > \frac{\pi}{2}$. This problem will be further discussed in Section 6 where we can use several methods to reduce the mobile density.

4.2 Moving Distance for Mobiles

In the hybrid network solution discussed above, we need to move mobiles to fill in the vacancies in each cell. In other words, we need to build up a one-to-one matching between mobile sensors and vacancies. In the following discussion, we show that the maximum distance that any mobile sensor will have to move is $O(\log^{3/4} L)$ w.h.p.

The matching is built in two steps: First, we match the mobiles to points on a grid with side length of $\frac{1}{\sqrt{\Lambda}}$. The maximum matching distance is $O(\frac{1}{\sqrt{\Lambda}}\log^{3/4}(\Lambda L))$ *w.h.p.*, as shown in Section 3. The function $\frac{1}{\sqrt{\Lambda}}\log^{3/4}(\Lambda L)$ is a decreasing function with Λ when Λ and L are larger than 1. Therefore, the matching distance decreases with Λ and it can be rewritten as $O(\log^{3/4} L)$, since $\Lambda \approx \sqrt{2\pi k} > 1$.

The second step is to match vacancies to the grid points on the grid with side length of $\frac{1}{\sqrt{\Lambda}}$. Unfortunately, the results from [11] cannot be directly applied since the vacancies are not uniformly distributed. We use the following theorem to bound the maximum matching distance:

Theorem 2. *Consider a square network with area L, where ΛL vacancies are distributed independently and identically, in cells with side length of $d_h = \frac{r}{\sqrt{2}}$ according to (5). Then, w.h.p., there exists a matching which has maximum matching distance of $O(\log^{3/4} L)$ between vacancies and the grid points on grids with side length of $\frac{1}{\sqrt{\Lambda}}$.*

Proof. See Appendix for the proof. □

Thus, the matching distance between vacancies and the grid points is also $O(\log^{3/4} L)$. Since the big O notation hides constant factors, the distance between the mobile and vacancy matched to the same grid point is also $O(\log^{3/4} L)$. This builds up the one-to-one matching between mobiles and vacancies with maximum distance between matched pairs as $O(\log^{3/4} L)$.

Compared to the all-mobile case, the moving distance of the hybrid network is $O(\sqrt{k})$ times larger. However, it is still much smaller than the network size.

5 Mobility Algorithm

We need a distributed locomotion schedule to relocate mobiles with maximum movement distance of $O(\log^{3/4} L)$. Simple greedy movement, such as moving mobiles to the nearest vacancy, may fail to fill all vacancies with short distance movements [9]. Since the matching problem is a special kind of network flow problem, we can use a network flow architecture to solve the movement schedule in a distributed manner.

5.1 Problem Formulation

Inspired by the network flow model used in [9], we formulate our movement schedule problem as follows: Suppose for each cell i there are n_i static sensors and m_i mobile sensors. The number of vacancies in cell i will be $v_i = [k - n_i]^+$. The problem of moving mobiles to fill the vacancies is similar to traffic flow problem in networks (see Fig. 2).

We construct a graph $G(\mathcal{V}, \mathcal{E})$ with each cell as a vertex. We add a directional edge (i, j) from cell i to j

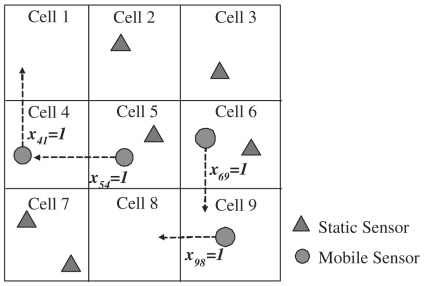


Fig. 2. Formulating the mobility problem as a network flow problem. The excess mobiles are "flowing" toward the vacancies.

when the distance between their center is smaller than $D = O(log^{3/4}L)$, which is the maximum distance that a mobile sensor can move. Thus, mobiles can move between two cells when the distance between them are smaller than D. Denote the number of mobiles which move from cell i to cell j as x_{ij}, then the movement schedule problem can be formulated as

$$Minimize \quad \sum_{i,j} c_{ij} \times x_{ij}, \tag{8}$$

$$s.t. \quad \sum_{j} x_{ji} - \sum_{j} x_{ij} \geq v_i - m_i \quad \forall i, \tag{9}$$

$$\sum_{j} x_{ij} \leq m_i \quad \forall i, \tag{10}$$

$$x_{ij} \geq 0 \qquad \forall i, j, \tag{11}$$

where c_{ij} is the movement cost. In this optimization problem, (9) is the flow conservation condition, which requires the net in-flow for cell i (number of mobiles moving into cell i minus number of mobiles moving out of cell i) to be larger than the number of mobiles it requires (the number of vacancies minus initial number of mobiles in cell i). This constraint guarantees that the final number of mobiles in cell i will be larger than the number of vacancies after the movement. Equation (10) shows that the total number of mobiles moving out of cell i should not be larger than the initial number of mobiles in cell i. The movement cost c_{ij} determines the metrics that need to be minimized. If we set all $c_{ij} = 1$, then the optimal solution will give the movement schedule which has minimum number of movements. If c_{ij} is selected as distance between cells, we will get the scheme with the minimum total moving distance. In this formulation, every mobile will move only once between cells which are not more than D apart. Note that our problem formulation is simpler compared to the formulation in [9].

This problem formulation can work for both the all-mobile network and hybrid network. In case of all-mobile network, we can simply set $v_i = k$ for all cells in the problem. This formulation can also be applied to irregularly shaped networks by the same graph construction methods as in square networks.

We next convert our problem to an equivalent standard network flow problem to show certain important properties of this problem. Note that the linear optimization problem of (8)-(11) is similar to the minimum cost flow problem [31], except for the flow conservation constraint of (9). Since in our problem the total number of mobiles $\sum_{i=1}^{2\pi L} m_i$ is slightly larger than the total number of vacancies $\sum_{i=1}^{2\pi L} v_i$ (see Section 4.1), we make the net in-flow to be larger or equal to $v_i - m_i$ instead of just equal to. If we add a super sink cell c_0 and slack variables x_{i0} to represent the excess number of mobiles, we will get an equivalent problem which is exactly the minimum cost flow problem with the few excess mobiles going to the super sink:

$$Minimize \quad \sum_{i,j} c_{ij} \times x_{ij}, \tag{12}$$

$$s.t. \quad \sum_{j \neq 0} x_{ji} - \sum_{j \neq 0} x_{ij} - x_{i0} = v_i - m_i, \forall i \neq 0, \tag{13}$$

$$\sum_{i} x_{i0} = \sum_{i=1}^{2\pi L} m_i - \sum_{i=1}^{2\pi L} v_i, \tag{14}$$

$$\sum_{j} x_{ij} \leq m_i \qquad \forall i, \tag{15}$$

$$x_{ij} \geq 0 \qquad \forall i, j. \tag{16}$$

This minimum cost flow problem has flow capacity constraints on nodes, i.e., the total flow through a node is limited. We can further convert such problem to a traditional network flow problem which has only edge capacity constraints using the well-known node-splitting method in network flow problems [31, pp. 41-42].

As the constraint matrices for network flow problems are Totally Unimodular [31, pp. 447-449], the optimal solution x_{ij}^* are integers since v_i and m_i are integers. Therefore, the optimal solution implies that we can just move x_{ij}^* mobile sensors from cell i to cell j to ensure that each cell has at least k sensors in total.

5.2 Distributed Solution

In this section, we describe a distributed algorithm to find the movement schedules for the mobiles. To better describe the algorithm, we first provide a distributed algorithm to solve a simpler problem. This algorithm only gives a feasible movement schedule to fill all vacancies without minimizing the total movement cost. We will later show that with several iterations of this algorithm, the minimum-cost flow can also be achieved. Details are at the end of this section.

The minimum cost flow problem described in Section 5.1 gives the optimal sensor movement schedule which minimizes the total movement costs. However, if our goal is only to use mobiles to fill all the vacancies (without minimizing the movement cost), we can just treat the problem as a maximum flow problem, i.e., maximizing the flow from the source to the destination. The solution of the maximum flow will be a feasible movement schedule for mobiles to fill all vacancies when such schedule exists. Several efficient algorithms for the maximum flow problem exist, such as Ford-Fulkerson augmenting path algorithm or the push-relabel algorithm

[32]. In this paper, we adopt the push-relabel structure which is a naturally distributed algorithm.

5.2.1 Assumptions for the Algorithm

We assume that each mobile or static sensor knows its location and knows which cell it is located in. After deployment, mobiles and static sensor in the same cell i communicate with each other to compute v_i and m_i. Each cell elects a mobile or static sensor as the delegator for the entire cell. This sensor stores the necessary information of the cell during the algorithm execution. The delegator of cell i also needs to communicate and exchange information with its neighbors in graph G described earlier. In case there are empty cells, which contain no mobile or static sensors, we can randomly assign a mobile in an adjacent cell as its delegator. Since we have shown the empty cell can be filled with a maximum moving distance of D *w.h.p.*, there will at least be one mobile within distance of D. In case that an empty cell makes the network disconnected, the nearest mobile moves to the empty cell to connect the network before executing the algorithm. In the following discussion, we just use the term "cell" instead of "the delegator of the cell" when an operation needs to be performed.

5.2.2 Algorithm Description

The basic idea for push-relabel algorithm is to iteratively *push* the excess flow of one vertex to neighboring vertices with lower "heights" or *relabel* itself, which is lift the height of itself, when a push cannot be performed. The push and relabel will be repeated until all cells have no excess flow. Details of the original push-relabel algorithm can be found in [32] and [33]. Here, we only discuss the difference between our algorithm and the original push-relabel algorithm:

- In our algorithm, we use the vacancies as the commodity instead of mobiles. In other words, we push the vacancies from the cells with fewer than k sensors to the cells with free mobiles.
- We have capacity limits on the total flow going through one cell to bound the number of mobiles which can move out of the cell. We adopt the node-splitting method to handle capacity bounds on nodes [31, pp. 41-42]. Each cell i will be split to two vertices i_{in} and i_{out}, the input vertex and the output vertex, respectively. The input vertex is connected with the output vertex by a unidirectional arc (i_{in}, i_{out}) with zero cost and capacity same as m_i, the upper bound on number of mobiles which can move out of cell i. Then, the output vertex i_{out} is connected with neighboring cell's input vertex j_{in} with a unidirectional arc (i_{out}, j_{in}) with cost defined as the c_{ij} in Section 5.1 and unlimited capacity. Therefore, each cell must maintain two vertices in the push-relabel algorithm. This node-splitting method directly comes from the network flow theory [31], and it is simpler than algorithms splitting each cell to three vertices as in [9].

The details of this algorithm are shown in Fig. 3. In this algorithm, cells only need to know the heights of vertices in neighboring cells within distance D to perform either push or relabel operation. The push process between i_{in} and i_{out} in the same cell is the same as between different cells except that no message needs to be sent. Note that the push and relabel operations only send messages between cells without actually moving the mobiles. The movements are performed at the end of the algorithm. Each cell will send mobiles to neighboring cells according to the in-flow of their input cells.

Mobility algorithm for cell i
01: Collect cell information of v_i and m_i
02: Set height of i_{in} and i_{out}, denoted as $h(i_{in})$ and $h(i_{out})$ to 0
03: Set excess of i_{in}, denoted as $e(i_{in})$ to 0, and $e(i_{out})$ to $v_i - m_i$
04: **while** there exists vertex with positive excess
05: Call *Push-relabel*(i_{in})
06: Call *Push-relabel*(i_{out})
06: Update heights of neighboring cells within distance D
07: **endwhile**
08: Send mobiles to cell j according the flow on arc (j_{out}, i_{in})

Push-relabel (vertex u)
01: **If** $e(u) > 0$
02: **while** $e(u) > 0$ *and* exists arc (u, w) *s.t.* $h(u) = h(w) + 1$ and the residual capacity of arc (u, w), $cap(u, w) > 0$.
03: Push amount of $y = \min\{e(u), cap(u, w)\}$ through arc (u, w) by sending a message to the cell associated to w
04: $e(u) = e(u) - y$; $e(w) = e(w) + y$; update $cap(u, w)$.
05: **endwhile**
06: **If** $e(u) > 0$
07: Update $h(u)$ as $1 + \min\{h(w) : cap(u, w) > 0\}$
08: Broadcast $h(u)$ to neighboring cells within distance D
09: **endIf**
10: **endif**

Fig. 3. Distributed mobility algorithm.

5.3 Algorithm Performance

In our algorithm, every cell needs to communicate only with cells within distance of D and the cell only requires knowledge about these neighboring cells to perform the push and relabel. The network graph has $|\mathcal{V}| = O(L)$ vertices and each vertex has at most $\pi D^2 = O(\log^{3/2} L)$ arcs. Therefore, the number of edges is $|\mathcal{E}| = O(L \log^{3/2} L)$. Since asynchronous distributed push-relabel algorithm runs in $O(|\mathcal{V}|^2)$ time and uses at most $O(|\mathcal{V}|^2|\mathcal{E}|)$ message exchanges [33], our algorithm takes at most $O(L^2)$ running time and the number of messages exchanged is $O(L^3 \log^{3/2} L)$. Later, via simulations, we will show that this bound is quite loose since the actual running time scales nearly linearly with the network size. Since our algorithm is executed in the delegators of the cells, the complexity of our algorithm scales with the network size (number of cells in the network) instead of the number of sensors in the network. When the network density increases, only the values of m_i and v_i in the algorithm change and the algorithm complexity remains the same.

If the minimum movement cost schedule is required instead of an arbitrary feasible movement schedule, we can use the cost scaling algorithm proposed by Goldberg and Tarjan [34]. The algorithm uses $O(\log(LC))$ iterations of push-relabel processes to refine the cost of the solution, where C is the maximum cost for any edge. In our problem, our edge cost has positive integer values bounded by $D = O(\log^{3/4} L)$. Therefore, finding the minimum movement cost schedule takes $O(\log L)$ times more computational time and message exchanges than finding an arbitrary feasible movement schedule.

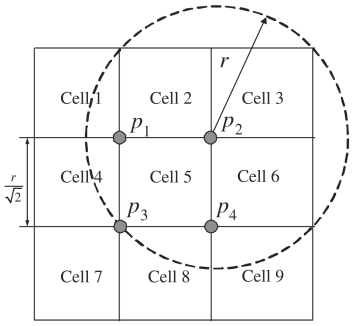


Fig. 4. Sharing the mobile sensor.

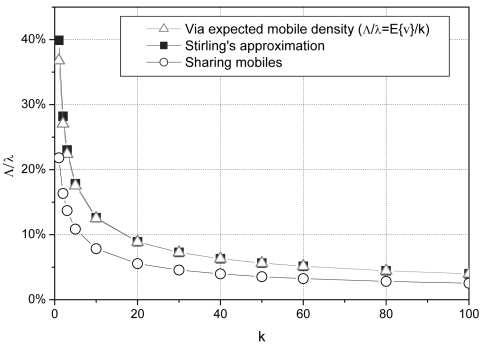


Fig. 5. Ratio of Λ/λ (with $\lambda = 2\pi k$).

6 Discussions

6.1 Reducing Mobile Density

The mobile density used in Section 4 is quite high especially when k is small. In this section, we provide several methods to reduce mobile density.

6.1.1 Sharing Mobiles

The cell used in our hybrid network has side length of $r/\sqrt{2}$, which is quite conservative compared to the sensing range of r. Actually, a mobile at the corner of a cell can cover four cells at the same time (see Fig. 4).

Consider the super cell which contains nine cells as shown in Fig. 4. We need to deploy the mobiles only at the four central points of p_i to provide coverage for the nine cells. For example, when $k = 1$, if there are no static sensors in the super cell, we can put one sensor at each p_i to provide full coverage on the nine cells while the basic algorithm discussed in Section 4 uses nine mobiles. If cell 1 has at least one static sensor, then at most three mobiles are needed to stay at p_2, p_3, and p_4 to cover the rest of cells. As each mobile can cover four adjacent cells, the mobile density can be reduced by a constant factor.

The density of mobiles required for sharing mobiles can be numerically calculated by enumerating possible vacancy distributions in the super cell. The reduced mobile density is shown in Fig. 5. We see that when k is small, this scheme can reduce the mobile density by half, compared to the original hybrid structure. In this case, our hybrid network can use fewer mobiles than all-mobile networks when $k = 1$. However, the improvement ratio reduces as k increases (see Fig. 5). When implementing the sharing mobile algorithm, we separate the network region to disjoint super cells. The number of mobiles required in each super cell can be determined when the number of static sensors in the nine small cells is known. We can apply the push-relabel algorithm to find the optimal solution in this case also.

6.1.2 Increasing Static Sensor Density

We can also reduce the density of mobiles by increasing the density of static sensors. Suppose that we increase the density of static sensors so that the average number of sensors in each cell of side length $\frac{r}{\sqrt{2}}$ is $g \geq k$.

The number of vacancies will be distributed as

$$\mathbb{P}\{\hat{v}_i = j\} = \begin{cases} \frac{g^{k-j}e^{-g}}{(k-j)!}, & 1 \leq j \leq k, \\ 1 - \sum_{m=0}^{k-1} \frac{g^m e^{-g}}{m!}, & j = 0, \\ 0, & otherwise. \end{cases} \tag{17}$$

Similar to (6), the expected number of vacancies in a cell will be

$$\begin{aligned} \mathbb{E}\{\hat{v}\} &= \sum_{l=0}^{k-1}(k-l)\frac{g^l e^{-g}}{l!} \\ &\leq \left(\frac{g}{k}\right)^{k-1} e^{-(g-k)} \sum_{l=0}^{k-1}(k-l)\frac{k^l e^{-k}}{l!} \\ &\leq e^{\frac{(k-1)(g-k)}{k}} e^{-(g-k)} \mathbb{E}\{v\} = e^{-\frac{g-k}{k}} \mathbb{E}\{v\}. \end{aligned} \tag{18}$$

The third step uses the inequality of $(1+\frac{x}{n})^n \leq e^x$ when $x > 0$ and $n > 0$, to get $(\frac{g}{k})^{k-1} = (1+\frac{g-k}{k})^{k-1} \leq e^{\frac{(k-1)(g-k)}{k}}$. Therefore, with an increased static sensor density, the density of mobile sensors is reduced at least exponentially as $e^{-\frac{g-k}{k}}$. Specifically, when the density of static sensor is doubled $(g = 2k)$, the density of mobiles can be reduced by at least e^{-1} which is close to one third. Note that the bound in (18) is tight only for small k, we can use even smaller number of mobiles when k is large. Note that we are still maintaining a constant overprovisioning factor in this solution.

6.1.3 Maximum Moving Distance

Suppose that we use one of the two methods discussed above and the mobile density is Λ. Using the same argument as in Section 4.2, we can first match the mobiles to grid points with side length $\frac{1}{\sqrt{\Lambda}}$, then match the vacancies to the grid points. However, the vacancy distributions are different from those in Section 4.2, and we have a different bound on moving distance here.

Theorem 3. *Consider a square network with area L with ΛL vacancies distributed independently and identically in each square cell with side length of d, where d is some constant. If the number of vacancies in each cell is upper bounded by k, then* w.h.p., *we can find a matching which has maximum matching distance as $O(\frac{k}{\Lambda}\log^{3/4} L)$ between vacancies and the grid points (on grids with side length of $\frac{1}{\sqrt{\Lambda}}$).*

Proof. See Appendix for the proof. □

Theorem 3 can be directly used for the increasing static sensor density scheme. For the first scheme which shares the mobiles, we can set d as the side length of the super cell, which is $\frac{3r}{\sqrt{2}}$. The number of mobiles required in each super

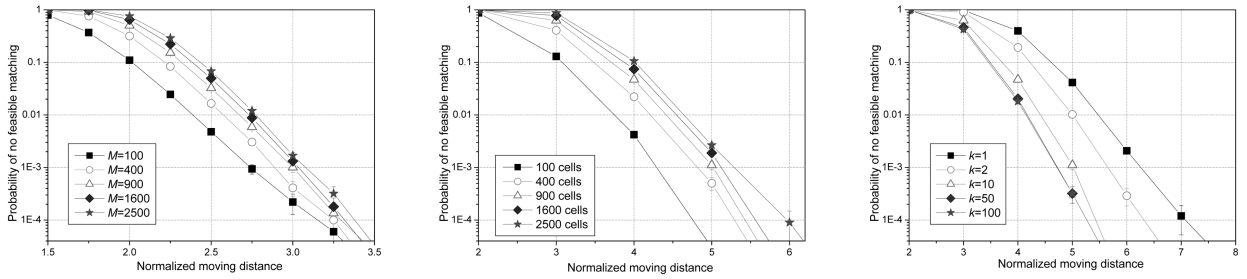


Fig. 6. Probability that no feasible matching exists for a given moving distance (confidence interval 95 percent). (a) All-mobile networks (moving distance normalized by $\sqrt{2}r$). (b) Hybrid networks of different size ($k = 10$, moving distance normalized by $r/\sqrt{2}$). (c) Hybrid networks of different k (900 cells, moving distance normalized by $r/\sqrt{2}$).

cell is independently and identically distributed. Also, each super cell needs at most $4k$ mobiles. Applying Theorem 3, we get the maximum moving distance of $O(\frac{k}{\Lambda}\log^{3/4} L)$. Therefore, when Λ and k are fixed, the maximum moving distances for both schemes still increases as $O(\log^{3/4} L)$ with the network size L. From Theorem 3, we also see that when the mobile density decreases, the moving distance for mobiles will increase.

6.2 Real-World Deployment

The sensing region may be imperfect due to barriers or noise in real deployments. Although our analysis in Sections 3 and 4 assumes the sensing regions are perfect disks with uniform radius, our results can be easily modified when sensing regions are imperfect. Our algorithm achieves a regular number of sensors in each square cell, irrespective of the shape of sensing region. If the sensing region is not a perfect disk, we can find a proper size for the cells so that each cell can be fully covered by any sensor within it. Our analysis and algorithm can also be applied to other types of grids such as triangular, rectangular, or hexagonal grids. We can choose among these cell shapes according to the sensing model used in deployment.

When certain areas require more sensors due to high sensing noise level, our algorithm can also be used to achieve a nonuniform deployment of sensors. Suppose more sensors are required to be deployed in a particular cell i than other cells. We can increase the number of vacancies, v_i, in cell i in our algorithm to find the movement schedule which sends more sensor to cell i. However, the maximum movement distance bounds may no longer hold in this case.

In previous sections, we assume that the sensing field is a $l \times l$ square. Our analysis and algorithm can also be extended to irregularly shaped network fields with barriers and holes. When the network field is not regular, we can divide the field into cells as in Section 5. In this case, the network size L will be defined as the number of cells in it. We can construct the graph for movement schedule by adding edges between cells that mobiles can move to as in Section 5. If there are barriers which mobiles cannot pass through, we just remove the edges crossing such barriers when constructing the graph. Then, the movement schedule can be found on the constructed graph by the algorithm described in Section 5.

The result of the maximum movement distance bound of $O(\log^{3/4} L)$ will still hold for certain irregularly shaped network fields. For example, the field can contain sparse convex holes with side length smaller than $O(\log^{3/4} L)$. When the maximum movement distance is $O(\log^{3/4} L)$, mobiles can bypass these holes when holes are not clustered together. Loosely speaking, the requirement on the shape of the field is that every subregion in the network field should have no less than a constant factor of its perimeter connected to other parts of the field (see the Proof of Theorem 2 in the Appendix). Therefore, the network field can be disks or convex polygons with side length larger than $O(\log^{3/4} L)$. However, if the field contains a thin strip with width smaller than $O(\log^{3/4} L)$ and length of $O(L)$, the maximum movement distance bound will no longer hold.

7 Numerical Results

7.1 All-Mobile Networks

We first consider the network where all sensors are mobiles. As shown in Section 3, providing k-coverage in an all-mobile network is the same as the 1-coverage case, but with the moving distance scaling as $1/\sqrt{k}$. Hence, we only consider the maximum matching distance for 1-coverage in our simulations.

In the simulation, $M = \Lambda L$ mobiles are uniformly and randomly scattered into the network with area of L, where Λ is fixed as $\frac{\pi}{2}$. Then, the mobiles are matched to M grid points on grids with side length of $d_s = \sqrt{2}r$ so that they can provide full coverage over the field (see Section 3.3). The matching is performed by a centralized linear programming algorithm described in Section 5.1. By repeating this over 10^5 randomly generated topologies, we find the probability that no feasible matching exists for a given maximum moving distance D.

Fig. 6a shows the probability that no feasible matching exists in different network sizes, where the moving distance is normalized by the grid size of d_s. From Fig. 6a, we see that the probability that no feasible matching exists quickly drops from 1 to 0 as the moving distance increases from $1.5d_s$ to $3.5d_s$. This phenomenon is a consequence of the fact that in random geometric graphs, monotone properties demonstrate critical threshold phenomena [35].

In this simulation, the network size is changed from 10×10 grids to 50×50 grids. Consider the moving distance which can ensure the network to be completely

covered by relocated mobiles with probability higher than 99.9 percent (where curves drop below 10^{-3} in Fig. 6a. We see that the moving distance is only increased by about $0.4d_s$, while the network size is increased by 25 times. Also, note that the maximum moving distance for large networks is small compared to the network size. For example, in a network with side length of $l = 50d_s$, the mobiles only need to move for at most $3.5d_s$ to form a regular grid deployment, which is less than one tenth of the network size. For larger networks, the difference will be even greater since the moving distance scales as $O(\log^{3/4} L)$. An online demo for networks smaller than 10×10 grids is available on [36].

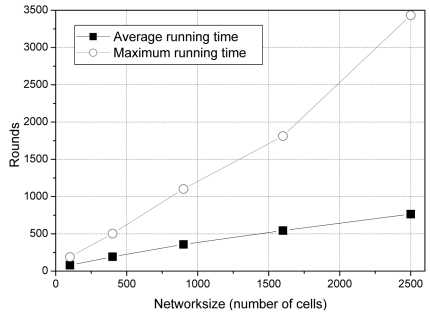
7.2 Hybrid Networks

In the simulation of hybrid networks, we divide the network area as cells with side length of $d_h = r/\sqrt{2}$, as in Section 4. We uniformly deploy $N = \lambda L$ static sensors and $M = \Lambda L$ mobiles in the network, where $\lambda = 2\pi k$, and M is selected so that there are exactly enough mobiles to fill all vacancies. The mobiles in one cell can move to cells within a distance of D. The following results are obtained by solving the linear program described in Section 5.1 on 10^5 randomly generated topologies for each network size.

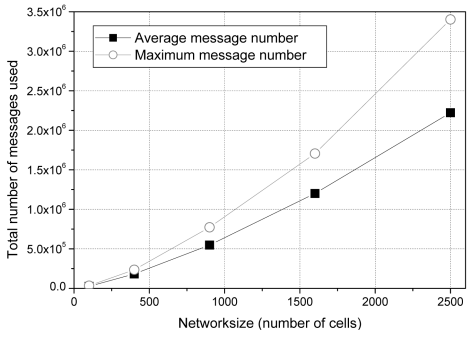
Fig. 6b shows the probability that there is no feasible mobility schedule to fill all vacancies under different network sizes when k is 10. In the hybrid network, the maximum moving distance for mobiles also increases slowly as the network size increases. Note that the moving distance in hybrid networks is normalized by d_h instead of d_s in the previous section. Since we have $d_h = 0.5d_s$, the actual moving distance for hybrid networks is comparable to the all-mobile case when $k = 1$. The moving distance for networks with varying k is plotted in Fig. 6c with network size of 900 cells. We see that the maximum moving distance required for hybrid networks slightly decreases as k increases, while the curve for $k = 50$ and $k = 100$ almost overlap. This shows that when k is small, the maximum moving distance is affected by both the matching distance from the mobile to the grid points and matching distance from the grid points to the vacancies. As k increases, the matching distance from the mobile to the grid points will decrease to zero as the mobile density increases (see Section 4). Then, the moving distance is dominated by the matching distance from the vacancies to the grid points which is not changed as k increases.

7.3 Performance of Push-Relabel Algorithm

We study the performance of the synchronous push-relabel algorithm which finds a feasible movement schedule without optimizing the total movement cost. The execution process is divided into rounds which contain two phases. In the first phase of each round, cells push excess flow to adjacent cells. If a relabel process is needed, the cell will relabel itself and inform neighboring cells at the second phase. In real networks, we can use an asynchronous algorithm which uses acknowledgment messages for push messages to relieve collisions in information updates [33]. Here, we study the synchronized version, which has similar performance as the asynchronous algorithm when collision rate is low. We execute the algorithm on 10^3 randomly generated topologies to get the average and the maximum running time in all topologies.



(a)



(b)

Fig. 7. Performance of push-relabel algorithm, $k = 10$ and $D = 6$ (confidence interval 95 percent). (a) Number of rounds used. (b) Total number of messages used.

Fig. 7a shows the number of rounds required by the algorithm. Although the upper bound of running time is $O(L^2)$ as shown in Section 5, the simulation shows that both the average and maximum running time increase linearly with L. For networks with 2,500 cells, we need to use on average 800 rounds to get the solution. Fig. 7b gives the number of messages used in the algorithm. By curve fitting, the number of messages increases empirically as $O(L^{1.4})$ when the network size increases, which is also much smaller than the bound. Note that the number of messages in Fig. 7b is the sum of messages sent by all cells in the network. When normalized by the number of cells in the network, the average number of messages sent by a single cell only increases sublinearly with the network size L. This hints that our problem is simpler than network flow problems on general graphs, since the mobiles only move to cells within distance of $O(\log^{3/4} L)$.

As this algorithm only executes once after the deployment, the transmission cost can be amortized over the lifetime of the network and become negligible in small networks. However, the algorithm may still consume considerable energy when the network size is extremely large. In that case, the role of delegator can be rotated between nearby sensors when a single cell is extremely highly loaded in the message exchange process.

8 Implementation

To demonstrate that our mobility algorithm is implementable and feasible in real-world situations, we implemented the push-relabel algorithm on real mobile sensor platforms.

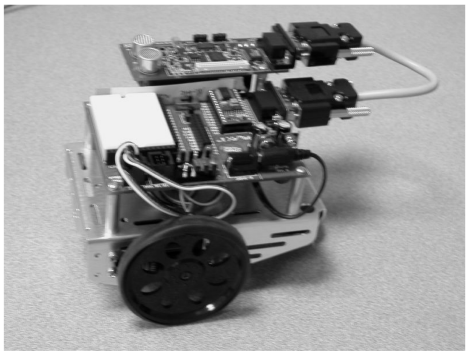


Fig. 8. The mobile sensor used in implementation.

Our mobile sensor platform uses an off-the-shelf mobile robot, called Boe Bot [13], to carry Cricket Motes developed at MIT [12]. Fig. 8 shows the mobile sensor platform. The Cricket Mote acts as the brain of the mobile sensor. It has one 4-MHz ATmega128L processor with 128-Kbyte instruction memory and 4-Kbyte RAM. It can also communicate with other mobile sensors through the embedded CC1000 radio transceiver. The Cricket platform is a popular indoor localization system for sensors. It measures time difference of arrival of the radio signal and ultrasound to perform range-based localization. The push-relabel algorithm is also implemented on the Cricket Mote. The Cricket Mote is connected to the Boe Bot through the serial port. The Boe Bot accepts simple instructions, such as "turn left" or "move forward", to perform actual movements.

The algorithm is implemented as described in Section 5. After deployment, the mobile sensors first calculate their location using the Cricket system.[2] Mobiles then send their locations to neighbors to obtain the number of mobiles in each cell. The delegates of each cell execute the push-relabel algorithm in a distributed fashion. Each cell maintains the height and excess of its own input and output vertexes. The cell also needs to keep track of the height of its neighboring cells defined as in Section 5. A push message is sent to the destination cell when a push operation is needed, and the push operation is successful only when the destination cell replies with a positive acknowledgment. If the cell relabels its vertices, it needs to broadcast the new height in a relabel message to its neighbors. The algorithm will terminate when the excess of all cells are zero. This can be inferred by individual mobiles when it has not heard any push or relabel message in the neighborhood for a long time. After that, the delegate of the cell will dispatch mobiles according to results of the distributed algorithm to achieve the optimal deployment.

We build up a test bed with nine mobile sensors on a field with 3×3 cells. Each mobile is allowed to move across one cell to achieve the optimal deployment which has exactly one mobile in each cell. In this case, at most four adjacent cells will be considered as neighbors. Since cells only need to keep track the height of neighboring cells, the memory requirement of our algorithm is quite low. As shown in Table 2, the push-relabel algorithm uses only 10,000-byte ROM and less than 300-byte RAM. In the field test, we see that mobiles can execute the algorithm and redeploy within 1 minute when the optimal solution exists (see the video available online [37]).

2. Unlike the original Cricket system, our triangulation algorithm is implemented on individual Cricket Motes.

TABLE 2
Size of the Mobility Algorithm

Program	ROM (bytes)	RAM (bytes)
Basic Cricket code (modified)	32,236	2,436
Added for mobility algorithm	10,218	267
Total	42,454	2,703

Communication links in the real world are unreliable, where mobiles may randomly lose data packets. Losing important messages, such as push, relabel, or push acknowledgment, may cause information inconsistency in neighboring cells. However, the problem can be solved by carefully designed algorithms. In our implementation, a cell always maintains the correct information about excess and height of itself. When a push message from neighboring cell contains incorrect information, the cell will reject the push and send back the correct information. Our tests in both real world and TOSSIM simulation environment show that the algorithm can work under a packet loss rate of more than 30 percent.

9 Conclusion

In this paper, we have investigated the distance that a mobile sensor will have to move in both all-mobile sensor networks and hybrid sensor networks. Our study formalizes the tradeoff that exists between an all-static network and a network with mobile sensors. Our results prove that from a scalability point of view, introducing mobility has significant advantages in providing coverage.

Appendix

Proof of Theorem 2. Our proof is similar to the proof given by Leighton and Shor for uniformly distributed points [11]. The major differences are that our vacancy distribution is not uniform and the vacancies are distributed in discrete cells rather than in a continuous field. In this proof, we focus on the differences and only give an outline for parts which are the same as in [11].

Consider the case of matching ΛL randomly distributed vacancies to the same number of grid points on the grid with side length of $\frac{1}{\sqrt{\Lambda}}$ by maximum matching distance of D. Define the neighborhood of a region R in the sensing field as $\mathcal{N}(R)$, the set of cells where all points are within a distance of D of at least one point in R. Note that R is also contained in $\mathcal{N}(R)$. It is easy to see that mobiles on grid points in $\mathcal{N}(R)$ can move into R by moving for a distance of D at most. By Hall's Theorem [38], there exists a perfect matching with maximum moving distance D between the vacancies and grid points *if and only if* for *every* subregion R in the field, the number of vacancies contained in R, denoted as V_R, is smaller or equal to the number of grid points in $\mathcal{N}(R)$, denoted as $W_{\mathcal{N}(R)}$.

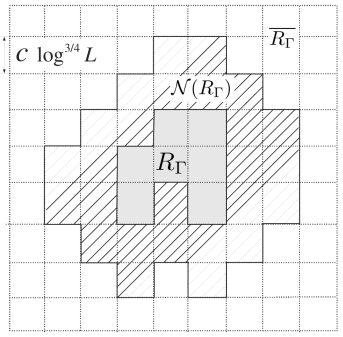


Fig. 9. A region and its neighborhood.

For our problem, we need to first define the number of vacancies in an arbitrary region which may not contain exactly an integer number of cells with side length $d_h = \frac{r}{\sqrt{2}}$. Denote the number of cells in region R as $A_R = Area(R)/d_h^2 = 2\pi Area(R)$. Suppose region R intersects with C_R cells, and it overlaps with only a fraction a_i of the total area of cell i, where $0 < a_i \leq 1$. Then, we have $A_R = \sum_{i=1}^{C_R} a_i$. Define the number of vacancies in R as $V_R = \sum_{i=1}^{C_R} a_i v_i$. In other words, if a region only covers part of the cell i, the number of vacancies contributed by the cell i will be $a_i v_i$. This definition "spreads" the vacancy in cell i uniformly on the area of a cell; thus, it has the property that the number of vacancies in a union of disjoint regions will be the sum of vacancies in individual regions.

Since we need to prove that $D = O(\log^{3/4} L)$, it is sufficient to only consider region R_Γ and $\mathcal{N}(R_\Gamma)$ with boundaries lying along the edges of squares Γ with side ength $c\log^{3/4} L$, where c is some constant [11], as shown in Fig. 9. We have

$$\begin{aligned} W_{\mathcal{N}(R_\Gamma)} &= \Lambda(Area(R_\Gamma) + Area(\mathcal{N}(R_\Gamma) \setminus R_\Gamma)) \\ &= \mathbb{E}\{V_{R_\Gamma}\} + \Lambda \times Area(\mathcal{N}(R_\Gamma) \setminus R_\Gamma), \end{aligned} \tag{19}$$

where $Area(R)$ is the area of region R. Define the *discrepancy* $\Delta(R) = |V_R - \mathbb{E}\{V_R\}|$. As shown in [11], we can make $Area(\mathcal{N}(R_\Gamma) \setminus R_\Gamma) \geq c_p Per(R_\Gamma)\log^{3/4} L$ for an arbitrarily constant c_p by setting $D = O(\log^{3/4} L)$, where $Per(R)$ is the perimeter of region R.[3] Thus, if we have $\Delta(R_\Gamma) \leq c_p \Lambda Per(R_\Gamma)\log^{3/4} L$ for all R_Γ, we can guarantee that $V_{R_\Gamma} \leq W_{\mathcal{N}(R_\Gamma)}$ also holds for all R_Γ.

By using the region decomposition method, it is shown in [11] that if for every region R (not necessarily having boundaries lying on Γ)

$$\mathbb{P}\{\Delta(R) \geq \delta\} < O\left(e^{-\frac{c_2\delta^2}{Area(R)}}\right) \; for \; \delta < Area(R), \tag{20}$$

$$\mathbb{P}\{\Delta(R) \geq \delta\} < O(e^{-c_2\delta}) \; for \; \delta > Area(R), \tag{21}$$

3. More strictly, this holds for at least one of R_Γ or its complement $\overline{R_\Gamma}$, so we need to show that $|V_R - \mathbb{E}\{V_{R_\Gamma}\}| \leq c_p \Lambda Per(R_\Gamma)\log^{3/4} L$ instead of $V_R - \mathbb{E}\{V_{R_\Gamma}\} \leq c_p \Lambda Per(R_\Gamma)\log^{3/4} L$. For details, see [11].

then *w.h.p.*, we will have $\Delta(R_\Gamma) \leq c_p Per(R_\Gamma)\log^{3/4} L$ for all regions R_Γ which have boundary lying on Γ. □

Therefore, we need the following lemma:

Lemma 1. *For any region R, we have the probability*

$$\mathbb{P}\{\Delta(R) \geq \delta\} < 2e^{-\frac{\delta^2}{4\pi Area(R)k}}, \tag{22}$$

when the vacancies are distributed according to (5).

Proof. Construct random variables $v_i' = v_i - \mathbb{E}\{v\}$ which have mean $\mathbb{E}\{v_i'\} = 0$. Accordingly, we have $V_R' = \sum_{i=1}^{C_R} a_i v_i' = V_R - \mathbb{E}\{V_R\}$ and $\Delta(R) = |V_R'|$. Since the number of vacancies in a cell cannot exceed k, we have $0 \leq V_R \leq A_R k$. Thus, we only need to consider $\delta < A_R k$, since $\mathbb{P}\{\Delta(R) \geq \delta\} = 0$ when $\delta \geq A_R k$.

We will bound the probability $\mathbb{P}\{\Delta(R) \geq \delta\}$ by the Chernoff bound. For $t > 0$, we have

$$\mathbb{P}\{V_R' \geq \delta\} \leq e^{-t\delta}\mathbb{E}\{e^{tV_R'}\}. \tag{23}$$

Since v_i' are independently distributed, we have

$$\mathbb{E}\{e^{tV_R'}\} = \prod_{i=1}^{C_R} \mathbb{E}\{e^{ta_i v_i'}\}. \tag{24}$$

Since $0 < a_i \leq 1$, the function $f(x) = x^{a_i}$ is concave when $x > 0$. By Jensen's inequality, we have $\mathbb{E}\{f(\mathbf{x})\} \leq f(\mathbb{E}\{\mathbf{x}\})$ for a concave function $f(x)$, we get

$$\mathbb{E}\{e^{ta_i v_i'}\} = \mathbb{E}\left\{(e^{tv_i'})^{a_i}\right\} \leq \left(\mathbb{E}\left\{(e^{tv_i'})\right\}\right)^{a_i} \tag{25}$$

for all i. Since v_i' are identically distributed, we have

$$\mathbb{P}\{V_R' \geq \delta\} \leq e^{-t\delta}\prod_{i=1}^{C_R}\left(\mathbb{E}\{e^{tv'}\}\right)^{a_i} = e^{-t\delta}\left(\mathbb{E}\{e^{tv'}\}\right)^{A_R}. \tag{26}$$

Using the vacancy distribution function of (5), we have

$$\begin{aligned} \mathbb{E}\{e^{tv'}\} &= \sum_{j=1}^{k} e^{t(j-\mathbb{E}\{v\})}\frac{k^{k-j}e^{-k}}{(k-j)!} + e^{-t\mathbb{E}\{v\}}\left(1 - \sum_{j=0}^{k-1}\frac{k^j e^{-k}}{j!}\right) \\ &< \sum_{l=0}^{k-1} e^{t(k-l-\mathbb{E}\{v\})}\frac{k^l e^{-k}}{l!} + e^{-t\mathbb{E}\{v\}} \\ &= e^{-t\mathbb{E}\{v\}}\left(e^{tk}\sum_{l=0}^{k-1} e^{-tl}\frac{k^l e^{-k}}{l!} + 1\right) \\ &< e^{-t\mathbb{E}\{v\}}\left(e^{tk-k}\sum_{l=0}^{\infty}\frac{(ke^{-t})^l}{l!} + 1\right) \\ &= e^{-t\mathbb{E}\{v\}}\left(e^{k(t+e^{-t}-1)} + 1\right), \end{aligned} \tag{27}$$

where the last equality comes from the expansion of $e^x = \sum_{l=0}^{\infty}\frac{x^l}{l!}$.

Consider two subcases:

1. $e^{k(t+e^{-t}-1)} \geq (e^{t\mathbb{E}\{v\}} - 1)^{-1}$.

The condition is equivalent to

$$e^{k(t+e^{-t}-1)} \geq \frac{e^{-t\mathbb{E}\{v\}}}{1 - e^{-t\mathbb{E}\{v\}}}. \tag{28}$$

Since $1 - e^{-t\mathbb{E}\{v\}} > 0$, (28) can be converted to

$$e^{k(t+e^{-t}-1)} \geq e^{-t\mathbb{E}\{v\}}\left(e^{k(t+e^{-t}-1)}+1\right) > \mathbb{E}\{e^{tv'}\}.$$

By (26), we get

$$\mathbb{P}\{V'_R \geq \delta\} < e^{kA_R(t+e^{-t}-1)-t\delta}.$$

Since $\delta < A_R k$, let $t = \log\frac{A_R k}{A_R k-\delta} > 0$, we get

$$\mathbb{P}\{V'_R \geq \delta\} < \exp\left((A_R k-\delta)\log\frac{A_R k}{A_R k-\delta}-\delta\right). \quad (29)$$

Using the inequality of $\log x \geq \frac{x^2-1}{2x}$ for $0 < x < 1$, we get

$$\begin{aligned}
&(A_R k-\delta)\log\frac{A_R k}{A_R k-\delta}-\delta \\
&= -A_R k\left(1-\frac{\delta}{A_R k}\right)\log\left(1-\frac{\delta}{A_R k}\right)-\delta \\
&\leq -A_R k\left(\frac{(1-\frac{\delta}{A_R k})^2-1}{2}+\frac{\delta}{A_R k}\right) \\
&= -\frac{\delta^2}{2A_R k}.
\end{aligned} \quad (30)$$

So, we get $\mathbb{P}\{V'_R \geq \delta\} < e^{-\frac{\delta^2}{2A_R k}}$.

2. $e^{k(t+e^{-t}-1)} < (e^{t\mathbb{E}\{v\}}-1)^{-1}$.

In this case, we have

$$\mathbb{E}\{e^{tv'}\} < e^{-t\mathbb{E}\{v\}}\left(\frac{1}{e^{t\mathbb{E}\{v\}}-1}+1\right) = \frac{1}{e^{t\mathbb{E}\{v\}}-1}.$$

By (26), we get

$$\mathbb{P}\{V'_R \geq \delta\} \leq \left(\frac{1}{e^{t\mathbb{E}\{v\}}-1}\right)^{A_R} e^{-t\delta}. \quad (31)$$

As $\mathbb{E}\{v\} \geq e^{-1}$ for any k (by (7)), we can select $t = 3$, so that $e^{t\mathbb{E}\{v\}} - 1 > e - 1 > 1$. We have

$$\begin{aligned}
\mathbb{P}\{V'_R \geq \delta\} &< \left(\frac{1}{e^{3\mathbb{E}\{v\}}-1}\right)^{A_R} e^{-3\delta} \\
&< e^{-3\delta} \leq e^{-3\frac{\delta^2}{A_R k}} < e^{-\frac{\delta^2}{2A_R k}}
\end{aligned}$$

due to $0 < \frac{\delta}{A_R k} < 1$.

Consider both cases, we have $\mathbb{P}\{V'_R \geq \delta\} < e^{-\frac{\delta^2}{2A_R k}}$ when $\delta < A_R k$. For $\delta \geq A_R k$, it is easy to see that $\mathbb{P}\{V'_R > \delta\} = 0$.

We can also get

$$\mathbb{P}\{V'_R \leq -\delta\} < e^{-\frac{\delta^2}{2A_R k}} \quad (32)$$

for the other side of the distribution with some different derivations. The details are omitted due to space limits.

Combining the results for both sides, we have $\mathbb{P}\{|V'_R| \geq \delta\} < 2e^{-\frac{\delta^2}{2A_R k}}$.

This is equivalent to

$$\mathbb{P}\{\Delta(R) \geq \delta\} < 2e^{-\frac{\delta^2}{4\pi Area(R)k}}, \quad (33)$$

as $A_R = 2\pi Area(R)$. □

Note that our discrepancy bound is $c_p\Lambda Per(R_\Gamma)\log^{3/4} L$ instead of $c_p Per(R_\Gamma)\log^{3/4} L$. Therefore, in our problem, the discrepancy should be scaled by a factor of Λ, which is the density of the grid points. We have $\Lambda \approx \sqrt{2\pi k}$. Put $\delta' = \Lambda\delta$ into (33), we can see that the factor of k in the exponent in (33) will be canceled by Λ^2. In this case, it is easy to see that the bound in Lemma 1 satisfies (20) and (21) for both cases of $\delta \leq Area(R)$ and $\delta > Area(R)$. Therefore, using the same region decomposition method as in [11], we have $\Delta(R_\Gamma) \leq c_1\Lambda Per(R_\Gamma)\log^{3/4} L$ w.h.p. This directly leads to the upper bound of $O(\log^{3/4} L)$ in maximum moving distance.

Proof of Theorem 3. The Proof of Theorem 3 is similar to the Proof of Theorem 2. Consider a region R with exactly A_R cells which has side length of d. Since the number of vacancies in each cell is bounded in $[0, k]$, by Hoeffding's inequality [39], we have

$$\mathbb{P}\{|V_R - \mathbb{E}\{V_R\}| \geq \delta\} < e^{-\frac{2\delta^2}{A_R k^2}}. \quad (34)$$

With similar arguments as the proof in Theorem 2, this bound can be extended to regions which do not contain exactly an integer number of cells, and we get

$$\mathbb{P}\{|V_R - \mathbb{E}\{V_R\}| \geq \delta\} < e^{-\frac{2d^2\delta^2}{Area(R)k^2}}. \quad (35)$$

Comparing (35) and (33), we see that any region will have discrepancy less than $c_p k Per(R_\Gamma)\log^{3/4} L$ *w.h.p.* This gives an upper bound on moving distance of $O(\frac{k}{\Lambda}\log^{3/4} L)$ since the grid point density is Λ. Note that the bound of Theorem 3 is $\sqrt{k}$ times larger than the bound in Theorem 2 when we set $\Lambda = O(\sqrt{k})$. This is because of the Chernoff bound we derived in Theorem 2 for the specific vacancy distribution is tighter than the Hoeffding bound used here. □

ACKNOWLEDGMENTS

A shorter version of this work has appeared in ACM MobiCom 2007 [1].

REFERENCES

[1] W. Wang, V. Srinivasan, and K.C. Chua, "Trade-Offs between Mobility and Density for Coverage in Wireless Sensor Networks," *Proc. ACM MobiCom '07*, pp. 39-50, 2007.

[2] S. Meguerdichian, F. Koushanfar, M. Potkonjak, and M. Srivastava, "Coverage Problems in Wireless Ad-Hoc Sensor Network," *Proc. IEEE INFOCOM '01*, pp. 1380-1387, 2001.

[3] C. Huang and Y. Tseng, "The Coverage Problem in a Wireless Sensor Network," *Proc. Second ACM Int'l Workshop Wireless Sensor Networks and Applications (WSNA '03)*, pp. 519-528, 2003.

[4] H. Zhang and J. Hou, "On Deriving the Upper Bound of α-lifetime for Large Sensor Networks," *Proc. ACM MobiHoc '04*, pp. 121-132, 2004.

[5] S. Kumar, T. Lai, and J. Balogh, "On k-Coverage in a Mostly Sleeping Sensor Network," *Proc. ACM MobiCom '04*, pp. 144-158, 2004.

[6] P.-J. Wan and C.-W. Yi, "Coverage by Randomly Deployed Wireless Sensor Networks," *IEEE Trans. Information Theory*, vol. 52, no. 6, pp. 2658-2669, 2006.

[7] G. Wang, G. Cao, and T.L. Porta, "Movement-Assisted Sensor Deployment," *Proc. IEEE INFOCOM '04*, pp. 2469-2479, 2004.

[8] Y. Zou and K. Chakrabarty, "Sensor Deployment and Target Localization Based on Virtual Forces," *Proc. IEEE INFOCOM '03*, pp. 1293-1303, 2003.

[9] S. Chellappan, X. Bai, B. Ma, and D. Xuan, "Sensor Networks Deployment Using Flip-Based Sensors," *Proc. Second IEEE Int'l Conf. Mobile Ad-Hoc and Sensor Systems (MASS),* 2005.
[10] J.T. Feddema, R.H. Byrne, J.J. Harrington, D.M. Kilman, C.L. Lewis, R.D. Robinett, B.P.V. Leeuwen, and J.G. Young, "Advanced Mobile Networking, Sensing, and Controls," Technical Report SAND2005-1661, Sandia Nat'l Laboratories, 2005.
[11] T. Leighton and P.W. Shor, "Tight Bounds for Minimax Grid Matching, with Applications to the Average Case Analysis of Algorithms," *Combinatorica,* vol. 9, no. 2, pp. 161-187, 1989.
[12] *MIT Cricket Platform,* http://cricket.csail.mit.edu/, 2008.
[13] *Parallax,* http://www.parallax.com/html_pages/robotics/boebot/boebot.asp, 2008.
[14] B. Liu, P. Brass, O. Dousse, P. Nain, and D. Towsley, "Mobility Improves Coverage of Sensor Networks," *Proc. ACM MobiHoc '05,* pp. 300-308, 2005.
[15] M. Zhang, X. Du, and K. Nygard, "Improving Coverage Performance in Sensor Networks by Using Mobile Sensors," *Proc. IEEE Military Comm. Conf. (MILCOM '05),* pp. 3335-3341, 2005.
[16] J. Teng, T. Bolbrock, G. Cao, and T.L. Porta, "Sensor Relocation with Mobile Sensors: Design, Implementation, and Evaluation," *Proc. Fourth IEEE Int'l Conf. Mobile Ad-Hoc and Sensor Systems (MASS),* 2007.
[17] J. Wu and S. Yang, "SMART: A Scan-Based Movement-Assisted Sensor Deployment Method in Wireless Sensor Networks," *Proc. IEEE INFOCOM '05,* pp. 2313-2324, 2005.
[18] J. Luo and J.P. Hubaux, "Joint Mobility and Routing for Lifetime Elongation in Wireless Sensor Networks," *Proc. IEEE INFOCOM '05,* pp. 1735-1746, 2005.
[19] W. Wang, V. Srinivasan, and K.C. Chua, "Using Mobile Relays to Prolong the Lifetime of Wireless Sensor Networks," *Proc. ACM MobiCom '05,* pp. 270-283, 2005.
[20] Z.M. Wang, S. Basagni, E. Melachrinoudis, and C. Petrioli, "Exploiting Sink Mobility for Maximizing Sensor Networks Lifetime," *Proc. 38th Ann. Hawaii Int'l Conf. System Science (HICSS),* 2005.
[21] S. Chellappan, W. Gu, X. Bai, and D. Xuan, "Deploying Wireless Sensor Networks under Limited Mobility Constraints," *IEEE Trans. Mobile Computing,* vol. 6, no. 10, pp. 1142-1157, 2007.
[22] Z. Lotker and A. Navarra, "Unbalanced Points and Vertices Problem," *Proc. First IEEE PERCOM Int'l Workshop Foundations and Algorithms for Wireless Networking (FAWN '06),* pp. 96-100, 2006.
[23] Z. Lotker and A. Navarra, "Managing Random Sensor Networks by Means of Grid Emulation," *Proc. Fifth Int'l IFIP-TC6 Networking Conf. (NETWORKING '06),* pp. 856-867, 2006.
[24] P. Hall, *Introduction to the Theory of Coverage Processes.* John Wiley & Sons, 1988.
[25] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill, "Integrated Coverage and Connectivity Configuration in Wireless Sensor Networks," *Proc. First ACM Conf. Embedded Networked Sensor Systems (SenSys '03),* pp. 28-39, 2003.
[26] R. Williams, *The Geometrical Foundation of Natural Structure: A Source Book of Design.* Dover Publications, 1979.
[27] G. Wang, G. Cao, and T.L. Porta, "Proxy-Based Sensor Deployment for Mobile Sensor Networks," *Proc. First IEEE Int'l Conf. Mobile Ad-hoc and Sensor Systems (MASS),* 2004.
[28] G. Wang, G. Cao, T.L. Porta, and W. Zhang, "Sensor Relocation in Mobile Sensor Networks," *Proc. IEEE INFOCOM '05,* pp. 2302-2312, 2005.
[29] E. Kratzel, *Lattice Points.* Kluwer Academic Publishers, 1989.
[30] A. Papoulis and S.U. Pillai, *Probability, Random Variables and Stochastic Processes,* fourth ed. McGraw Hill, 2002.
[31] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin, *Network Flows: Theory, Algorithms, and Applications.* Prentice Hall, 1993.
[32] T.H. Cormen, C.E. Leiserson, R.L. Rivest, and C. Stein, *Introduction to Algorithms,* second ed. MIT Press and McGraw-Hill, 2001.
[33] A. Goldberg and R. Tarjan, "A New Approach to the Maximum-Flow Problem," *J. ACM,* vol. 35, no. 4, pp. 921-940, 1988.
[34] A. Goldberg and R. Tarjan, "Finding Minimum-Cost Circulations by Successive Approximation," *Math. of Operations Research,* vol. 15, no. 3, pp. 430-466, 1990.
[35] A. Goel, S. Rai, and B. Krishnamachari, "Sharp Thresholds for Monotone Properties in Random Geometric Graphs," *Proc. 36th ACM Symp. Theory of Computing (STOC '04),* pp. 580-586, 2004.
[36] *Web Based Demo for Mobile Sensors,* http://cnds.ece.nus.edu.sg/mobile/mobile.html, 2008.
[37] *Video of Mobile Implementations,* http://cnds.ece.nus.edu.sg/mobile/mobile4.mpg, 2008.
[38] S. Janson, T. Luczak, and A. Ruciński, *Random Graphs.* John Wiley & Sons, 2000.
[39] W. Hoeffding, "Probability Inequalities for Sums of Bounded Random Variables," *J. Am. Statistical Assoc.,* vol. 58, no. 301, pp. 13-30, 1963.

Wei Wang received the BEng and MSc degrees in electronics science and engineering from Nanjing University, China, in 1997 and 2000, respectively. He is currently working toward the PhD degree in the Department of Electrical and Computer Engineering, National University of Singapore, Singapore. His research interests include wireless sensor networks. He is a student member of the IEEE.

Vikram Srinivasan received the BS degree in physics from the University of Chennai in 1994, the ME degree in electrical and communications engineering from the Indian Institute of Science, Bangalore in 1998, and the PhD degree in electrical and computer engineering from the University of California, San Diego, in 2003. From 2003 to 2007, he was an assistant professor in the Department of Electrical and Computer Engineering, National University of Singapore. He is currently with Bell Labs Research, Bangalore, India. His research interests include wireless networks. He is a member of the IEEE.

Kee-Chaing Chua received the PhD degree in electrical engineering from the University of Auckland, New Zealand, in 1990. He joined the National University of Singapore (NUS) as a lecturer in 1990 and is currently a professor in the Department of Electrical and Computer Engineering. He served as the Faculty of Engineering's vice dean for research from June 2003 to March 2006. From 1995 to 2000, he was seconded to the Center for Wireless Communications (now part of the Institute for Infocomm Research), a national telecommunication R&D center funded by the Singapore Agency for Science, Technology, and Research as its deputy director. From 2001 to 2003, he was on leave of absence from NUS to work at Siemens Singapore, where he was the founding head of the Mobile Core R&D Department funded by Siemens' ICM Group. Since March 2006, he has been seconded to the National Research Foundation as a director. He has carried out research in various areas of communication networks and has published more than 200 papers in these areas in international refereed journals and conferences. His current research interests include wireless networks (in particular, wireless sensor networks) and optical burst switched networks. He has also been an active member of the IEEE and is a recipient of an IEEE Third Millennium medal.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.