Sentry Selection in Sensor Networks: Theory and Algorithms

Paul Balister[*‖], Béla Bollobás[†*‖**], Martin Haenggi[‡], Amites Sarkar[§], Mark Walters[¶]

[*]Department of Mathematical Sciences, University of Memphis, Memphis TN 38152, USA. pbalistr@memphis.edu
[†]Trinity College, Cambridge CB2 1TQ, UK. B.Bollobas@dpmms.cam.ac.uk
[‡]Department of Electrical Engineering, University of Notre Dame, Notre Dame IN 46556, USA. mhaenggi@nd.edu
[§]Department of Mathematics, Western Washington University, Bellingham WA 98225, USA. amites.sarkar@wwu.edu
[¶]School of Mathematical Sciences, Queen Mary University of London, London E1 4NS, UK. M.Walters@qmul.ac.uk
[‖]Supported by NSF grant DMS 1301614 [**]Supported by MULTIPLEX grant 317532

***Abstract*—We study the geometric aspects of the sentry selection problem for a sensor network with randomly distributed sensors each of which covers a small disc. The problem is to partition the sensors into a predetermined number of groups, so that, for each group, the sensing regions of the sensors in that group together cover the region of interest. After presenting some theoretical results, we include descriptions of two fast partitioning algorithms, together with the results of simulating them on moderate-sized networks.**

Keywords Wireless sensor networks; sentry selection; Gilbert model; random coverings; recolouring algorithms; geometric probability.

Paul Balister is a Professor and the Associate Chair of the Department of Mathematical Sciences at the University of Memphis. He obtained his PhD from the University of Cambridge, England in 1992, and subsequently held positions at Harvard and Cambridge. He has worked in algebraic number theory, combinatorics, graph theory and probability theory.

Béla Bollobás FRS is a Senior Research Fellow of Trinity College, Cambridge, England, and holds the Jabie Hardin Chair of Excellence in Graph Theory and Combinatorics at the University of Memphis. He received PhDs from Eötvös Loránd University, Budapest, Hungary, and the University of Cambridge, England. He works in extremal and probabilistic combinatorics, polynomials of graphs, combinatorial probability and percolation theory. He has written ten books, including Extremal Graph Theory, Random Graphs and Modern Graph Theory, and over 400 research papers. He has supervised over 50 doctoral students, many of whom hold leading positions in prominent universities throughout the world.

Martin Haenggi is a Professor of Electrical Engineering and a Concurrent Professor of Applied and Computational Mathematics and Statistics at the University of Notre Dame, Indiana, USA. He received the Dipl.-Ing. (MSc) and Dr.sc.techn. (PhD) degrees in electrical engineering from the Swiss Federal Institute of Technology in Zurich (ETH). He has written two books, Interference in Large Wireless Networks, and Stochastic Geometry for Wireless Networks. His scientific interests include networking and wireless communications, with an emphasis on ad hoc, cognitive, cellular, sensor, and mesh networks.

Amites Sarkar is an Associate Professor at Western Washington University. He obtained his PhD from the University of Cambridge in 1998, and he has worked in combinatorics and probability theory.

Mark Walters is a Senior Lecturer at Queen Mary University of London. He obtained his PhD from the University of Cambridge in 2000, and was a Research Fellow of Trinity College, Cambridge. He has worked in combinatorics and probability theory.

I. Introduction

In this paper we shall study the problem of *sentry selection* in a network consisting of many small wireless sensors scattered at random over a large region for the purpose of monitoring it. This is currently a well studied topic for two reasons. Firstly, the sensors themselves have become progressively smaller and increasingly mobile, so that it now makes sense to assume that their locations are *random*, rather than given by some predetermined arrangement, such as a grid. Secondly, for many applications, the sensor batteries are not rechargable. Once most of the sensor batteries have failed, the network is permanently disabled. Even when the batteries are rechargable, they cannot be recharged while the sensor is sensing [12]. Consequently, in both cases, it is very important to conserve energy, switching off sensors as they become redundant and only switching them on again when absolutely necessary. It turns out that the natural redundancy in a random network makes this possible.

Most of the previous work in this area (see [10], [21] and the references therein) has focused on protocol design. Our purpose here is to examine the geometric aspects of the problem. We will consider a random *static* arrangement of sensors with a common sensing range r, and our analysis will be based purely in terms of the abstract set of sensing discs drawn in the plane. Indeed, a good way to visualize the basic setup is to picture a large number of identical small thin discs scattered over a square tabletop: each disc represents the sensing area of a sensor, and all other aspects of the sensors (such as mobility and interference) are ignored.

It is important to distinguish between those monitoring applications that require *coverage* and those which only require *percolation* (termed "detectability" in [14]). If the sensors are intended to detect a fire originating anywhere in a large forest, every point of the forest has to lie within range of some sensor, i.e., in one of the sensing discs. If, however, the sensors are intended to detect a cougar moving through the forest, full coverage is not necessary: all we need is that the

cougar cannot wander across the (square) forest from left to right without falling within range of a sensor. If this is the case then we say that there is *percolation* in the network. For percolation, the sensing discs do not have to cover the whole region, but some of them must form a (perhaps quite circuitous) barrier between two opposite sides.

In this paper we shall consider those applications requiring coverage, so that we are interested in detecting fires rather than cougars. We remark, however, that for the cougar detection problem, it is actually much simpler to ensure that no cougars even enter the forest. This can be achieved by merely guarding the forest perimeter, and we are thus led to the topic of *barrier coverage* in thin strips, which is the subject of [3] and [4].

The distinction between percolation and coverage is far greater for random networks than for grid-based networks. For instance, consider a sensor network based on a large square grid, with sensors located at points (i, j) in the plane, where i and j are integers with $0 \leq i, j \leq 1000$, say. In this model, $r > 1/2$ guarantees "percolation" (in the sense described above), and $r > 1/\sqrt{2}$ guarantees coverage of the square $[0, 1000]^2$. But for random networks, coverage comes at a much higher price. Let us suppose that we place $n = 10^6$ sensors uniformly at random in the same square $[0, 1000]^2$, so that we have one sensor per unit area, as before. It turns out that we need $r \approx 0.6$ for percolation (see [6] for this and related results), but, for coverage, r has to be much larger: we need $r \approx 2.3$. Moreover, the threshold value $r \approx 0.6$ for percolation is an absolute constant (for sensors distributed at random with one per unit area), while the threshold for coverage is an *increasing function* of n, the total number of sensors.

The reason for this is that coverage, unlike percolation, is determined by the "holes" in the network. As the number of sensors and the size of the region increase (keeping one sensor per unit area), so does the size of the largest hole. We will not have full coverage unless this hole is covered, and, if there are n sensors, the largest hole is typically so large that we need to make r about $\sqrt{\log n/\pi}$ (more precisely we need $\pi r^2 \approx \log n + \log\log n$ – see later) to cover it. Of course, in our model, r is the same for each sensor, so that for most of the rest of the region this measure will be a tremendous overkill. Indeed, if $\pi r^2 = \log n$, a typical point in our region will be covered by $\log n$ sensing discs, so that the vast majority of the sensors will actually be redundant.

However, perhaps we can exploit this redundancy in the following manner. We would like to devise a rota system so that each sensor can *sleep* for most of the time. We plan to partition the set of sensors into k groups, and arrange that only the sensors in group ℓ are active in the ℓ^{th} time slot. After k time slots have expired, we will repeat the process. In order to detect an event (e.g. a fire) occurring anywhere and at any time, it is necessary that the sensors in each group themselves form a single cover of the sensing region. So, finally, here is our basic question:

(I) For fixed n and k, how large should the sensing radius r be to ensure that our n randomly placed sensors can be partitioned into k groups, each of which covers the sensing region A (of area n)?

We call this the problem of *sentry selection*, since each of the groups is a group of sentries keeping watch over the region while the others are sleeping. Clearly, if we can devise such a partition, the lifetime of each sensor, and consequently the entire network, will increase by almost a factor of k (the sensors probably consume *some* energy in sleep mode).

Of course, our hope is that the answer to Question (I) will be "We only need to make r large enough to ensure coverage", a hope justified by the high level of average coverage provided by such an r. However, there is a problem. If we can partition the set of sensors into k groups, each of which provides coverage, then the original set of sensors must provide k-coverage: every point of the sensing region must lie within the sensing range of k sensors. So r also has to be large enough to ensure this. Fortunately, it turns out that the necessary increase in r is very small – ignoring boundary effects, the threshold for k-coverage is only $\pi r^2 \approx \log n + k \log\log n$.

In any case, we are led to the following reformulation of (I):

(II) Suppose that we have obtained a k-cover $\mathcal{C}$ of a large sensing region A using randomly placed sensors. What is the probability that we can partition $\mathcal{C}$ into k single covers of A?

Note that this is essentially the same as (I), except that we are now explicitly concerned with the gap between k-coverage and "k-partitionability". Also, note that we have not yet specified the exact model and parameters, details of which appear in the next section.

Leaving aside the random aspect of (I) and (II) for the moment, we turn briefly to the topic of partitioning a geometric k-cover into $l \leq k$ single covers, which has a rich mathematical history. It was introduced by L. Fejes Tóth in the 1970s, and, although there are now many results in the area (see [18] and [19] for a sample), it is surprising how little is known about some of the most basic questions. Pach and Pálvölgyi [17] proved only in 2013 that, for all k, there exists a k-cover of the plane with unit discs that is not 2-partitionable. While their constructions are complicated, there are simple examples of 2-covers which are not 2-partitionable,

and these examples play a central rôle in the solution of Question (II). In fact, as part of our study, we discuss a recent classification theorem for (non-random) non-2-partitionable 2-covers of the plane with half-planes, which, somewhat surprisingly, has implications for random k-covers with discs and, consequently, the solution of Question (II).

In three dimensions, a much older result of Mani-Levitska and Pach [15] states that for any k, there exists a k-cover of space with unit balls that is not 2-partitionable. Of course, there is no need to consider just unit balls, and the papers [17], [18] and [19] contain results on planar covers with convex polygons, infinite strips and straight lines. Many of these results are negative, in the sense that there is usually, for arbitrarily large k, a k-cover $\mathcal{C}_k$ of the plane with certain shapes that is not even 2-partitionable. However, when studying random coverings, such as those arising in (I) and (II), we have a distinct advantage: perhaps all the bad covers $\mathcal{C}_k$ occur with very small probability. It turns out that this is indeed the case, at least for discs, so that our eventual answer to (II) will be: "The probability tends to one as the number of sensors increases". Detailed results are given in the next section.

From a practical point of view, it is no use knowing that there is a partition if we cannot find it quickly. Therefore, we must consider the following algorithmic problem:

(III) Suppose we know that a certain k-cover $\mathcal{C}$ of a large sensing region A using randomly placed sensors is k-partitionable. Is there a fast algorithm for finding the partition?

We will think of colouring the sensors (or, equivalently, the sensing discs) with k colours, so that the sensors of each colour form the parts of the partition. In this paper we present two colouring/partitioning algorithms, and our results suggest that the answer to (III) is "Yes".

A. Related work

One of the earliest relevant papers was written by Slijepcevic and Potkonjak [20], who discuss Question (III), and provide a partitioning algorithm which runs in time $O(n^2)$, based on the idea of minimizing the coverage level of sparsely covered areas within one cover (throughout this paper, n is the number of sensors). Abrams, Goel and Plotkin [2] consider a variant of the problem, where the objective is to partition the sensors into covers so that the number of covers that include an area, summed over all areas, is maximized. Finally, Liu and Haenggi [13] present a lattice-based scheme for selectively activating and de-activating randomly placed sensors to conserve energy: however, this doesn't result in disjoint covers of the region, since a sensor might be active in two different time slots.

More recently, in [22], the authors analyze the performance of a randomized scheduling algorithm with the aim of maximizing the probability of intruder detection; in contrast, we propose an adaptive scheduling algorithm for which the probability of intruder detection is one. Adaptive scheduling algorithms have also been considered in [7] and [1]. Ding, Wang and Xiao [7] aim to partition the sensors in a network so that the sensors of each group retain network *connectivity* (rather than coverage); thus their work has little overlap with ours. AbdelSalam and Olariu [1] consider an asynchronous network where the sensors adjust their sleep time based on their remaining energy and that of neighboring sensors; however, their network is a *tasking network* rather than a simple sensing network, and their aim was to extend network lifetime while retaining a certain quality of service, rather than to guarantee complete coverage. To summarize, although several related ideas and algorithms have been considered in the literature, they all deal with different problems, and so none of this previous work is directly comparable with ours.

The rest of this paper is organized as follows. In Section II, we will first of all state problems (I) and (II) precisely, before giving some theoretical results. Then, in Section III, we will describe two colouring algorithms, and present the results from both of them in detail. We will see that there are surprising connections between the theoretical and computational aspects of the problem.

II. Theoretical results

Here are the exact specifications of our model. We consider n points (representing sensors) placed uniformly at random in a unit (1×1) *torus* T, so that the coordinates of every point are taken modulo one. Boundary effects are not always negligible in problems of this kind: however in this paper we shall avoid them for simplicity. Surround each of our n points p_i in T by the open disc $D_r(p_i)$ of radius $r \ll 1$ centred at p_i. We say that the union of these discs $\mathcal{C}$ forms a k-cover of T if each point $t \in T$ lies in at least k discs of $\mathcal{C}$, and that $\mathcal{C}$ is k-partitionable if the discs of $\mathcal{C}$ can be coloured with k colours so that the discs of each colour themselves form a 1-cover of T. Suppressing the dependence on n, we will write E_r^k for the event that $\mathcal{C}$ is a k-cover, and F_r^k for the event that $\mathcal{C}$ is k-partitionable. Note that problems (I) and (II) of the introduction are, in this context, "What is $\mathbb{P}(F_r^k)$?" and "What is $\mathbb{P}(F_r^k|E_r^k)$?" respectively.

Note also that we are using a different normalization in this section: in the introduction we took one sensor per unit area so as to compare our model to a lattice-based model, whereas it is more convenient to perform simulations in a fixed area, with an increasing number (and, consequently, density) of sensors. Thus the ex-

pected number of sensors in a sensing region is now $\pi r^2 n$, as opposed to πr^2.

Our first priority is to estimate $\mathbb{P}(E_r^k)$. The following theorem was proved by Janson [11], who extended a result of Hall [9], who applied a method of Gilbert [8] to a question of Moran and Fazekas de St Groth [16], which was motivated by a problem in biology (on antibodies).

Theorem 1. *If*

$$\pi r^2 n = \log n + k \log\log n + x,$$

then

$$\mathbb{P}(E_r^k) \to e^{-e^{-x}/(k-1)!}$$

as $n \to \infty$.

The problem of determining $\mathbb{P}(E_r^k)$ and $\mathbb{P}(F_r^k)$ exactly seems hopelessly intractable. Hence, one goal of our simulations is to estimate these probabilities for moderate values of n. Although results such as Theorem 1 are only asymptotic, our simulation results suggest that the convergence is usually pretty rapid.

Here is the answer to Question (II):

Theorem 2. *With* $r \in \mathbb{R}$ *and* $n, k \in \mathbb{N}$,

$$\mathbb{P}(E_r^k \setminus F_r^k) \leq \frac{c_k}{\log n}.$$

Consequently, as long as $\mathbb{P}(E_r^k) = \Theta(1)$, $\mathbb{P}(F_r^k | E_r^k) = 1 - o(1)$. Putting this together with Theorem 1, we obtain the following answer to Question (I).

Theorem 3. *If*

$$\pi r^2 n = \log n + k \log\log n + x,$$

then

$$\mathbb{P}(F_r^k) \to e^{-e^{-x}/(k-1)!}$$

as $n \to \infty$.

The proof of Theorem 2, which is rather complicated, is given in [5]. The same paper also contains a short proof of Theorem 1. Several steps in the proof of Theorem 2 are also used implicitly in our colouring algorithms, so we will postpone a more detailed discussion until the next section. For now, we only remark that Theorem 2 is best possible, up to the value of the constant c_k.

III. Algorithms and obstructions

As we have already remarked, if r is high enough to guarantee k-coverage, most of our sensing region T is very heavily covered. Consequently, we can just colour the sensors with k colours completely at random (so that each colour is used on each sensor with probability $1/k$) and hope for the best. If some part B of T is covered c times by the discs $D_r(p_i)$, then the probability p_{fail} that our random colouring fails at B, that is, the probability that B is not covered by discs of each colour, satisfies

$$p_{\text{fail}} \leq k \left(1 - \frac{1}{k}\right)^c \approx k e^{-c/k},$$

and so is usually very small since c will generally be much larger than k. Indeed, this observation, together with the Lovász local lemma, can be used to show that Theorem 2 holds as long as r is not too close to the threshold for k-coverage. However, we are most interested in precisely this threshold range. For values of r just above the threshold, there will be several "atomic" regions of T (topological components of $T \setminus \bigcup_i \partial D_r(p_i)$) which are covered, say, between k and $10k$ times. For these regions, p_{fail} will be (approximately!) somewhere between $10e^{-10}$ and e^{-1}, and, sooner or later, our colouring will in fact fail on one of them. The following sections describe two ways of avoiding this problem. The first, which is fully distributed, involves recolouring, while the second, centralized, algorithm starts by colouring the thinly covered regions first.

A. Threshold recolouring

For this algorithm, the first step is to randomly colour each sensor as above. Then, for each *sensor* p, we measure the total received power at p (assuming a path loss law of the form $P = d^{-\alpha}$), if all the sensors receiving the same colour as p were to transmit simultaneously. If this power is above a certain threshold, we recolour p. We do this simultaneously for all n sensors (or, alternatively, in k rounds, one for each colour class), and repeat. The hope is that successive iterates of this process will converge to a more balanced colouring. Note that this method ensures that two nearby sensors coloured identically in one round are likely to receive different colours in the next. Exactly what we mean by "nearby" and "likely" is determined by the path loss exponent α, which we can use to fine-tune the algorithm. Also, for the simulation results below, the threshold was chosen so that 30% of sensors were recoloured in the first round: this threshold was then fixed for subsequent rounds, up to a maximum of 10 rounds. Consequently, the algorithm runs in time $O(1)$, and requires $O(n)$ operations in total. The pseudecode for each trial run of the algorithm is presented as Algorithm 1.

This algorithm has several noteworthy features. First, there is some reason to hope that it will converge to a colouring which is *balanced*, meaning that each colour class not only covers T, but covers it uniformly, in some sense. Second, there is no requirement that the nodes can communicate: each node only needs to be able to measure received power, and is independent in all other aspects of its operation.

Algorithm 1 Threshold recolouring

randomly place n sensors on the torus
color each sensor p randomly with one of k colours
if F_r^k does not occur **then**
    declare "random fail"
    $S \leftarrow$ colour sequence
    calculate received power P_p from sensors of same colour
    set threshold θ to 70-th percentile of (P_p)
    $b \leftarrow |\{p\colon P_p > \theta\}|$ % number of "bad" sensors
    $b_{\min} \leftarrow b$
    $i \leftarrow 0$
    while $i < 10$ and $b > 0$ **do**
        $i \leftarrow i + 1$
        re-colour each sensor for which $P_p > \theta$
        calculate received power P_p from sensors of same colour
        $b \leftarrow |\{p\colon P_p > \theta\}|$
        if $b < b_{\min}$ **then**
            $b_{\min} \leftarrow b$
            $S \leftarrow$ colour sequence % save best colouring
        end if
    end while
end if
restore color sequence S
if F_r^k occurs **then**
    declare "recoloured"
end if

$1/r$	k	n	random fail	recoloured	successes
8	2	375	255	151	896
8	3	600	255	205	950
8	4	800	301	252	951
8	5	1050	261	230	969
16	2	2000	114	80	966
16	3	2600	444	343	899
16	4	3750	332	293	961
16	5	5000	245	226	981
32	2	8000	340	229	889
32	3	12000	484	381	897

TABLE I
THRESHOLD RECOLOURING WITH $\alpha = 2$ (1000 TRIALS)

$1/r$	k	n	random fail	recoloured	successes
8	2	375	272	171	899
8	3	600	244	182	938
8	4	800	300	245	945
8	5	1050	252	214	962
16	2	2000	105	73	968
16	3	2600	451	338	887
16	4	3750	314	268	954
16	5	5000	223	210	987
32	2	8000	356	264	908
32	3	12000	481	399	918

TABLE II
THRESHOLD RECOLOURING WITH $\alpha = 4$ (1000 TRIALS)

Table I and Table II show the results of simulating this algorithm with path loss exponents $\alpha = 2$ and $\alpha = 4$ respectively. For each combination of values of r, k and n, 1000 runs were performed (with simultaneous recolouring in each iteration). The column headed "random fail" indicates the number of random arrangements of discs where random colouring failed, the column headed "recoloured" indicates the number of remaining arrangements that were successfully coloured by iterating the threshold recolouring algorithm, and the final column indicates the number of successes after recolouring. The program did not check whether the original arrangement of (all) discs formed a k-cover, although theory suggests that this is greater than 0.99985 even in the worst case for the parameters chosen. The results show that the recolouring was successful at least half of the time (when random colouring failed) for every combination of parameters tested, and, owing to the possible presence of non-k-covers in the cases where it failed, this is an underestimate of the algorithm's success rate.

The simulations were carried out using Matlab® on a MacPro equipped with two 2.8 GHz quad-core Intel Xeon processors and 8 GB RAM. To give an indication of the execution time (on a single core) of the algorithm, the simulation for $r = 1/16$, $k = 2$, $n = 2000$ takes about 4 s per trial.

B. Freedom recolouring

The following colouring algorithm runs in time $O(n \log n \log\log n)$, assuming reasonable distribution of the locations of the sensors.

We first divide the torus into $2r \times 2r$ boxes, so that the sensing region of any sensor in a box b only overlaps with the sensing regions of sensors within one of the nine boxes that are either equal, adjacent, or diagonally adjacent to b. We shall base our running time estimates on the assumption that no box contains more than $C \log n$ sensors, for some constant C. On average, each box contains $\Theta(\log n)$ sensors in the critical regime, and, for suitably large C, no box will contain more than $C \log n$ sensors with high probability. The algorithm proceeds in three phases.

Phase 1: We form a list of atomic regions that are *thinly covered*. In our simulations, we took this to mean regions with coverage at most $s = 2k+1$: however the parameter s is adjustable. To create this list, for each sensor p, we list the neighbours $p_1, \ldots, p_l$ of p, that is the sensors whose sensing regions intersect that of p. For each p, only sensors in nearby boxes need to be checked, so that, under the above assumption, this will take time $O(\log n)$ for each sensor.

The sensing region of each p_i intersects the boundary of the sensing region of p in an arc. Let θ_i^- and θ_i^+ be the angles of the beginning and end of this arc, measured from p. We record in an array $(\eta_1, \ldots, \eta_l)$ whether or not this arc crosses the ray $\theta = 0$, setting $\eta_i = 1$ if it does cross $\theta = 0$, and $\eta_i = 0$ otherwise. Then the total coverage at the point $\theta = 0$ on $\partial D(p)$ is just $c = \sum_{i=1}^{l} \eta_i$.

We order the angles $\theta_i^{\pm} \in [0, 2\pi]$, and then consider

each angle $\theta_i^{\pm}$ in increasing order. If we encounter a θ_i^-, we set $\eta_i = 1$ and increase c by 1. If we encounter a θ_i^+, we set $\eta_i = 0$, and decrease c by 1. Thus the arrays (η_i) and c track the sensor discs and coverage level respectively, as we traverse $\partial D(p)$. Each time the coverage level c is at most the coverage limit s, we record the set $S = \{p_i : \eta_i = 1\}$ of sensors that cover this part of $\partial D(p)$ in an array.

To increase efficiency, we only need to record S if the coverage is at a local minimum, i.e., just after a θ_i^+ and just before a θ_j^-. Ordering the angles takes $O(l \log l)$ time, and all other steps here take $O(l)$, so the total running time is now $O(n \log n \log\log n)$ over all sensors. (By the above assumption, $l \leq 9C \log n$.)

Phase 2: We place the sets S in an array in order of their *freedom*. The freedom of S is defined as the number of uncoloured sensors in S minus the number of missing colours, that is, colours that do not occur as a colour of some $p \in S$. Initially, no sensor is coloured and all k colours are missing, so the freedom of each set S is just $|S| - k$. Note that there are only $s-k+1 = O(1)$ possible values for the freedom, and so we can maintain the sets in order of freedom with only a $O(1)$ time penalty whenever we add or delete a new set. There are also at most $O(n \log n)$ sets, although in practice there are far fewer if s is chosen to be not too large. Indeed, in practice we choose s so that the next phase of the algorithm is not too slow.

Take a set S with smallest freedom. If this freedom is negative, then we stop as the colouring has failed. If all sensors in S are coloured, we discard S. Otherwise, we pick an uncoloured sensor $p \in S$, and assign to p at random any of the missing colours of S, that is, any colour that has not been assigned to any $p' \in S$. We update the freedoms of the (at most $O(\log n)$) other sets containing p, and repeat until either the colouring fails or there are no sets left. (A slight improvement in the algorithm can be achieved by first constructing the set M of missing colours that are also missing in every other set $S' \ni p$ of freedom 0. Then, if $M \neq \emptyset$, we colour p with a random colour from M. This avoids reducing the freedom of another set to below 0, resulting in the failure of the algorithm.)

The above colouring takes time $O(\log n)$ per colour assignment, so at most time $O(n \log n)$ overall. We have now coloured the sensors so as to cover the "difficult" regions, unless the algorithm has failed.

Phase 3: The final stage of the colouring is to repeat the first phase, finding the neighbours $p_1, \ldots, p_l$ of each sensor in turn. As before, we calculate the angles $\theta_i^{\pm}$ and initialize the array (η_i). This time, however, we also maintain an array $(n_1, \ldots, n_k)$ of the number n_j of discs of colour i covering an arc. As before, we traverse $\partial D(p)$ updating this array, incrementing n_j when we encounter a θ_i^- and p_i has colour j, and decrementing n_j when we encounter a θ_i^+ and p_i has colour j. If any n_j becomes zero, we greedily assign colour j to any uncoloured sensor p_i that covers this interval (i.e., $\eta_i = 1$), incrementing n_i in the process. If no such uncoloured sensor exists then the colouring fails and we stop.

The result, if successful, is a partial colouring of the sensors such that each point is covered by sensors of each colour.

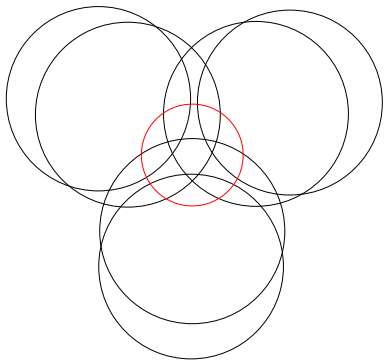
The simulation was written in C, and the random numbers were generated by a 16-bit version of ARC4. All coordinate arithmetic was done to full double precision.

C. Obstructions to partitionability

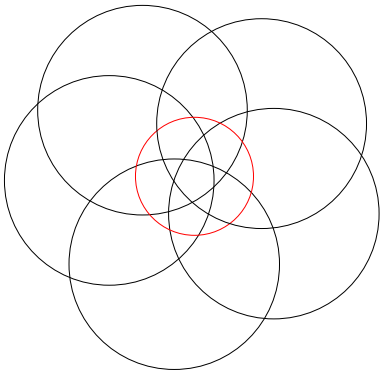
Whatever algorithm we use, some k-covers cannot be k-coloured. Let us initially concentrate on the case $k = 2$. Figure 1(a), Figure 1(b) and Figure 1(c) illustrate three non-2-partitionable 2-covers. Although there are other obstructions to partitionability, these three together appear to account for the majority of cases – see Table III. Referring to the figures, we are only interested in the *central region* inside the small central circle (which is not one of the disc boundaries $\partial D_r(p_i)$). In each case, the central region is 2-covered, but any attempt to partition the discs into two covers is doomed to failure. For Figure 1(a), two of the three "inner" discs (those which intersect at the very centre) must be coloured identically, which means that the discs of the other colour class cannot cover the entire central region. In Figure 1(b), some two discs which are adjacent in the cyclic order must be coloured identically, leading to the same conclusion. Finally, in Figure 1(c), the discs must be coloured alternately with the two colours as we go round the figure, which means that the very central atomic region is only covered by discs of one colour.

The first two configurations, which we term C_3 and C_5 respectively , are different from the third, A_3. (An explanation of this notation will be given in the next section.) The reason is that the central region can be made as small as one likes, as the configuration also exists with half-planes. A_3, however, owes its existence to the curvature of the discs, and there is no equivalent half-plane version. For this reason, when n is large, the C_3 and C_5 configurations will dominate over the A_3 configurations. Indeed, for large n, it is very unlikely that the circular central region is large and intersects no other discs. Consequently, the central region is usually small, and, on its scale, the disc boundaries are almost straight lines. For C_3 and C_5 this is possible, but for A_3, the small central circle is forced to be of a comparable size to the discs themselves, making A_3 less and less likely to occur as n increases.

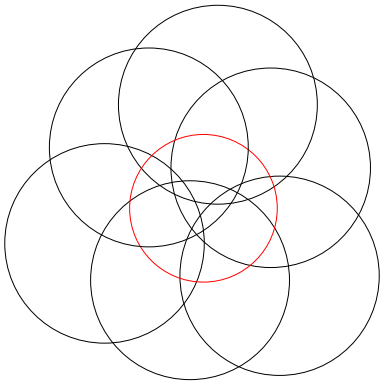
These facts point us towards the investigation of non-2-partitionable 2-covers with half-planes. In [5], the following theorem is proved.



(a) C3



(b) C5



(c) A3

Fig. 1. Obstructions to 2-partitionability

Theorem 4. *Suppose $\mathcal{C}$ is a finite 2-cover of $\mathbb{R}^2$ with half-planes that is not 2-partitionable. Then $\mathcal{C}$ contains a C_3 or a C_5 configuration.*

C_3s and C_5s are the configurations which occur with probability $\Theta(1/\log n)$ and make Theorem 2 best possible for $k = 2$. In fact, for $k \geq 3$, it turns out that the principal obstructions to k-partitionability in a k-cover are just non-2-partitionable 2-covers in which the entire central region is covered by $k-2$ common discs. All other obstructions occur with a lower asymptotic frequency. To summarize, C_3s and C_5s (possibly buried beneath other discs) are the principal obstacles to k-colouring a k-cover when n is large.

D. Freedom recolouring results and statistics of obstructions

The results of 100000 runs of the freedom recolouring algorithm for each case are displayed in Table III. The key for the column headings is as follows. "ft$k$c" indicates that the maximum number of discs allowed by the program failed to k-cover T, C_i indicates that the program found a C_i configuration, "fail" indicates that the program succeeded in colouring the k-covered regions, failed to k-colour the discs, and failed to find an obstruction, and "succeed" indicates that the program succeeded in k-colouring the discs.

The program also identified several different types of "asymptotically low frequency" obstructions (such as A_3), listed in the middle columns of Table III. The general notation C_n refers to a configuration containing n k-covered regions covered by $k-2$ common discs, where, after these common discs are removed, each region is only covered a pair of discs (D_i, D_{i+1}) of sensor discs, where n is odd, and where the subscripts are taken modulo n. Only C_3 and C_5 have half-plane variants, although, as can be seen from the table, several disc-C_7s and even 4 disc-C_9s were detected overall. The remaining codes describe families of obstructions. To describe them, we first define the *equality graph*. This is a graph whose vertices are the discs, and in which two vertices D_i and D_j are joined by an edge if there are some two exactly k-covered regions A and B such that the symmetric difference of the sets of discs covering A and B is exactly $\{D_i, D_j\}$ (see Figure 2). The reason for the terminology is that two vertices joined by an edge correspond to discs which must be coloured with the same colour. With this convention, B_n refers to the existence of a k-covered region in which two of the covering discs are forced to be the same colour due to a path of length $\frac{1}{2}(n-1)$ in the equality graph, and A_n refers to the existence of an n-covered region where too many discs are forced to be the same colour by the equality graph. Actually, C_n is a special case of B_n, which is a special case of A_k, but the program only lists the highest ranked bad configuration with respect to the order

$$C_3 > C_5 > \cdots > B_3 > B_5 > \cdots > A_3 > \cdots > KC.$$

Finally, KC indicates that the program failed to colour the discs so that the exactly k-covered regions were covered by discs of each colour, but that none of the above bad configurations was detected.

In all the simulations, there were no instances where 3-coverage did not imply 2-partitionability, although such instances do in fact exist, from the results in [17]. There were, however, some instances where 4-coverage might not have implied 3-partitionability. Figure 3 shows graphs of $\mathbb{P}(E_r^k)$ and $\mathbb{P}(F_r^k)$ estimated from the simulations.

$1/r$	k	ftkc	C_3	C_5	KC	A_3	A_4	A_5	A_6	A_7	A_8	B_3	B_5	B_7	B_9	C_7	C_9	fail	succeed
4	2	0	4042	839	0	278	2	0	0	0	0	0	0	0	0	60	4	207	94568
8	2	0	3442	414	0	125	0	0	0	0	0	0	0	0	0	4	0	69	95946
16	2	0	2822	324	0	56	0	0	0	0	0	0	0	0	0	0	0	38	96760
32	2	0	2322	215	0	48	0	0	0	0	0	0	0	0	0	0	0	16	97399
64	2	0	2116	188	0	33	0	0	0	0	0	0	0	0	0	0	0	8	97655
128	2	0	1857	149	0	22	0	0	0	0	0	0	0	0	0	0	0	7	97965
4	3	0	4050	334	18	0	564	4	0	0	0	1528	601	30	1	0	0	5630	87240
8	3	0	3584	314	2	0	365	5	0	0	0	1176	257	1	0	0	0	2807	91489
16	3	0	3180	226	2	0	303	2	0	0	0	903	156	0	0	0	0	1857	93371
32	3	0	2931	223	1	0	284	0	0	0	0	738	106	0	0	0	0	1223	94494
64	3	0	2681	198	0	0	202	3	0	0	0	552	67	0	0	0	0	933	95364
128	3	1	2478	169	0	0	125	1	0	0	0	454	56	0	0	0	0	782	95934
4	4	0	3763	246	229	0	0	637	16	1	0	2501	621	25	0	0	0	14584	77377
8	4	0	3606	241	118	0	0	501	7	0	0	1945	326	1	0	0	0	8330	84925
16	4	0	3382	216	73	0	0	368	6	0	0	1610	213	0	0	0	0	5325	88807
32	4	0	3127	176	55	0	0	335	3	0	0	1307	165	0	0	0	0	3736	91096
64	4	0	2959	159	36	0	0	307	1	0	0	1121	129	0	0	0	0	2947	92341
128	4	2	2767	175	35	0	0	245	4	0	0	931	98	0	0	0	0	2471	93272
4	5	0	3645	208	430	0	0	0	614	20	0	2979	593	15	1	0	0	25702	65793
8	5	0	3545	190	260	0	0	0	541	12	0	2528	357	1	0	0	0	15841	76725
16	5	0	3365	191	163	0	0	0	458	7	1	2104	274	0	0	0	0	10526	82911
32	5	0	3235	168	141	0	0	0	399	8	0	1727	196	0	0	0	0	7812	86314
64	5	1	3092	149	95	0	0	0	315	8	0	1481	186	0	0	0	0	6675	87998
128	5	8	2879	167	76	0	0	0	296	9	0	1340	121	0	0	0	0	5878	89226

TABLE III
SIMULATION RESULTS FOR FREEDOM RECOLOURING ALGORITHM

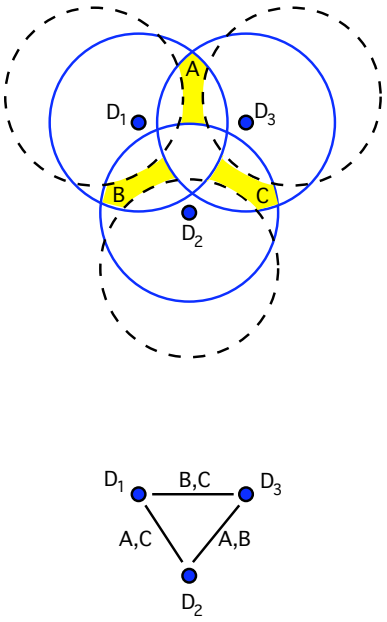


Fig. 2. The equality graph

Note that for these simulations, discs were only added until k-coverage was achieved, so that the algorithm was always run on a "worst-case" k-cover. This is in contrast to the threshold recolouring simulation results presented earlier.

IV. Conclusion

The sentry selection problem in wireless sensor networks is of both theoretical and algorithmic interest. For the setup where n nodes are placed uniformly at random on the unit torus, a recent result states that randomly generated k-covers are k-partitionable asymptotically almost surely. In other words, the gap between k-coverage and k-partitionability vanishes as the number of nodes increases. Together with an old result on k-coverage, this shows that the area of the sensing disc πr^2 needs to be just slightly larger than $(\log n + k \log\log n)/n$ for k-partitionability. Hence the increase in r necessary to have k disjoint sets of sensors, each covering the area, instead of just single coverage, is rather small as n gets large – and the benefit is a k-fold increase in network lifetime.

In this paper, we have examined some of the primary obstructions to k-partitionability. The probability of such obstructions occurring goes to zero as $1/\log n$. On the algorithmic side, we have introduced an improvement over random colouring that is based on measuring the total power received from the nodes with the same colour, and recolouring if the power exceeds a threshold. This simple scheme "saves" about 80% of the cases where random colouring was not successful. We have also proposed an efficient centralized colouring algorithm that runs in time $O(n \log n \log\log n)$ and succeeds in almost all cases. Finally, we have presented simulation results which provide detailed statistics on the frequency of various obstructions to partitionability. These shed light on recent theoretical results and suggest directions for future research on geometric partitioning problems.

References

[1] H. AbdelSalam and S. Olariu, Toward adaptive sleep schedules for balancing energy consumption in wireless sensor networks,

IEEE Transactions on Computers **61** (2012), 1443–1458.

[2] Z. Abrams, A. Goel and S. Plotkin, Set k-cover algorithms for energy efficient monitoring in wireless sensor networks, *Information Processing in Sensor Networks*, Berkeley, California (2004), 424–432.

[3] P. Balister, B. Bollobás and A. Sarkar, Barrier coverage, *Random Structures and Algorithms*, to appear.

[4] P. Balister, B. Bollobás, A. Sarkar and S. Kumar, Reliable density estimates for coverage and connectivity in thin strips of finite length, *ACM MobiCom*, Montréal, Canada (2007), 75–86.

[5] P. Balister, B. Bollobás, A. Sarkar and M. Walters, Sentry selection in wireless networks, *Advances in Applied Probability* **42** (2010), 1–25.

[6] P. Balister, B. Bollobás and M. Walters, Continuum percolation with steps in the square or the disc, *Random Structures and Algorithms* **26** (2005), 392–403.

[7] Y. Ding, C. Wang and L. Xiao, An adaptive partitioning scheme for sleep scheduling and topology control in wireless sensor networks, *IEEE Transactions on Parallel and Distributed Systems* **20** (2009), 1352–1365.

[8] E.N. Gilbert, The probability of covering a sphere with N circular caps, *Biometrika* **56** (1965), 323–330.

[9] P. Hall, On the coverage of k-dimensional space by k-dimensional spheres, *Annals of Probability* **13** (1985), 991–1002.

[10] J. Hui, Z. Ren and B. Krogh, Sentry-based power management in wireless sensor networks Lecture Notes in Computer Science **2634** (2003), 458–472.

[11] S. Janson, Random coverings in several dimensions, *Acta Mathematica* **156** (1986), 83–118.

[12] K. Kar, A. Krishnamurthy and N. Jaggi, Dynamic node activation in networks of rechargeable sensors, *IEEE/ACM Transactions on Networking* **14:1** (2006), 15–26.

[13] X. Liu and M. Haenggi, Toward quasiregular sensor networks: topology control algorithms for increased energy efficiency, *IEEE Transactions on parallel and distributed systems* **17** (2006), 975–986.

[14] B. Liu and D. Towsley, On the coverage and detectability of large-scale wireless sensor networks, *Proc. Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt)* (2003).

[15] P. Mani-Levitska and J. Pach, Decomposition problems for multiple coverings with unit balls, unpublished manuscript (1986). Available at http://www.math.nyu.edu/~pach/publications/unsplittable.pdf

[16] P.A.P. Moran and S. Fazekas de St Groth, Random circles on a sphere, *Biometrika* **49** (1962), 389–396.

[17] J. Pach and D. Pálvölgyi, Unsplittable coverings in the plane, arXiv:1310.6900.

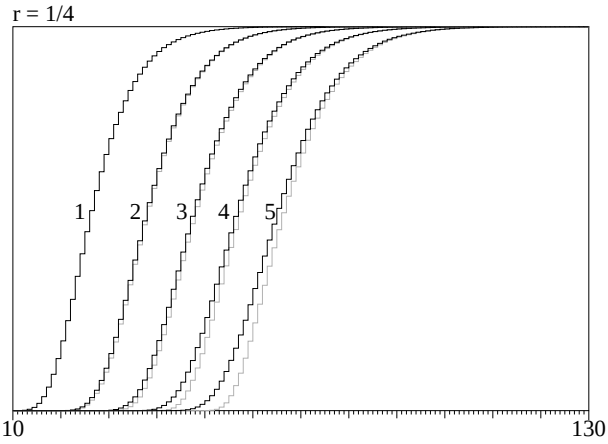
[18] J. Pach, G. Tardos and G. Tóth, Indecomposable coverings, Lecture Notes in Computer Science **4381** (2007), 135–148.

[19] J. Pach and G. Tóth, Decomposition of multiple coverings into many parts, *23rd ACM Symposium on Computational Geometry*, ACM Press, New York (2007), 133–137.

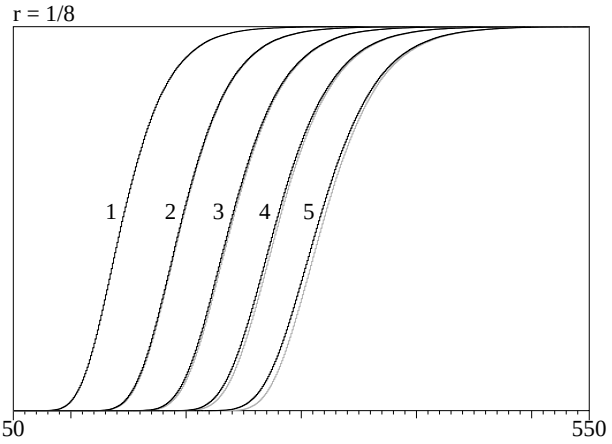
[20] S. Slijepcevic, M. Potkonjak, Power efficient organization of wireless sensor networks, *IEEE International Conference on Communications*, Helsinki, Finland (2001), 472–476.

[21] D. Tian and N. Georganas, A coverage-preserving node scheduling scheme for large wireless sensor networks, *Proc. ACM Workshop on Wireless Sensor Networks and Applications* (2002), 32–41.

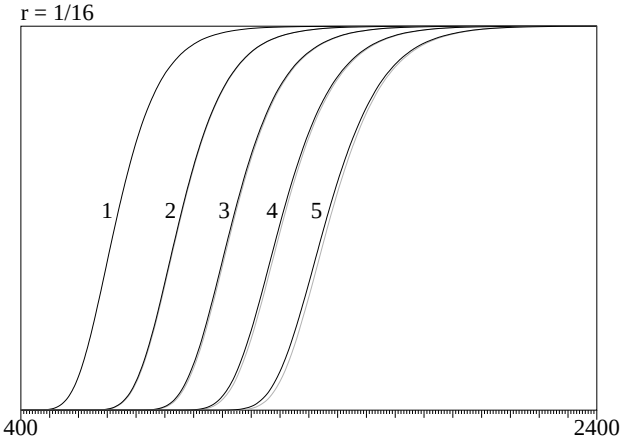
[22] Y. Xiao, H. Chen, K. Wu, B. Sun, Y.Zhang, X. Sun and C. Liu, Coverage and detection of a randomized scheduling algorithm in wireless sensor networks, *IEEE Transactions on Computers* **59** (2010), 507–521.



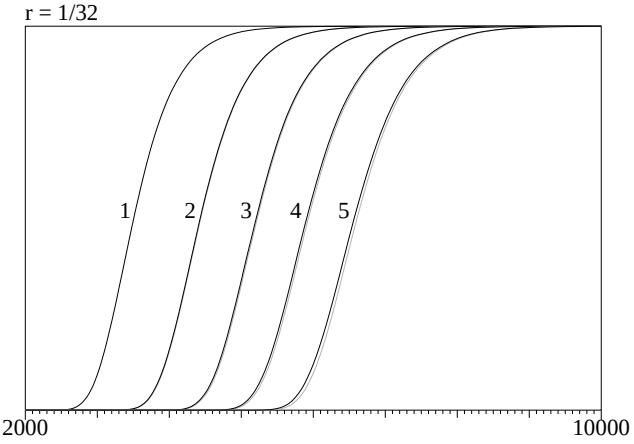
(a) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/4$



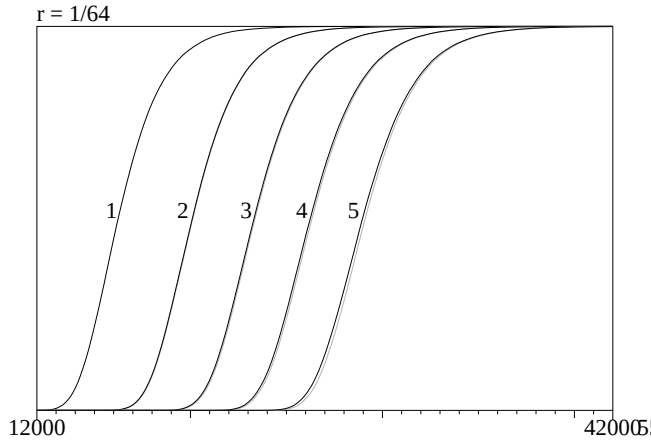
(b) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/8$



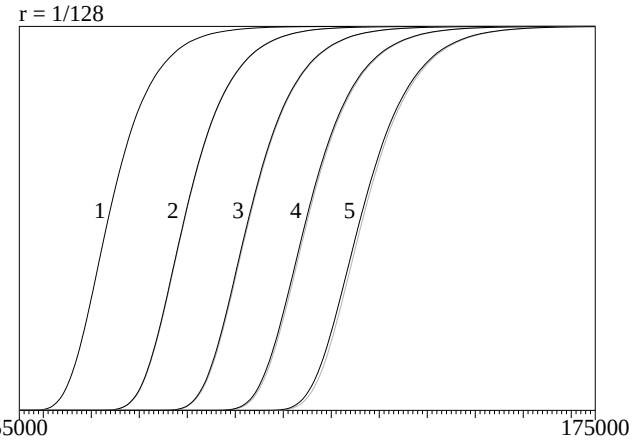
(c) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/16$



(d) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/32$



(e) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/64$



(f) $\mathbb{P}(E_r^k)$ (black) and $\mathbb{P}(F_r^k)$ (gray) for $r = 1/128$

Fig. 3. Probabilities of coverage (black) and partitionability (gray)