5-2015

Average Case Network Lifetime on an Interval with Adjustable Sensing Ranges

Amotz Bar-Noy
City University of New York

Benjamin Baumer
Smith College, bbaumer@smith.edu

Noname manuscript No.
(will be inserted by the editor)

Average Case Network Lifetime on an Interval with Adjustable Sensing Ranges

Amotz Bar-Noy · Ben Baumer

Received: date / Accepted: date

Abstract Given n sensors on an interval, each of which is equipped with an adjustable sensing radius and a unit battery charge that drains in inverse linear proportion to its radius, what schedule will maximize the lifetime of a network that covers the entire interval? Trivially, any reasonable algorithm is at least a 2-approximation for this SENSOR STRIP COVER problem, so we focus on developing an efficient algorithm that maximizes the expected network lifetime under a random uniform model of sensor distribution. We demonstrate one such algorithm that achieves an expected network lifetime within 12% of the theoretical maximum. Most of the algorithms that we consider come from a particular family of ROUNDROBIN coverage, in which sensors take turns covering predefined areas until their battery runs out.

Keywords wireless sensor networks · adjustable range · restricted strip cover · lifetime · area coverage · barrier coverage

A preliminary version of this work appeared in Proceedings of the 7th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities (ALGOSENSORS 2011), Lecture Notes in Computer Science 7111, Springer, pp. 28-41 [2].

Amotz Bar-Noy
Department of Computer Science
The Graduate Center of the City University of New York
Brooklyn College
Tel.: 212-817-8194
E-mail: amotz@sci.brooklyn.cuny.edu

Ben Baumer
Department of Mathematics & Statistics
Smith College
Tel.: 413-585-3440
E-mail: bbaumer@smith.edu

1 Introduction

Suppose that we wish to cover a one-dimensional region (or interval) with a wireless sensor network. Given the locations and initial battery charges of n sensors deployed on that interval, we have the ability to set the sensing radius of each sensor to an arbitrary length finitely many times. However, the battery of each sensor drains in inverse linear proportion to its sensing radius, so that the larger we make the radius, the faster the battery drains. Our goal is to maximize the *lifetime* of the network, which is the length of time that the entire interval is covered until all of the batteries run out. What schedule will achieve this?

In the most general version of the problem, which we call Sensor Strip Cover[1], the initial battery charges are not necessarily the same, the scheduler is free to change the radius of any sensor finitely many times, and there are no restrictions on how many sensors may work together at one time or over the lifetime of the network. In this paper, we focus on linear time approximation algorithms with high average-case performance, under the additional assumptions: 1) the sensor locations are drawn at random from a uniform distribution; and 2) each sensor has the same initial battery charge.[2]

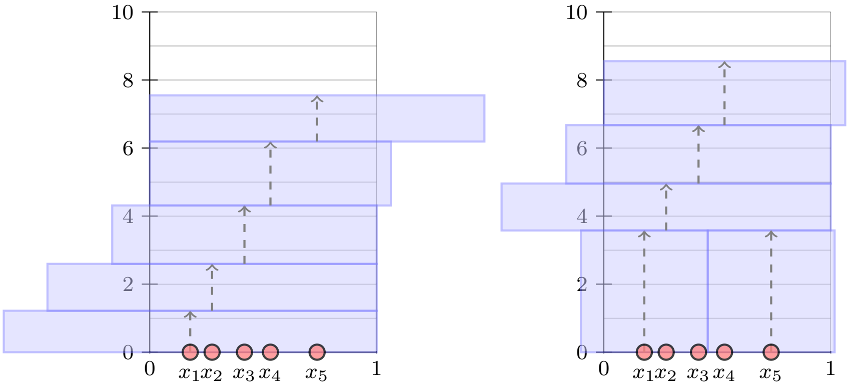
1.1 Problem definition

Formally, let $U = [0, 1]$ be a real interval to be covered, and suppose that n sensors are deployed on U with locations $X = \{x_1, ..., x_n\}$ such that $x_1 \leq x_2 \leq \cdots \leq x_n$. Since each sensor has the same initial battery charge, we assume for convenience that each initial charge is 1. An instance thus consists of the set X, and a solution is an assignment of radii, activation times, and deactivation times to sensors. More specifically a solution (or *schedule*) is a finite list S of coverage assignments (i, ρ, τ, σ), each of which tells the i^{th} sensor to activate at time τ with a *radius* of ρ, and maintain that coverage until time σ. For a list of length m, let S_j indicate the j^{th} coverage assignment for $1 \leq j \leq m$.

Since we allow the coverage assignment of each sensor to be changed only finitely many times, any schedule can be visualized by a space-time diagram in which each coverage assignment is represented by a rectangle. It is customary in such diagrams to view the sensor locations as forming the horizontal axis, with time extending upwards vertically. In this case, the coverage of a sensor located at x_{i_j} and assigned the radius ρ_j beginning at time τ_j and ending at time σ_j is depicted by a rectangle with lower-left corner $(x_{i_j} - \rho_j, \tau_j)$ and upper-right corner $(x_{i_j} + \rho_j, \sigma_j)$. Let the set of all points contained in this rectangle be denoted as $Rect(S_j)$. A point $(u, t) \in U \times [0, \infty)$ in space-time

[1] We use the terms *strip* and *interval* interchangeably to refer to the one-dimensional coverage region.

[2] Please see Remark 1 and the Open Problems section for a discussion of conditions under which the latter assumption can be lifted without loss of generality.



(a) Lifetime = 7.55

(b) Lifetime = 8.55

Fig. 1 Graphical depictions of valid schedules for a randomly generated instance of 5 unit-battery sensors. Here, $X = (0.178, 0.275, 0.417, 0.532, 0.737)$. At left, the schedule produces a lifetime of 7.55 time units, while on the right, a superior lifetime of 8.55 time units is achieved by a different schedule. Neither schedule reaches the maximum theoretical lifetime of 10 time units, although in this case the solution at right is the best possible.

is *covered* by a schedule S if $(u, t) \in \bigcup_j Rect(S_j)$. Examples of two possible schedules for a randomly generated instance are shown in Figure 1.

In addition to the *coverage* constraint described above, a solution to Sensor Strip Cover must also satisfy a *battery* constraint. That is, no sensor i can consume more than 1 battery unit. During the j^{th} coverage assignment, sensor i_j consumes $\rho_j \cdot (\sigma_j - \tau_j)$ battery units. Thus, a valid schedule must have $\sum_{i_j=i} \rho_j \cdot (\sigma_j - \tau_j) \leq 1$ for all $1 \leq i \leq n$.

The *lifetime* of the network in a solution S is the maximum value T such that every point $(u, t) \in U \times [0, T]$ is covered, and no sensor i has consumed more than 1 battery unit. Graphical depictions of valid schedules are shown in Figures 1, 2, 4, 5, and 6.

1.2 Motivation

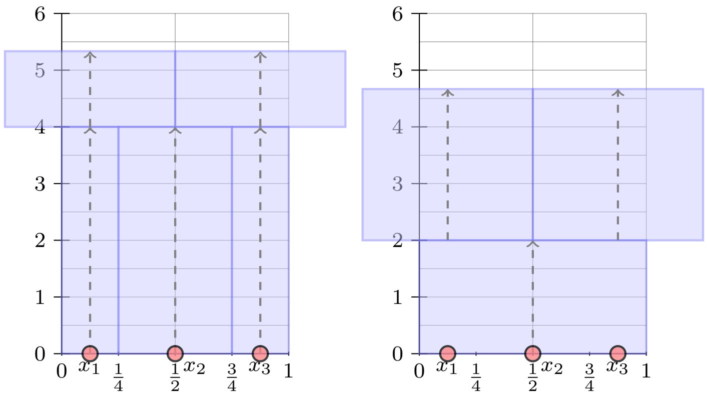
This type of scheduling problem arises in many applications, often in relation to problems of *barrier coverage* (see [9,21] for surveys). Suppose that we have a highway, supply line, or fence in territory that is either hostile or difficult to navigate. While we want to monitor activity along this line, conditions on the ground make it impossible to systematically place wireless sensors at specific locations. However, it is feasible and inexpensive to deploy adjustable range sensors along the line by, say, dropping them from an airplane flying overhead (e.g. [8,19,20]). Once deployed, the sensors send us their locations via GPS, which we then analyze and respond with a list of coverage assignments. Replacing the battery in any sensor is infeasible. How do we construct a set of assignments that will keep this vital supply line completely monitored for as long as possible?

While worst-case analysis of algorithms is certainly of interest, since (as we will show) a 2-approximation for Sensor Strip Cover is trivial, we have focused our efforts on finding simple, linear time algorithms with good *average-case* performance against uniform random sensor deployments. If the sensors are in fact dropped from an airplane overhead, then this is a reasonable model. Furthermore, in this event average-case performance is likely to be of greater practical importance than worst-case performance, which might occur in only a few extremely unlikely instances.

1.3 Previous Research

Buchsbaum et al. [7] initiated this line of research by defining Restricted Strip Cover (RSC) as a special case of a more general sensor cover problem. In RSC, the locations, sensing radii, and battery charges of n sensors placed on an interval are given, and the problem is to compute an optimal set of activation times, so as to maximize the network lifetime. By taking advantage of the packing-covering duality, and using a reduction from Dynamic Storage Allocation, they showed that this problem is NP-hard. They presented an $O(\log\log n)$-approximation algorithm, which was later improved to a constant factor approximation algorithm by Gibson and Varadarajan [15]. For the special case of RSC in which the battery charges are all the same, Buchsbaum et al. presented a simple greedy algorithm that yields an optimal solution in polynomial time. However, they considered only the *non-preemptive* case, in which a sensor cannot be reactivated once it is turned on and then off. In some cases, such as the one we show in Figure 2, pre-emptive scheduling can increase the achievable lifetime.

The problem of finding the optimal set of radial assignments for sensors deployed on an interval, rather than the activation times, is more tractable. Whereas Sensor Strip Cover requires *area* coverage (i.e. a requirement to cover *all* points in an interval), Lev-Tov and Peleg [16] studied a similar problem with *target* coverage (i.e. a requirement to cover only a finite number of specific points in an interval). In Minimum Sum of Radii Cover (MSRC), the input is a set of n sensors and a finite set of m points on the interval that are to be covered, and the goal is to find the radial assignments with the minimum sum of radii, such that all m points are covered. They used dynamic programming to devise a polynomial time algorithm for the one-dimensional case, and a polynomial time approximation scheme for the two-dimensional case. Note that the emphasis here is on energy conservation (through minimizing the sum of radii), and not maximizing lifetime explicitly, since they seek only a single coverage assignment. This is akin to a "one-shift" solution. Bar-Noy et al. [4] found a polynomial time solution to the one-shift problem with area coverage on an interval. In Observation 2 we show that a greedy algorithm based on this criteria provides at best a $\frac{6}{5}$-approximation.



(a) $T = 5\frac{1}{3}$, pre-emptive

(b) $T = 4\frac{2}{3}$, non-preemptive

Fig. 2 Illustration of the advantages of pre-emptive scheduling for $X = (\frac{1}{8}, \frac{1}{2}, \frac{7}{8})$. At left, pre-emptive scheduling is employed to reset the radial assignments of x_1 and x_3 after 4 time units have passed. The resulting lifetime of $5\frac{1}{3}$ time units surpasses the best non-preemptive solution, shown at right.

Bilò et al. [6] generalized MSRC to cases where the battery draining rate α is not linear with respect to the radius, and showed NP-hardness for the case where $\alpha \geq 2$. Alt et al. [1] found constant factor approximation algorithms and extended the hardness result to $\alpha > 1$. Gibson et al. found a polynomial time solution to a related clustering problem using pinned disks [14].

Worst-case analysis of Sensor Strip Cover and several closely-related variants was conducted by Bar-Noy et al. [2,4,3]. They showed the RoundRobin algorithm we discuss in this paper has a tight $\frac{3}{2}$-approximation guarantee. The instance that produces this lower bound produces the same bound for the best one circle solution to the Connected Range Assignment (CRA) problem. In this problem, studied by Chambers et al. [12], radii are assigned to points in the plane in order to obtain a connected disk graph.

Working in the plane, Wu and Yang [23] introduced the notion of networks that use sensors with adjustable ranges, and studied energy consumption under a random deployment model. Cardei and Du [8] proposed the Disjoint Set Covers problem, in which n sensors monitor m target points, and the goal is find the maximum number of disjoint covers. They showed that this problem is NP-complete, and that any polynomial-time algorithm is at best a 2-approximation. Heuristics were presented for solving a version of the problem that had been converted into a maximum-flow problem. In subsequent work, Cardei et al. [10,11] extended this problem to the Adjustable Range Set Cover problem by lifting the restriction that the covers be disjoint. An important difference between our work and theirs is that they assumed that the sensing ranges came from a finite discrete set, rather than a continuous range, as we allow.

Algorithm	$\mathbb{E}[T]$	$Var[T]$	AC	WC_{lb}	WC_{ub}
RoundRobin	1.386	0.078	1.443	1.5	1.5 [4]
k-RoundRobin	1.386	0.078	1.443	1.5	1.5
log-RoundRobin	1.738	0.022	1.151		
Optimized log-RoundRobin	1.791		1.117		

Table 1 Summary of results for Sensor Strip Cover with unit batteries. $\mathbb{E}[\bar{T}]$ is the expected lifetime per sensor, if the sensors are deployed uniformly at random. AC is the approximation ratio in the average case, while WC_{lb} and WC_{ub} are lower and upper bounds for the worst-case approximation ratios, respectively.

Unlike the duty cycling approach taken by Cardei et al., Berman et al. [5, 13] sought to maximize the true lifetime in the Sensor Network Lifetime Problem (SNLP), which seeks to cover m target points with n adjustable range sensors. They propose a Linear Program that is exponential in n, but achieve an $O(\log n)$-approximation by using the Garg-Könemann algorithm for solvings LPs. Sensing ranges are allowed to vary continuously up to a maximum cutoff distance. Sensor Strip Cover differs from SNLP in that it requires area coverage on a one-dimensional region.

1.4 Our Contribution

Our extension of Restricted Strip Cover is the first to consider the true lifetime for area coverage on the line with adjustable sensing ranges. Since any reasonable algorithm is at least a 2-approximation, most of our efforts are focused on raising the approximation ratio in the *average case*, which in an application scenario, is likely to be of greater value. Our main result is a constructive proof that a linear time algorithm exists that achieves an approximation ratio of 1.117 in the average case. We accomplish this by employing RoundRobin coverage on a hierarchical system of pre-defined coverage areas. Although we allow pre-emptive scheduling, we do not explicitly use it in our algorithms. Thus, our results are also valid for the special case in which pre-emptive scheduling is not allowed. A summary of our results is shown in Table 1.

2 Preliminaries

Much of our analysis is predicated on a particularly simple algorithm: RoundRobin, which forces the sensors to take turns covering U. Namely it assigns, for every i,

$$\rho_i = \max\{x_i, 1 - x_i\}, \qquad \tau_i = \sum_{j=1}^{i-1} \frac{1}{\rho_j} \ . \tag{1}$$

The lifetime of RoundRobin is thus

$$\mathrm{RR}(X) \triangleq \sum_{i=1}^{n} \frac{1}{\rho_i} \ .$$

Note that RoundRobin gives a valid solution for *every* variant of the Sensor Strip Cover problem.

The best possible lifetime of any instance is contrained by the total battery charge in the system, as well as the location of the sensors.

Lemma 1 *The maximum lifetime of a unit battery instance X is at most $2n$.*

Proof Consider an optimal solution S for X with lifetime T. Due to the battery constraint, the total area of space-time covered by sensor i is at most 2, for all $1 \leq i \leq n$, since the width of each rectangle is twice the assigned radius. Due to the coverage constraint, the lifetime T of the network is at most the total area of space-time covered by the sensors, which is at most $2n$.

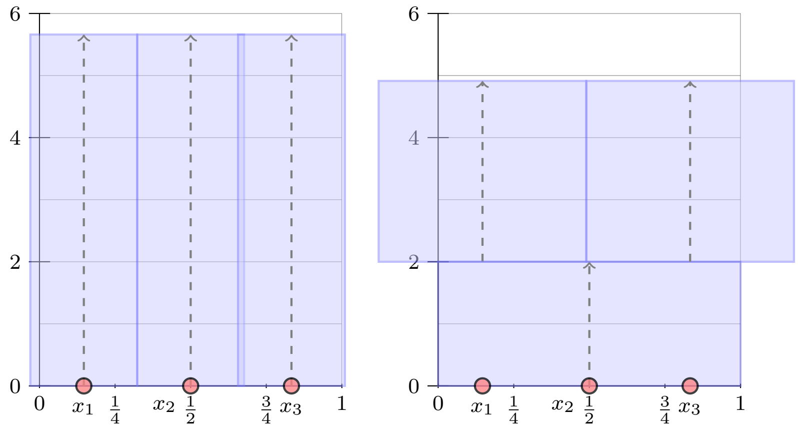
For any set of sensor locations X, we assume that there exists some optimal schedule that will produce the longest possible lifetime Opt.

Remark 1 Since the scheduler is allowed to change the sensing radius of each sensor finitely many times, we contend that the assumption of uniform initial battery charges can be made without loss of generality under mild regularity conditions. Consider an instance (X, B) of Sensor Strip Cover with sensor locations X and corresponding non-uniform battery charges B, where $b_i \in \mathbb{Z}^+$ is the initial battery charge of the sensor located at x_i, and not all of the b_i's are the same. Then if the product XB is uniformly distributed, the performance of most algorithms on this instance is roughly the same as it would be on a unit battery instance, with b_i sensors placed at x_i.

Remark 2 It is natural to wonder whether an algorithm that successively schedules the least wasteful coverage until a sensor runs out of battery life is optimal. Here, we show a worst-case approximation bound for such a Greedy algorithm.

Observation 2 *The approximation ratio of* Greedy *is at best $\frac{6}{5}$.*

Proof Consider $X = \left(\frac{1}{6} - \epsilon, \frac{1}{2}, \frac{5}{6}\right)$, for some $\epsilon > 0$. Greedy chooses to activate the middle sensor by itself on the entire interval first, since that is the only perfect assignment possible. This produces a lifetime approaching 5 as $\epsilon \to 0$, but the optimal lifetime of 6 is achievable in the limit by scheduling all three sensors together (see Figure 3).



(a) OPT$(X) \to 6$ (b) GREEDY$(X) \to 5$

Fig. 3 Depiction of $\frac{6}{5}$ lower bound for GREEDY. For the instance $X = (\frac{1}{6} - \epsilon, \frac{1}{2}, \frac{5}{6})$, OPT produces a lifetime that approaches 6 as $\epsilon \to 0$, but GREEDY achieves a lifetime that only approaches 5. In this example $\epsilon = 0.02$, and the lifetime of OPT is 5.66 while the lifetime of GREEDY is 4.91.

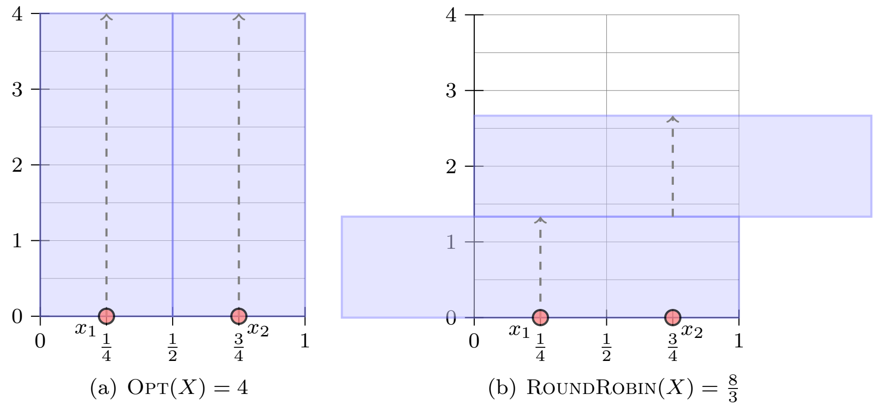
Remark 3 In [2], a $\frac{3}{2}$ lower bound on the performance of ROUNDROBIN relative to OPT was demonstrated using the example shown in Figure 4. This same lower bound applies to each of the algorithms discussed in this paper, in fact from the same example. Consequently for the remainder of this paper we focus on bounds for the average-case performance of algorithms under the assumption that sensor locations are distributed uniformly at random. That is, we assume that X is a uniform random variable on $[0, 1]$. We will abuse notation by allowing X to represent both the r.v. and a set of n observations of that r.v.

3 Average-case analysis

For any solution, let $\bar{T} = T/n \in [1, 2]$ be the average network lifetime per sensor. For a group of sensors working simultaneously, it is often convenient to discuss the *normalized* lifetime $\hat{T}$, which is the average lifetime of a particular coverage group.

3.1 ROUNDROBIN

Clearly, ROUNDROBIN performs best when sensors are located close to 1/2, where the lifetime is close to 2, and poorly for sensors near 0 and 1, where the lifetime is 1. We analyze the average case by assuming that X is a uniform random variable over $[0, 1]$. Then the function $T_{0,1}(X) = \frac{1}{\max{(X, 1-X)}}$ yields a



(a) $\text{Opt}(X) = 4$

(b) $\text{RoundRobin}(X) = \frac{8}{3}$

Fig. 4 Depiction of $\frac{3}{2}$ lower bound for RoundRobin. The lifetime of the network is shown on the vertical axis, while location is shown on the horizontal axis. Each sensor is indicated by a red dot, and each rectangle represents a coverage assignment. The dashed arrows indicate periods of activity. Note that the total area of space-time consumed by each sensor is exactly 2. For the instance $X = (\frac{1}{4}, \frac{3}{4})$, Opt produces a lifetime of 4, but RoundRobin achieves a lifetime of only $\frac{8}{3}$.

new r.v. giving the lifetime of an *individual sensor*[3]. It is easy to calculate its mean

$$\mu_T \triangleq \mathbb{E}[T_{0,1}(X)] = \int_0^1 \frac{dx}{\max(x, 1-x)} = 2\int_{\frac{1}{2}}^1 \frac{dx}{x} = 2\ln x\Big|_{\frac{1}{2}}^1 = 2\ln 2\,, \qquad (2)$$

and variance

$$\sigma_T^2 \triangleq \mathbb{E}[T_{0,1}^2(X)] - \mu_T^2 = \int_0^1 \frac{dx}{(\max(x, 1-x))^2} - \mu_T^2 = 2 - 4\ln^2 2\,. \qquad (3)$$

We will develop algorithms that improve on this expected lifetime of μ_T.

Central Limit Theorem. Of course, with n sensors, we are more interested in the distribution of $\bar{T}$, as opposed to that of T. Since we know μ_T and σ_T^2, the Central Limit Theorem implies that the distribution of $\bar{T}$ approaches a normal distribution with mean μ_T and variance σ_T^2/n as $n \to \infty$. For this reason we report the variance but focus most of our attention on the expected average lifetime of each algorithm.

Theorem 1 *If the sensor locations are distributed uniformly at random over* $[0, 1]$*, then* RoundRobin *achieves an approximation ratio of* $1/\ln 2 \approx 1.443$ *in the average case.*

Proof Since $\text{Opt}(X) \leq 2n$ by Lemma 1, the result follows from equation 2.

[3] We will occasionally abuse notation by using T to refer to either the lifetime of the system, or the random variable giving the lifetime of a sensor. The precise meaning should be clear from context.

3.2 k-RoundRobin

A natural extension of RoundRobin is to partition U into k equally-spaced subintervals, and run it independently on each of those. Somewhat surprisingly, the performance is no better in either the worst or the average case. A generalized version of RoundRobin is shown in Algorithm 1.

Let k be a fixed positive integer, and let $U_k(i) = [\frac{i-1}{k}, \frac{i}{k}]$ for $i = 1, ..., k$ define a partition of U. We define k-RoundRobin to be the algorithm that runs RoundRobin independently on each subinterval $U_k(i)$; maintaining k parallel queues. However, over any subinterval $[a, b] \subseteq U$, the r.v. giving the lifetime of a sensor in $U_k(i)$ is simply a rescaling of T from the original RoundRobin.

Algorithm 1 RoundRobin

```
Input: An ordered set of n sensor locations $X$
Optional Parameters: a start time $t_0$, locations of the $left$ and $right$ edges of the coverage interval
Output: A schedule $S$
if $t_0$ is null then
    $t_0 = 0$
end if
if $left$ is null then
    $left = 0$
end if
if $right$ is null then
    $right = 1$
end if
if $n > 0$ then
    $\tau_1 = t_0$
    $\rho_1 = \max(x_1 - left, right - x_1)$ {Set the radius to touch the furthest endpoint}
    $\sigma_1 = \tau_1 + 1/\rho_1$
    if $n > 1$ then
        for $i$ from 2 to n do
            $\tau_i = \sigma_{i-1}$
            $\rho_i = \max(x_i - left, right - x_i)$
            $\sigma_i = \tau_i + 1/\rho_i$
        end for
    end if
    return ([n], $\rho, \tau, \sigma$)
else
    return null
end if
```

Remark 4 For any interval $[a, b] \subseteq U$, the expected lifetime $T_{a,b}(X)$ of a sensor running RoundRobin on $[a, b]$ is $\frac{\mu_T}{b-a}$ with variance $(\frac{\sigma_T}{b-a})^2$.

With $b - a = 1/k$, the expected lifetime of each sensor in k-RoundRobin is $k\mu_T$, with a maximum lifetime of $2k$. However, in order to cover the whole line, we have to run k parallel queues, so that the expected *normalized* lifetime of each sensor is $\mathbb{E}[\hat{T}] = \mu_T$. For a set of n sensors, the total expected lifetime is $n\mu_T$, so the expected *average* network lifetime $\mathbb{E}[\bar{T}]$ is μ_T. Similar calculations

Algorithm 2 k-RoundRobin

```
Input: An ordered set of sensor locations $X$, an optional depth parameter $k$
Output: A schedule $S$
if $k$ is null then
    $k \leftarrow \lfloor \max(1, \frac{n}{3 \ln n}) \rfloor$
end if
for $i$ from 1 to $n$ do
    $q_i = \lfloor k \cdot x_i \rfloor$ {Determine to which of the $k$ queues each sensor belongs}
end for
$S = \{\}$
for $\ell$ from 1 to k do
    $Q = \{x_i \in X : q_i = \ell\}$
    $S = S \cup$ RoundRobin$(Q, 0, \frac{\ell-1}{k}, \frac{\ell}{k})$
end for
return $S$
```

show that the variance of each sensor's lifetime is $(k\sigma_T)^2$, while the normalized variance is σ_T^2 and the variance of the mean is $Var(\bar{T}) = \sigma_T^2/n$.

Load Balancing. Since we are maintaining k parallel queues that must work together to cover U, our calculations are sensitive to the requirement that the lifetime be the same in each queue. Figure 5 illustrates the dangers of improper load balancing.

Following [17], we can think of the observation of each sensor location as an independent Poisson trial, and use a Chernoff bound to ensure that the probability of a sub-interval $U_k(i)$ getting too few sensors is $o(1)$. Let N_i be a r.v. denoting the number of sensors in $U_k(i)$. Then for any $k < \frac{n}{3 \ln n}$, we have that

$$\Pr\left[\left| N_i - \frac{n}{k} \right| \geq \sqrt{\frac{3n \ln n}{k}} \right] \leq 2 \exp\left\{ -\frac{1}{3} \frac{n}{k} \frac{3k \ln n}{n} \right\} = \frac{2}{n} .$$

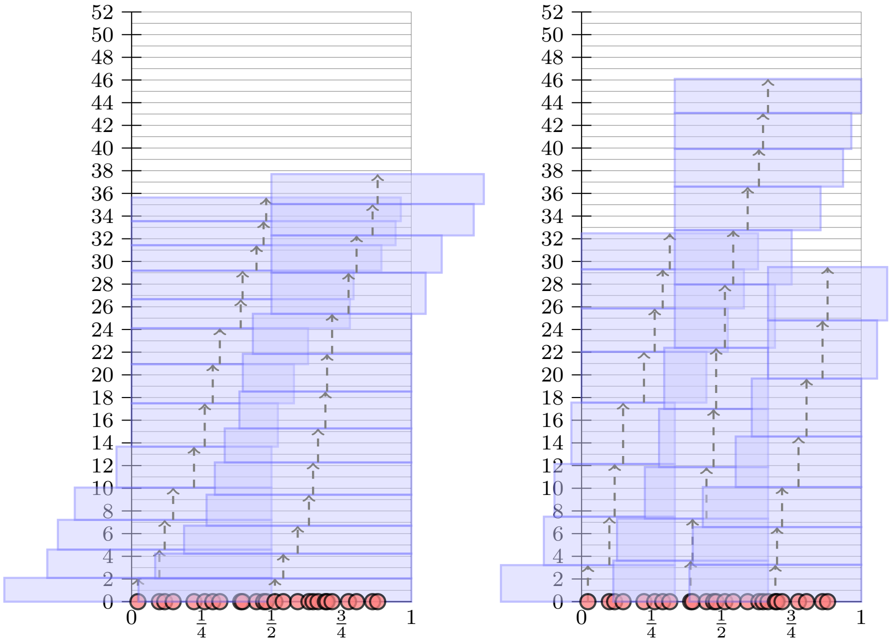
In our case, we need to bound the probability that *some* $U_k(i)$ has too few sensors in it, but using a union bound, the probability of this is at most $\frac{2k}{n}$, which still goes to 0 as $n \to \infty$ for a fixed k. This shows that with high probability, the deviations from the mean number of sensors in each interval are on the order of $O(\sqrt{n \ln n})$ for a fixed k.

Set $n = n_1 + n_2$, where $n_1 = k \cdot \min_{1 \leq i \leq k} N_i$. Our scheduler allows the n_1 sensors to run k-RoundRobin on perfectly balanced stacks, and then throws the n_2 leftover sensors away. Thus, the actual expected average lifetime of the algorithm is

$$\mathbb{E}[\bar{T}_{actual}] = \frac{n_1}{n} \cdot \mathbb{E}[\bar{T}] + \frac{n_2}{n} \cdot 0 \to \mathbb{E}[\bar{T}] = \mu_T \text{ , as } n \to \infty ,$$

since $n_2 = O(\sqrt{n \ln n})$ and thus $\frac{n_2}{n} \to 0$ as $n \to \infty$.

Observation 3 k-RoundRobin *provides the same worst-case and average-case performance as* RoundRobin*.*



(a) 2-RoundRobin(X) = 35.63 (b) 3-RoundRobin(X) = 29.50

Fig. 5 k-RoundRobin for $k = 2, 3$ against a random instance with 26 sensors. Even with proper load balancing, there are many inefficient sensors. The resulting average lifetimes are 1.370 (left) and 1.135 (right).

3.3 log-RoundRobin

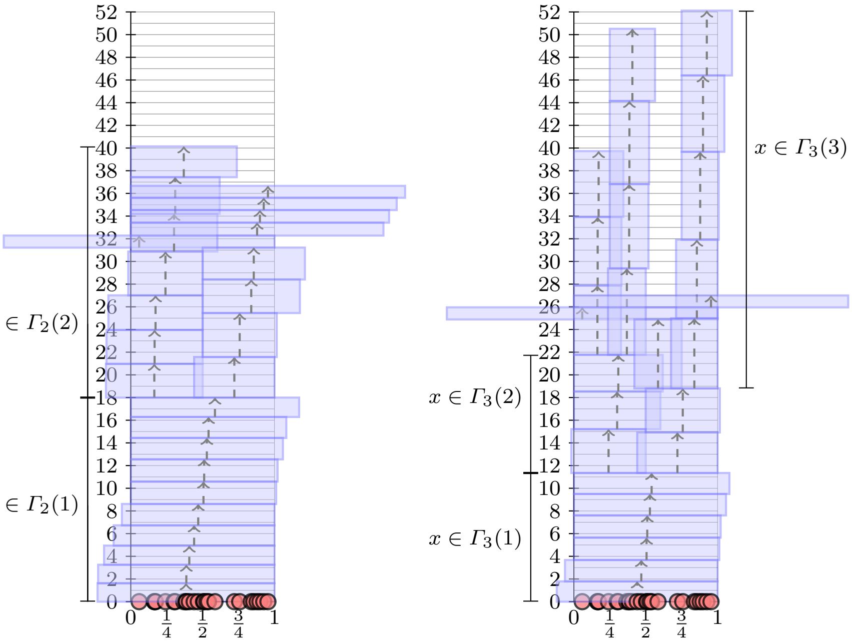
Nevertheless, clever applications of RoundRobin can yield efficient algorithms. While the expected lifetime of a sensor in RoundRobin is independent of the length of the interval it covers, it still performs better when it is near the center of the interval. Specifically, the expected lifetime of a sensor covering an interval $[a, b]$, that is located within a subinterval $U_{a,b}(\delta) = [\frac{b+a}{2} - \delta, \frac{b+a}{2} + \delta] \subseteq [a, b]$ for some $0 \leq \delta \leq \frac{b-a}{2}$, is given by

$$\mathbb{E}[T_{a,b}(X;\delta)] = \frac{1}{2\delta}\int_{\frac{b+a}{2}-\delta}^{\frac{b+a}{2}+\delta} \frac{dx}{\max(x-a, b-x)} = \frac{1}{\delta}\ln\left(1+\frac{2\delta}{b-a}\right). \tag{4}$$

Since the maximum lifetime is $2/(b-a)$, the expected normalized lifetime is $\mathbb{E}[\hat{T}_{a,b}(X;\delta)] = \frac{b-a}{\delta}\ln\left(1+\frac{2\delta}{b-a}\right)$, and the normalized variance is:

$$Var(\hat{T}_{a,b}(X;\delta)) = 4\left[1 - \frac{1}{1+\frac{b-a}{2\delta}} - \left(\frac{b-a}{2\delta}\cdot\ln\left(1+\frac{2\delta}{b-a}\right)\right)^2\right]. \tag{5}$$

Within the framework of using RoundRobin on subintervals $[a, b]$, but selecting only those sensors that are closest to the midpoints of those intervals,



(a) log-RoundRobin with depth 2

(b) log-RoundRobin with depth 3

Fig. 6 log-RoundRobin with depth 2 and 3 against a random instance with 26 sensors. The depth parameter of 2 depicted at left offers better load balancing than the parameter of 3 at right for 26 sensors. The schedule on the left achieves a lifetime of 36.66, while the one on the right achieves only 27.0

an algorithm emerges naturally: partition U into subintervals, but employ RoundRobin only on those sensors that are close to the midpoint of each subinterval. To make efficient use of each sensor, we construct a hierarchical series of such partitions. We call this algorithm log-RoundRobin, and it is indexed by a depth parameter k, which indicates the number of partitions it employs.

Formally, for a fixed positive integer k, we partition U into $2^k + 1$ subintervals $U_k(i) = [\frac{i}{2^k} - \frac{1}{2^{k+1}}, \frac{i}{2^k} + \frac{1}{2^{k+1}}] \cap U$ for $i = 0, 1, ..., 2^k$.[4] Each sensor thus belongs to one subinterval $U_k(i)$. The collection of sensors belonging to the same subinterval run RoundRobin over a wider coverage interval. To figure out its width, note that every integer between 1 and $2^k - 1$ can be written as 2^j times some odd integer between 1 and $2^k - 1$, for some $j = 0, \ldots, k$. We define

$$\eta(i) = \max_{0 \leq j \leq k} \{j : i = 2^j \cdot y, 1 \leq y \leq 2^k - 1, y \text{ is odd}\},$$

[4] Note that the first and last intervals, $U_k(0) = [0, 2^{-k-1}]$ and $U_k(2^k) = [1 - 2^{-k-1}, 1]$, respectively, are only half as wide as the others, all of which have width 2^{-k}.

Algorithm 3 log-RoundRobin

```
Input: An ordered set of sensor locations $X$, an optional depth parameter $k$
Output: A schedule $S$
if $k$ is null then
    $k \leftarrow \lfloor \max(1, \ln n) \rfloor$
end if
for $i$ from 1 to $n$ do
    $q_i = round(2^k \cdot x_i)$ {Determine to which of the $2^k + 1$ queues each sensor belongs}
end for
for $\ell$ from 1 to $2^k - 1$ do
    $j_\ell = k - \eta(\ell)$ {To which coverage group does this queue belong?}
    $parent_\ell = \{q_i \in q : q_i$ is the parent queue of $q_\ell\}$ {Which queue runs immediately before this one?}
    $left_\ell = \frac{\ell}{2^k} - 2^{-j_\ell}$
    $right_\ell = \frac{\ell}{2^k} + 2^{-j_\ell}$
    $Q_\ell = \{x_i \in X : q_i = \ell\}$
    $S_\ell = \text{RoundRobin}(Q_\ell, 0, left_\ell, right_\ell)$
    $T_\ell = \max_{S_\ell} \sigma$
end for
for $j$ from 1 to $k$ do
    for all $\ell$ such that $j_\ell = j$ do
        if $parent_\ell$ is null then
            $t_\ell = 0$
        else
            $t_\ell = T_{parent_\ell}$
            for all $i$ listed in $S_\ell$ do
                $\tau_i = \tau_i + t_\ell$
                $\sigma_i = \sigma_i + t_\ell$ {Push back each schedule by that of its parent}
            end for
        end if
        $T_\ell = \max_{S_\ell} \sigma$ {Reset the ending time of each queue}
    end for
end for
$S = \bigcup_{1 \leq \ell \leq 2^k - 1} S_\ell$
$T =$ lifetime of $S$
$S = S \cup \text{RoundRobin}(\{x_i \in X : q_i = \{0, 2^k\}\}, T, 0, 1)$
return $S$
```

to be the largest such integer.[5] If sensor $x \in U_k(i)$, then x is responsible for covering the interval around $i/2^k$ with radius $2^{\eta(i)-k}$. For example, the middle interval occurs when $i = 2^{k-1}$. Thus, since $\eta(2^{k-1}) = k - 1$, any sensor that lies in $U_k(2^{k-1})$ is assigned to cover all of U. Similarly, sensors within 2^{-k-1} of either $\frac{1}{4}$ or $\frac{3}{4}$ are assigned to cover the subintervals $[0, \frac{1}{2}]$ and $[\frac{1}{2}, 1]$, respectively. In this manner all sensors are near the center of their coverage intervals. A graphical depiction of the normalized sensor network lifetime as a function of location is shown in Figure 7.

[5] If the integers from 1 to $2^k - 1$ were placed in a binary tree, $\eta(i)$ tell us how high up i is in that tree.

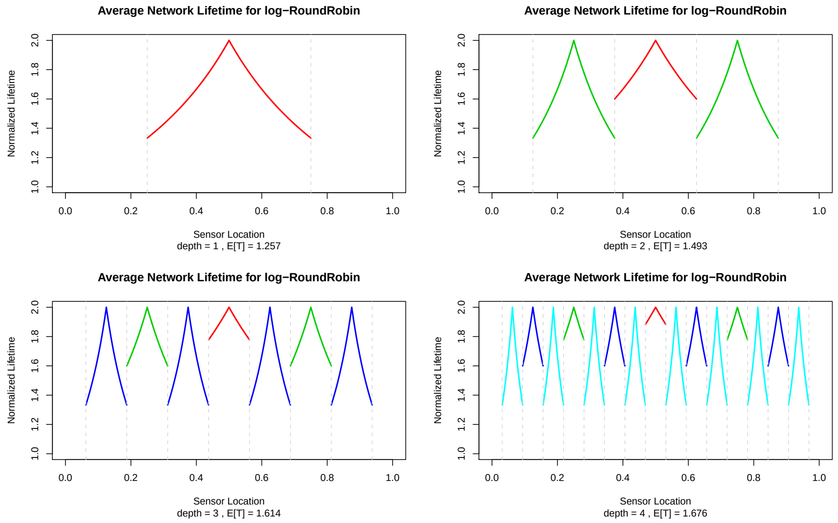


Fig. 7 Normalized sensor network lifetime for $k = 1, 2, 3, 4$ using the log-RoundRobin algorithm. Each color represents the lifetime of the sensors in $\Gamma_k(j)$. Note that while the actual lifetime of a sensor in $\Gamma_k(j)$ may reach 2^j, it must run in parallel with 2^{j-1} partners, so the normalized lifetime *of the group* is at most 2. The expected average lifetime of the network approaches 1.737752 as $k \to \infty$.

For $j = 1, ..., k$, we define $\Gamma_k(j)$ to the be the set of intervals that comprise the j^{th} level of the algorithm. Formally, we denote

$$\Gamma_k(j) = \left\{ \bigcup_{i=1}^{2^k-1} U_k(i) : j = k - \eta(i) \right\} .$$

Note that $\Gamma_k(j)$ consists of 2^{j-1} disjoint intervals, each of width 2^{-k}.[6] Thus $\Gamma_k(j)$ occupies 2^{j-k-1} of U. We can compute the expected normalized lifetime for $\Gamma_k(j)$ using Equation 4

$$\mathbb{E}[\hat{T}_k(j)] = \mathbb{E}[\hat{T}_{0,2^{-j+1}}(X; 2^{-k-1})] = 2^{k-j+2} \ln\left(1 + 2^{j-k-1}\right),$$

and the variance using Equation 5:

$$Var(\hat{T}_k(j)) = 4\left[1 - \frac{1}{1 + 2^{k-j+1}} - \left(2^{k-j+1} \cdot \ln\left(1 + 2^{j-k-1}\right)\right)^2\right] .$$

Summing over the $\Gamma_k(j)$'s to find the total expected normalized lifetime, we obtain

$$\mathbb{E}[\hat{T}_k] = \sum_{j=1}^{k} \frac{\mathbb{E}[\hat{T}_k(j)]}{2^{k-j+1}} = 2 \ln \prod_{j=1}^{k} \left(1 + 2^{j-k-1}\right) = 2 \ln \prod_{\ell=1}^{k} \left(1 + 2^{-\ell}\right) . \tag{6}$$

[6] We let $\Gamma_k(0)$ be the set of sensors assigned to $U_k(0)$ or $U_k(2^k)$, and have those run RoundRobin on U after all other sensors complete. Their contribution to the network lifetime becomes negligible as $k \to \infty$, so we omit it from our calculations.

The analogous infinite product is a q-series [22], denoted here by $\left(-1; \frac{1}{2}\right)_\infty$, for which we can compute an approximate limiting value. This leads directly to the expected average lifetime:

$$\mu_T^* \triangleq \mathbb{E}[\hat{T}] = \lim_{k \to \infty} \mathbb{E}[\hat{T}_k] = 2 \ln \left(\prod_{\ell=1}^{\infty} 1 + 2^{-\ell} \right) \approx 1.737752 \,.$$

The mean normalized variance satisfies

$$\mathbb{E}[Var(\hat{T}_k)] = \sum_{j=1}^{k} \frac{Var(\hat{T}_k(j))}{2^{k-j+1}} = 4 \left[\sum_{\ell=1}^{k} \frac{1}{1+2^\ell} - 2^\ell \cdot \ln^2 \left(1 + 2^{-\ell}\right) \right] ,$$

which has the approximate limit of 0.02202547 as $k \to \infty$. Computation of the total variance is omitted, since it requires extensive calculation that adds little elucidation, but it will converge to the above as $k \to \infty$.

Furthermore, it is clear from Figure 7 that the worst-case lifetime occurs when a sensor in $\Gamma_k(k)$ lies near one of the endpoints of the interval on which it is active. The normalized lifetime at this point is $4/3$, a constant. This provides the same worst-case performance as RoundRobin. Note that this result only holds for balanced loads – if loads are unbalanced, the worst-case performance of log-RoundRobin is unbounded.

Load Balancing, revisited. In log-RoundRobin, each set $\Gamma_k(j)$ for $j = 1, ..., k$ maintains 2^{j-1} parallel queues. Proper functioning of our algorithm requires balanced loads across these queues, but the hierarchical structure of log-RoundRobin alleviates the load balancing issue, since the $\Gamma_k(j)$'s are scheduled in ascending order of j. To see this, suppose that the left half of $\Gamma_k(2)$ runs out, while the right half is still going. U remains covered if the left half of $\Gamma_k(3)$ starts running alongside the right half of $\Gamma_k(2)$. In this manner load imbalances are averaged out over the k levels of the algorithm.

Nevertheless, a Chernoff bound analogous to the one used above for k-RoundRobin will show that for $k < \ln n$, with high probability N_i will deviate from its mean of $\frac{n}{2^k}$ by $O(\sqrt{n \ln n})$. Setting $n_1 = 2^k \cdot \min_{1 \le i \le 2^k - 1} N_i$ yields

$$\mathbb{E}[\bar{T}_{actual}] \geq \frac{n_1}{n} \cdot \mu_T^* + \frac{n_2}{n} \cdot 0 \to \mu_T^*, \text{ as } n \to \infty \,.^7$$

Theorem 2 *For sufficiently large n, the* log-RoundRobin *algorithm achieves an approximation ratio of* 1.151 *in the average case with high probability.*

[7] The inequality is justified by the preceding argument that in practice, the actual load balancing will work at least this well.

3.4 Optimizations of log-RoundRobin

Still, it is clear from Figure 7 that efficiency is highest in $\Gamma_k(1)$ and lowest in $\Gamma_k(k)$. We can show that in fact, the relative efficiency of $\Gamma_k(k)$ is the constant $\frac{1}{2\ln\frac{3}{2}} \approx 1.233$. On the other hand, it is easy to see that the relative efficiency of $\Gamma_k(1)$ approaches 1 as $k \to \infty$. Therefore, we can improve the efficiency of log-RoundRobin by shrinking the intervals over which $\Gamma_k(k)$ is active. Note that since every $\Gamma_k(j)$ for $j = 1, ..., k-1$ borders $\Gamma_k(k)$ on both sides, we maintain balanced loads across each $\Gamma_k(j)$ even as we shrink the width of $\Gamma_k(k)$. Let $0 \le \epsilon = \epsilon(k) \le 1$ be a parameter measuring the inward shift of the boundaries of $\Gamma_k(k)$. Then using Equation 4, the expected normalized lifetime becomes

$$\mathbb{E}[\hat{T}_k(j,\epsilon)] = \mathbb{E}\left[\hat{T}_{0,2^{-j+1}}\left(X; \frac{1+\epsilon}{2^{k+1}}\right)\right] = \frac{2^{k-j+2}}{1+\epsilon}\ln\left(1+(1+\epsilon)2^{j-k-1}\right)$$

for $j = 1, ..., k-1$, and

$$\mathbb{E}[\hat{T}_k(k,\epsilon)] = \mathbb{E}\left[\hat{T}_{0,2^{-k+1}}\left(X; \frac{1-\epsilon}{2^{k+1}}\right)\right] = \frac{4}{1-\epsilon}\ln\left(\frac{3-\epsilon}{2}\right).$$

Taking the weighted average again, we have a generalization of Equation 6 that can be expressed as another q-series:

$$\mathbb{E}[\hat{T}_k(k,\epsilon)] = 2\ln\left(\frac{3-\epsilon}{2}\right)\prod_{i=2}^{\infty} 1+(1+\epsilon)2^{-i} = 2\ln\frac{(3-\epsilon)\left(-(1+\epsilon);\frac{1}{2}\right)_\infty}{(\epsilon+3)(\epsilon+2)}.$$

We can find the optimal $\epsilon(k)$ using elementary calculus, but unfortunately a general solution requires factoring a polynomial of degree $k-1$:

$$T_k'(\epsilon) = 0 \Rightarrow \frac{1}{3-\epsilon} = \sum_{j=1}^{k-1}\frac{1}{2^{j+1}+1+\epsilon}. \quad (7)$$

However, since $T_k'(0) > 0$ for $k > 3$, and $T_k'(1) < 0$ for $k > 0$, the derivative has a root between 0 and 1 for $k > 3$ by the Intermediate Value Theorem. Moreover the Second Derivative Test confirms that for $k > 1$, each of these roots is a local maximum.

Numerical approximations of some relevant roots of this polynomial are shown in Table 2, alongside the expected network lifetime of the optimized algorithm. These optimizations improve the expected average network lifetime by more than 3% above that of log-RoundRobin.

Theorem 3 *For sufficiently large n, the optimized* log-RoundRobin *algorithm achieves an approximation ratio of* 1.117 *in the average case with high probability.*

Convergence. The Ratio Test, combined with L'Hôpital's Rule, will show that both series $T_k(\epsilon)$ and $T_k'(\epsilon)$ converge as $k \to \infty$ for any fixed $\epsilon \in [0,1]$. As we have not found a closed functional form for either limit, we cannot prove that the optimal ϵ converges to a limit.

k	ϵ	$T_k(0)$	$T_k(\epsilon)$	Gain %	$\|U_k(k;\epsilon)\|$%
2	0	1.492783	1.492783	0	50.00
3	0	1.614033	1.614033	0	50.00
4	0.211103	1.675576	1.696157	1.23	39.44
5	0.371297	1.706584	1.743439	2.16	31.44
6	0.448178	1.722149	1.767123	2.61	27.59
7	0.485871	1.729946	1.778990	2.84	25.71
8	0.504537	1.733848	1.784931	2.95	24.77
10	0.518459	1.736777	1.789391	3.03	24.08
12	0.521929	1.737509	1.790506	3.05	23.90
15	0.522941	1.737723	1.790831	3.06	23.85
20	0.523081	**1.737752**	1.790876	3.06	23.85

Table 2 Numerical approximations for optimized log-RoundRobin. Note that $T_{20}(0)$ equals $T_\infty(0) = \mu_T^*$ to six digits. The rightmost column shows the percentage of U that is covered by $\Gamma_k(k;\epsilon)$.

4 Summary

Our emphasis on the *lifetime* of a network addresses application considerations more directly than previous attempts to maximize the number of covers, or conserve energy in a single cover. Moreover, our study of the expected performance of algorithms in the average case is likely to be more relevant in many realistic applications.

4.1 Discussion and Open Problems

While it is known that RSC is NP-hard [7], and that Sensor Strip Cover is NP-hard if each sensor's radius is allowed to be set only once [4], it is not known whether Sensor Strip Cover itself is NP-hard. Furthermore, both of these hardness proofs require non-uniform initial battery charges, and thus it is not clear whether Sensor Strip Cover with uniform batteries is NP-hard, or even if a gap exists between it and the set-once restricted version. Note that RSC is NP-hard in general, but admits a polynomial-time solution in the uniform battery case [7].

In this paper we found the expected performance of several algorithms under the assumptions that the sensor locations X were distributed uniformly at random, and the initial battery charges B were the same. The assumption of uniform battery charges was made without loss of generality, since we have provided a rationale for translating a non-uniform battery instance into a uniform battery instance on which most algorithms will have roughly the same performance. However, this argument ignored the question of load-balancing that is necessary for several of our algorithms to function properly. Thus, a better assumption might be that the product XB is uniformly distributed. In this event, we contend that our results will still hold for sufficiently large n. On the other hand, it is not clear how our results would change under radically different assumptions about the distribution of sensor locations *or* battery charges (e.g. a binomial distribution). This type of analysis could be performed for any pair of probability distributions with finite support.

In this average-case analysis, we measured the quality of approximation against the theoretical maximum determined by the total battery charge of the system ($2n$). This is only an upper bound on the performance of Opt, since it is clear that there are instances in which even Opt cannot achieve a lifetime of $2n$ (e.g. $X = (\frac{1}{3})$). Thus, the expected performance of Opt against uniform random deployment is unknown. It may be the case that the optimized log-RoundRobin algorithm is in fact very close to being optimal.

This problem setting is rich, in that there are many variations in the setup that can alter the resulting analysis dramatically. In this paper we have assumed that the battery charges dissipate in direct inverse proportion to the assigned sensing radius (e.g. $\tau = 1/\rho$). It is natural to suppose that an exponent could factor into this relationship, so that, say, the radius drains in quadratic inverse proportion to the sensing radius (e.g. $\tau = 1/\rho^2$). This may correspond more closely to a realistic dissipation of power.

We have assumed that once deployed, the sensors cannot be moved. Phelan et al. [18] have considered a variation of Sensor Strip Cover in which the sensors are allowed to be moved, at a certain cost. This line of research could have interesting connections with our findings.

Finally, while we have restricted our attention to a one-dimensional coverage region, one could consider a variety of similar problems in higher dimensions. For example, one might keep the sensor locations restricted to the line, but consider a two-dimensional coverage region (i.e. *beach* coverage). Conversely, the sensors could be located in the plane, while the coverage region remains one-dimensional (i.e. *road* coverage). Of course, from an application point-of-view the most important variation would allow both the sensor locations and the coverage region to be two-dimensional.

References

1. Alt, H., Arkin, E.M., Brönnimann, H., Erickson, J., Fekete, S.P., Knauer, C., Lenchner, J., Mitchell, J.S.B., Whittlesey, K.: Minimum-cost coverage of point sets by disks. In: Proceedings of the 22nd ACM Symposium on Computational Geometry, pp. 449–458 (2006)
2. Bar-Noy, A., Baumer, B.: Maximizing network lifetime on the line with adjustable sensing ranges. In: Proceedings from the 7th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, pp. 28–41 (2011)
3. Bar-Noy, A., Baumer, B., Rawitz, D.: Changing of the guards: Strip cover with duty cycling. In: Proceedings from the 19th International Colloquium on Structural Information and Communication Complexity, pp. 36–47 (2012)
4. Bar-Noy, A., Baumer, B., Rawitz, D.: Set it and forget it: Approximating the set once strip cover problem. Arxiv preprint arXiv:1204.1082v1 (2012)
5. Berman, P., Calinescu, G., Shah, C., Zelikovsky, A.: Efficient energy management in sensor networks. Ad Hoc and Sensor Networks, Wireless Networks and Mobile Computing **2**, 71–90 (2005)
6. Bilò, V., Caragiannis, I., Kaklamanis, C., Kanellopoulos, P.: Geometric clustering to minimize the sum of cluster sizes. In: Proceedings from the 13th Annual European Symposium on Algorithms, pp. 460–471 (2005)

7. Buchsbaum, A.L., Efrat, A., Jain, S., Venkatasubramanian, S., Yi, K.: Restricted strip covering and the sensor cover problem. In: Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms, pp. 1056–1063 (2007)
8. Cardei, M., Du, D.Z.: Improving wireless sensor network lifetime through power aware organization. Wireless Networks **11**(3), 333–340 (2005)
9. Cardei, M., Wu, J.: Coverage in wireless sensor networks. In: Handbook of Sensor Networks: Compact Wireless and Wired Sensing Systems, chap. 19, pp. 432–446. CRC Press (2004)
10. Cardei, M., Wu, J., Lu, M.: Improving network lifetime using sensors with adjustable sensing ranges. International Journal of Sensor Networks **1**(1/2), 41–49 (2006)
11. Cardei, M., Wu, J., Lu, M., Pervaiz, M.O.: Maximum network lifetime in wireless sensor networks with adjustable sensing ranges. In: Proceedings from the 3rd IEEE International Conference on Wireless and Mobile Computing, Networking and Communications, pp. 438–445 (2005)
12. Chambers, E.W., Fekete, S.P., Hoffmann, H.F., Marinakis, D., Mitchell, J.S.B., Srinivasan, V., Stege, U., Whitesides, S.: Connecting a set of circles with minimum sum of radii. In: Proceedings from the 12th International Symposium on Algorithms and Data Structures, pp. 183–194 (2011)
13. Dhawan, A., Prasad, S.K.: A distributed algorithmic framework for coverage problems in wireless sensor networks. International Journal of Parallel, Emergent and Distributed Systems **24**(4), 331–348 (2009)
14. Gibson, M., Kanade, G., Krohn, E., Pirwani, I.A., Varadarajan, K.R.: On clustering to minimize the sum of radii. SIAM J. Comput. **41**(1), 47–60 (2012)
15. Gibson, M., Varadarajan, K.R.: Optimally decomposing coverings with translates of a convex polygon. Discrete & Computational Geometry **46**(2), 313–333 (2011)
16. Lev-Tov, N., Peleg, D.: Polynomial time approximation schemes for base station coverage with minimum total radii. Computer Networks **47**(4), 489–501 (2005)
17. Mitzenmacher, M., Upfal, E.: Probability and computing - randomized algorithms and probabilistic analysis. Cambridge University Press (2005)
18. Phelan, B., Terlecky, P., Bar-Noy, A., Brown, T., Rawitz, D.: Should i stay or should i go? maximizing lifetime with relays. In: Proceedings from the 8th IEEE International Conference on Distributed Computing in Sensor Systems, pp. 1–8 (2012)
19. Saipulla, A., Westphal, C., Liu, B., Wang, J.: Barrier coverage of line-based deployed wireless sensor networks. In: Proceedings from the 28th IEEE International Conference on Computer Communications, pp. 127–135 (2009)
20. Taniguchi, Y., Kitani, T., Leibnitz, K.: A uniform airdrop deployment method for large-scale wireless sensor networks. International Journal of Sensor Networks **9**(3/4), 182–191 (2011)
21. Wang, L., Xiao, Y.: A survey of energy-efficient scheduling mechanisms in sensor networks. Mobile Networks and Applications **11**(5), 723–740 (2006)
22. Weisstein, E.W.: q-series. From MathWorld–A Wolfram Web Resource. URL http://mathworld.wolfram.com/q-Series.html. Visited on 04/23/2012
23. Wu, J., Yang, S.: Energy-efficient node scheduling models in sensor networks with adjustable ranges. International Journal of Foundations of Computer Science **16**(1), 3–17 (2005)