Energy Efficient Monitoring in Sensor Networks[*]

Amol Deshpande[**], Samir Khuller[***], Azarakhsh Malekian[***], Mohammed Toossi[†]

Computer Science Department, University of Maryland,
A.V. Williams Building, College Park, MD 20742, U.S.A.
{amol,samir,malekian}@cs.umd.edu, mtoossi@gmail.com

Abstract. In this paper we study a set of problems related to efficient energy management for monitoring applications in wireless sensor networks. We study several generalizations of a basic problem called Set k-Cover. The problem can be described as follows: we are given a set of sensors, and a set of regions to be monitored. Each region can be monitored by a subset of the sensors. To increase the lifetime of the sensor network, we would like to partition the sensors into k sets (or time-slots), and activate each set of sensors in a different time-slot, thus extending the battery life of the sensors by a factor of k. The goal is to find a partitioning that maximizes the total coverage of the regions for a given k. This problem is known to be NP-hard. We develop an improved approximation algorithm for this problem using a reduction to Max k-Cut. Moreover, we are able to demonstrate that this algorithm is practical, and yields almost optimal solutions in practice.

We also consider generalizations of this problem in several different directions. First, we allow each sensor to be active in α different sets (time-slots). This means that the battery life is extended by a factor of $\frac{k}{\alpha}$, and allows for a richer space of solutions. We also consider different coverage requirements, such as requiring that all regions, or at least a certain number of regions, be covered in each time slot. In the Set k-Cover formulation, there is no requirement that a region be monitored at all, or in any number of time slots. We develop a randomized rounding algorithm for this problem.

We also consider extensions where each sensor can monitor only a bounded number of regions in any time-slot, and not all the regions adjacent to it. This kind of problem may arise when a sensor has a directional camera, or some other physical constraint might prevent it from monitoring all adjacent regions even when it is active. We develop the first approximation algorithms for this problem.

1 Introduction

Wireless sensor networks, comprising of tiny, radio-enabled sensing devices, open up new opportunities for low-cost monitoring and data collection in a wide variety of applications [2]. This has been a very active research area in recent years, and the work
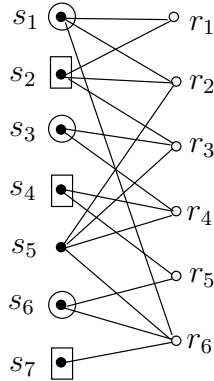

Fig. 1. Graph H and solution for $k = 3$ is $S_1 = \{s_2, s_4, s_7\}$, $S_2 = \{s_1, s_3, s_6\}$ and $S_3 = \{s_5\}$ and $R_1 = \{r_1, r_2, r_3, r_4, r_5, r_6\}$, $R_2 = \{r_1, r_2, r_3, r_4, r_5, r_6\}$ and $R_3 = \{r_2, r_3, r_4, r_6\}$.

in this area has spanned a range of issues including designing tiny radio-equipped sensor boards [20] and network protocols to handle the noisy environments and resource constraints [18], building general purpose applications that can run on the sensors [16, 23, 29], and finally, deployment of large-scale sensor networks in real-world scenarios [24, 9, 19].

Perhaps the primary challenge in designing protocols and algorithms for wireless sensor networks is efficient energy management, since we desire the wireless devices to communicate and to continue to function effectively for long periods of time. In this paper we study a basic set of problems dealing with energy efficient *target monitoring* in sensor networks. One particular question of this type was first formalized in a paper by Slijepcevic and Potkonjak [25], in which they asked for a collection of disjoint set covers given a sensor network.

Let $H = (S \cup R, E)$ denote a bipartite graph in which nodes in set S correspond to sensors, and nodes in set R correspond to regions (see Figure 1 for an example). There is an edge between node s_i and node r_j if sensor s_i can monitor region r_j. By keeping all sensors activated all the time, clearly all the regions can be monitored continuously (assuming no nodes in R have zero degree). The problem with this solution is that the sensors may not last very long. One might now wonder if a better solution can be obtained by exploiting the redundancy in the sensor network. There are multiple ways in which we could formulate this problem. One approach is to partition the nodes in S into k sets $S_1, \ldots, S_k$, such that the sensors in set S_i cover all the regions in R. This is also referred to as the domatic set problem [12] for which a randomized approximation algorithm with factor $O(\log n)$ has been proposed. An alternative formulation (called Set k-cover), studied by Abrams et al. [1], instead asks for a partitioning that maximizes the total regions covered. More formally, let $R_i \subseteq R$ denote the regions covered by the sensors in S_i; in other words, $R_i = \{r | (s, r) \in E \wedge s \in S_i\}$. The goal is to maximize $\sum_{i=1}^{k} |R_i|$.

Given such a partitioning, the idea then is to cycle through the k sets S_i in a round-robin fashion. When we activate all sensors in set S_i, we cover the regions in R_i. Thus the objective function tries to maximize the coverage of the regions in the different time slots. We will consume significantly less energy this way, as each sensor is activated in only one of the k sets. Moreover there is evidence to suggest that the battery life of sensors is increased significantly when batteries are used in short bursts, rather than being used continuously [5]. We note that the cost of deactivating and reactivating a sensing device can be non-negligible; however that cost will be amortized away as long as the time duration of the slots is sufficiently large.

At the same time we will maximize the coverage over time. Ideally each region will belong to R_i for each value of i. Notice that we have relaxed the requirement that each region is always monitored. In practice, if k is not very large it is entirely possible that we can actually monitor all regions at all times. Of course, the larger the value of k, the longer we extend the lifetime of the system, while paying a penalty of lowering the coverage level within each time-slot. The average coverage obtainable also depends on the redundancy of coverage, i.e., the average number of sensors that monitor each region.

1.1 Problem Variations and Generalizations

In this paper, we identify and address several generalizations of this problem, many of which, to our best knowledge, have not been addressed before. One generalization we define is called the Set (k, α)-Cover problem. We would like to find k sets, $S_1, S_2, \ldots, S_k$. As before, $S_i \subset S$. We require that each vertex in S belong to at most α such sets. More formally, for all $s_j \in S$ we require that $|\{i | s_j \in S_i\}| \leq \alpha$. A solution for this problem can be mapped to a sensor schedule in which each sensor is active in α of the k time-slots. We have relaxed the requirement that the S_i sets are disjoint which corresponds to the case when $\alpha = 1$. In Section 3 we show that this problem is also NP-hard.

Since each sensor now belongs to α sets, the battery life is extended by a factor of $\frac{k}{\alpha}$. Our main goal really is to maximize the battery life, subject to adequate coverage requirements. However, for the development of the algorithms, it is easier to fix the parameters k and α and to develop algorithms that work with these parameters. In practice one would consider a spectrum of solutions produced by our algorithms for different choices of k and α.

We then consider a generalization where sensors have *capacity constraints*. In many scenarios, even though a sensor may be able to cover multiple regions, at any given time, it may only be able to actually monitor one or a small number of them. For example, a pan-and-tilt camera can monitor only one region (or a few regions) at any time. We call this a capacity constraint, and denote by $c(s_i)$ the maximum number of regions sensor s_i can cover in one time slot. The goal is to solve the Set k-Cover or Set (k, α)-cover problems given such capacity constraints.

Finally, given a k or a (k, α) pair, a sensor cover problem asks for the best designation of the sensors to k time slots (along with a designation of which regions to monitor for the capacity-constrained version) that optimizes some coverage property. The optimization goal itself could be one of the following three:

- **avg-coverage:** The average coverage over the k time slots, i.e., $\frac{\sum_{i=1}^{k} |R_i|}{k}$.

- **min-coverage-time:** The minimum value of the fractional coverage in any time slot, i.e., $\min_{i=1}^{k} \frac{|R_i|}{|R|}$.

- **min-coverage-region:** The minimum over all regions, of the fraction of time a region is covered. This is important for applications where high coverage of regions is required over time.

Different applications may demand support for different optimization goals. Further, the problems may be framed in different ways depending on the optimization goal. For example, for the min-coverage-time version, we could fix a coverage requirement, by specifying that $|R_i| \geq \gamma |R|$ for some $1 \geq \gamma > 0$ and ask to maximize k.

The above discussion suggests a classification of monitoring problems using three dimensions: Set k-cover vs Set (k, α)-cover, capacitated (C) vs non-capacitated (NC), and avg-coverage vs min-coverage-time vs min-coverage-region. Under this classification, Abrams et al. [1] study the {Set k-cover, NC, avg-coverage} version of the problem.

We now consider a slightly different view of the Set k-Cover problem. Given the bipartite graph H describing the sensor-region relationship, we construct the following hypergraph $G = (V, E)$. Each node in V corresponds to a sensor. For each region r, we create a hyperedge e that contains the set of sensors that cover the region. We assume that each region is covered by at most d sensors, i.e., the hyper edge has size d. The goal now is to color the nodes of the graph with k colors (this is simply a way to view the partitioning into k sets). The objective is to maximize the total *benefit* of all hyper edges. The benefit of a hyper edge is the number of different colors that the nodes in the hyper edge are colored with. If all nodes in this hyper edge have the same color, then the benefit is 1 since they are all in the same set. If the nodes have k different colors, then this region is always monitored and its benefit is k.
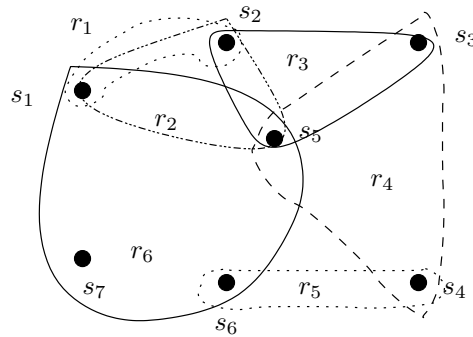

Fig. 2. Hyper edge structure for the example from Fig. 1.

The practical problem is of interest for small d, so it is worth studying this case in more detail, since we do not expect too many sensors to cover the same region, otherwise this suggests that the density of sensors is too high. For $d = 2$ this problem is clearly related to the well studied max k-cut problem for which a SemiDefinite Programming (SDP) based algorithm does very well [13, 14]. The max k-cut problem asks for a partitioning of the vertices of a graph into k groups so as to *maximize the number of edges across the cut* (edges connecting vertices in two different groups). While both problems ask for a partitioning of the vertices, the precise objective functions are different.

1.2   Prior Work

Designing sleep schedules to maximize the network lifetime while guaranteeing coverage has been one of the most active research areas in wireless sensor networks. Cardei and Wu [8] survey the work in this area, and identify two types of coverage problems, *area* coverage (where the goal is to maximally cover the area the sensor network is deployed in), and *target* coverage (where the goal is to cover a set of targets).

The problem we address in this paper can be seen as a target coverage problem, and we briefly review the prior work on this problem. Slijepcevic and Potkonjak [25] pose the problem with full coverage requirement; given a sensor network, the goal is to identify mutually exclusive sets of sensor nodes such that the members of each set cover the monitored regions (targets) completely. They provide several heuristics for this problem. Abrams, Goel and Plotkin [1] develop approximation algorithms for the Set k-Cover problem, where k is provided, and the goal is to find a partitioning that maximizes the coverage. They present a simple randomized algorithm, where each sensor is assigned to one of the k sets, and they show that the resulting solution approximates the optimal solution within a factor of $1 - \frac{1}{e}$. In fact their bound is 0.75 when $k = 2, d = 2$ and approaches $(1 - \frac{1}{e})$ for large d and k. They also show that it is NP-hard to get a polynomial time approximation algorithm with a factor better than $\frac{15}{16} + \epsilon$ for any $\epsilon > 0$. This is shown by a direct reduction from the E4-SET SPLITTING problem for which a $\frac{7}{8} + \epsilon$ hardness has been shown by Hastad [15]. However, the gap between $1 - \frac{1}{e}$ and $\frac{15}{16}$ is significant and our goal is to try and consider other approaches that can be used to narrow this gap further. They also present a distributed greedy algorithm that is a $\frac{1}{2}$ approximation for the problem. Hsin and Liu [17] analyze two types of sleep schedules for sensor networks, *random sleep* and *coordinated sleep*, and consider their effects of network coverage. Lu et al. [22] consider a similar problem of partitioning the sensor into disjoint sets, with the goal to minimize the end-to-end communication delay.

More recently, Buchsbaum et al. [6] considered a geometric version of the sensor cover problem where each sensor monitors a contiguous region in the area the sensor network is deployed in; the goal is to find, for each sensor, the time to activate it (after activating a sensor, it continues to be active until its battery is exhausted) so as to maximize the total time for which the entire area is covered. They exploit the geometric nature of the problem to develop several approximation algorithms for this problem.

Aloupis et al. [4] consider a related problem of coloring a set of points in a 2D or 3D space, so that every region of a certain family (e.g. unit disks, corresponding to the regions to be covered in our problem) that contains at least a certain number

of points (sensors) with different colors. They consider the problem of identifying the minimum of number of colors $p(k)$ needed to guarantee that each region contains at least k different colors.

Set k-cover problem can be seen as a special case of the welfare maximization problem where the goal is to allocate a set of items (in our case, sensors) to a set of players (time-slots), given utility functions for each player (see e.g. Feige [11], Vondrak [26]). Since this work addresses a more general problem, the approximation ratios of the proposed algorithms are typically worse than the bounds obtained for the Set k-cover problem [1]. For instance, Feige provides an algorithm with an approximation ratio of $1/2$ for subadditive utility functions, whereas Vondrak develops a $(1 - 1/e)$-approximation for submodular utility functions under a value oracle model (set cover is both subadditive and submodular).

We are not aware of any work that has considered the Set (k, α)-cover problem as described in this paper, or the capacity-constrained versions of this problem. Cardei et al. [7] consider a version of the problem that is similar to the Set (k, α)-cover problem, where they allow sensors to belong to multiple sets, and allow the sets to be active for different durations. They present several heuristics for solving the problem. Another related paper is [21] in which the unit capacity case is considered.

Several researchers have considered the issues of maintaining connectivity while designing sleep schedules (e.g. [30, 27]). Wang et al. [27] analyze the relationship between (area) coverage and connectivity, and show that if the communication radius is at least twice as large as the sensing radius, then full coverage guarantees connectivity. In this paper, we focus on the coverage issues, and leave the exploration of connectivity issues to future work.

1.3 Contributions

Our first algorithm (see Section 2) shows how to "reduce" the set k-cover problem to the max k-cut problem so we can apply known approximation methods for the latter. In fact, this approach gives rise to an extremely fast and practical method to solve the problem. We also prove some improved approximation factors for small d using this approach (see Section 2). For $d \leq 3$, this gives significantly improved worst case approximation factors compared to the randomized approach in [1]. However, the key point is that this approach gives almost optimal solutions in practice, even for larger values of d. We also present a worst case analysis of this method for large d.

We next consider a variation in which each sensor can be active in a bounded number of time slots and the goal is to cover each region in each time slot. In Section 3 we develop an LP based randomized rounding algorithm for approximating the lifetime of the network for the {Set (k, α)-cover, NC, min-coverage-time} version of the problem (an extension to the min-coverage-region case is straightforward). We fix a (k, α) pair. Assuming that a feasible integral solution exists, we are guaranteed to find a feasible fractional solution. Once we obtain the fractional solution, we round it. We use a scaling parameter s to do the rounding. This may increase the number of sets a node belongs to, with the expected number being $s\alpha$. We can prove that the probability that each region is covered is very high and at least $1 - \frac{1}{e^s}$.

For the case where each sensor can only cover one region (unit capacity) when it is active, we develop a polynomial time algorithm that computes an optimal solution for the capacitated set (k, α)-cover problem (see Section 4) for maximizing avg-coverage. For the case of *arbitrary* capacities we develop a polynomial time $(1-\frac{1}{e})$ approximation since we can show that the problem is NP-hard even when the capacities are three.

In Section 5 we illustrate the data sets as well as the performance of our set k-cover method. On the data sets in which we were able to compute the optimal solution, we found that our approach gave almost optimal solutions almost every time.

In addition, we also implemented and tested our algorithms for the version of the problem with capacities. For this case, we simply compare the quality of our solution to an *upper* bound on the optimal solutions since we were unable to compute the optimal solutions. Again, we found that the algorithms performance is significantly better than the worst case bounds we derive in Section 4.

One interesting feature about our algorithms is that they take the structure of the problem into account. The randomized approach in [1] is oblivious to the actual graph structure.

2　Max *k*-cut Approach

We now discuss the Set k-Cover problem. We are given a hyper graph $G = (V, E)$, where each node in V corresponds to a sensor. For each region r we create a hyper edge e that contains the set of sensors that cover the region. The goal now is to color the nodes of the graph with k colors. The objective is to maximize the total *benefit* of all hyper edges. The benefit of a hyper edge is the number of different colors that the nodes in the hyper edge are colored with.

Our algorithm works as follows. We replace each hyper edge $e = \{a_1, \ldots, a_p\}$ by edges (a_i, a_j) for $i \neq j$, essentially replacing each hyper edge by a clique (see Fig. 3). We denote this new (multi-graph) by $G' = (V', E')$. Let C^* denote an optimal solution for the Max k-cut problem for the graph G', and let E_{C^*} denote the corresponding set of edges that cross the cut.

We apply the SDP based Max k-Cut approximation algorithm [13] that tries to maximize the number of edges across the cut after partitioning the vertices into k sets. We use the partitioning produced by this algorithm, even though our original objective function is different. Let C' be the cut produced by the max k-cut algorithm, and $|E_{C'}|$ the number of edges across the cut.

Let α_k be the approximation ratio of the SDP based algorithm [13] for Max k-Cut. Frieze and Jerrum showed that α_k satisfies the following.

(i) $\alpha_k > 1 - \frac{1}{k}$

(ii) $\alpha_k - (1 - \frac{1}{k}) \sim 2\frac{\ln k}{k^2}$

(iii) $\alpha_2 \geq 0.878, \alpha_3 \geq 0.8, \alpha_4 \geq 0.85, \alpha_{10} \geq 0.926, \alpha_{100} \geq 0.99$.

In the next two subsections we present a worst case analysis of this method. We first give a simple analysis for the case $d = 2$ and this will convey some intuition about why this scheme works well. The analysis for general d is more complex, and we present it subsequently.

2.1 Analysis for $d = 2$

For the case of $d = 2$ (where each region is monitored by 2 sensors), we are able to use an α_k approximation for Max k-cut to obtain an approximation guarantee of $\frac{1}{2}(1+\alpha_k)$.

For this case, the optimal solution has total benefit $(E - E_{C^*}) + 2E_{C^*} = E + E_{C^*}$ (each edge that is cut corresponds to a region monitored in two time slots). We know that we will get at least $\alpha_k E_{C^*}$ edges across the cut once we run the Max k-Cut algorithm. Thus we get a benefit of at least $2\alpha_k E_{C^*} + (E - \alpha_k E_{C^*}) = E + \alpha_k E_{C^*}$. Taking the ratio, and using the fact that $E_{C^*} \leq E$ gives the desired bound. This is significantly better than the oblivious approach of randomly coloring the nodes, regardless of the structure of the graph. If we plug in $\alpha_2 = 0.878$ [14] then we obtain a bound of 0.939. If we plug in $\alpha_3 = 0.8$ [13], we get a bound of 0.9. (In contrast the randomized method of [1] gives a bound of $(1 - \frac{1}{2k})$, which is 0.75 and 0.83 for $k = 2, 3$. Note that α_k also improves as k increases [13]).
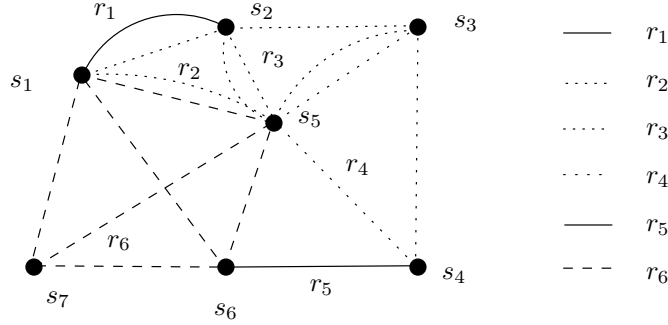

Fig. 3. Converting the hyper graph into a multi-graph for the example from Fig. 2.

2.2 Analysis for general d (regular hypergraphs)

The next theorem presents the approximation ratio obtained by the Max k-Cut approach.

Theorem 1. *The benefit obtained from the method based on Max k-cut for d-regular hypergraphs is at least $\frac{1}{d} + \frac{2\alpha_k}{d+2}(1 - \frac{1}{d})$ fraction of the maximum benefit for the Set k-cover problem.*

Let E be the set of hyper edges in G. Let E_j^* be the hyper edges in E that intersect j sets, in an optimal solution for the Set k-Cover instance. When $j > k$ we assume that $E_j^* = \emptyset$.

We first give a lower bound on $|E_{C^*}|$ as a function of the total benefit B^* of an optimal solution for the set k-cover problem.

Lemma 1.

$$|E_{C^*}| \geq \frac{d}{2}(B^* - |E|)$$

Proof. By definition, note that $B^* = \sum_{j=1}^{d} j|E_j^*|$ and $|E| = \sum_{j=1}^{d} |E_j^*|$.

The optimum solution for set k-cover corresponds to a valid solution for the Max k cut problem in G'. We derive a lower bound on the number of edges that cross the cut for this specific partition. Each hyper edge in E_j^* contributes at least $\frac{(j-1)(2d-j)}{2}$ edges since it intersects j sets. The minimum is achieved when we have one node in each of $j-1$ sets and the remaining $d-(j-1)$ nodes in the last set. Thus we get

$$|E_{C^*}| \geq \sum_{j=1}^{d} \frac{(j-1)(2d-j)}{2}|E_j^*|.$$

Simplifying (since $d \geq j$), we obtain

$$|E_{C^*}| \geq \frac{d}{2}\sum_{j=1}^{d}(j-1)|E_j^*|.$$

Thus

$$|E_{C^*}| \geq \frac{d}{2}(B^* - |E|).$$

Let us briefly consider a diversion for the case $d = 3$. Lemma 1 shows that $B^* \leq |E| + \frac{2}{3}|E_{C^*}|$. Before we prove Lemma 2 in general, we prove a lower bound on the quality of the obtained solution. There are two main reasons for this. The first is that in fact we get a slightly better bound when $d = 3$ as follows (in contrast the bound obtained by [1] is 0.70). The second reason is that it provides some intuition for the case when d is arbitrary, but this proof is easier to understand.

Let E_j' be the set of hyper edges that have j different colors in the solution based on Max k-cut. If $j > k$, then $E_j' = \emptyset$.

Theorem 2. *For the case when $d = 3$ (uniform hypergraphs), we obtain an approximation factor of $\frac{1}{3} + \frac{1}{2}\alpha_k$.*

Proof. Essentially each hyper edge of size three is reduced to a triangle. Note that when all three nodes have different colors, the benefit function is 3, and we have 3 edges crossing the cut. When the nodes of this hyper edge belong to two sets, we also have two edges crossing the cut.

The benefit from the max k-cut approach is $B_{C'} = \sum_{i=1}^{3} |E_i'|(i-1) + |E|$. The number of edges across the cut C' (in G') is $|E_{C'}|$. Clearly $|E_{C'}| = 2|E_2'| + 3|E_3'| = 2(|E_2'| + \frac{3}{2}|E_3'|)$. Hence $\frac{|E_{C'}|}{2} = |E_2'| + \frac{3}{2}|E_3'|$. So we can conclude that $B_{C'} = |E| + |E_2'| + 2|E_3'| \geq |E| + \frac{|E_{C'}|}{2} \geq |E| + \frac{\alpha_k|E_{C^*}|}{2}$.

We also know that $|E_{C^*}| \leq 3|E|$, where $|E|$ is the number of hyperedges in G. Putting these equations together with the upper bound on B^* of $|E| + \frac{2}{3}|E_{C^*}|$ gives us a lower bound on $\frac{B_{C'}}{B^*}$. Using the bound from Lemma 1, we will show that the benefit

from the solution based on the Max k-cut approach is at least $(\frac{1}{3}+\frac{1}{2}\alpha_k)$ of the maximum benefit in the optimal solution.

$$\frac{B_{C'}}{B^*} \geq \frac{|E| + \frac{\alpha_k}{2}|E_{C^*}|}{|E| + \frac{2}{3}|E_{C^*}|} = \frac{1 + \frac{\alpha_k}{2}\frac{|E_{C^*}|}{|E|}}{1 + \frac{2}{3}\frac{|E_{C^*}|}{|E|}}.$$

We now utilize the observation that $\frac{1+ax}{1+bx}$, with $a < b$, is a monotonically decreasing function of x. Since $x \leq 3$, the result follows.

When $d = 3$ our bound depends on α_k and is at least 0.828 for large k. When $k = 3$, α_k is 0.8 and we get a bound slightly better than 0.733.

Recall that E'_j be the hyper edges that contribute j to the objective function in the max k-cut solution, namely the number of distinct colors used by the vertices in each hyper edge. Then:

Lemma 2.

$$\sum_{j=1}^{d}(j-1)|E'_j| \geq \frac{|E_{C'}|}{\Delta}, \quad where \ \ \Delta = \frac{d(d+2)}{4}.$$

Proof. Suppose a hyper edge with d nodes that contributes j to the objective function must contribute at most $E(d,j)$ edges to $E_{C'}$, the total number of edges across the cut. Thus: $|E_{C'}| \leq \sum_{j=1}^{d} E(d,j)|E'_j|$.

We next show that for each j, we have $E(d,j) \leq \Delta(j-1)$, and the lemma follows. This is obtained basically by spreading out all d nodes into j sets as evenly as possible, with each group having either $\lceil \frac{d}{j} \rceil$ nodes or $\lfloor \frac{d}{j} \rfloor$ nodes. Each of the d nodes has degree at most $\delta = (d - \lfloor \frac{d}{j} \rfloor)$. The total number of edges $E(d,j)$ thus is at most $\frac{1}{2}d\delta$. We need to prove that this is at most $\Delta(j-1)$. This is easy to verify by setting the value of Δ to $\frac{d(d+2)}{4}$ (this breaks into two cases, when $j = 2$ and when $j \geq 3$, both are easy to verify).

Proof (Proof Of Theorem 1). The contribution of E'_j to the objective function $B_{C'}$ can be written as $B_{C'} = \sum_{j=1}^{d} j|E'_j|$. Note that $|E| = \sum_{j=1}^{d} |E'_j|$. From this we get that:

$$B_{C'} = |E| + \sum_{j=1}^{d}(j-1)|E'_j| \ \geq \ |E| + \frac{|E_{C'}|}{\Delta} \ \geq \ |E| + \alpha_k\frac{|E_{C^*}|}{\Delta}$$

Using this and Lemma XXX and considering the fact that $|E| \geq \frac{|E_{C^*}|}{\binom{d}{2}}$ it can be shown that the benefit from the solution based on the Max k-cut approach is at least $\frac{1}{d} + \frac{2\alpha_k}{d+2}(1 - \frac{1}{d})$ fraction of the maximum benefit in the optimal solution.

3 Set (k, α)-cover Problem

In this section, we consider the Set (k, α)-cover problem. Recall that here each sensor is allowed to be active in α of the k sets. For the optimization goal of maximizing the

average coverage, we can obtain a $(1 - \frac{1}{e})$-approximation by replicating each sensor α times and using the algorithms for the Set k-cover problem [1]. In this section we consider the problem of minimizing the maximum number of time slots a sensor is active in (α), given a fixed k, with the requirement that every region be monitored in each time slot.

3.1 NP-Hardness

We prove the NP-completeness of the decision problem of finding a partitioning of the sensors so that each target is covered in each time slot, for a given k and α. Similar constructions can be used to prove other versions of this problem (e.g. maximizing the average coverage across time-slots) NP-hard.

We first prove that the special case of Set $(2, 1)$-cover problem is NP-hard. Here there are only two time slots, and each sensor can be active in exactly one time slot. The question is: is there a partitioning of the sensors so that each target is covered in each time slot?

We show that this problem is NP-hard by a reduction from 3-Satisfiability (3-SAT). Assume that we have a 3-SAT instance with n variables $x_1, \ldots x_n$ and m clauses $C_1, \ldots, C_m$.

For the reduction we create a set of sensors and a set of targets. We will show that there is a valid schedule for activating sensors in one of the two time slots such that all targets are covered if and only if a solution to the satisfiability instance exists.

We create two special sensors F and T. Assume that we have two time slots 0 and 1. We create a target t_0 that is only adjacent to both F and T. W.l.o.g. assume that t_0 is covered by activating F in time slot 0 and T in time slot 1. Clearly both T and F have to be active in different time slots since no other sensor can cover t_0. For each variable x_i we create two sensors s_i^T and s_i^F and create a new target t_i that is adjacent to these two sensors. Target t_i is covered by activating these two sensors in different time slots. For each clause C_j we create a new target t_{n+j}. This target is adjacent to sensor F and also sensors corresponding to the literals in this clause. Suppose the clause $C_j = (x_{j_1} \cup x_{j_2} \cup \overline{x}_{j_3})$, then we make the target adjacent to $s_{j_1}^T, s_{j_2}^T$ and $s_{j_3}^F$.

If s_i^T is active in time slot 1, then this corresponds to the variable x_i being TRUE and $\overline{x}_i$ being FALSE. If s_i^T is active in time slot 0, then this corresponds to the variable x_i being FALSE and $\overline{x}_i$ being TRUE. It is easy to see that each target t_{n+j} is covered if and only if one of its neighbors is active in time slot 1 (other than the F neighbor which is active in time slot 0).

Thus we have shown that the Set $(2, 1)$-cover problem is NP-complete since membership in NP is trivial. This can be used to inductively prove the NP-hardness of Set $(k, 1)$-cover problem as follows. Given an instance of the Set $(k - 1, 1)$-cover problem with a set of sensors $S = s_1, \ldots, s_n$ and a set of regions $R = r_1, \ldots, r_m$, add a new sensor s' to the problem instance that can monitor all the regions. Solving the Set $(k, 1)$-cover problem on this new problem instance is equivalent to solving the original Set $(k - 1, 1)$-cover instance (the new sensor s' will cover all regions in one time-slot, leaving us to partition the remaining sensors into $k - 1$ time-slots).

Next we show a reduction from Set $(k, k - 1)$-cover problem to the Set $(k + 1, k)$-cover problem, from which the NP-hardness of the Set $(k+1, k)$-cover problem follows

(by induction). Consider an instance of the Set $(k, k-1)$-cover problem as above. We introduce one new sensor s', and n new regions, $r'_1, \ldots, r'_n$, with r'_i monitored by sensors s_i and s'. Consider a feasible solution to the Set $(k+1, k)$-cover problem on this new problem instance. Say s' is active in sets 1 through k (w.l.o.g.), covering the regions $r'_i, i = 1, \ldots, n$ in those partitions. Sensors s_i must then be active in time slot $k+1$ in order to cover those regions. The residual problem is exactly the original Set $(k, k-1)$-cover problem.

A generalization of the above two constructions can be used to show the NP-hardness of any Set (k, α)-cover problem where k and α are relatively prime. An alternate construction (omitted for brevity) shows the NP-hardness of Set $(2k, 2)$-cover problem for any k, and we conjecture that the general case is also NP-hard.

3.2   A Bicriteria Approximation Algorithm and Analysis

In this section we consider the problem of minimizing the maximum number of time slots a sensor is active in (α), given a fixed k, while maximizing average coverage of all regions.

We start with the following Integer Program (IP) formulation (for fixed k).

We define a 0/1 variable x_{ij}, for $i = 1 \ldots k$, $j = 1 \ldots n$. When $x_{ij} = 1$ it means that sensor j is active in time-slot i.

$$\min \alpha$$

Constraints are as follows

$$\forall j \in S \ \sum_{i=1}^{k} x_{ij} \leq \alpha$$

$$\forall r \in R \ \forall i = 1 \ldots k \ \sum_{(r,p) \in E} x_{ip} \geq 1$$

$$x_{ij} \in \{0, 1\}$$

The first constraint simply states that each sensor may belong to at most α time-slots. The second constraint states that each region j is covered in each time slot.

Clearly there is no benefit to increasing a variable beyond 1. We can relax this IP to an LP by simply requiring that $x_{ij} \geq 0$, which can be solved efficiently.

We now use randomized rounding to obtain an integral solution in which with high probability each region is covered in all the time slots. This is a bicriteria approximation algorithm in a sense that the approximation factor increases when we look for solutions with higher probability for coverage. To do the randomized rounding, we first scale all the x_{ij} variable by a scale factor s and then we round up x_{ij} to 1 with probability equal to the scaled x_{ij}, call the new variable $x'_{ij} = \min(1, s \cdot x_{ij})$. In fact, the analysis of the algorithm does not provide a probability of covering all regions in all time slots since the events of covering two regions in two time slots may be negatively correlated. Each sensor will be active in $s\alpha$ time-slots (in expectation); and each region will be covered in each time-slot with probability at least $(1 - \frac{1}{e^s})$. Let x''_{ij} be the rounded value of the variable x_{ij}. For each sensor $j \in S$, the expected value of $E(\sum_{i=1}^{k} x''_{ij}) \leq$

$\sum_{i=1}^{k} sx_{ij} = s\sum_{i=1}^{k} x_{ij} \leq s\alpha$. So the expected cost of the new integral solution is at most $s \cdot OPT$. Now we show that each region will be covered in all the time slots with high probability. Probability that a given region r in a given time slot i is not covered is equal to the probability that none of the x_{ip} variables $\forall (r, p) \in E$ is rounded up to one. In other words, $\Pr[r$ is not covered at time $i] = \prod_{(r,p)\in E}(1 - sx_{ip}) \leq \frac{1}{e^s}$. This uses the fact that the left-hand side is maximized subject to $\sum_{(r,p)\in E} x_{ip} \geq 1$ when all x_{ip}'s are equal. So each region in each time slot will be covered with probability at least $1 - \frac{1}{e^s}$.

3.3 Non-constant Integrality Gap

One might wonder if there is a rounding that satisfies all the constraints with constant s, thus giving a constant approximation. However, it can be shown that there is a non-constant integrality gap for this problem. Consider a matrix M of size $\binom{k}{k/2} \times k$. The rows contain all the binary strings of length k that have exactly $\frac{k}{2}$ 1's. Consider each column as a sensor and each row as a region. So if $M_{ij} = 1$, sensor j will cover region i and otherwise not. Suppose we have k time slots. One (fractional) solution is to activate each sensor fractionally for $\frac{2}{k}$ in each time slot so that each location is covered since we have $\frac{k}{2}$ sensors covering each region. This gives us $\alpha = 2$ for the fractional solution. At each time step, at least $(\frac{k}{2} + 1)$ sensors have to be active (otherwise a region is uncovered). Thus the total number of active sensors (summing over all times) is $\Omega(k^2)$. Hence at least one sensor is active in $\Omega(k)$ time slots, and has $\alpha = \Omega(k)$. This shows an integrality gap of $\Omega(k)$, and we can make k as large as $\Omega(\frac{\log n}{\log \log n})$, while still having a polynomial number of regions.

Extensions for the min-coverage-region are straightforward.

4 Sensors with Capacity Constraints

In many applications, sensors have constraints as to how many locations they can monitor even when the sensor is active. For example, a camera sensor s_i may have the *capability* to monitor a set of regions $N(s_i)$, but in a time slot when s_i is active it can only monitor at most $c(s_i)$ regions. For a fixed camera sensor, in fact $c(s_i)$ may just be 1. For a moving sensor, it is possible that the sensor can cover multiple regions. In this case, we also have to come up with an assignment of each sensor to at most $c(s_i)$ regions in a time-slot when the sensor is active.

4.1 NP-Completeness

We examine several cases and either provide polynomial time algorithms, or approximation algorithms when the problem can be shown to be NP-complete.

We first show that even when the capacities are as low as 3, even the basic problem is NP-complete for $k = 2$ and each region being covered by exactly two sensors.

First recall that Max-Cut[1] is NP-complete for graphs with degree at most 3 [28]. The question is - is there a way to partition the nodes of a graph $G = (V, E)$ into two groups so that at least Δ edges cross the cut?

---

[1] This is the Max k-cut problem when $k = 2$.

Next we briefly describe the reduction. Let $S = \{s_i | v_i \in V\}$. Each edge $(v_i, v_j) \in E$ corresponds to a region in R that has neighbors s_i and s_j. Since each node has degree at most 3, in any case a sensor covers at most 3 regions so with a capacity of 3 each sensor can cover all regions adjacent to it. Set $k = 2$. There is a partition in which the total coverage is at least $|E| + \Delta$ if and only if there is a solution to Max Cut with at least Δ edges across the cut.

4.2   Unit Capacity

We consider the set (k, α)-cover problem with the objective to maximize average coverage. We show that this problem can be solved optimally in polynomial time. Our goal is to maximize the number of regions that can be covered. Recall that we need to define k subsets S_i such that each sensor belongs to at most α subsets.

We first construct the following bipartite multi-graph. Let $H = (S, R, E')$. We create a vertex in S for each sensor and a vertex for each region in R. We put α parallel edges between each sensor and region pair if that the sensor can cover the region. We now select a maximum *bounded degree subgraph*[2] of H with the maximum number of edges. The degree bound on sensor nodes in S is exactly α and the degree bound on region nodes in R is k. This problem can be solved in polynomial time for bipartite graphs using network flows.Once we find a maximum subgraph (it is not necessarily unique) H^*, we then find an edge coloring [3] of H^* using at most k colors where k is the maximum degree (since $\alpha \leq k$). An edge coloring of a graph is an assignment of colors to the edges such that no pair of edges that are incident on a common vertex have the same color. Each color class forms a matching in the bipartite graph and corresponds to a time slot. The sensor nodes will be members of the α color classes corresponding to the colors of the edges incident on the sensor nodes.

Theorem 3. *The running time of the algorithm is constrained by the time taken to compute the bounded degree subgraph with the maximum number of edges. This takes $O(n^3)$ time in the worst case where n is the number of vertices in the graph.*

4.3   General Capacity

In this case, we consider the set (k, α)-cover problem. Each sensor can be activated in at most α sets and the goal is to maximize the average coverage. Unlike the previous case, each sensor s_i can cover $c(s_i)$ regions in each time slot in which it is active. We develop a randomized $(1 - \frac{1}{e})$-approximation algorithm for this problem. As in the previous section we first construct a bipartite graph. Let $H_c = (S_c, R_c, E'_c)$. For each sensor s_i we create α vertices $s_i^1, \ldots, s_i^\alpha \in S_c$. We put an edge from each of the α copies of a sensor to a region node if that the sensor can cover that region. Next, we select a bounded degree subgraph of H_c with the maximum number of edges. The bound on the degree of each $s_i^t \in S_c$ is $c(s_i)$ and the bound on the degree of each $r \in R_c$ is k. As

---

[2] This problem can be viewed as a generalization of the problem of finding a matching in a graph. Each node v has a upper bound of $b(v)$ on the number of chosen edges in the subgraph. With this restriction we wish to compute a subgraph with the maximum number of edges.

in the previous case, we can find the subgraph with the maximum number of edges in polynomial time using network flow. Call the subgraph H_c^*. In this subgraph, each s_i^t node will belong to a different time slot and cover the subset of regions that correspond to the (at most $c(s_i)$) adjacent nodes. It is easy to prove that the number of edges in H_c^* is an upper bound on OPT. To actually find the schedule we use the following randomized algorithm:

For each sensor, randomly, choose α of the k available time slots (without replacement).

Theorem 4. *The expected value of the coverage given by the randomized algorithm is at least $1 - \frac{1}{e}$ of the number of edges in H_c^*.*

Proof. The argument used here is similar to the one given in [1] with some adaptations to work for the new method. We first compute the probability that a region r is not covered in a given time slot t. We use N_r to denote the set of neighbors of r in H_c^*. Also N_{rs} refers to the copies of sensor s that belongs to N_r.

We first show that the probability that r is not covered in a specified time slot t is $(1 - \frac{1}{k})^{N_r}$. For each sensor s, the probability that none of the copies of s belonging to N_{rs}, have been covered it, is at most $(1 - \frac{|N_{rs}|}{k})$. For $k \geq 1$, $(1 - \frac{|N_{rs}|}{k}) \leq (1 - \frac{1}{k})^{|N_{rs}|}$. The probability that a specified region r is not covered at a given time slot t is the probability that it is not covered by any of its neighbors. Since for sensors i, j, the two events that N_{ri} is not covering r and N_{rj} is not covering r are independent events, the probability that r is not covered in t is the product of all these probabilities for all N_{rs} sets. So the probability that r is not covered is bounded by $(1 - \frac{1}{k})^{\sum_{s \in S} |N_{rs}|} = (1 - \frac{1}{k})^{|N_r|}$. Since we know that N_r is the union of all N_{rs} sets for $s \in S$. In other words, $N_r = \cup_{s \in S} N_{rs}$.

The probability that r is covered in each time slot is at least $1 - (1 - \frac{1}{k})^{N_r}$. Let l_r be the number of sensors covering region r, in our solution. We can see that $E(l_r) \geq k - k(1 - \frac{1}{k})^{|N_r|}$. The optimal solution can be bounded by the number of edges in the H_c^* which is $\sum_r |N_r|$. As shown in [1], for each region r, $\frac{E(l_r)}{|N_r|} \geq (1 - \frac{1}{e})$ which completes the proof.

Since $H_c^* \geq OPT$, we have a $(1 - \frac{1}{e})$ approximation.

5 Experimental evaluation

In this section, we present the results of a comprehensive experimental study that compares the performance of the algorithms we proposed. The salient points of our study can be summarized as follows:

– The Max k-Cut algorithm for solving the Set k-Cover problem achieves schedules very close to optimal, and much better than the prior randomized algorithm.
– Allowing sensors to belong to multiple sets (Set (k, α)-cover) results in increased lifetime in several cases.
– Our Max-Flow based algorithms for the capacity-constrained sensor cover problem consistently perform significantly better than the theoretical worst case bounds.

We ran experiments on a wide variety of synthetic datasets, and report the results for a representative sample of them.

– **(DS1, DS3) - Random Uniform Datasets**: A set of n sensors and m locations were created (for varying values of n and m). For each location v, a degree d_v is chosen uniformly in $[d_{\min}, d_{\max}]$. Each of the d_v sensors covering v is then randomly selected.

  We show experimental results for two datasets of this type. **DS1** was generated with $n = 20$, $m = 50$, and $[d_{min}, d_{max}] = [8, 15]$. The value of k was set to 11 for the experiments with these graphs. **DS2** was generated with $n = 20, m = 50$ and $[d_{min}, d_{max}] = [3, 5]$, and the experiments were run with $k = 5$.

– **(DS2, DS4) - Geometric (2D and 3D) Datasets**: A set of n sensors and m locations are created as random points in the unit square (**DS2**) or unit cube (**DS4**) . A sensor covers all locations within distance r. Locations with degree less than $d_{\min}$ $(= 4)$ are deleted. Once again we use $n = 20$ sensors and $m = 50$ locations, and the experiments were run with $k = 5$. The value of r were set to 0.4 and 0.6 for **DS2** and **DS4** respectively.

– **(DS5) - Colored Dataset**: This dataset is designed to capture multi-modal, heterogenous sensor deployments. Given a connectivity graph and c colors, each location is given a color (*modality*) uniformly selected in range $[1, c]$. Each sensor can monitor a subset of colors. This color set is chosen uniformly from all non-empty subsets of $1, 2, \cdots, c$. The connectivity graph is then filtered so that sensors are only connected to locations with colors in their color set. Locations with degree less than $d_{\min}$ are deleted.

  We conducted experiments on colored version of all of **DS1, DS2, DS3**, and **DS4**. We report the results for the colored variant on the **DS2** (geometric 2D dataset) with c set to 3.

5.1 Set k-Cover

With our first set of experiments, we compare the performance of the new Set k-Cover algorithms that we developed in this paper. For each of the datasets we generated 20 random instances, and compared the performance of the Max-Cut algorithm and the randomized algorithm by Abrams et al. [1] (called Random henceforth), with the optimal solution found by solving the integer program using the CPLEX software. Note that solving the integer program is not feasible in most cases; for some of the problem instances we tested, the algorithm took days to finish, and in several cases, it did not terminate at all. Max-Cut runs in a few seconds, and Random is even faster since the algorithm runs in linear time.

For Random, as with all the randomized algorithms that we test, we ran the algorithm 100 times and took the best obtained solution over these runs.

In Figure 4, we plot the percentage of the regions *not covered* by the solution found using these algorithms. As we can see from Figure 4, the solution found by Max-Cut is consistently almost as good as the optimal solution in all but one case (for dataset DS1), for which we could not find the optimal solutions for all graphs.
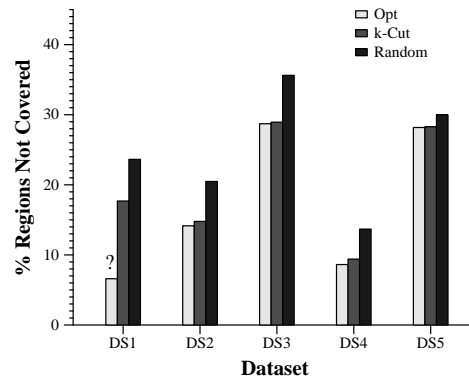
Fig. 4. Comparing Max-Cut, Random, and Opt for different datasets. The "?" for DS1 for Opt indicates that the optimal solution could not be found; we instead plot a weak lower bound.

5.2 Effect of α

We study the effect of allowing for larger values of α. To try and understand the benefits of allowing larger values of α, we ran the following experiment. We compute (using an IP formulation) the largest possible value of k while asking for full coverage (in other words, we would like each region covered in every time slot) after fixing $\alpha = 1$. In other words, we require a partitioning of the sensors into k sets such that each partition covers all the sensors.

We then compute the maximum value of k for each choice of $\alpha = 2 \ldots 5$ such that we can create k subsets $S_1, \ldots, S_k$ with each sensor belonging to at most α sets. The maximum value of $\frac{k}{\alpha}$ gives us the maximum lifetime that is possible. Surprisingly for geometric and colored graphs in many cases, there was no significant improvement. For random graphs, on the other hand, we found many cases with improvements upto 25%. These prelimiary results indicate that the approach of using larger values of α can result in significant improvements, and should be pursued further.

5.3 Sensors with Capacity Constraints

With our final set of experiments, we compare the performance of our randomized algorithm for the capacity-constrained version of this problem. Recall that our algorithm finds the optimal solution in the case when capacities are equal to 1; when capacities are greater than or equal to 3, the problem is NP-hard, and our algorithm is a $\left(1 - \frac{1}{e}\right)$ approximation to the optimal solution. We compare the performance of our algorithm against an optimistic bound on the solution – number of edges in H_c^* (see Section 4.3).

Figure 5 shows the results of our experiments as we increase the sensor capacities for the five datasets described above. As we can see, the solution achieved by our algorithm is within 80% of the upper bound, much better than the worst case approximation bound. Note that the upper bound that we compute is highly optimistic and is unlikely to be achieved by the optimal solution either.
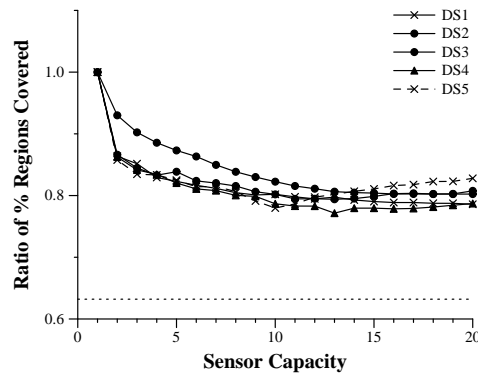
Fig. 5. Comparing the solution found by our algorithm for the capacity-constrained case with an optimistic upper bound (note that the y-axis starts at 0.6). Dashed line denotes the approximation ratio guaranteed by our algorithm.

6 Conclusions

Most of the prior work on energy minimization for sensor networks is based on algorithms based on random partitioning of sensors into sets. For the *first time*, we have shown a better scheme which actually takes the structure of the graph into account by converting the formulation to a hyper-graph formulation, then reducing that to a graph formulation and applying a Max k-Cut algorithm on that. And as the experiments suggest this approach gives almost optimal solutions in practice. We also propose a direct SDP based approach, that currently is slow (albeit polynomial) and we believe would be the approach to use for developing worst case approximation bounds.

General capacitated versions of this problem are introduced and studied for the first time as well. The natural approach for this problem would be to first partition the sensors into sets, and then figure out the assignment of which regions each sensor should monitor. The use of network flows as a pre-processing step is crucial in making this algorithm work. In addition, it is slightly surprising that the problem can be solved in polynomial time with unit capacities but with capacity three it is already NP-complete.

Finally we note that we have also developed a *direct* SDP-based algorithm (see [10]) for this problem (similar to the approach used to develop algorithms for max k-cut). This gives rise to an algorithm that runs in polynomial time for constant d. The solutions produced are almost as good as the solutions produced by the max k-cut approach but the algorithm is slower compared to the direct reduction to max k-cut. However we believe that this approach will eventually give a better worst case approximation bound for the case when the hyper-edges are large.

Some of the other interesting open problems include designing online, incremental algorithms to handle changes to the sensor network (nodes leaving or joining the network). Further, we currently assume that the time is slotted, and that a sensor cannot be partially active in a given slot. It may worth investigating if dropping this assumption would lead to more energy savings. In some sense, the Set (k, α)-cover version of the

problem that we introduce here is an attempt to approximate that. However we may be able to develop algorithms for that problem directly using other techniques.

Bibliography

[1] Z. Abrams, A. Goel, and S. Plotkin. Set k-cover algorithms for energy efficient monitoring in wireless sensor networks. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 424–432, 2004.

[2] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Computer Networks*, 38, 2002.

[3] N. Alon. A simple algorithm for edge-coloring bipartite multigraphs. *Inf. Process. Lett.*, 85(6):301–302, 2003.

[4] G. Aloupis, J. Cardinal, S. Collette, S. Langerman, and S. Smorodinsky. Coloring geometric range spaces. In *In Proceedings of the 8th Latin American Theoretical Informatics (LATIN'08)*, 2008.

[5] L. Benini, G. Castelli, A. Macii, E. Macii, M. Poncino, and R. Scarsi. A discrete-time battery model for high-level power estimation. In *DATE '00: Proceedings of the conference on Design, automation and test in Europe*, pages 35–41, New York, NY, USA, 2000.

[6] A. L. Buchsbaum, A. Efrat, S. Jain, S. Venkatasubramanian, and K. Yi. Restricted strip covering and the sensor cover problem. In *SODA '07*, 2007.

[7] M. Cardei, M.T. Thai, Yingshu Li, and Weili Wu. Energy-efficient target coverage in wireless sensor networks. In *IEEE Infocom*, 2005.

[8] M. Cardei and Jie Wu. Energy-efficient coverage problems in wireless ad-hoc sensor networks. *Computer Communications*, pages 413–420, 2006.

[9] A. Cerpa, J. Elson, D.Estrin, L. Girod, M. Hamilton, and J. Zhao. Habitat monitoring: Application driver for wireless communications technology. In *Proceedings of ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, 2001.

[10] A. Deshpande, S. Khuller, A. Malekian, and M. Toossi. Energy efficient monitoring in sensor networks. In *LATIN 2008: Theoretical Informatics, 8th Latin American Symposium, Búzios, Brazil, April 7-11, 2008, Proceedings*, pages 436–448, 2008.

[11] U. Feige. On maximizing welfare when utility functions are subadditive. In *STOC*, 2006.

[12] U. Feige, M. Halldorsson, G. Kortsarz, and A. Srinivasan. Approximating the domatic number. *SIAM J. on Comput.*, 32:172–195, 2002.

[13] A. M. Frieze and M. Jerrum. Improved approximation algorithms for max k-cut and max bisection. In *Proceedings of the 4th International IPCO Conference*, pages 1–13, 1995.

[14] M. X. Goemans and D. P. Williamson. .879-approximation algorithms for max cut and max 2sat. In *STOC '94: Proceedings of the twenty-sixth annual ACM symposium on Theory of computing*, pages 422–431, New York, NY, USA, 1994. ACM Press.

[15] J. Hastad. Some optimal inapproximability results. *JACM*, 48:798–859, 2001.

[16] J. Hill, R. Szewczyk, A. Woo, S. Hollar, and D. Cullerand K. Pister. System archi-tecture directions for networked sensors. In *Proceedings of ASPLOS*, November 2000.

[17] C. Hsin and M. Liu. Network coverage using low duty-cycled sensors: random & coordinated sleep algorithms. In *IPSN '04: Proceedings of the third international symposium on Information processing in sensor networks*, pages 433–442, New York, NY, USA, 2004. ACM Press.

[18] C. Intanagonwiwat, R. Govindan, and D. Estrin. Directed diffusion: A scalable and robust communication paradigm for sensor networks. In *Proceedings of ACM MOBICOM*, Boston, MA, August 2000.

[19] P. Juang, H. Oki, Y. Wang, M. Martonosi, L. Pehand, and D. Rubenstein. Energy-efficient computing for wildlife tracking: Design tradeoffs and early experiences with zebranet. In *Proceedings of ASPLOS*, October 2002.

[20] J. M. Kahn, R. H. Katz, and K. S. J. Pister. Mobile networking for smart dust. In *ACM MOBICOM*, Seattle, WA, August 1999.

[21] H. Liu, X. Jia, P. Wan amd C. Yi, S. Makki, and N. Pissinou. Maximizing lifetime of sensor surveillance systems. *IEEE/ACM Trans. on Networking*, 15:172–195, 2007.

[22] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel. Delay efficient sleep scheduling in wireless sensor networks. In *IEEE Infocom*, 2005.

[23] S. Madden, M.J. Franklin, J.M. Hellerstein, and W. Hong. TAG: A tiny aggre-gation service for ad-hoc sensor networks. In *5th Annual Symp. on Operating Systems Design and Implementation (OSDI)*, 2002.

[24] A. Mainwaring, J. Polastre, R. Szewczyk, and D. Culler. Wireless sensor networks for habitat monitoring. In *ACM Workshop on Sensor Networks and Applications*, 2002.

[25] S. Slijepcevic and M. Potkonjak. Power efficient organization of wireless sensor networks. In *IEEE International Conference on Communications (ICC'01)*, 2001.

[26] J. Vondrak. Optimal approximation for the submodular welfare problem in the value oracle model. In *STOC*, 2008.

[27] X. Wang, G. Xing, Y. Zhang, C. Lu, R. Pless, and C. Gill. Integrated coverage and connectivity configuration in wireless sensor networks. In *SenSys '03: Proceed-ings of the 1st international conference on Embedded networked sensor systems*, pages 28–39, 2003.

[28] M. Yannakakis. Node-and edge-deletion NP-complete problems. In *STOC '78: Proceedings of the tenth annual ACM symposium on Theory of computing*, pages 253–264, New York, NY, USA, 1978. ACM Press.

[29] Y. Yao and J. Gehrke. Query processing in sensor networks. In *Proceedings of Conference on Innovative Data Systems Research (CIDR)*, 2003.

[30] Z. Zhou, S. Das, and H. Gupta. Connected k-coverage problem in sensor networks. In *Intl Conference on Computer Communications and Networks (ICCCN)*, 2004.