

Sensor-Aided Overlay Deployment and Relocation for Vast-Scale Sensor Networks

Guanqun Yang, Bin Tong, Daji Qiao and Wensheng Zhang
Iowa State University, Ames, IA 50011
{gqyang, tongbin, daji, wzhang}@iastate.edu

Abstract—The overlay-based network architecture has been recognized as an effective way to deal with the *funneling effect* in sensor networks, where sensors closer to the sink are usually responsible for relaying more network traffic. Such funneling effect is particularly harmful when the number of sensors in the network is vast. In an overlay-based sensor network, a special type of resource-rich multi-radio mobile wireless devices (we call them *syphons*) are deployed along with sensors. Syphons form an overlay network and help nearby sensors relay their data to the sink via the overlay network, thus mitigating the funneling effect. In this paper, we study one of the fundamental challenges in overlay-based sensor networks: *syphon deployment problem*, i.e., how to deploy a limited number of syphons to cover a vast sensing field while maintaining the connectivity and balanced loads among them. We propose a novel Sensor-aided Overlay Deployment and Relocation (SODaR) protocol as a possible solution. The key idea is to take advantage of sensors' assistance and to relocate syphons by circling them around the sink in an orderly manner until all syphons are connected. Simulation results show that, with SODaR, syphons are able to self-form and self-maintain a connected tree structure which provides excellent load balancing among syphons with modest message and movement overhead.

I. INTRODUCTION

The ultimate goal of deploying a wireless sensor network is to monitor a target field and, upon occurrence of events of interest, to deliver the sensing data back to the sink at the desired fidelity level. However, data dissemination in sensor networks exhibits a unique *funneling effect* [1] where, as data are forwarded towards the sink, the traffic load intensifies along the forwarding paths. As a result, sensors closer to the sink consume their energy at higher rates and once they use up their energy, the network is partitioned. The funneling effect becomes more significant in a vast-scale sensor network where tens of thousands of sensors are deployed to monitor a huge target area. Moreover, due to the funneling effect, it becomes more difficult or even impossible for sensors (with a limited bandwidth) near the sink to provide a reliable and high-bit-rate data transportation (e.g., for delivery of the real-time surveillance video) to the sink. In such a network, the funneling effect could quickly render the network non-operational even when proper data aggregation and congestion control mechanisms have been employed to reduce the traffic load.

In recent years, mobility has been introduced into sensor networks to deal with the funneling effect [2]–[7]. In [8], the authors classified these works into three categories: *mobile-base-station based*, *mobile-data-collector based*, and *rendezvous based*. In mobile-base-station based approaches [2], [3], mobile base stations (i.e., mobile sinks) need to move to different locations periodically to balance the network traffic. The frequent movement of base stations may incur large amount of

energy consumption for maintaining the paths between sensors and the base stations; furthermore, base stations may not be allowed to move under certain circumstances (e.g., when they are wired to the Internet). In mobile-data-collector based approaches [4]–[6], a set of mobile data collectors traverse the network periodically to collect the data generated and buffered at sensors, while in rendezvous based approaches [7], sensors send their data to designated rendezvous nodes, which are visited by mobile data collectors periodically for data collection. Although these two types of solutions consume less energy than mobile-base-station based approaches, they do not suit well in vast-scale sensor networks because the incurred delay could be unacceptably large.

In [1], the authors proposed an interesting solution by deploying special static virtual sinks randomly into the sensor network. Each virtual sink is equipped with a long-range and high-rate 802.11 radio in addition to a conventional sensor radio. When the congestion occurs in the network, sensors can redirect their traffic to virtual sinks which in turn forward the data to the physical sink via the 802.11 communication links. Although the random deployment of these virtual sinks can reduce the funneling effect, the efficiency could be limited if they are not connected to the physical sink via the 802.11 links, in which case they may still need additional sensors to forward their data to the physical sink. How to deploy virtual sinks into the sensing field and how to maintain the connectivity among them were not addressed in [1].

Inspired by the above mobility-based and static virtual-sink-based approaches, and aiming to address their limitations, we investigate an *overlay-based* architecture for sensor networks to mitigate the funneling effect, which consists of the following three types of wireless devices:

- *Sink*: a static wireless station that collects and processes the sensing data; it is equipped with two wireless network interfaces that operate at non-interfering frequency bands or channels and have sharply contrasting communication characteristics: one is able to communicate over long range (denoted as R_c) at high transmission rates, e.g., an 802.11 interface, while the other communicates over shorter distance (denoted as r_c , and usually $r_c \ll R_c$) with lower transmission rates but is compatible with wireless interfaces equipped on small-factor sensors, e.g., an 802.15.4 interface;¹
- *Sensor*: a static wireless device that is able to sense the environment, perform light-weight computation, and communicate via its (only) 802.15.4 interface;
- *Syphon*: a special mobile wireless device that is equipped with an 802.11 interface and an 802.15.4 interface; it

The research reported in this paper was supported in part by the Information Infrastructure Institute (iCube) of Iowa State University, and the NSF under Grants No. CNS 0520102 and No. CNS 0716744.

¹To simplify the presentation, we use 802.11 and 802.15.4 to refer to these two types of wireless interfaces in the rest of the paper, although many different types of wireless interfaces could be used in practice.

communicates with sensors via the 802.15.4 interface but with syphons and the sink via the 802.11 interface; syphons and the sink form an overlay 802.11 syphon network over the underlying 802.15.4 sensor network with the presence of such an overlay network, each syphon forwards its sensing data to the nearest syphon and data is then relayed to the sink via the overlay network; syphons are resource-rich nodes and may have replenishable energy supplies (e.g., solar energy cells).

There are many unique research challenges related to overlay-based sensor networks. We address the fundamental *syphon deployment* problem in this paper. Given that the number of syphons deployed is limited due to their high cost, we propose a Sensor-aided Overlay Deployment and Relocation (SODaR) protocol to form a connected syphon network, which does not require localization or time synchronization in the network. The key idea of SODaR is to take advantage of the assistance of sensor nodes and to relocate mobile syphons by circling them around the sink in an orderly manner until all syphons automatically form an aggregation tree. There are a few works related to mobile sensor deployment [9]–[11], which is quite different from the issues we address in this paper. In particular, [9], [10] aim to maximize the coverage of a sensor network by relocating mobile sensors to uncovered regions, while the authors of [11] proposed a scan-based protocol to deploy mobile sensors.

The rest of the paper is organized as follows. Section II overviews the proposed SODaR protocol and Section III describes the protocol details. Section IV presents the simulation results and the paper concludes in Section V.

II. SYSTEM MODEL AND SODAR OVERVIEW

We consider a vast sensing field. Sensors and syphons are deployed at the same time into the field. We assume that sensors are deployed highly densely in the field hence the underlying sensor network is always connected with a high average node degree, while syphons are deployed sparsely and uniformly randomly in the field. Therefore, upon initial syphon deployment, it is most likely that only a few syphons are connected to the sink. Moreover, we assume that each syphon is equipped with a smart antenna [12] so that it can detect the source direction of a signal and move towards it.

The goal of our proposed SODaR protocol is to guide the syphons to self-form a connected overlay network with minimum syphon movement while balancing the network load that is measured by the number of sensors served by each syphon. To achieve this goal, after forming an initial syphon tree upon deployment, SODaR relocates off-tree syphons by circling them around the sink (to keep syphon-to-sink distances relatively unchanged) in an orderly manner until they are all attached to the syphon tree. Fig. 1(a) and (b) illustrate, respectively, the unconnected syphons upon initial deployment and a connected syphon tree after SODaR has been executed. Moreover, SODaR does not require syphons to have localization capabilities (e.g., GPS) or time synchronization among them, which facilitates its practical application.

III. SODAR – SYPHON TREE FORMULATION

In [13], we present detailed theoretical analysis to determine the number of syphons required to guarantee the desired probability of forming a syphon tree with SODaR. We also

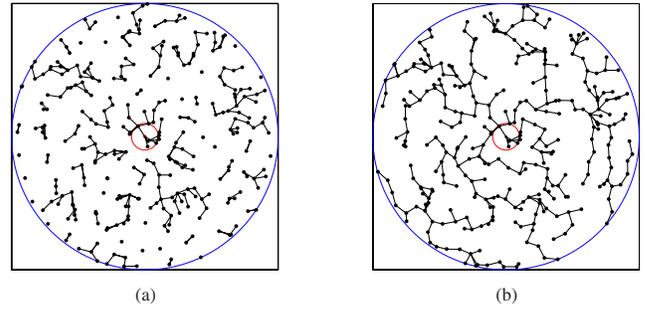


Fig. 1. Examples of (a) the unconnected syphons upon initial deployment and (b) a connected syphon tree after SODaR has been executed. The sink sits at the center of the field. A pair of syphons (black dots) are connected with a straight line if there is a communication link between them. Sensors are not shown in the figure.

give details in [13] about the syphon tree maintenance as well as practical issues that should be taken into account when deploying an overlay syphon network. After the required number of syphons have been distributed into the sensing field, each syphon executes SODaR to form a connected syphon tree, which consists of the following three phases.

A. Phase 1: Initialization

This phase has two objectives. Firstly, each sensor or syphon finds out its distance to the sink in terms of the number of 802.15.4 transmission hops, which is referred to as the *sensor hop count (SHC)*. Then, each set of sensors with the same *SHC* form a *ring belt*; hence multiple ring belts are formed encircling the sink. Secondly, all the syphons that can communicate with the sink via the 802.11 communication links form an *initial syphon tree* rooted at the sink. As detailed in Algorithm 1 and Algorithm 2, these objectives are accomplished through the flooding of *init-SHC* and *init-syphon-tree* messages.

Algorithm 1 Determining Sensor Hop Count (SHC) for Each Sensor/Syphon

For the sink (with ID 0)

1: broadcast *init-SHC* message $msg(ID = 0, SHC = 1)$.

For sensor or syphon (with ID i)

Initialization:

1: $Node_i.SHC \leftarrow \infty$

Upon receiving *init-SHC* message $msg(ID, SHC)$:

2: **if** $Node_i.SHC > msg.SHC$ **then**
 3: $Node_i.SHC \leftarrow msg.SHC$
 4: **if** this is the first time receiving *init-SHC* message **then**
 5: start a *timer* with interval τ /* τ is a system parameter*/
 6: **end if**
 7: **else if** the timer has been fired **then**
 8: broadcast $msg(ID = i, SHC = Node_i.SHC + 1)$
 9: **end if**

Upon timer is fired:

10: broadcast $msg(ID = i, SHC = Node_i.SHC + 1)$

Algorithm 2 Constructing the Initial Syphon Tree

For the sink (with ID 0)

1: broadcast *init-syphon-tree* message $msg(ID = 0)$

For syphon (with ID i)

Initialization:

1: $Node_i.PID \leftarrow nil$ /*initialize its parent ID to nil*/

Upon receiving the first *init-syphon-tree* message $msg(ID)$:

2: $Node_i.PID \leftarrow msg.ID$
 3: broadcast $msg(ID = i)$

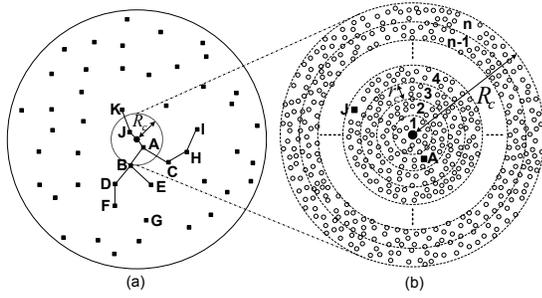


Fig. 2. Illustration of the sink (black dot at the center), syphons (black squares) and sensors (empty dots) in a vast-scale overlay sensor network.

Fig. 2(b) shows that Algorithm 1 organizes sensors into ring belts. *Ring belt i* is referred to as the belt containing all sensors with $SHC = i$. Fig. 2(a) shows that, after the execution of Algorithm 2, syphons A, \dots, F, H, \dots, K are on the initial syphon tree, while syphon G and others are not. We refer to the syphons on the syphon tree as the *on-tree syphons* and others as the *off-tree syphons*. To make full use of off-tree syphons in relaying data for sensors, it is desirable that they are relocated to join the syphon tree. The off-tree syphon relocation will be performed in the next phase.

B. Phase 2: Expansion of Syphon Tree

For an off-tree syphon to join the syphon tree, a straightforward solution is that, it first gets the location of an on-tree syphon, and then relocates to attach to that syphon. However, this approach does not work in our system because we do not require syphons to have the localization capability. Furthermore, it is nontrivial for an off-tree syphon to find out the locations of on-tree syphons in a vast sensing field.

By contrast, our proposed SODaR protocol does not require localization capabilities from syphons, and relocates off-tree syphons in the following three steps: *sensor-aided matching of on-tree and off-tree syphons*, *sensor-aided discovery of relocation paths*, and *sensor-aided relocation of off-tree syphons*. Next, we discuss the preparations that should be performed by all syphons and then elaborate the details of the three steps.

1) *System Preparation*: Each syphon uses its 802.15.4 interface (with the communication range of r_c) to initiate a message broadcast within a circle with radius of $\hat{R}_c = \alpha R_c$ where $\alpha \geq 1$ is a design parameter. As the message is propagated in a similar way as that in Algorithm 2, a *temporary syphon-sensor tree* rooted at the syphon is established, and each sensor records its parent on the tree and the root of the tree (i.e., the syphon). The propagation stops at the sensors which are $\lceil \hat{R}_c / r_c \rceil$ hops away from the root, and these sensors become leaf nodes of the temporary syphon-sensor tree. Note that the temporary syphon-sensor tree is only maintained during Phase 2.

2) *Step I: Sensor-Aided Matching of On-Tree and Off-Tree Syphons*: Matching on-tree and off-tree syphons is similar to the classic publish/subscribe [14] problem, for which there are three types of solutions: pull-based, push-based and hybrid methods. In SODaR, we choose a hybrid solution since the other two types of solutions either require time synchronization or incur high communication overhead. Moreover, since off-tree syphons are not connected with on-tree syphons via the 802.11 communication links, sensors are used to bridge the connections between them, which distinguishes our solution

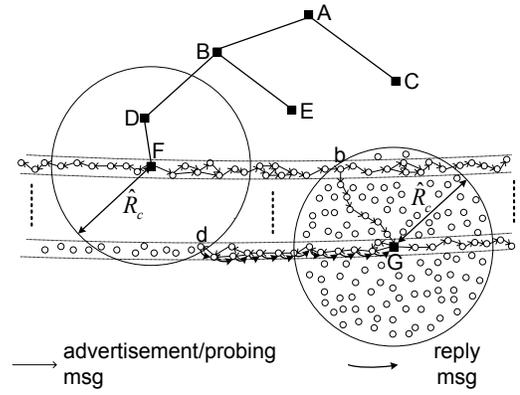


Fig. 3. Illustration of SODaR's syphon tree expansion scheme.

from those to typical publish/subscribe problems. Specifically, our solution includes the following two sub-steps:

- *Advertising On-Tree Syphons*: As shown in Fig. 3, each on-tree syphon (e.g., F) broadcasts an *advertisement* message using its 802.15.4 interface within the ring belt it belongs to. That is, only sensors within the ring belt forward the message. Note that this message travels within the ring belt in two opposite directions.
- *Capturing Advertisement Messages*: When an advertisement message arrives at a leaf node of a temporary syphon-sensor tree rooted at an off-tree syphon (e.g., b in Fig. 3), in addition to being forwarded within the belt, the advertisement message is also forwarded to the syphon along the temporary syphon-sensor tree. Upon receiving the captured advertisement, the off-tree syphon is aware of the existence of an on-tree syphon as well as an estimated number of sensor hops between them (i.e., the SHC carried by the advertisement message).

3) *Step II: Sensor-Aided Discovery of Relocation Paths*: After receiving a captured advertisement, the off-tree syphon initiates the process of relocation path discovery, which includes the following sub-steps:

- *Path Probing*: The off-tree syphon broadcasts a *probing* message using its 802.15.4 interface, and the message is flooded within the ring belt it belongs to. Different from the advertisement message, the flooding of probing messages is more bounded. Since the off-tree syphon recorded SHC in the captured advertisement, it can set the TTL of its probing message to $2 \times SHC$ to limit the flooding scope. TTL is decreased by one at each forwarding sensor node, and when the TTL reaches 0, the probing message is dropped.
- *Path Establishment*: When the probing message reaches a leaf node of a temporary syphon-sensor tree rooted at an on-tree syphon (e.g., d in Fig. 3), the sensor will send back a *path reply* message, which is unicast along the path reverse to the probing message's forwarding path. The path reply message also carries an SHC whose initial value is zero. Upon receiving the message, each sensor along the path records the SHC value, increments SHC by one, and then forwards it to the next sensor along the path. These on-path sensors will later serve as *landmarks* when the off-tree syphon relocates to join the syphon tree.

4) *Step III: Sensor-Aided Relocation of Off-Tree Syphons*: After receiving a path reply message, the off-tree syphon starts

the actual relocation procedure as follows. It periodically broadcasts a beacon using its 802.15.4 interface. On-path sensors (i.e., landmarks) that hear the beacon reply with their *SHC* values. The smart antenna equipped on the syphon decides the direction of the sensors that have replied, and then the syphon always chooses the sensor with the smallest *SHC* and moves towards it. The off-tree syphon also periodically broadcasts a beacon using its 802.11 interface. When an on-tree syphon hears the beacon, it replies with another beacon. The beacon exchange confirms that the off-tree syphon has joined the syphon tree successfully. The above relocation procedure stops as soon as the off-tree syphon hears a beacon reply from an on-tree syphon, or when it arrives at the location of the sensor with zero *SHC*.

5) *Termination of Syphon Tree Expansion*: The above three steps form a loop that can be performed repetitively. That is, after an off-tree syphon attaches to an on-tree syphon, itself becomes a new on-tree syphon and immediately starts advertising itself in order to match and relocate other off-tree syphons. The loop terminates when there are no more off-tree syphons that can be added to the syphon tree using this protocol. The syphon tree expansion is performed in a distributed manner. So even if an on-tree syphon cannot discover any new off-tree syphon near the ring belt it belongs to, it is not sure whether the syphon tree expansion has terminated, and hence cannot proceed to the next phase. To address this problem, we propose the following approach for each on-tree syphon to determine whether the expansion has terminated:

- (i) As an on-tree syphon advertises itself, it starts a timer *TimerC*. The advertisement message traverses at most half of the perimeter (in terms of sensor hop count) of the ring belt it belongs to. When an off-tree syphon receives this advertisement, it sends out a probing message. When the probing message reaches a sensor within \hat{R}_c from an on-tree syphon, the sensor forwards the probing message to this syphon. For example, in Fig. 3, sensor *d* forwards the probing message to syphon *F*. The probing message traverses at most twice the number of the hop counts that the advertisement message does. So by setting a proper *TimerC*, the on-tree syphon knows when it should expect a probing message if there exists an off-tree syphon. If it does not receive any probing message before *TimerC* expires, it goes to (iii); otherwise it goes to (ii).
- (ii) When an on-tree syphon receives a probing message, it gets the *SHC* value in the message. Based on the *SHC* value and the (known) maximum velocity of syphons, the on-tree syphon estimates when the off-tree syphon will arrive and join the tree, and starts a timer *TimerM* with a proper value. If the on-tree syphon does not hear from the expected off-tree syphon over its 802.11 interface before *TimerM* expires, it assumes that the off-tree syphon is pinned for some reason and then stops waiting for it.
- (iii) If the on-tree syphon is a leaf node of the current syphon tree, it sends a report to its parent on the syphon tree. Otherwise, after it has received reports from all of its children, it sends a report to its parent syphon or starts broadcasting an *expansion terminated* message to all on-tree syphons if it has no parent (i.e., it is the sink).

C. Phase 3: Setting up Data Forwarding Path to Syphon Tree

Once the syphon tree expansion terminates, each sensor needs to find out the nearest syphon and to set up a path towards

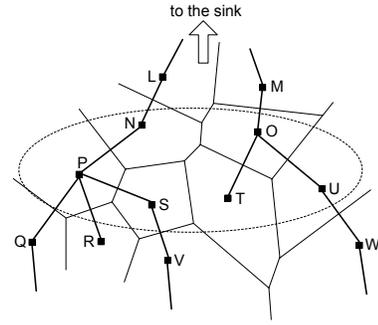


Fig. 4. Syphons (black squares) and Voronoi Diagram. Only Voronoi diagrams for syphons within the dotted eclipse are shown.

it so that its sensing data can be forwarded to the syphon and then relayed to the sink via the syphon overlay network. To achieve this goal, SODaR adopts a scheme based on the Voronoi diagram [15], with which the entire sensor network is partitioned into Voronoi cells and each on-tree syphon is located at the center of a Voronoi cell, as shown in Fig. 4. As a result, each sensor belongs to a Voronoi cell and forwards its data to the syphon located at the center of the cell. The scheme is detailed in Algorithm 3. Note that Algorithm 3 does not require any location information.

Algorithm 3 Setting Up Forwarding Paths Between Sensors and On-Tree Syphons

For every syphon (with ID *i*)
1: broadcast *Voronoi-setup-message* *msg*(*SyID* = *i*, *SHC* = 1, *SeID* = *SyID*)
/* *SyID*: syphon ID,
SHC: number of sensor hops away from this syphon,
SeID: ID of sensor forwarding this message */

For every sensor (with ID *j*)
Initialization:
1: $Node_j.SHC \leftarrow \infty$
/* number of sensor hops to the nearest on-tree syphon */
2: $Node_j.list \leftarrow nil$
/* used for recording syphon info */

Upon receiving *Voronoi-setup-message* *msg*(*SyID*, *SHC*, *SeID*)
3: **if** $Node_j.SHC > msg.SHC$ **then**
4: $Node_j.SHC \leftarrow msg.SHC$
/* record sensor hop count to the nearest on-tree syphon */
5: $Node_j.SyID \leftarrow msg.SyID$
/* record the ID of the nearest on-tree syphon */
6: $Node_j.SeID \leftarrow msg.SeID$
/* record the ID of the next-hop sensor towards the nearest on-tree syphon */
7: forward *msg*(*Node_j.SyID*, *Node_j.SHC* + 1, *j*)
8: **end if**
9: **if no** (*msg.SyID*, *, *) in *Node_j.list* **then**
10: put (*msg.SyID*, *msg.SHC*, *msg.SeID*) to *Node_j.list*
11: **end if**
12: **if there is** (*msg.SyID*, *x*, *) (*x* > *msg.SHC*) in *Node_j.list* **then**
13: replace (*msg.SyID*, *x*, *) with
(*msg.SyID*, *msg.SHC*, *msg.SeID*)
14: **end if**

IV. PERFORMANCE EVALUATION

We develop a custom simulator based on MATLAB and use it to evaluate the performance of the proposed SODaR protocol. In the simulation, we randomly deploy 80000 sensors and 300 syphons in a disk-shape field with a radius of 5000 meters. The sink sits at the center of the field. The 802.15.4 transmission range is $r_c = 50$ meters and the 802.11 transmission range is $R_c = 500$ meters. We choose α to 1.1 and thus $\hat{R}_c = \alpha R_c = 550$ meters. Through simulation, we investigate the communication overhead, the syphon movement overhead and the balance of the syphon distribution.

In SODaR, the communication overhead is incurred in both overlay syphon network and sensor network. Since the syphon network has much higher network bandwidth than the sensor network and syphons have more power supplies than sensors, our simulation focuses on measuring the communication overhead in terms of *the number of control messages generated or forwarded by each sensor* during the deployment and relocation of syphons. Fig. 5(a) plots the simulation results. As one can see, more than 80% of the sensors forward less than 8 messages, and only about 3% of the sensors forward 12 or more messages. Considering that SODaR is only executed once at the beginning of the network setup, such communication overhead is negligible. We also measure the movement distance of each syphon, and use it as the metric to evaluate the movement overhead of SODaR. Fig. 5(b) shows the cdf of the movement distance for each syphon. It can be seen that approximately 20% of the syphons do not need to move at all, and most of the syphons move less than 3000 meters. Since most syphons only need to move once during the network lifetime, and the movement brings significant benefits in forwarding the sensing data to the sink more efficiently, such movement overhead is a reasonable cost.

cdf of the number of sensors in each syphon-centered Voronoi cell (i.e., the number of sensors served by each syphon). The results are shown in Fig. 5(d), which clearly demonstrate that SODaR can significantly improve the balance of the syphon distribution. Note that about 7% of “cells” do not contain any sensors. These “cells” are not actual Voronoi cells; instead, they represent the fraction of syphons that are not connected to the sink via the 802.11 links even after SODaR has been executed.

V. CONCLUSION

In this paper, we propose a novel Sensor-aided Overlay Deployment and Relocation (SODaR) protocol to deploy and relocate a limited number of syphons to cover a vast sensing field, in order to alleviate the undesired funneling effect exhibited in sensor networks. Syphons are a special type of wireless devices that are mobile, resource-rich, and equipped with multiple wireless network interfaces and a smart antenna. Upon initial syphon deployment, SODaR directs syphons to circle around the sink in an orderly manner until an overlay syphon network is formed. Subsequently, each syphon serves as the virtual sink for its nearby sensors and forwards data to the physical sink via the overlay network. Simulation results demonstrate the effectiveness of the proposed SODaR protocol in terms of message/movement overhead, load balancing, and self-forming and self-healing capabilities.

REFERENCES

- [1] C.Y. Wan, S.B. Eisenman, A.T. Campbell, and J. Crowcroft, “Siphon: overload traffic management using multi-radio virtual sinks in sensor networks,” in *Proc. of SenSys*, 2005.
- [2] W. Wang, V. Srinivasan, and K.C. Chua, “Using mobile relays to prolong the lifetime of wireless sensor networks,” in *Proc. of ACM MobiCom*, 2005.
- [3] J. Luo and J. P. Hubaux, “Joint mobility and routing for lifetime elongation in wireless sensor networks,” in *Proc. of IEEE INFOCOM*, 2005.
- [4] R. C. Shah, S. Roy, S. Jain, and W. Brunette, “Data MULEs: modeling and analysis of a three-tier architecture for sparse sensor networks,” *Elsevier Ad Hoc Networks*, vol. 1, no. 2-3, 2003.
- [5] A. A. Somasundara, A. Ramamoorthy, and M. B. Srivastava, “Mobile element scheduling for efficient data collection in wireless sensor networks with dynamic deadlines,” in *Proc. of RTSS*, 2004.
- [6] Y. Gu, D. Bozdag, E. Eylem, O. Fusun, and C.G. Lee, “Partitioning based mobile element scheduling in wireless sensor networks,” in *Proc. of IEEE SECON*, 2005.
- [7] D. Jea, A. A. Somasundara, and M. B. Srivastava, “Multiple controlled mobile elements (data mules) for data collection in sensor networks,” in *Proc. of DCOSS*, 2005.
- [8] E. Eylem, Y. Gu, and D. Bozdag, “Mobility-based communication in wireless sensor networks,” *IEEE Communications Magazine*, vol. 44, no. 7, July 2006.
- [9] G. Wang, G. Cao, and T. L. Porta, “A bidding protocol for sensor deployment,” in *Proc. of IEEE ICNP*, 2003.
- [10] G. Wang, G. Cao, and T. L. Porta, “Movement-assisted sensor deployment,” in *Proc. of IEEE INFOCOM*, 2004.
- [11] J. Wu and S. Yang, “Smart: A scan-based movement assisted sensor deployment method in wireless sensor networks,” in *Proc. of IEEE INFOCOM*, 2005.
- [12] T.K. Sarkar, M.C. Wicks, M. Salazar-Palma, and R.J. Bonneau, *Smart Antennas*, Wiley-IEEE Press, 2003.
- [13] “Tech. rep.,” <http://www.public.iastate.edu/~gqyang/sodar.pdf>.
- [14] W. Chen, J.F. Kurose, D.F. Towsley, and Z. Ge, “Optimizing event distribution in publish/subscribe systems in the presence of policy-constraints and composite events,” in *Proc. of IEEE ICNP*, 2005.
- [15] F. Aurenhammer, “Voronoi diagrams – a survey of a fundamental geometric data structure,” *ACM Computer Surveys*, vol. 23, no. 3, 1991.
- [16] R. Jain, *The art of computer systems performance analysis*, John Wiley and Sons, 1991.

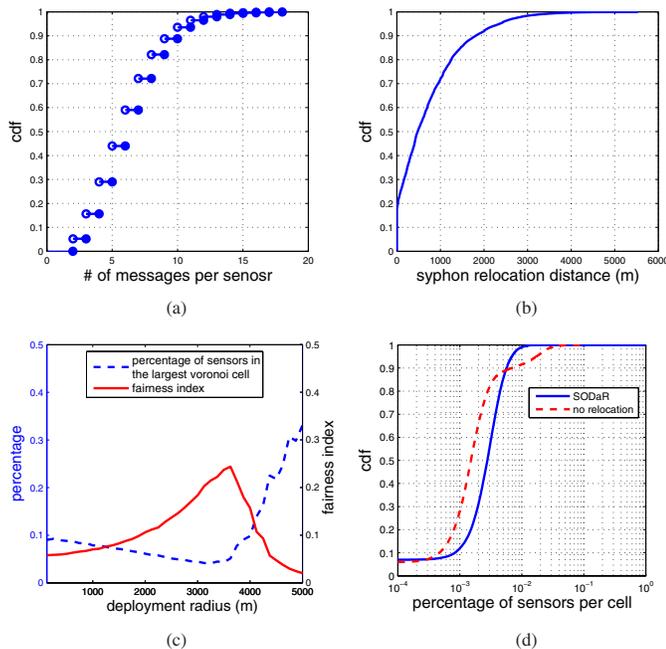


Fig. 5. (a) Communication overhead. (b) Movement overhead. (c) Percentage of sensors in the largest Voronoi cell and Jain’s fairness index. (d) Balance of the syphon distribution with or without relocation.

Next, we study the impact of the syphon relocation on the balance of the syphon distribution. Firstly, we measure the balance of the syphon distribution without relocation. We vary the radius of the deployment field for syphons from 125 to 5000 meters, and compute two metrics: the percentage of sensors in the largest Voronoi cell and Jain’s fairness index [16]. As shown in Fig. 5(c), when the radius is about 3500 meters, Jain’s fairness index attains its maximum value, and at the same time, the percentage of sensors in the largest Voronoi cell attains its minimum value. Secondly, we run SODaR to relocate syphons deployed in the field with radius of 5000 meters, and compare to a network without syphon relocation (deployment radius is fixed to 3500 meters) in terms of the