A Virtual Infrastructure for Mitigating Typical Challenges in Sensor Networks

Hady S. Abdel Salam
Old Dominion University

A VIRTUAL INFRASTRUCTURE FOR MITIGATING TYPICAL CHALLENGES IN SENSOR NETWORKS

by

Hady S. Abdel Salam
B.S. June 1999, Alexandria University, Egypt
M.S. June 2004, Alexandria University, Egypt

A Dissertation Submitted to the Faculty of
Old Dominion University in Partial Fulfillment of the
Requirement for the Degree of

DOCTOR OF PHILOSOPHY

COMPUTER SCIENCE

OLD DOMINION UNIVERSITY
November 2010

Approved by:

[illegible]
Stephan Olariu (Director)

[illegible]
Kurt Maly

Hussien Abdel-Wahab

[illegible]
Larry Wilson

Ivan Stojmenovic

ABSTRACT

A VIRTUAL INFRASTRUCTURE FOR MITIGATING TYPICAL CHALLENGES IN SENSOR NETWORKS

Hady S. Abdel Salam
Old Dominion University, 2010
Director: Dr. Stephan Olariu

Sensor networks have their own distinguishing characteristics that set them apart from other types of networks. Typically, the sensors are deployed in large numbers and in random fashion and the resulting sensor network is expected to self-organize in support of the mission for which it was deployed. Because of the random deployment of sensors that are often scattered from an overflying aircraft, the resulting network is not easy to manage since the sensors do not know their location, do not know how to aggregate their sensory data and where and how to route the aggregated data. The limited energy budget available to sensors makes things much worse. To save their energy, sensors have to sleep and wake up asynchronously. However, while promoting energy awareness, these actions continually change the underlying network topology and make the basic network protocols more complex.

Several techniques have been proposed in different areas of sensor networks. Most of these techniques attempt to solve one problem in isolation from the others, hence protocol designers have to face the same common challenges again and again. This, in turn, has a direct impact on the complexity of the proposed protocols and on energy consumption. Instead of using this approach we propose to construct a lightweight backbone that can help mitigate many of the typical challenges in sensor networks and allow the development of simpler network protocols.

Our backbone construction protocol starts by tiling the area around each sink using identical regular hexagons. After that, the closest sensor to the center of each of these hexagons is determined – we refer to these sensors as *backbone sensors*. We define a ternary coordinate system to refer to hexagons. The resulting system provides a complete set of communication paths that can be used by any geographic routing technique to simplify data communication across the network.

We show how the constructed backbone can help mitigate many of the typical challenges inherent to sensor networks. In addition to sensor localization, the network

backbone provides an implicit clustering mechanism in which each hexagon represents a cluster and the backbone sensor around its center represents the cluster head. As cluster heads, backbone sensors can be used to coordinate task assignment, workforce selection, and data aggregation for different sensing tasks. They also can be used to locally synchronize and adjust the duty cycle of non-backbone sensors in their neighborhood.

Finally, we propose "Backbone Switching", a technique that creates alternative backbones and periodically switches between them in order to balance energy consumption among sensors by distributing the additional load of being part of the backbone over larger number of sensors.

I dedicate this thesis to the soul of my mother,

May ALLAH shower all his mercy and blessings upon her and grant her a place in Jannat Alfirdaus . . . Amen

ACKNOWLEDGMENTS

This work could not be completed without the help of many individuals to whom I would like to express my appreciation. First and foremost, I would like to thank my advisor, Prof. Stephan Olariu, who has put a great deal of his time and effort into the guidance of this work. Prof. Olariu used to treat me as his son and not as his student. He used to keep his office opened in front of me in order to help me solve any technical or personal problem I had during the program. He funded me through his research grants and never stopped encouraging me till I could complete my PhD.

Next, I would like to convey my sincere thanks to other members of my PhD committee, Prof. Kurt Maly, Prof. Hussein Abdel-Wahab, Prof. Larry Wilson, and Prof. Ivan Stojmenović. Their expertise, thorough reviewing, continuous support, and valuable suggestions have led to a greatly improved dissertation.

I am also grateful to my family for their encouragement and support. Finally, special thanks to my father and mother for their understanding and patience during the time I spend completing this work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Electro-mechanical sensors have been used for relatively long time in different control systems. In a typical such system, a small number of sensors are deployed in predetermined positions in order to provide readings about important system parameters. Through wired-based networking of sensors, sensory data are transmitted to a central processing unit which analyzes received data and take decisions that control the functionality of the system.

Advances in nano-technology and wireless communications have enabled the development of a new generation of sensor-based networks. In particular, technology allowed the massive production of low-cost low-power multi-functional sensors. Although, these sensors usually have limited sensing, computational and communication capabilities, they can be networked to provide services for a vast spectrum of applications. The past decade has witnessed a phenomenal proliferation of sensor network applications ranging from battlefield surveillance [1], to border monitoring [2], to fire detection and habitat monitoring [3], to home automation [4], to traffic control [5], to health-care [6], and to body sensor networks [7], among many others.

I.1 DESIGN CHALLENGES

Designing a reliable and energy aware sensor network has been always challenging due to many factors inherent to the modest sensor resources and to the nature of the sensor network itself. A considerable amount of research has been conducted, and is still ongoing, on the topic of developing protocols and solutions to overcome these challenges. We now highlight some of the main challenges specific to the design of efficient protocols for sensor networks:

- Ad hoc nature: sensor networks are ad hoc in nature with no underlying infrastructure. It is the responsibility of individual sensors to identify their connectivity to other sensors and to decide what routing mechanism should be used to forward information to their intended destination. Moreover, traditional routing schemes may not be useful here because of the dynamic topology and energy considerations;

This dissertation follows the style of *The Physical Review*

- Limited energy budget: the sensors are powered by a modest, non-renewable on-board energy source. Once its energy is depleted, a sensor becomes totally non-functional. Hence, sensor energy must be treated as a precious resource that has to be used wisely, if network longevity is to be promoted;

- Energy hole problem: sensory data collected from different parts of the network need to be routed towards one of the network sinks for processing/aggregation. The additional routing load imposed on network sensors around sinks, would result in the depletion of their energy much faster than other sensors. Once the energy of these sensors is totally depleted, the sinks are disconnected from the rest of the network by holes that contain only non-functional sensors. In time, sensory data can not be routed to the sinks and the network fails;

- Location unawareness: the sensors are usually deployed in regions that have no infrastructure at all. A typical way of deployment is to scatter the sensors from airplanes. This kind of deployment does not allow sensors to be aware of their positions. Moreover, assuming that the sensors can be equipped with a relatively expensive and energy-hungry GPS chips does not seem to be a feasible or, indeed, an acceptable assumption for these low-cost low-power devices;

- The sensors must work unattended: due to the massive deployment of sensors, it is entirely impractical to devote attention to individual sensors. Once deployed, the sensors must work unattended with no external intervention;

- Limited computing power: the sensors are designed to be low-cost, low-power devices. Thus, the computing capabilities of sensors are very limited in terms of the processing speed and available memory. This imposes additional restrictions on the type of protocols that can be run by sensors;

- Small transmission range: wireless communication are known to consume a large portion of sensor energy; indeed, running the radio interface causes the largest energy expenditure incurred by individual sensors. Supporting sensors with long-range transmission capability can excessively consume their energy which it turn reduces their lifetime dramatically. To promote the functional longevity of the network, the sensors should perform their tasks with the minimum possible sensor-to-sensor or sensor-to-sink communication;

- Dynamic topology: to save their energy, the sensors alternate between *sleep* and *awake* periods. Due to clock drift and lack of communication, the sleep and awake cycles (duty cycles) of different sensors are assumed to occur asynchronously. When a sensor wakes up, it might find many of its neighboring sensors still sleeping. This behavior continuously changes the connectivity of the network creating a dynamic topology that complicates many of the network tasks, including routing and coverage;

- Unlike classical networks, where the main target is to maximize channel throughput or link utilization, the main target in sensor networks is to extend the network lifetime without sacrificing coverage, connectivity, and reliability of the network.

I.2 MOTIVATION

Several techniques have been proposed to address each of the challenges mentioned earlier (i.e. localization, clustering, routing, data aggregation, etc). The main goal of these techniques was to make the network more tractable by solving one of the inherent network challenges. A major problem when using this approach, each of these techniques tries to solve one problem separately from other problems. Protocol designers have to face the same common challenges every time they solve any of these problems. This in turn has its direct impact on the complexity of the proposed protocols and energy consumption.

Instead of solving each of these problems individually facing the same common challenges with each problem, we propose to construct what we call a network skeleton that is constructed immediately after network deployment and provides some kind of an infrastructure that makes the network more tractable. The skeleton provides sensors with coarse localization information that enables them to associate their sensory data with the geographic location in which the data was measured. Moreover, it promotes a geographic routing scheme that simplifies data communication across the network through skeleton sensors. By hypothetically tiling the deployment area using identical hexagons, the construction algorithm clusters sensors based on their locations into hexagons(clusters). Skeleton sensors which are chosen to be the closest sensors to the centers of these hexagons represent cluster heads and can play a crucial rule in coordinating task assignment, workforce selection, and data aggregation.

I.3 SENSOR NETWORK MODEL

We assume a typical sensor network which includes a massive deployment of tiny sensors each perhaps no larger than a dime. These sensors are deployed uniformly at random across the deployment area. In addition to the tiny sensors, we assume the existence of more powerful devices, referred to as *sinks.* While the tiny sensors are static, the sinks may be be able to move in support of the mission at hand. Further, while the sensors have a modest non-renewable energy budget, the sinks are assumed to be energetically self-sufficient and/or to have rechargeable batteries. Finally, while the number of sensors is large, perhaps in the tens of thousands, the number of sinks is many orders of magnitude smaller.

For precise reference, we now list our assumptions about the capabilities of sensors and sinks.

I- Sensors:

- The sensors are tiny, inexpensive devices with very limited sensing, computational and communication capabilities;
- The sensors are powered by a non-renewable on-board energy source; when the energy budget is exhausted the sensor becomes in-operational. Hence, the sensors sleep and wake up alternatively to save their energy. Sleep and wake-up cycles for different sensors are assumed to occur asynchronously;
- Once deployed, the sensors must work *unattended.* Although, the sensors may have fabrication-time identities, they should be treated as if they were anonymous as it is either impractical or infeasible to devote attention to individual sensors;
- The sensors are assumed to be static (immobile) and, at least initially, unaware of their location;
- Each sensor has a maximum transmission range, denoted by t_x, assumed to be much smaller than the width or the length of the deployment area. [1] This implies that messages transmitted by a sensor can only reach recipients in its proximity;

[1]Of course, t_x depends on the particular type of sensors deployed. Under technology available at the time of this writing, t_x is about 30m for micro-sensors.

- The reception circuitry of sensors is able to determine the strength of received signals (RSS). This feature allows each sensor to estimate its distance to the transmitter. Unfortunately, this estimate is often inaccurate due to the irregularity of signal propagation resulting from surrounding noise, signal reflection and refraction, and multi-path fading. This limitation is inherent to all RSSI-based distance measurements as mentioned in [8]. However, in our protocol, we try to get around this inaccuracy by:

  (a) Making distance estimates based on the average value of the received signal strength of several transmissions,

  (b) Avoiding any RSSI-based distance measurements for long-range transmissions in order to reduce the probability of being affected by surrounding noise.

II- Sinks:

- In addition to the tiny sensors, the network contains a small number of sinks. These sinks are responsible for tasking the sensors and for collecting the aggregated results;

- The sinks are much more powerful than the tiny sensors. They have no energy constraints either by being connected to a steady energy supply or by being powered by rechargeable batteries;

- As they do not have energy constraints, the sinks are assumed to be awake all the time;

- The sinks may be static or mobile, however they are always aware of their positions (i.e they might be equipped with GPS devices or their positions can be entered manually);

- Each sink is equipped with two transceivers with transmission range T_x that exceeds the maximum transmission range t_x of a sensor. The first transceiver is omnidirectional while the second is unidirectional with a narrow beam width. The sink nodes are aware of their orientation, and can rotate their unidirectional transmitter to any angle $[0, 2\pi]$ to transmit in any direction.

I.4 ROADMAP

The remainder of this dissertation is organized as follows, in Chapter II, we briefly summarize related work pertaining to research to solve challenges in sensor networks. Then in Chapter III, we present the details of our backbone construction protocol. Starting from section III.5, more emphasis is given to show how the proposed backbone can help mitigate many of the typical challenges in sensor networks including sensor localization, data clustering, data aggregation, and geographic routing. Chapter IV is dedicated for backbone-based task assignment and workforce selection. We propose one centralized and another distributed protocols that can be implemented on top of the proposed backbone. The chapter is concluded by a family of data aggregation protocols that can be integrated with the proposed task management process. In Chapter V, we present a backbone guided sleep scheduling scheme that can be used to balance sensor energy consumption. Finally, in Chapter VI, we conclude our work and highlight on future research directions.

CHAPTER II

STATE OF THE ART

Although, advances in technology enable the massive production of inexpensive sensors that can be deployed in large geographical areas, it raises numerous challenges on the protocols needed to interact with these sensors efficiently. In this chapter, we provide an overview of the work that has been reported in the literature and that is related to the construction of backbones in sensor networks. We also highlight on the state of the art techniques which were proposed to solve the most important challenges in sensor networks including localization, scheduling and routing.

II.1 SENSOR BACKBONES AND INFRASTRUCTURES

Wadaa *et al.* [9] proposed a virtual infrastructure for a massively-deployed collection of anonymous sensor nodes. They defined a coordinate system that provides an interesting clustering scheme for anonymous sensors, and referred to the process in which sensors learn their coordinates as *sensor training.* Sensor training can be repeated in a scheduled or ad-hoc basis to provide robustness and dynamic reorganization. During the training process, the deployment area around each sink is divided using a number of equiangular sectors (wedges) and concentric circles (coronas) centered at that particular sink. Olariu and Stojmenović [10] have shown that the radii of the coronas can be determined to optimize the efficiency of sensors-to-sink transmission. The group of sensors which reside in the region determined by the intersection of a specific corona and a specific wedge maps into one cluster. Although sensor training can simplify many of network management tasks like routing and data fusion, it has an inherent scalability shortcoming. In particular, as we move from the sink node outward, the cluster sizes increase from one corona to the next. After certain point, sensors within the same cluster may not be within the communication range of each other, hence more complex data fusion and leader election protocols are needed to handle these clusters.

More recently, Bertossi *et al.* [11] have enhanced the training protocol presented in [9]. The new approach outperforms the original approach in terms of the overall time for training by lowering it from linear to a square-root function of the size of the coordinate system used for location awareness.

Srinivas *et al.* [12] presented a novel hierarchical wireless networking approach in which some of the nodes are more capable than others. In their model, the more capable nodes serve as mobile backbone nodes and provide a backbone over which end-to-end communication can take place. In their approach they try to control the mobility of the backbone nodes in order to maintain connectivity. They formulate the problem of minimizing the number of backbone nodes and refer to it as the Connected Disk Cover problem. They show that it can be decomposed into the geometric disk cover (GDC) problem and the steiner tree problem with minimum number of Steiner points (STP-MSP). They provide approximation solutions to both problems. Although, mobile backbones can solve many of network management problems, they assume the existence of nodes with more advanced capabilities which may not be available in many sensor network deployment scenarios.

Frey *et al.* [13] showed that any sensor network graph can be transformed into an aggregated form which is a virtual overlay graph (e.g. an infinite mesh of regular hexagons) whose nodes are the centers of all nonempty clusters (hexagons). Two nodes C_1 and C_2 of that overlay graph are connected by an edge if there exists at least one connected pair of network nodes with one node located in C_1 and the other located in C_2. In their work, Frey *et al.* show that network connectivity does not suffer from generalizing the concept of sensing coverage to arbitrary clusters. Hence, geographic routing can be used on cluster-base and not on node-base.

II.2 SENSOR LOCALIZATION

Most WSN applications require in a way or the other to associate sensor readings with the geographic location in which the readings were taken. Getting location information through recording positions manually or through an expensive GPS chip are not valid options for sensor networks. To address the localization problem and to estimate a good approximation of the position of each sensor node, many techniques have been developed, each of them has its merits and demerits. Up to the time of writing these lines, there is no specific algorithm on top of others. Hence, we briefly summarize the technical foundations of the most important localization techniques proposed in the literature.

In general, localization techniques can be classified into two main categories: **Range-based** and **Range-free**. Range-based techniques depend on range estimation between nodes that know their positions and nodes that do not. Ranges are usually estimated based on measurements of received signal strength (RSS), time of arrival (TOA), time difference of arrival (TDOA) and angle of arrival (AOA)). Although Range-based techniques are more accurate than Range-free techniques, they share a common drawback, they require additional hardware to be available with each node to be able to take the required measures. Range-free techniques have been proposed to overcome the stringent hardware requirements of range-based techniques. The main idea behind these techniques is to exploit radio connectivity information among neighboring nodes to infer rough estimates of their positions without taking any range measurements. This way range-free techniques eliminate the need to equip each node with any specialized hardware allowing the manufacturing cost of these nodes to be low.

In the Centroid method [14], a sensor node estimates its position as the centroid of the polygon whose vertices are positioned at the anchors it could receive messages from. Anchors are location aware nodes with a transmission range that is usually longer than the transmission range of regular nodes. If anchors were positioned uniformly, localization error can be reduced however this can not be guaranteed in ad hoc or non static deployments. An obvious problem of this technique it localizes all the nodes that receive messages from the same subset of anchors (which is typically large number of nodes) to the same position which increases average localization error of nodes. An enhanced however range based variation of this technique evaluates node position as the centroid of anchor positions weighted by the strength of signals received from these anchors. Another similar approach for complex shapes was proposed in [15].

The APIT method [16] divides the deployment area into triangular sections using anchor nodes. Each sensor applies an approximate PIT test to decide whether it is inside each possible triangle or not. After that it uses a grid scan algorithm to estimate the maximum likelihood area within which it resides. In fact, APIT could achieve good localization accuracy, however it has two major drawbacks. The grid scan algorithm is very time and resource consuming especially if the grid size is small. Also, APIT does not localize nodes outside the convex hull of the anchor nodes accurately.

The Ad-hoc Positioning System (APS) [17] was proposed to allow non-GPS enabled nodes to estimate their locations in a hop by hop fashion. Three different methods were investigated, and the DV-Hop algorithm was the best to perform in most cases. In the DV-hop algorithm, each anchor sends a message that contains its location information to all the sensors around it. Sensors that receive these messages forward them to their neighbors and so the messages are flooded through the whole network. Within each message, there is a field that indicates number of hops this message has been forwarded. Basically, this field is initialized to 1 at the anchor node, and incremented at each hop. This way, sensors can determine how far they are (in terms of number of hops) from different anchors. The average distance per hop is calculated and each sensor can estimate its distance to different anchors. After that, a sensor node can use the estimated distances between itself and three or more different anchors to localize itself. Although DV-HOP is known to be one of the best known range free protocols, it has several drawbacks: first, the flooding nature of the protocol consumes much of sensor energy, second, it is not always easy and sometimes infeasible to use trilateration to estimate node position using approximate distances to anchors. The Amorphous [18] localization protocol depends on a similar idea. The location coordinates of the anchor are flooded throughout the network with the number of hops to the source anchor tracked in each message. This enables each node to maintain a list of hop-count to each anchor along with the location of that anchor. Each node that does not know its location can use this list to estimate its location. Unfortunately, Amorphous still has the same issues of DV-HOP.

Cricket [19], an indoor location support system proposed by MIT, allows nodes to learn their physical location by using listeners that hear and analyze information from anchors. Anchors concurrently use radio and ultrasonic signals to send their location information. The listener inference modules on the node overcome multipath and interference and improve localization accuracy. AHLoS [20] is similar to Cricket and uses RF and ultrasound for indoor localization. TDOA techniques like these techniques rely on extensive hardware that might be expensive and energy consuming, making it less suitable for sensor networks. Another drawback of TDOA techniques that use ultrasound, they require dense deployment as ultrasound signals propagate for a limited distance only.

The lighthouse system [21] uses a rotating anchor that produces a parallel light beam of fixed width. A sensor node detects the light beam for a period of time that depends on the distance between the anchor and the sensor. If the rotation speed and the width of the beam are known to the sensors, then each sensor can measure the time it detects the light beam and estimates the distance and the angle to the anchor.

Acoustic-based ranging techniques like BeepBeep [22] was proposed to localize sensors based on the two-way time of flight of the beeps between two communicating devices. Other special purpose localization techniques have also been proposed. Underwater 2D and 3D localization was proposed in [23], [24]. APL was suggested in [25] to address localization in Road Networks where most Range based localization techniques fail due to the sparse nature of deployment. APL uses binary vehicle-detection timestamps to obtain distance estimates between any pair of sensors on roadways.

II.3 SLEEPING SCHEDULE

To save their energy, sensors spend its whole life switching between two modes, sleeping mode (power consumption is minimum) and wake up mode (power consumption is relatively high). Different deterministic and probabilistic schemes can be used to determine the schedule based on which sensors sleep and wake up. Next, we summarize the technical foundations of the most important approaches we found in the literature.

In [26], a new scheduling protocol was proposed to maximize network lifetime for rare event target surveillance systems. The protocol provides a schedule that guarantees that each point in the environment is sensed within some maximum interval of time, called the detection delay. However the protocol does not handle QoS requirements that might require each point to be monitored by k sensors ($k > 1$) for more reliable readings.

Another sleep scheduling protocol was proposed in [27], however it requires synchronization between nodes which is hard to achieve in sensor networks. Analytical analysis of the maximum achieved upper bound on network lifetime was presented in [28].

The authors of [29] proposed a balanced-energy scheduling scheme for clustered sensor networks. Their scheme aims to evenly distribute the energy load of the sensing and communication tasks among all the nodes in the cluster, thereby extending the network lifetime. However, their scheme does not provide any guarantees on QoS requirements expressed in terms of the number of sensors participating in each task.

Another algorithm was proposed in [30] for large-scale wireless sensor networks. The algorithm allows each sensor to probabilistically schedule its own activity based on its node degree to guarantee a minimum level of connectivity. Unfortunately, k-connectivity does not imply k-coverage, and so their algorithm does not provide any guarantees that k-coverage based QoS requirements are satisfied.

CHAPTER III

BACKBONE CONSTRUCTION PROTOCOL

The main goal of this chapter is to describe our approach for constructing a network backbone that can be used to simplify many of the problems inherent to sensor networks. In Section III.5 and in the following chapters, we show how the proposed backbone can provide solutions to many of the typical challenging problems in sensor networks including localization, clustering, data aggregation, routing, scheduling, workforce selection among others.

III.1 COMMUNICATION BACKBONES

The concept of "network backbone" can be broadly generalized in sensor networks to include any subnetwork of sensors. Referring to any of these backbones, it is straightforward to realize that the network can be easily clustered by associating each non-backbone sensor to its closest backbone sensor. Figure 1, shows the different clusters constructed when using randomly selected backbone sensors. The reader should note that the clusters are bounded by the Voronoi diagram whose vertices are the selected backbone sensors.

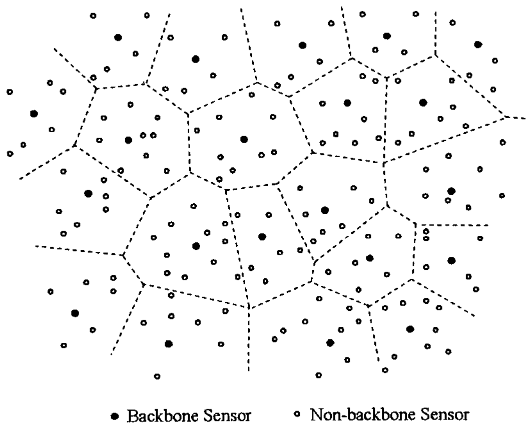


FIG. 1: *Clusters constructed from randomly selected backbone sensors*

Due to the generality of the definition, one would expect a typical sensor network to have a large number of backbones, although most of these backbones would have no practical value. Apparently, in order to be practically useful, a network backbone has to satisfy certain conditions. Our purpose is to highlight on the minimum conditions a backbone needs to satisfy in order to be practically useful for communication purposes. This would definitely give us more insight about the best strategy to follow in order to construct such backbones.

A network backbone would be practically useful for communication purposes if it satisfied the following conditions:

1. The backbone subnetwork is connected so it can be used to forward messages between different parts of the network. Moreover, to reduce the number of hops, the distance between any two backbone sensors should be as close as possible to sensor maximum transmission range, t_x.

2. Backbone sensors are well distributed across the deployment area. Any non-backbone sensor is within the transmission range of at least one backbone sensor. Furthermore, non-backbone sensors should be evenly distributed across different backbone clusters. For uniform sensor deployments, this implies that the areas of backbone clusters are equal.

3. The number of backbone neighbors for each backbone sensor is maximized in order to maximize the number of different communication paths between any two nodes.

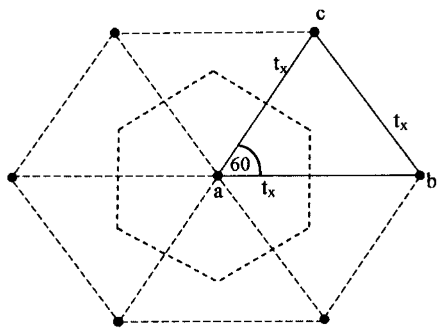


FIG. 2: *Hexagonal clusters maximize the number of backbone neighbors*

Assuming extremely dense network, where it is possible to find at least one sensor as close as possible to any point in the deployment area, we are interested to find the geometric shape of backbone clusters which will satisfy the conditions mentioned above. We start by placing our first backbone sensor a around the center of the deployment area (see Figure 2). From condition (1), since the distance between any two backbone sensors is t_x, the triangle $\triangle abc$ must be equilateral. Moreover, to satisfy condition (3), we need to maximize the number of backbone neighbors around sensor a which can be done by replicating the triangle $\triangle abc$ as shown in Figure 2 confirming that each backbone sensor can have up to six neighbor backbone sensors. Furthermore, Figure 2 shows that the backbone cluster around sensor a is in fact a regular hexagon with side length equals $\frac{t_x}{\sqrt{3}}$.

III.2 THE BACKBONE CONSTRUCTION PROTOCOL

Since, in our proposed protocol, the network backbone is constructed starting from the sink nodes outwards, we find it more appropriate to start by showing how the protocol works around a single sink node. It will then become clear that each sink performs the same backbone construction in a disk of radius R_D around itself.

Consider an arbitrary sink; we are interested in setting up a backbone in a disk D of radius R_D, $(R_D \leq T_x)$, around this particular sink. We note that, as a rule, the area covered by the disk is a small fraction of the deployment area. Nonetheless, to simplify the presentation, we shall refer to the sensor network built on the sensors in this disk as *the network.*

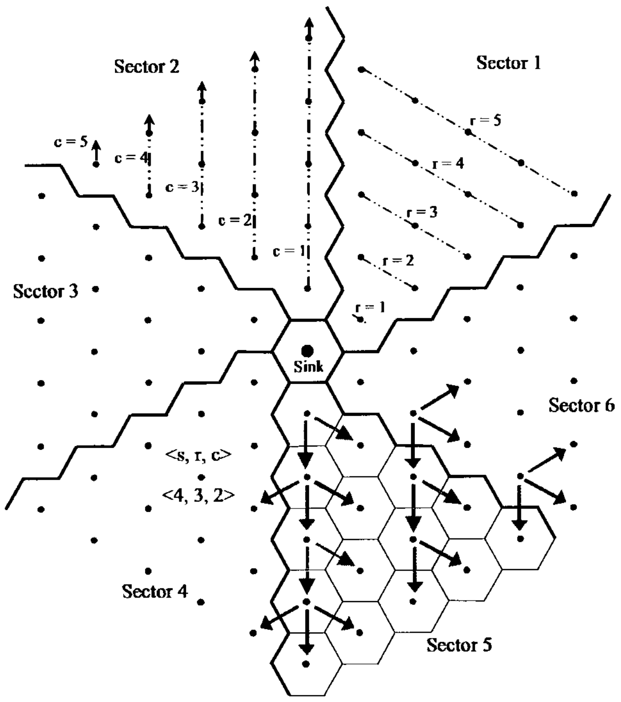


FIG. 3: *Partitioning the area around the sink into sectors*

We summarize the main steps of our construction protocol [31, 32] in the following phases:

- Tiling phase: in this phase the disk D around the sink is tiled using a set of identical regular hexagons which makes the area around the sink look like a beehive (see Figure 3);
- Backbone selection phase: in this phase the closest sensor to the center of each hexagon is determined. These sensors are referred to as backbone sensors;
- Non-backbone sensor classification phase: after being selected, backbone sensors announce themselves as well as the hexagon they represent to other nodes. Non-backbone sensors use the received signal strength to determine the hexagon to which they belong.

Tiling the disk D starts at the sink outwards. The first hexagon is positioned such that the center of the hexagon coincides with the sink. The side length of the tiling hexagons is taken to be $\frac{t_x}{\sqrt{3}}$, where t_x is sensor maximum transmission range.

In practice, we replace t_x by $\hat{t_x} = (1-\delta)t_x$, where $\delta \approx 0.1$. The reason behind this is to allow the selection of backbone sensors which are very close to the target hexagon center however they reside on the other side of the center and may not be reached if we used t_x.

Referring to Figure 3, the tiling continues by placing hexagons side by side in six different directions (i.e. $\frac{\pi}{6}$, $\frac{3\pi}{6}$, $\frac{5\pi}{6}$, $\frac{7\pi}{6}$, $\frac{9\pi}{6}$, and $\frac{11\pi}{6}$). We refer to these angles as "orientation angles". As shown in Figure 3, the area is divided into six sectors. In each sector, the hexagons are stacked in rows. In the first row there is only one hexagon (column), in the second row there are two hexagons, in the third row there are three hexagons, and so on.

We propose a ternary coordinate system to uniquely identify the various hexagons in the tiling above. Specifically, the hexagon in column c of row r in sector s is uniquely identified using the tuple $\langle s, r, c\rangle$. It is worthwhile to mention that although several addressing schemes for hexagonal networks [33, 34] have been proposed, our coordinate system seems to be more appropriate for our construction protocol.

The remainder of this section is divided into the following subsections: in Subsection III.2.1 we describe how the necessary angles and associated trigonometric functions are theoretically computed and practically measured; in Subsection III.2.2 we specify the order in which backbone sensors are selected; finally, in Subsection III.2.3 we present the technical details of the backbone selection process.

III.2.1 Computing and Measuring Angles

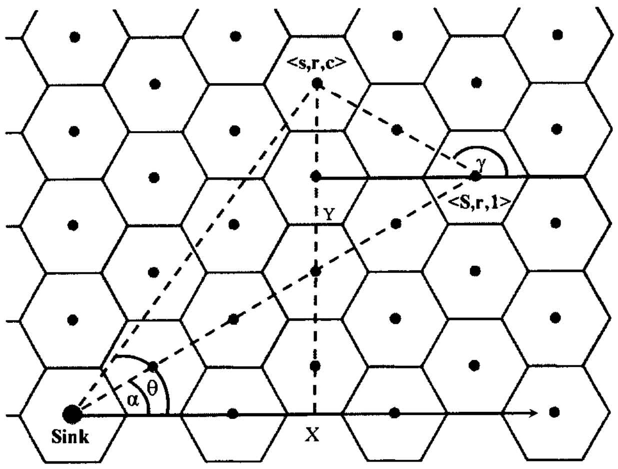


FIG. 4: *Estimation of the position of hexagon centers*

For a given sector s, the tangent of the angle $\theta_{\langle s,r,c\rangle}$ subtended by the positive x-axis and the line connecting the sink to the center $S_{\langle s,r,c\rangle}$ of the hexagon $\langle s,r,c\rangle$ is

$$\tan\theta_{\langle s,r,c\rangle} = \frac{r\sqrt{3}\tan\frac{\pi s}{3} + (2c - r - 2)}{r\sqrt{3} + (r - 2c + 2)\tan\frac{\pi s}{3}}. \tag{1}$$

To justify (1), we begin by evaluating the coordinates (x, y) of the center $S_{\langle s,r,c\rangle}$ of the hexagon $\langle s,r,c\rangle$, where the sink is assumed to be located at $(0,0)$. Referring to Figure 4, the distance between $S_{\langle s,r,1\rangle}$ and the sink is rt_x, and the distance between $S_{\langle s,r,1\rangle}$ and $S_{\langle s,r,c\rangle}$ is $(c-1)t_x$, where t_x is the distance between the centers of any two adjacent hexagons. Moreover, the angle α between the line connecting $S_{\langle s,r,1\rangle}$ to the sink and the positive x-axis is the orientation angle of sector s, and the angle γ can be evaluated geometrically to be $\alpha + \frac{2\pi}{3}$. With this preamble out of the way, we can evaluate x and y as follows

$$\begin{aligned} x &= rt_x\cos\alpha + (c-1)t_x\cos\gamma \\ &= rt_x\cos\frac{(2s-1)\pi}{6} + (c-1)t_x\cos\frac{(2s+3)\pi}{6} \end{aligned} \tag{2}$$

and

$$\begin{aligned} y &= rt_x\sin\alpha + (c-1)t_x\sin\gamma \\ &= rt_x\sin\frac{(2s-1)\pi}{6} + (c-1)t_x\sin\frac{(2s+3)\pi}{6}. \end{aligned} \tag{3}$$

However, recalling that

$$\begin{aligned} \sin\frac{(2s-1)\pi}{6} &= \sin\left(\frac{s\pi}{3} - \frac{\pi}{6}\right) \\ &= \sin\frac{\pi s}{3}\cos\frac{\pi}{6} - \cos\frac{\pi s}{3}\sin\frac{\pi}{6} \\ &= \frac{\sqrt{3}}{2}\sin\frac{\pi s}{3} - \frac{1}{2}\cos\frac{\pi s}{3}, \end{aligned}$$

$$\begin{aligned} \cos\frac{(2s-1)\pi}{6} &= \cos\left(\frac{s\pi}{3} - \frac{\pi}{6}\right) \\ &= \cos\frac{\pi s}{3}\cos\frac{\pi}{6} + \sin\frac{\pi s}{3}\sin\frac{\pi}{6} \\ &= \frac{\sqrt{3}}{2}\cos\frac{\pi s}{3} + \frac{1}{2}\sin\frac{\pi s}{3}, \end{aligned}$$

$$\begin{aligned} \sin\frac{(2s+3)\pi}{6} &= \sin\left(\frac{\pi s}{3} + \frac{\pi}{2}\right) \\ &= \sin\frac{\pi s}{3}\cos\frac{\pi}{2} + \cos\frac{\pi s}{3}\sin\frac{\pi}{2} = \cos\frac{\pi s}{3}, \text{ and} \end{aligned}$$

$$\begin{aligned}\cos\frac{(2s+3)\pi}{6} &= \cos\left(\frac{s\pi}{3}+\frac{\pi}{2}\right)\\ &= \cos\frac{\pi s}{3}\cos\frac{\pi}{2}-\sin\frac{\pi s}{3}\sin\frac{\pi}{2} = -\sin\frac{\pi s}{3},\end{aligned}$$

we can rewrite equations (2) and (3) as

$$x = \frac{t_x}{2}\left[r\sqrt{3}\cos\frac{\pi s}{3}+(r-2c+2)\sin\frac{\pi s}{3}\right] \qquad (4)$$

$$y = \frac{t_x}{2}\left[r\sqrt{3}\sin\frac{\pi s}{3}+(2c-2-r)\cos\frac{\pi s}{3}\right]. \qquad (5)$$

Using (4) and (5), combined, we can write

$$\begin{aligned}\tan\theta_{\langle s,r,c\rangle} &= \frac{r\sqrt{3}\sin\frac{\pi s}{3}+(2c-2-r)\cos\frac{\pi s}{3}}{r\sqrt{3}\cos\frac{\pi s}{3}+(r-2c+2)\sin\frac{\pi s}{3}}\\ &= \frac{r\sqrt{3}\tan\frac{\pi s}{3}+(2c-r-2)}{r\sqrt{3}+(r-2c+2)\tan\frac{\pi s}{3}},\end{aligned}$$

confirming that (1) holds.

We wish to point out that if for all s, $(1 \le s \le 6)$, the values $\sin(\frac{\pi s}{3})$, and $\cos(\frac{\pi s}{3})$ are tabulated, then each sensor can readily evaluate the coordinates (x, y) of the center of hexagon $< s, r, c >$ as well as $\tan\theta_{\langle s,r,c\rangle}$ without evaluating any trigonometric functions. This is very important as it reduces the energy expended by individual sensors.

Practical measuring of the angle between the positive x-axis and the line connecting the sink to any sensor is more challenging. As mentioned in the network model, we assume that the sink is capable of both omnidirectional and directional transmission. The directional antenna at the sink has a small beam-width and can be rotated toward any direction. Recall that antenna physics states that the transmission pattern of directional antenna consists of a major lobe which is oriented in the direction of the transmission and several smaller (back and side) lobes [35]. The received transmission power is maximum at the center of the major lobe and reduces as we go far from the center. For the purpose this work, we can simplify the antenna transmission pattern by representing it as a narrow sector with a small angle that is divided in half by the transmission direction beam (see Figure 5).

Initially, the sink uses its omnidirectional antenna to send a sequence of WAKEUP messages to wake up sleeping sensors so they can measure their angle to the sink. Obviously, the number of WAKEUP messages should be sufficiently large so that each sensor within the disk D will receive at least one copy of the message. Moreover

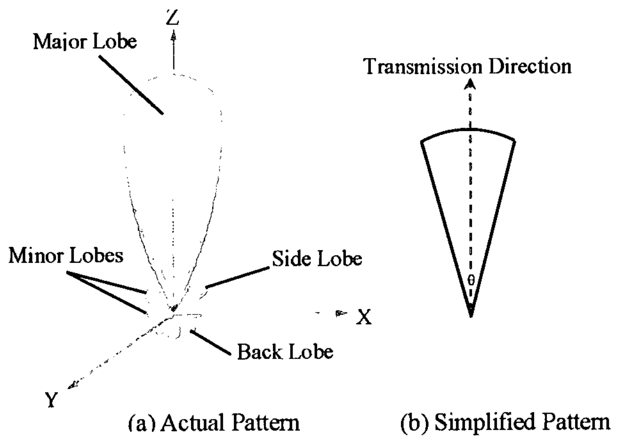


FIG. 5: *Radiation lobes and beamwidths of directional antenna pattern*

to save sensors energy, we suggest that the number of remaining WAKEUP messages be specified within each message, so when a sensor receives a WAKEUP message in an early stage and realizes that there will be more WAKEUP messages to follow, it can save energy by turning off its sensory and reception circuitry, switching to the sleep mode after adjusting its internal timer to wake up on time.

In addition to waking up sensors, the last WAKEUP message should also provide some level of synchronization among sensors. Although this kind of synchronization may not be accurate due to different transmission, propagation and processing delays at each node, the achieved level of synchronization (within a few milliseconds) is more than sufficient for our purpose especially in the existence of the large delays due to the mechanical rotation of the directional antenna.

After the last WAKEUP message, each sensor turns on its reception circuitry and waits. At the same time, the sink uses its directional antenna to start transmitting angle estimation messages starting from an initial angle θ_0. After transmitting a message, the sink rotates its antenna by a small angle $\Delta\theta$, then it transmits the next message and so on. Although angle estimation messages are very short, they convey useful pieces of information to sensors. The most important among these pieces is the current angle of transmission θ. When a sensor receives a recognizable angle estimation message (i.e received signal strength p_r is larger than some threshold value p_{th}), it stores the angle θ along with the power of received signal p_r. When the antenna of the sink returns back to the initial angle θ_0, it can either stop, or start another cycle using a different value for the rotation angle $\Delta\theta$ in order to enhance the accuracy of the estimated angles. Obviously, there is a trade-off between the accuracy of the estimated angles and the number of cycles needed which will definitely affect the time and energy consumption. After the last angle estimation cycle, each sensor estimates its angle as the average of the received angles weighted by their received signal strength p_r. Mathematically, this can be written as

$$\theta = \frac{\sum_{m=1}^{n} p_{rm}\theta_m}{\sum_{m=1}^{n} p_{rm}}, \qquad (6)$$

where n is the total number of received messages, θ_m is the angle transmitted in message m, and p_{rm} is the received signal strength of message m. Recall that the power of radio signals decays proportionally with the inverse of the traveled distance raised to the path loss exponent (≥ 2). Typically, reflected signals travel a distance that is longer than the distance traveled by direct LOS signals. Hence, if the initial transmission power p_0 remains the same, then the received power of reflected signals should be smaller than the received power of direct LOS signals. Consequently, when angle θ_m is weighted by the received power p_{rm}, we reduce the impact of reflected signals on the accuracy of estimated angle.

Algorithm 1 Evaluate $\sin\theta$ and $\cos\theta$

Input: Angle θ

Output: $\sin\theta$ and $\cos\theta$

```
1: $\theta^2 = \theta \cdot \theta$ ;
2: $F[] := \{\frac{1}{0!}, \frac{1}{1!}, \frac{-1}{2!}, \frac{-1}{3!}, \frac{1}{4!}, \frac{1}{5!}, \frac{-1}{6!}, \frac{-1}{7!}, \frac{1}{8!}, \frac{1}{9!}\}$ ;
3: sin := F[9] ;
4: cos := F[8] ;
5: for (i=7; $i > 0$ ;) do
6:     sin := $\theta^2$ · sin + F[i−−] ;
7:     cos := $\theta^2$ · cos + F[i−−] ;
8: end for
9: sin := sin · $\theta$ ;
10: return sin, cos ;
```

After evaluating its angle to the sink using equation (6), each sensor evaluates the sine and the cosine of its angle using the well known MacLaurin expansion. By adding the first 5 terms of the MacLaurin expansion of the sine and the cosine, we obtain an accuracy of up to 4 decimal places which is more than sufficient for our purpose. Moreover, we can rewrite the approximated expansions of the sine and the cosine functions using Horner's rule [36] to reduce the number of multiplications as follows

$$\begin{aligned}
\sin\theta &\approx \theta - \frac{\theta^3}{3!} + \frac{\theta^5}{5!} - \frac{\theta^7}{7!} + \frac{\theta^9}{9!} \\
&\approx \left(\left(\left(\left(\frac{\theta^2}{9!} - \frac{1}{7!}\right)\theta^2 + \frac{1}{5!}\right)\theta^2 - \frac{1}{3!}\right)\theta^2 + 1\right)\theta \\
\cos\theta &\approx 1 - \frac{\theta^2}{2!} + \frac{\theta^4}{4!} - \frac{\theta^6}{6!} + \frac{\theta^8}{8!} \\
&\approx \left(\left(\left(\frac{\theta^2}{8!} - \frac{1}{6!}\right)\theta^2 + \frac{1}{4!}\right)\theta^2 - \frac{1}{2!}\right)\theta^2 + 1
\end{aligned}$$

Furthermore, we can combine the evaluation of the two series and reduce required number of multiplication to 10 operations only as shown in Algorithm 1.

By implementing Algorithm 1, and through 10 multiplications only, each sensor can evaluate the sine and the cosine of its angle to the sink node. We draw the attention of the reader that using this evaluation method is more appropriate and consistent with the limited computational power available to these tiny devices.

III.2.2 Order of Selection of Backbone Sensors

The selection process of backbone sensors starts when the sink selects the six backbone sensors in the first row. After that, the process continues recursively where the sensors in any row select sensors in the next row. This continues for a sufficient number of rows necessary to cover the desired disk D. In practice, we expect the maximum number of rows to be typically around 5 rows.

Backbone sensors can be selected in many ways and in different order. However, to avoid redundant selections, minimize collisions, and save sensors energy, we propose a set of rules that determine the order and the selection responsibility of backbone sensors, these rules are:

1. Only the sensors with odd column coordinate are allowed to select.

   (i.e. sensor $S_{\langle s,r,c\rangle}$ is allowed to select $\iff$ $c \equiv 1 \pmod 2$).

2. If $[(c = 1)$ and $(r \equiv 0 \pmod 2)]$ Then

   $S_{\langle s,r,1\rangle}$ selects $S_{\langle s,r+1,1\rangle}$, $S_{\langle s,r+1,2\rangle}$ and $S_{\langle s-1,r+1,r+1\rangle}$.

   Else

   $S_{\langle s,r,2c-1\rangle}$ selects $S_{\langle s,r+1,2c-1\rangle}$ and $S_{\langle s,r+1,2c\rangle}$.

3. Assuming, that the selection of a single backbone sensor takes one time epoch, then to reduce collisions and interference between transmissions, selection in odd sectors (i.e. $s = 1, 3, 5$) occurs in different time epochs than selection in even sectors (i.e. $s = 2, 4, 6$).

Figure 3 shows selection order and responsibilities for sector 5.
Relative to the above rules for backbone selection we note the following:

- Backbone sensors with even column coordinate do not select other backbone sensors;
- Selection in odd rows in any sector requires 2 time epochs, however selection in even rows requires 3 time epochs;
- The total number of time epochs needed to search in all the six sectors is 4 time epochs for odd rows and 6 time epochs for even rows.

III.2.3 The Details of Backbone Selection

In Subsection III.2.2, we specified the order and the selection responsibilities according to which backbone sensors are selected. In this section, we give the technical details of the backbone selection process.

The protocol starts when the searching entity (i.e. the sink or some sensor) uses equation (1) to compute $\tan\theta_{\langle s,r,c\rangle}$, the tangent of the angle subtended by the positive x-axis and the line connecting the sink to the center of the target hexagon identified by the tuple $\langle s, r, c\rangle$. After that, it broadcasts a message to all the sensors in its neighborhood asking for the closest sensor to the center of the target hexagon to declare itself. Recall that sensors at this point are aware of their angle to the sink node θ_s as described in Subsection III.2.1. The sensors that receive the message, check if the difference $|\tan\theta_s - \tan\theta_{\langle s,r,c\rangle}|$ is within acceptable range (i.e less than certain threshold). The sensors within the range use RSS to estimate their distance to the searching sensor S. They also use additional information within the message transmitted by S to estimate e_s^2, the square of their distance to the center of the target hexagon (calculation of e_s^2 is presented in detail later). After that, each sensor initializes a countdown timer to $\frac{Ke_s^2}{t_x^2}$ units and waits till its timer expires. When the timer of any of these sensors expires, the sensor realizes that it is the closest sensor to the center of the target hexagon. Consequently, it broadcasts a message to all

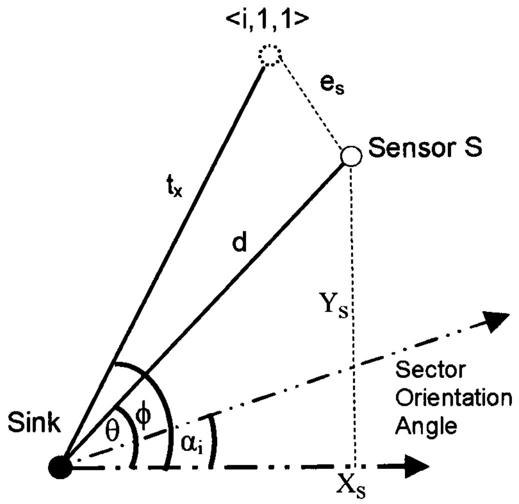


FIG. 6: *Case 1: estimation of* $e_s{}^2$

its neighbors declaring itself as the backbone sensor representing the new hexagon. The sensors that receive the message stop their timers and use this message along with messages they receive from other backbone sensors to determine the hexagon to which they belong. Sensors which evaluate e_s^2 to be larger than some threshold value (i.e. e_{th}^2) do not initialize their internal timers. This provides the stopping criterion upon which the boundaries of deployment area is reached.

Although collisions might well occur, they do not represent a big problem as ties between colliding sensors can always be broken by any contention-based mechanism. One way of doing this is to let colliding sensors wait a random amount of time before transmitting again. The first sensor to transmit is selected to be the backbone sensor.

As we mentioned earlier, during the selection of backbone sensors, each candidate sensor needs to estimate e_s^2, the square of the distance between the sensor and the center of the target hexagon. To evaluate this value, we distinguish between two cases. The first case handles the evaluation for the six backbone sensors around the sink (i.e. in row 1), while the second case handles the evaluation for backbone sensors in other rows.

Case 1: for each sector i, the sink has to select backbone sensor s to represent the hexagon $\langle i, 1, 1\rangle$. The selection error of sensor s is e_s and it represents the distance between s and the center of the hexagon $\langle i, 1, 1\rangle$. Basically, the criterion to select s is to keep e_s minimum. Using Figure 6, it can be readily verified that the position

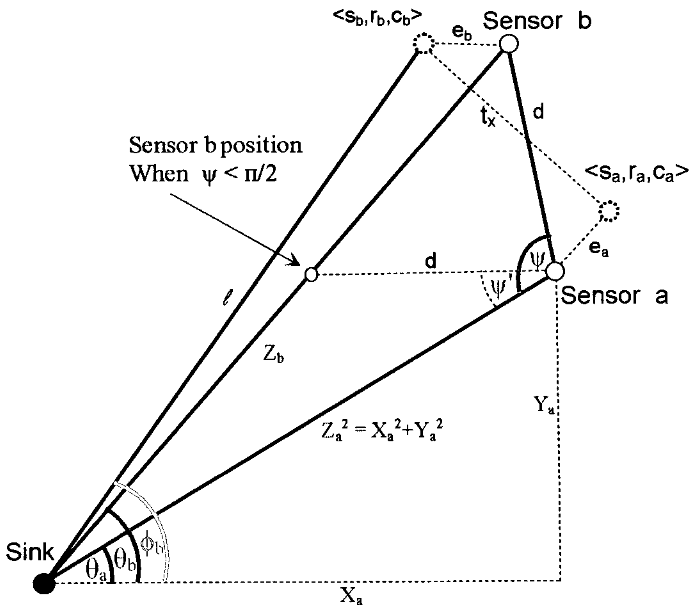


FIG. 7: *Case 2: estimation of* $e_b{}^2$

(X_s, Y_s) of sensor s relative to the sink and e_s^2 can be evaluated as

$$X_s = d \cdot \cos\theta \tag{7}$$

$$Y_s = d \cdot \sin\theta \tag{8}$$

$$\begin{aligned} e_s^2 &= t_x{}^2 + d^2 - 2t_x d \cos(\phi - \theta) \\ &= t_x{}^2 + d^2 - 2t_x d\,(\cos\phi\cos\theta + \sin\phi\sin\theta) \\ &= t_x{}^2 + d^2 - \frac{2t_x d\,(\cos\theta + \tan\phi\sin\theta)}{\sqrt{1+\tan^2\phi}} \end{aligned} \tag{9}$$

where d is the distance between the sink and the sensor and is estimated using RSSI. *Case 2:* as shown in Figure 7, we assume the existence of backbone sensor a that was previously selected by the protocol to represent the hexagon $\langle s_a, r_a, c_a\rangle$. The selection error of sensor a is denoted by e_a and represents the distance between a and the center of the hexagon $\langle s_a, r_a, c_a\rangle$. We recall that a was selected such that $e_a{}^2$ is minimum. Now it is sensor a's turn to select another backbone sensor b to represent the hexagon $\langle s_b, r_b, c_b\rangle$. Again, b should be selected such that the selection error $e_b{}^2$ is minimum (e_b is the distance between b and the center of the hexagon $\langle s_b, r_b, c_b\rangle$).

Our goal is to provide an expression for e_b^2 that can be evaluated by each sensor independently. The evaluation process starts at the searching sensor, a, when it broadcasts a message asking for the closest sensor to the center of the target hexagon to declare itself. Within this message, sensor a includes:

1. $Z_a = \sqrt{X_a^2 + Y_a^2}$, the Euclidean distance between the sink node and sensor a;

2. $\sin\theta_a$ and $\cos\theta_a$, the sine and the cosine of the angle between the positive x-axis and the line connecting the sink to sensor a;

3. In addition to this, sensor a evaluates and sends ℓ and $\tan\phi_b$, where ℓ is the Euclidean distance between the center of the target hexagon and the sink, while ϕ_b is the tangent of the angle subtended by the positive x-axis and the line connecting the sink to the center of the target hexagon. Clearly, ℓ and $\tan\phi_b$ can be evaluated using

$$\begin{aligned} \ell &= \sqrt{X^2 + Y^2} \\ \tan\phi_b &= \frac{r_b\sqrt{3}\tan\frac{\pi s_b}{3} + (2c - r - 2)}{r_b\sqrt{3} + (r_b - 2c_b + 2)\tan\frac{\pi s_b}{3}}, \end{aligned}$$

   where X and Y are given by the equations (4), and (5) respectively.

After receiving the message transmitted by sensor a, each sensor continues the evaluation of e_b on its own. Given that

- Z_a, $\sin\theta_a$, $\cos\theta_a$, ℓ and $\tan\phi_b$ are known from the message received from sensor a;

- d is estimated through the strength of the signal received from sensor a;

- $\sin\theta_b$ and $\cos\theta_b$, (the sine and the cosine of the angle subtended by the positive x-axis and the line connecting the sensor b to the sink) were estimated at protocol initialization through WAKEUP messages, a generic sensor b can estimate the distance Z_b between itself and the sink by elementary trigonometry as follows:

$$\begin{aligned}
\frac{Z_a}{\sin(\pi - (\theta_b - \theta_a + \psi))} &= \frac{d}{\sin(\theta_b - \theta_a)} \\
\sin(\theta_b - \theta_a + \psi) &= \frac{Z_a \cdot \sin(\theta_b - \theta_a)}{d} \\
\cos\psi + \frac{\sin\psi}{\tan(\theta_b - \theta_a)} &= \frac{Z_a}{d} \\
\frac{1 + 2A\tan\psi + A^2\tan^2\psi}{1 + \tan^2\psi} &= \frac{Z_a^2}{d^2} \\
1 + 2A\tan\psi + A^2\tan^2\psi &= \frac{Z_a^2}{d^2} + \frac{Z_a^2}{d^2}\tan^2\psi, \qquad (10)
\end{aligned}$$

where

$$\begin{aligned}
A &= \frac{1}{\tan(\theta_b - \theta_a)} = \frac{\cos(\theta_b - \theta_a)}{\sin(\theta_b - \theta_a)} \\
&= \frac{\cos\theta_b\cos\theta_a + \sin\theta_b\sin\theta_a}{\sin\theta_b\cos\theta_a - \cos\theta_b\sin\theta_a} \qquad (11)
\end{aligned}$$

Because in our protocol we always assume that sensors in any row select sensors in the next row, the value of the angle ψ is larger than $\frac{\pi}{2}$. For the case, when $\tan\theta_b = \tan\theta_a$, which implies for our scenario that $\theta_b = \theta_a$. Hence, $\psi = \pi$, and $Z_b = Z_a + d$. For the general case, we solve the quadratic equation (10) for $\tan\psi$,

$$\begin{aligned}
(A^2d^2 - Z_a^2)\tan^2\psi + 2Ad^2\tan\psi + d^2 - Z_a^2 &= 0 \\
\frac{-2Ad^2 \pm \sqrt{4A^2d^4 - 4(A^2d^2 - Z_a^2)(d^2 - Z_a^2)}}{2(A^2d^2 - Z_a^2)} &= \tan\psi
\end{aligned}$$

After a bit of algebra,

$$\tan\psi = \frac{Ad^2 + Z_a\sqrt{d^2 + A^2d^2 - Z_a^2}}{Z_a^2 - A^2d^2} \qquad (12)$$

Here, we chose the negative root to guarantee that $\tan\psi$ is negative since $A > \frac{Z_a}{d}$. Now, we can evaluate Z_b as,

$$\begin{aligned}
\frac{Z_b}{\sin\psi} &= \frac{d}{\sin(\theta_b - \theta_a)} \\
Z_b &= \frac{d \cdot \sin\psi}{\sin\theta_b\cos\theta_a - \cos\theta_b\sin\theta_a} \\
&= \frac{d \cdot \tan\psi}{(\sin\theta_b\cos\theta_a - \cos\theta_b\sin\theta_a)\sqrt{1 + \tan^2(\psi)}} \qquad (13)
\end{aligned}$$

Finally, after evaluating Z_b, each sensor can apply the trigonometric law of cosines to evaluate e_b^2 as

$$\begin{aligned} e_b^2 &= \ell^2 + Z_b^2 - 2\ell Z_b \cos(\phi_b - \theta_b) \\ &= \ell^2 + Z_b^2 - 2\ell Z_b \left(\cos\phi_b \cos\theta_b + \sin\phi_b \sin\theta_b\right) \\ &= \ell^2 + Z_b^2 - \frac{2\ell Z_b \left(\cos\theta_b + \sin\theta_b \tan\phi_b\right)}{\sqrt{1+\tan^2\phi_b}} \qquad (14) \end{aligned}$$

After evaluating e_b^2, each sensor initializes its internal timer using $\frac{Ke_b^2}{t_x^2}$ as described earlier. The winning sensor, i.e. the sensor whose timer expires first, declares itself as the selected backbone sensor by broadcasting a message to other candidate sensors. The selected backbone sensor estimates its position relative to the sink node using

$$\begin{aligned} X_b &= Z_b \cdot \cos\theta_b \qquad (15) \\ Y_b &= Z_b \cdot \sin\theta_b. \qquad (16) \end{aligned}$$

III.3 BACKBONE SWITCHING

One of the major advantages of our proposed backbone is that it provides an implicit clustering mechanism where hexagons can be viewed as clusters and backbone sensors are cluster heads. Although this can simplify many network management tasks including data aggregation, leader election and routing, it usually results in uneven energy consumption among sensors. In particular, it imposes higher tasking load on backbone sensors much more than it does on other sensors which results in depleting their energy much faster.

One way to overcome this problem is through changing the cluster head periodically in order to distribute the additional load on different sensors. In this section, we propose *backbone switching* as a solution to the energy balancing problem.

The main idea behind backbone switching is to construct disjoint backbones and to periodically switch between these backbones to balance their energy consumption. Initially, using the approach described in Section III.2, each sink constructs its first backbone. For lack of a better term, we refer to this as the main backbone. As described earlier, the main backbone is constructed by selecting sensors in six different directions (i.e. $\frac{\pi}{6}$, $\frac{3\pi}{6}$, $\frac{5\pi}{6}$, $\frac{7\pi}{6}$, $\frac{9\pi}{6}$, and $\frac{11\pi}{6}$). Now, what happens if the sink node rotated its positive x direction by some angle θ such that $0 < \theta < \frac{\pi}{3}$? Theoretically, and as shown in Figure 8, the rotation should result in selecting a completely different

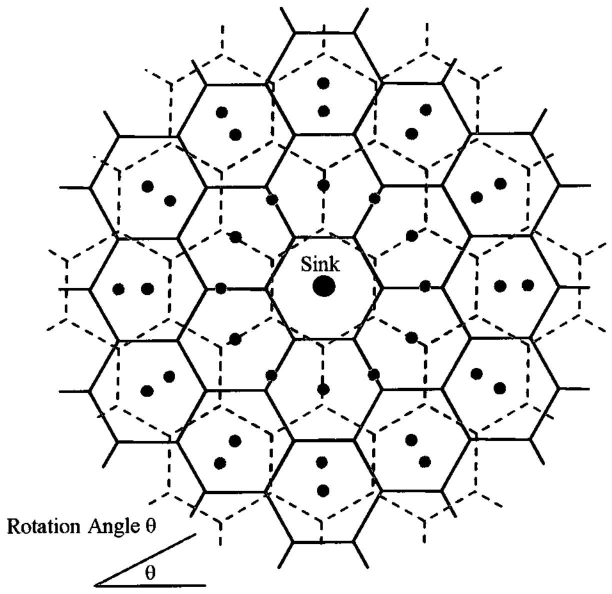


FIG. 8: *Balancing energy consumption using backbone switching*

set of sensors (i.e an alternative backbone). While theoretically correct, this view may not be entirely correct from a practical perspective. Recall that backbone sensors are selected to be the closest sensors to the hexagon centers they represent. So it is possible that even after rotation, the same sensor may still be the closest sensor to the new hexagon center, hence it can be selected to be part of the alternative backbone. Fortunately, our selection protocol only selects sensors that nominate themselves to be part of the network backbone (by participating in the countdown process). Hence, if sensors which are already part of another backbone do not nominate themselves, they will not be selected as part of the new backbone giving chance to other sensors to join the new backbone. This trick provides a simple solution to guarantee that alternative backbones are disjoint.

After constructing the main backbone, each sink constructs a set of alternative backbones using appropriately selected angles θ_i. Each backbone is associated with an ID assigned by its sink node. At any point of time, only one backbone should be active. It is the responsibility of the sink to periodically broadcast messages to change the current active backbone giving a chance to sensors in other backbones to save their energy.

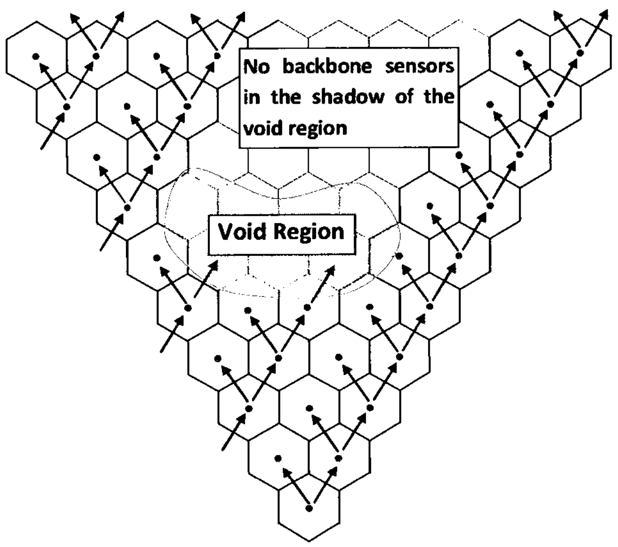


FIG. 9: *Unlocalized sensors in the shadow of void regions*

III.4 RECOVERING FROM SENSOR VOIDS

In certain applications the sensors are deployed in rough environments in which there exist some spots in the deployment area where sensors can not be deployed. We refer to these spots as *voids*. Voids can be created naturally by physical obstacles (e.g. lakes, streams, large rocks, deep holes, steep slopes, etc.). Voids can be created as well when all the sensors within a certain spot expire due to energy depletion.

The main goal of this section is to discuss how our proposed construction protocol would perform in the presence of such voids. Initially, we point out to the problems that might arise due to the existence of these voids. After that, we show how the proposed protocol can overcome these problems.

Figure 9 shows how voids can prevent the propagation of the backbone selection process. If no backbone sensors can be selected in the void region and, consequently, the selection process stops and no backbone sensors are selected in the shadow of the void region. To get around this problem, we propose the "Even Neighbor Replacement" rule and enhance it later by adding "Backward Selection".

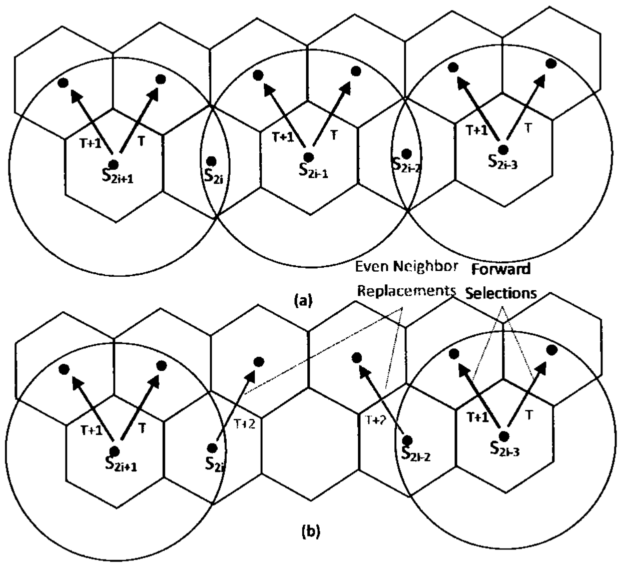


FIG. 10: *Illustrating even neighbor replacement*

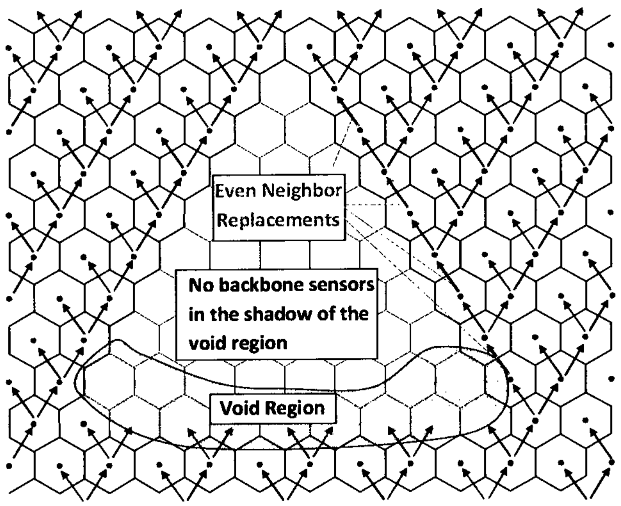

Even Neighbor Replacements
No backbone sensors in the shadow of the void region
Void Region

FIG. 11: *Recovering from voids using even neighbor replacement*

III.4.1 Even-Neighbor Replacement – the Details

Recall that in our basic backbone selection rules, in any row, only sensors with odd column coordinate are allowed to select backbone sensors in the next row. Moreover, as shown in Figure 10-(a), every even neighbor can receive selection messages transmitted from its two immediate odd neighbors. If the initial selection rules failed to select an odd backbone sensor due to the existence of a void, then all backbone sensors belonging to the tree rooted at the missing sensor are pruned out. This kind of behavior blocks the propagation of backbone selection in the shadow area behind the void.

The idea behind the even neighbor replacement rule is to allow the immediate even neighbor sensors to replace any missing odd sensors in order to continue the selection chain. Figure 11, shows an example of how our construction protocol would work when applying the even neighbor replacement rule. Although, the protocol can recover from the void region, its recovery rate is relatively slow which leaves a large region of the deployment area uncovered by backbone sensors. This motivates for our next selection rule that we add to our protocol rules in order to expedite the rate by which voids are recovered.

III.4.2 Backward Selection – the Details

The idea behind this rule is simple. If the selection of a backbone sensor was initiated by a sensor other than the one determined by the basic rules (i.e. through even-neighbor replacement or another backward selection), then the selection responsibility is reversed and the newly selected sensor carries the responsibility of selecting the odd sensor that was supposed to select it. Figure 12 shows how the rule is applied.

Initially, sensor S_{2i-1} transmits a message calling for the closest sensor to the center of hexagon H_3 to announce itself. As expected, sensor S_{2i} receives the message transmitted by sensor S_{2i-1}, however, it does not receive a similar message from sensor S_{2i+1}. This motivates sensor S_{2i} to apply the even neighbor replacement rule and transmits a message calling for the closest sensor to the center of the hexagon (H_2) to announce itself. Sensor A_2 receives the message and announces itself to other sensors. Moreover, from the information embedded within the message that triggered its selection, sensor A_2 realizes that the message was transmitted by sensor

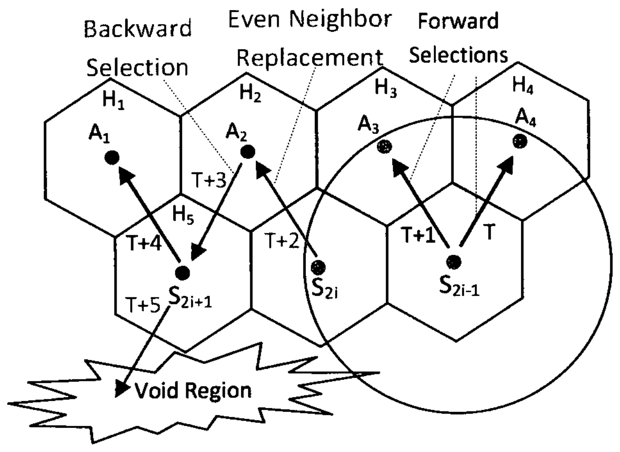


FIG. 12: *Even-neighbor replacement with backward selection*

S_{2i+1} and not sensor S_{2i} as determined by the basic selection rules. This motivates sensor A_2 to apply backward selection rule and transmits a backward message calling for the closest sensor to the center of the hexagon (H_5) to announce itself. Sensor S_{2i+1} receives the message and announces itself to other sensors. Following the basic selection rules, sensor S_{2i+1} transmits a message calling for he closest sensor to hexagon H_1 to announce itself. Sensor A_1 responds to this message announcing itself to other sensors. After fulfilling its forward selection obligations, and because it was not selected according to basic forward selection rules, sensor S_{2i+1} continues its backward selection toward the void region.

Figure 13, illustrates an example of how our proposed construction protocol works when applying the even-neighbor replacement rule with backward selection. Obviously, the protocol can recover from void region efficiently.

III.5 MITIGATING NETWORK CHALLENGES

Sensor networks have their own distinguishing characteristics that set them apart from other types of networks. The ad-hoc nature of deployment, location unawareness, modest non-renewable energy budget, limited computing and communication capabilities, along with the dynamically changing topology induced by the sleep-awake cycles are only few examples of the typical challenges faced by WSN protocol

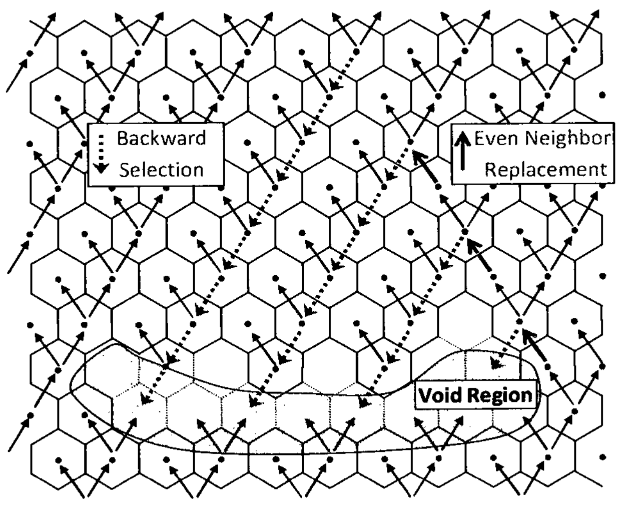


FIG. 13: *Voids recovery using even neighbor replacement/backward selection*

designers. Instead of solving each of the aforementioned problems individually, facing the same common challenges with each problem, we show how our proposed backbone can be very useful in collectively simplifying solutions for these problems.

Our network backbone provides some form of virtual infrastructure that allows the sensors to acquire *coarse-grain location awareness* and promotes *dynamic clustering*. Thus, on the one hand, the infrastructure provides the sensors with necessary information that enables them to associate their sensory data with the geographic location in which the data was measured and, on the other hand, it simplifies the task of clustering the sensors in support of various network tasks. Once such an infrastructure is in place, entire protocol suites can leverage the infrastructure, resulting in ease of programming and energy savings. In particular, by tiling the area around sinks using identical hexagons, the construction algorithm clusters sensors based on their locations into hexagons (clusters). Backbone sensors represent cluster heads and can play a crucial rule in data aggregation, workforce selection, task management, leader election, duty cycle scheduling, and local synchronization.

In the following subsections, we show how our proposed backbone can simplify sensor localization [37], local data aggregation, geographic routing, and clustering. We also point to how using mobile sinks on top of our backbone can reduce the energy holes growth rate within the network. We dedicate Chapter IV for backbone-based task management and workforce selection. We start Chapter V by a rigorous analysis on awake sensor density and its relation to different scheduling schemes. After that, we propose a backbone guided energy-aware scheduling scheme for balancing sensor energy consumption.

III.5.1 Sensor Localization

As the exact position of a sink (X_{sink}, Y_{sink}) can be broadcast to all the sensors in a disk D of interest, as an additional field in WAKEUP messages, we assume without loss of generality that (X_{sink}, Y_{sink}) is known to all the sensors in D. Moreover, we have shown earlier in Subsection III.2.3 that the sensors can estimate their position relative to the sink through equations (7),(8), (15), and (16). Using the available information, each sensor can estimate its absolute position (X,Y) as:

$$X = \begin{cases} X_{sink} + d \cdot \cos\theta & \text{for case 1} \\ X_{sink} + Z_b \cdot \cos\theta_b & \text{for case 2} \end{cases}$$

$$Y = \begin{cases} Y_{sink} + d \cdot \sin\theta & \text{for case 1} \\ Y_{sink} + Z_b \cdot \sin\theta_b & \text{for case 2} \end{cases}$$

As expected and confirmed by simulation in Section III.6, the localization accuracy decreases almost linearly with the distance between a sensor and the sink. When the sinks are mobile or in the case of a relatively large number of sinks, we can use these sinks to enhance the achieved level of accuracy as follows: the sensors that reside close to one of the network sinks will typically belong to a hexagon that has a small row coordinate, hence these sensors should be localized accurately. On the other hand, sensors that reside far from the network sinks and close to the boundaries of the localization regions of different sinks will typically receive localization messages triggered by each of these sinks. From the received messages, boundary sensors can localize themselves relative to each of these sinks. These sensors should estimate their final position as the weighted average of the positions estimated from each sink individually. The weight of each position is evaluated based on the row coordinate of the hexagon that contains the sensor relative to the sink used to evaluate this position. The positions associated with small row coordinates are expected to be more accurate, hence they are assigned higher weights than positions associated with large row coordinates. Mathematically, this can be expressed as follows,

$$\begin{aligned} w_i &= 1 - \frac{r_i}{R+1} \\ X &= \frac{\sum_{i=1}^{S} w_i \cdot X_i}{\sum_{i=1}^{S} w_i} \\ Y &= \frac{\sum_{i=1}^{S} w_i \cdot Y_i}{\sum_{i=1}^{S} w_i} \end{aligned}$$

where

- S is the number of sink nodes used to localize the sensor,
- (X_i, Y_i) is the position of the sensor as estimated through sink i,
- R is the maximum number of rows allowed within the disk centered at the sink,
- r_i is the row coordinate of the hexagon that contains the sensor when localized through the sink i.

III.5.2 Clustering and Leader Election

Our backbone implicitly clusters the sensors based on their geographic location. Each hexagon represents a cluster and the backbone sensor around the center of each hexagon is the cluster head which can be always elected as the leader to coordinate between sensors in its hexagon for any centralized protocol. For instance, backbone sensors can play an important rule in workforce selection and task management for all sensing tasks issued in the hexagons they represent. More details about this approach are presented in Chapter IV. Furthermore, backbone sensors can be treated as elected coordinators for any centralized synchronization or scheduling protocols for sensors within their hexagons. This is discussed in more details in Chapter V. We note here that when it comes to selecting the backbone sensor in a given hexagon, nothing prevents us from extending the selection protocol in the obvious way to select a *committee* of several possible backbone sensors that may collectively act as cluster leaders or, indeed, may take turns serving any assigned tasks.

III.5.3 Geographic Routing

Given the coordinates of the hexagon that contains the source sensor in the ternary system defined by our backbone protocol, it is straightforward to find a path from the source hexagon to the sink by hopping through backbone sensors representing the hexagons in between (see Figure 14). The details behind the selection of the route through which data flows toward the sink are presented in the next subsection. Here, it is worthwhile to mention that by controlling the mobility of sink nodes, we can tremendously reduce the growth rate of energy holes within the network.

III.5.4 Data Aggregation:

Non-backbone sensors within any hexagon can report their sensory data to the backbone sensor in their hexagon which can locally aggregate the data before forwarding the aggregated result to the next backbone toward the sink node.

Using backbone hexagons, sensory data aggregation and routing aggregated results toward sink nodes can be straightforward. In particular, when a sensor participates in any task, it transmits its results to the nearest backbone sensor where sensory data can be locally aggregated. After that, backbone sensors in row r forward their aggregated results to backbone sensors in row $r - 1$ and so on toward the sink node. Backbone sensors in row r use a reversed version of the same rules they followed during backbone selection in order to determine which backbone sensor in row $r - 1$ to forward their data to. Figure 14 illustrates an example of data aggregation and routing toward the sink node.

The reader should note that our backbone does not impose any restrictions on the order or the type of local data aggregation inside the hexagons. The aggregation process runs completely under the supervision of the backbone sensor within the hexagon. Furthermore, routing aggregated results toward the sink node by reversely following the path determined during backbone selection automatically provides a workaround for routing problems that might arise due to the existence of energy holes or void regions.

III.6 SIMULATION RESULTS

In order to evaluate the performance of our proposed backbone, we have built a simulator that implements our backbone construction protocol. In our simulation, we have run several experiments using different network parameters and configurations. In general, we assumed a rectangular deployment area where a number of sinks were

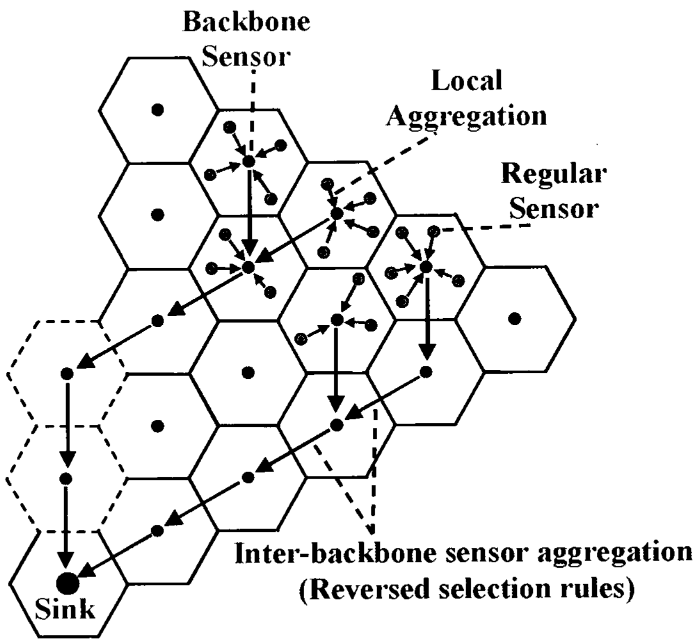

Backbone Sensor
Local Aggregation
Regular Sensor
Inter-backbone sensor aggregation (Reversed selection rules)
Sink

FIG. 14: *Data aggregation and routing through our backbone*

placed uniformly across the deployment area. We used a standard uniform pseudo-random generator to distribute sensors with required density in the deployment area.

We estimated the backbone selection error as the average Euclidean distance between the position of the selected sensor and the position of the center of the hexagon it represents. Mathematically,

$$\text{Error} = \frac{1}{n}\sum_{i=1}^{n}\sqrt{(x_{s_i} - x_{h_i})^2 + (y_{s_i} - y_{h_i})^2},$$

where (x_{s_i}, y_{s_i}) is the position of backbone sensor representing hexagon $i = \langle a, s, r, c\rangle$ and (x_{h_i}, y_{h_i}) is the position of center of the hexagon i.

To verify the correctness of our simulation implementation, we initially run our simulation assuming exact distance and angle measurements. Basically, in the absence of distance and angle measurement errors, there should be no errors in sensor localization (except for minor truncation errors), also the closest sensor to the center of each hexagon should be always selected to be the backbone representative of this hexagon (we intentionally put sensors at these positions and verified that they are appropriately selected). Figure 15 shows a plot of average localization error for different rows and verifies the correctness of our implementation.

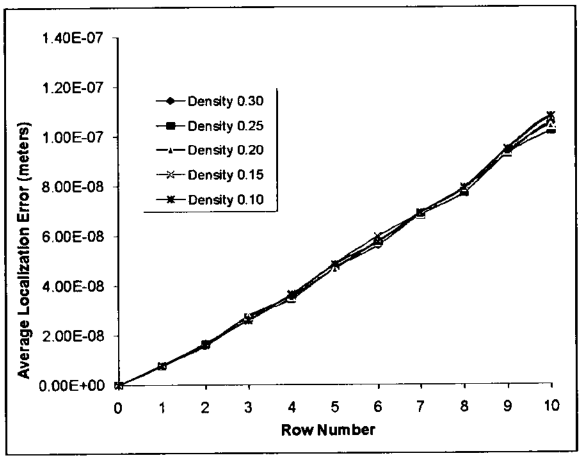


FIG. 15: *Average localization error of backbone sensors vs. row number for different network densities using a single sink in the absence of measurements errors*

After verifying the correctness of our implementation, we conducted several experiments to test the performance of our proposed backbone in the existence of errors in distance estimations and angle measurements. To account for errors in distance measurements due to the irregularity of signal propagation, we represented estimated distance between the transmitter and the receiver as a random variable that follows the Gaussian distribution with mean equals the exact distance and standard deviation equals 0.1 of the maximum transmission range (around $3m$ when $t_x = 30m$). In a similar fashion, we represented measured angle between a sensor and the sink node as a Gaussian random variable with mean equals the exact angle and a standard deviation equals 3 radian degrees.

In our first experiment, we were interested to know what the actual hexagons produced by our protocol look like. Figure 16 shows the actual hexagons produced by our simulation when the deployment area was set to a ($200m$ x $200m$) square, a single sink node is placed at $(0, 0)$, network density was set to $\rho = 0.3$ sensors/m^2, and sensor maximum transmission range $t_x = 30m$. Although the boundaries of the hexagons are completely distorted as they do not look like hexagons, we are still able to distinguish the spots they occupy. For the same experiment, Figure 17 shows the positions of hexagon centers and the positions of the corresponding backbone sensors representing them.

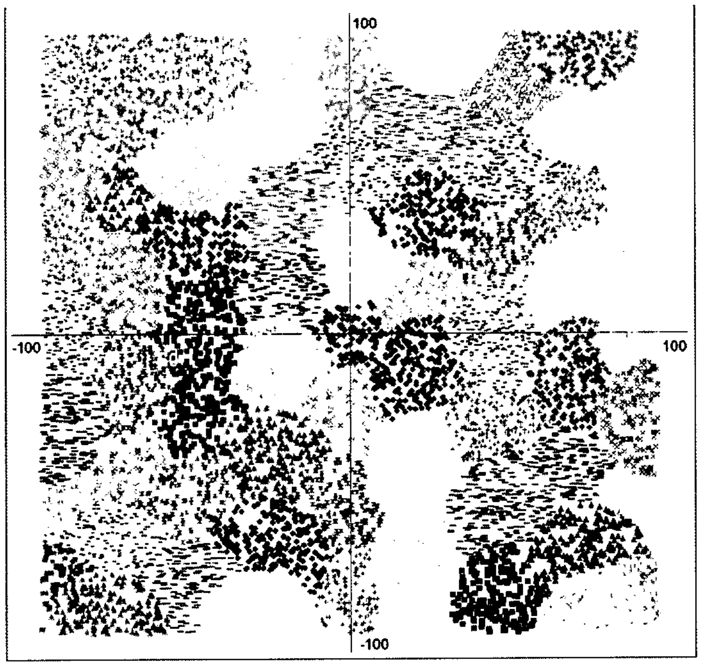


FIG. 16: *Actual hexagons produced by simulation*

After that, we conducted several experiments to measure the impact of distance/angle measurement errors on the localization accuracy. Figure 18-(a) shows the average localization error under different network densities (0.05, 0.10, 0.15, and 0.20) assuming only distance measurement errors. The figure shows that the backbone selection error increases almost linearly with the row number. When we repeated the same experiment under the same simulation parameters in the existence of angle measurement errors. Figure 18-(b) shows that the average localization error still increases linearly with the row number, however the slope of the curve nearly doubles its value in the distance-noise case. This can be explained by the fact that an angle measurement error of $\Delta\theta$ for a sensor that is away from the sink node by distance d, will result in a selection error proportional to $d \cdot \Delta\theta$. So for the same angle error, localization error increases as the value of d increases (i.e the row number increases). We repeated the same experiment a third time, however this time we considered both distance and angle measurement errors. As expected, Figure 18-(c) shows a similar linear relationship but with a larger slope.

We also conducted another experiment to compare the localization accuracy

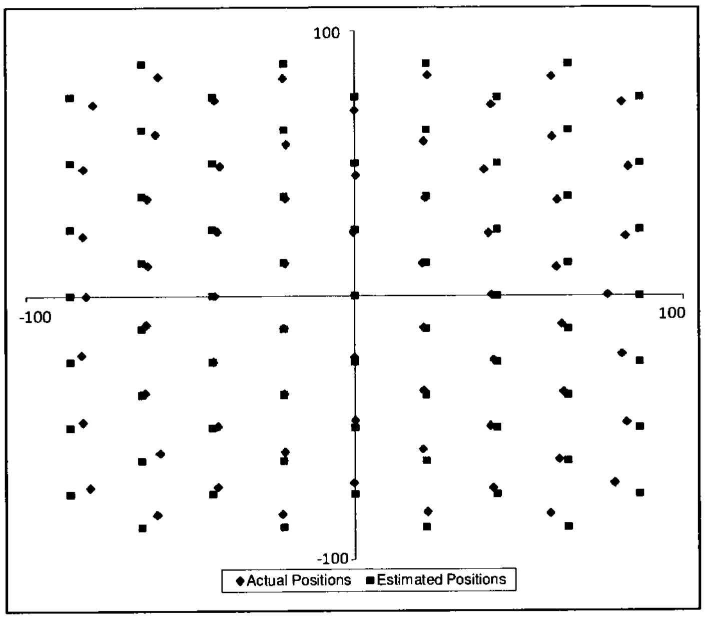
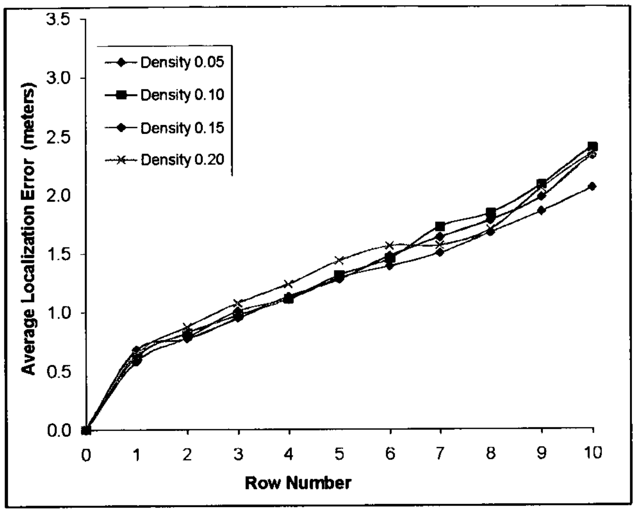


FIG. 17: *Actual vs estimated positions of backbone sensors.*

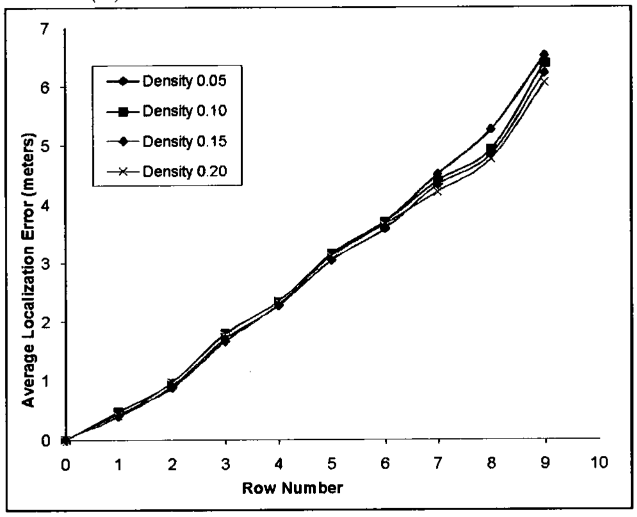
achieved when using our backbone against other localization techniques known in the literature. In particular, we compare the performance of our protocol to other RSSI based localization techniques like weighted centroid and APIT. [1]. We also compare our technique to two range-free protocols: the centroid and DV-HOP.

Figure 19 shows a quantitative comparison between average localization error for different localization protocols. The figure confirms that backbone-based localization has a better accuracy compared to other protocols.

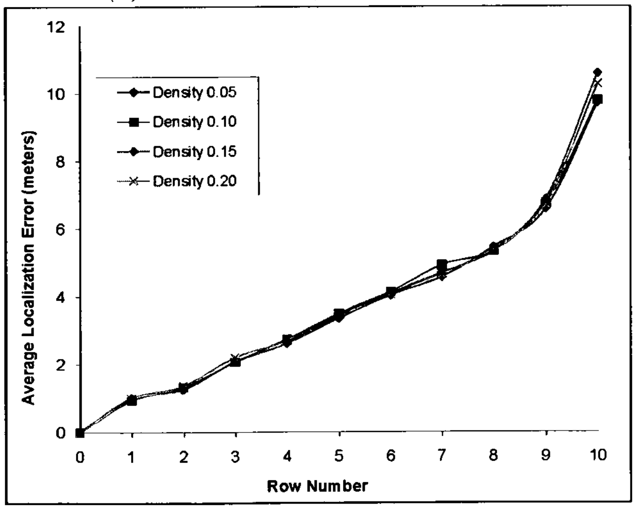
[1]APIT uses RSSI to approximate the PIT test



(a) Errors in distance measurements



(b) Errors in angle measurements



(c) Errors in distance and angle measurements

FIG. 18: *Average localization error in the presence of errors in distance and/or angle measurements for different network densities.*

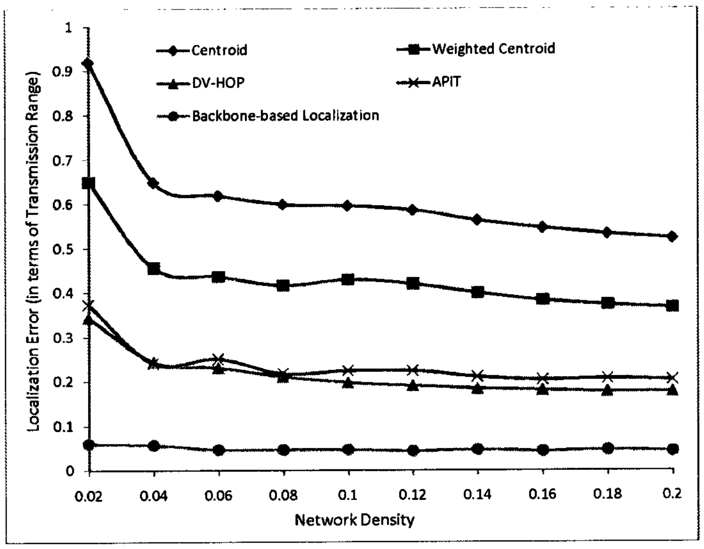


FIG. 19: *A comparison between average localization error for different localization protocols using different network densities*

CHAPTER IV

BACKBONE-BASED TASK MANAGEMENT

Sensors can perform their sensing tasks in two different modes: **Proactive Mode** and **Reactive Mode**. In the proactive mode, sensors periodically report their sensory data to one of the sink nodes that are distributed across the deployment area. The periodic reporting behavior of sensors makes the proactive mode more appropriate for surveillance and monitoring applications. On the other hand, in the reactive mode, sensors provide information only in response to queries or tasks assigned to them by task issuing entities (TIEs)(e.g sink nodes). This makes the reactive mode more appropriate for query based applications (e.g elderly aid navigation systems).

In networks adopting the reactive mode, the first step to perform a task is to recruit a workforce of sensors in the area of interest, these sensors will be responsible for the actual execution of the task. The workforce size is usually determined according to QoS requirements expressed in terms of the minimum number of sensors participating in each task. Although, some network designers may decide to skip workforce selection by allowing all the sensors in the task neighborhood to participate in the task execution, such decision would be completely inefficient from energy point of view especially for tasks whose QoS requirements can be satisfied by only a few number of sensors. Due to the modest and non-renewable energy budget available to sensors, it is highly not recommended to skip workforce selection in reactive sensor network applications. In general, task assignment and workforce selection can be challenging and handling it improperly may eventually result in many problems as shown in the following example.

Assume that sensors were deployed as shown in Figure 20-(a). Tasks T_1 and T_2 are issued in the indicated locations, each task is assumed to consume one unit of energy of each recruited sensor and requires the cooperation of at least three sensors (in order to increase the reliability of the readings). Furthermore, we assume that the sensing range for sensors is R, so sensors within range R from the task location are allowed to participate in the task. Other sensors can not participate because the monitored phenomenon is outside their sensing range. The chart in Figure 20-(b) shows the remaining energy of sensors S_1 through S_9. For illustration purposes, we assume that sensors S_1 through S_9 will be awake during sensor recruiting for tasks T_1 and T_2.

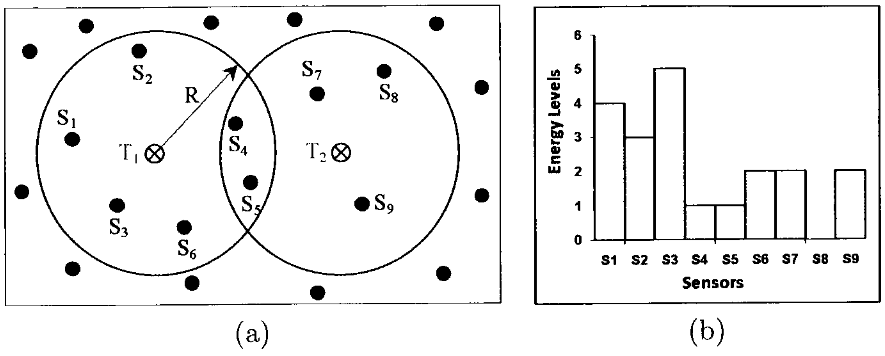


FIG. 20: *An example of improper task assignment*

Now, assume that the recruiting protocol for task T_1 selected sensors S_4, S_5, and S_6. Although the recruited sensors still have enough energy to complete task T_1, the energy of sensors S_4 and S_5 will be totally depleted after the task completion, hence they will not be able to participate in any upcoming task. Moreover, there is no way to recruit three sensors for task T_2. On the other hand, if the recruiting protocol for task T_1 had selected sensors S_1, S_2, and S_3, then sensors S_4 and S_5 would have saved their energy to participate in task T_2. A more clever protocol would not recruit sensors S_4 and S_5 together for task T_2, instead it would recruit sensors S_7, S_9 and either S_4 or S_5. Should there be a third task T_3 at the same location of task T_2, enough sensors will still be available to execute the task.

In spite of its simplicity, the previous example shows that task assignment can be very tricky and doing it improperly can cause many problems starting from reducing network density in different areas of the network and ending with the creation of energy holes that can eventually partition the network into disconnected islands. This in turn has its impact on network reliability and durability through increasing task failure rate and being unable to satisfy QoS requirements. The example also shows the importance of taking the difference in sensor energy into consideration while selecting the required workforce for any task.

Next, we propose two different techniques that can be used throughout our proposed backbone to efficiently recruit sensors for different network tasks. Both techniques consider sensor remaining energy in the workforce selection process giving priority to sensors with high energy over other sensors. While the first technique is centralized [38, 39, 40, 41] and depends on backbone sensors to coordinate workforce selection, the second technique is fully distributed and does not require a central entity for coordination [42, 43]. In the following sections, we present the details of both techniques highlighting the merits and the demerits of each of them.

IV.1 CENTRALIZED TASK MANAGEMENT

In this approach, workforce selection occurs in a centralized fashion coordinated by the nearest backbone sensor to the task position. We refer to this sensor as "Task Coordinator". Tasks are issued to sensors by TIEs (i.e sink nodes) according to the following tasking model.

IV.1.1 Centralized Tasking Model

Each sensing task is associated with a certain position that is chosen to be at the center of the area of interest. Using the localization protocol described earlier in subsection III.5.1, sensors should be aware of their positions. Based on the position of the task center, sensor position, sensor remaining energy, and sensing range, each sensor can determine whether it can participate in a given task or not. We assume that each task requires the cooperation of a workforce of w sensors and consumes one unit of energy of each participating sensor.

The sensors have a maximum transmission range denoted by t_x, and a maximum sensing range denoted by R. The communication and sensing ranges of sensors are governed by the inequality ($t_x \geq 2R$). This inequality guarantees that whenever two sensors are within the same sensing range (i.e. they can cooperate in the same task), they will be able to communicate with each other during the workforce selection process.

If the distance between the centers of two tasks is less than $2(t_x+R)$, then they are not allowed to run concurrently to avoid interference between messages transmitted during workforce selection (see Figure 21). This constraint can be partially removed if the TIEs can communicate between themselves through a separate channel. If this communication channel is available and if the distances between the centers of the tasks to be performed concurrently are relatively small, then these tasks can be combined into one single task. The QoS requirements of the combined task is chosen to be the maximum of the QoS requirements of the comprising subtasks. Only one

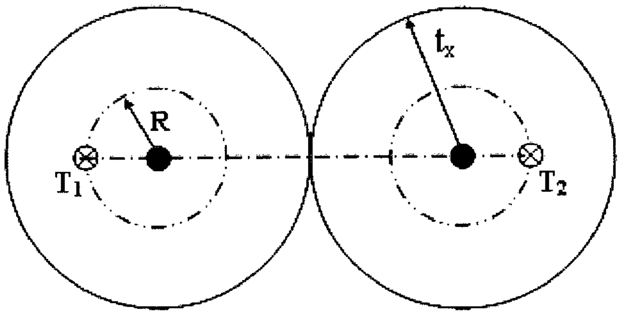


FIG. 21: *Minimum distance between two concurrent tasks*

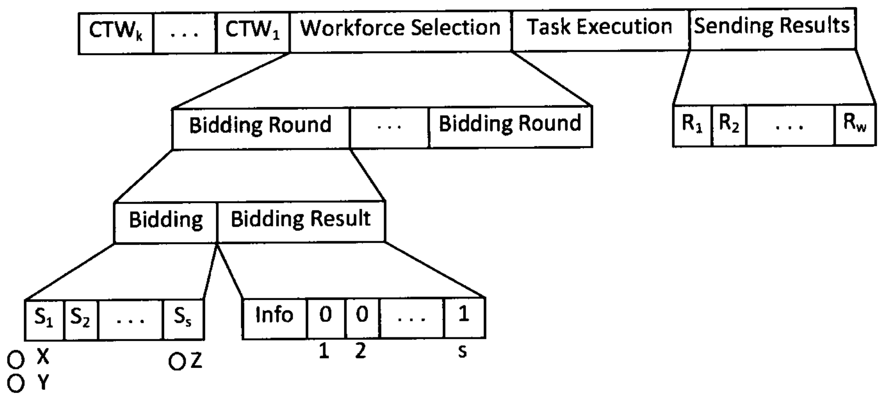


FIG. 22: *Tasking model for the centralized approach*

TIE should be responsible for running the combined task. Once the TIE gets the aggregated result, it can forward the result directly to other TIEs through their separate communication channel.

Under the assumptions described above, TIEs issue tasks for sensors and later they receive the aggregated results for taking decisions. Figure 22 shows the different stages of running a task under the centralized model. A task starts, when the TIE sends a sequence of Call To Work (CTW) messages to get the attention of a sufficiently large number of sensors that we will refer to as "candidate sensors". After that, a contention-based workforce selection mechanism is used to recruit required workforce based on sensors remaining energy. The workforce selection process involves one or more bidding rounds to be coordinated by the nearest backbone sensor. After collecting the required workforce, task execution starts immediately by the recruited sensors. Task execution time might vary based on the type of the sensing task and the sensor itself. When the sensors complete the execution of their tasks, they start sending their sensory data back to the nearest backbone sensor in the same order they joined the workforce. In the next few sections, we provide the technical details of each of these steps.

IV.1.2 CTW Messages

Recall that in our tasking model, we assume that the TIE sends a sequence of k CTW messages to attract the attention of a sufficient number of sensors for the next task. An important parameter that the TIE has to evaluate is the parameter k.

Assuming that the network density is ρ $sensors/m^2$, the sensing area of any task is πR^2, and so the expected number of sensors in the sensing area of a given task is given by $N = \pi \rho R^2$. Furthermore, assuming that the probability that a sensor is awake to receive a CTW message is p, it is easy to see that the expected number of awake sensors collected by the first CTW message is Np. On the second CTW message, the probability that a sensor is awake remains p, however the number of remaining sensors becomes $(N - Np)$, hence the expected number of sensors collected by the second CTW message is

$$(N - Np)p = (1 - p)Np$$

Similarly, on the third CTW message, the expected number of collected sensors is

$$(N - Np - (1 - p)Np)p = (1 - p)^2 Np$$

An easy inductive argument shows that on the k^{th} CTW message, the expected number of collected sensors is given by

$$(1 - p)^{k-1} Np$$

Consequently, the expected value of the total number of collected sensors at the end of k CTW messages is

$$\begin{aligned}\text{Collected Sensors} &= \sum_{i=0}^{k-1}(1-p)^i Np \\ &= \rho\pi R^2 p \cdot \sum_{i=0}^{k-1}(1-p)^i \\ &= \rho\pi R^2 p \cdot \frac{(1-p)^k - 1}{(1-p)-1} \\ &= \rho\pi R^2 \cdot \left[1-(1-p)^k\right] \qquad (17)\end{aligned}$$

The TIE can use equation (17) to estimate k, the number of CTW messages needed to attract the attention of c candidate sensors as follows

$$\begin{aligned}c &\geq \rho\pi R^2 \cdot \left[1-(1-p)^k\right] \\ (1-p)^k &\geq 1 - \frac{c}{\rho\pi R^2} \\ k &\geq \frac{\log(1-\frac{c}{\rho\pi R^2})}{\log(1-p)} \\ k &= \left\lceil \frac{\log(1-\frac{c}{\rho\pi R^2})}{\log(1-p)} \right\rceil \qquad (18)\end{aligned}$$

At this point it is important to observe that c does not represent the number of sensors required for the task; instead, c represents the number of candidate sensors from which the required workforce w is to be selected. Hence, if the TIE wants to collect a workforce of w sensors for the next task, it substitutes c in equation (18) by $c = f(w) \geq w$ (the derivation of the function $f(w)$ is presented later). This way the protocol probabilistically attracts the attention of more than w candidate sensors. From these sensors, only w sensors will be selected based on the difference in sensor remaining energy.

Next, we show how the TIE estimates p, the probability that a sensor is awake. Assume that a sensor during its whole life and before its energy is totally depleted goes through m sleep/awake duty cycles. We assume that in each cycle, a sensor sleeps for a random amount of time uniformly distributed in the range $[T_s, T_S]$ and stays awake for a random amount of time uniformly distributed in the range $[T_l, T_L]$. If m, the total number of cycles, is sufficiently large, then the expected total awake time, the expected total sleeping time, and the expected total lifetime of a sensor can be expressed as

$$\begin{aligned}
\text{Awake Time} &= \frac{m(T_l + T_L)}{2} && (19)\\
\text{Sleep Time} &= \frac{m(T_s + T_S)}{2} && (20)\\
\text{Life Time} &\doteq \frac{m(T_l + T_L)}{2} + \frac{m(T_s + T_S)}{2} \\
&= \frac{m(T_l + T_L + T_s + T_S)}{2} && (21)
\end{aligned}$$

From (19) and (21), we can evaluate p by writing

$$p = \frac{\frac{m(T_l+T_L)}{2}}{\frac{m(T_l+T_L+T_s+T_S)}{2}} = \frac{T_l + T_L}{T_l + T_L + T_s + T_S} \qquad (22)$$

Interestingly, the expression in (22) shows that the probability that a sensor is awake is independent of time.

IV.1.3 Workforce Selection

Workforce selection starts immediately after the last CTW message through one or more bidding rounds. A bidding round is a contention-based mechanism used to select a subset of sensors from a larger set based on a certain criterion. Each bidding round has a number of bidding slots which is explicitly specified in CTW messages or within bidding result messages of the previous rounds. Candidate sensors willing to participate in a task show their interest by bidding randomly in one of the bidding slots. The task coordinator (the nearest backbone sensor to the task center) is responsible for coordinating the bidding process (i.e. it announces the winning bidders at the end of each round, determines whether there is a need for another bidding round and announces the number of bidding slots in the next bidding round). Only single-bidder slots are considered winning slots. Once the task coordinator announces the winning slots in a round, the winning bidders (sensors which bid on any of the winning slots) immediately join the workforce. If the required workforce is not fully recruited, bidding continues for another round. In the new bidding round, not only losers in the previous round are eligible to bid but also additional sensors who received the bidding results message but were asleep during the CTW stage can place their bids. If the bidding extends beyond the maximum number of bidding rounds allowed, there are two options available: (1) cancel the task, (2) execute the task with the currently recruited workforce. Either way the TIE must be informed that the required level of Qos will not be satisfied.

Immediately after the last CTW message (for the first bidding round) or the bidding result message (for the following bidding rounds), the time line is divided into a number of bidding slots; each bidder selects one of these slots at random and transmits a short frame that contains the sensor current energy level and the slot number (to avoid any confusion due to the lack of accurate synchronization between sensors).

Candidate sensors participate in bidding with probabilities that are proportional to the difference between their current energy and the maximum energy among candidate sensors E_{max}. TIEs send their estimate of the value of E_{max} in the neighborhood of the required task within CTW messages. In the first bidding round of the first task within a certain area, the TIE does not know E_{max}, so candidate sensors participate in bidding with probabilities $\frac{1}{2}$ (we justify for this choice later). For subsequent rounds, the task coordinator can estimate E_{max} from the bids received so far and transmits the estimated value of E_{max} within the bidding results message. This value is also transmitted with the aggregated result to the TIE which uses the received values to update an internal two dimensional matrix that keeps track of the current estimate of E_{max} in different regions of the network.

Each bidding slot can have zero, one, or multiple bids. Slots with no bids are useless while those with multiple bidders result in garbled messages and also are useless and ignored. Only messages in single-bidder slots can be received correctly. At the end of each bidding round, the task coordinator calculates the number of single-bidder slots G. If G is greater than the required workforce w, then the task coordinator selects required workforce by adding sensors based on their remaining energy starting by those with higher energy first. If G is less than w, then it selects all the winning sensors and announces for another bidding round to collect the remaining workforce.

Each bidding result message contains a vector v that has s elements corresponding to the s bidding slots in the preceding bidding round. For bidding slot i, the corresponding element $v[i]$ is set to 0 to indicate that the bidding sensor was not selected at this round, otherwise $v[i]$ is set to a non zero temporary ID to indicate that the bidding sensor was selected by the task coordinator to join the workforce. A sensor starts task execution immediately after it joins the workforce. When sensors finish the execution of the required task and based on their temporary IDs they sequentially send their results to the task coordinator for local data aggregation. Finally, the task coordinator sends the aggregated results to the TIE for further processing if necessary.

Next, we show how the number of bidding slots in any bidding round is estimated. Assuming that for a general bidding round we have n bidders and s slots, we calculate the expected number of single-bidder slots in this round. We assume that a bidder can bid on any slot with equal probability, more precisely $\frac{1}{s}$, and since we have n bidders that bid independently of each other, the probability that a specific slot has only one bidder equals $n(\frac{1}{s})(1-\frac{1}{s})^{n-1}$. Since we have s slots, the expected number of single-bidder slots, G, can be expressed as

$$E[G] = n\left(1-\frac{1}{s}\right)^{n-1}$$

Recall that, in our workforce selection only single-bidder slots are counted. Hence, maximizing the number of these slots will definitely reduce the number of bidding rounds needed. Since n is a discrete variable, we define the continuous variable x such that $x = n$ for all the values of n. Now, we can differentiate $E[G]$ with respect to x as follows.

$$\begin{aligned} E[G] &= x\left(1-\frac{1}{s}\right)^{x-1} \\ \frac{\partial E[G]}{\partial x} &= \left(1-\frac{1}{s}\right)^{x-1}\left[1+x\cdot\ln\left(1-\frac{1}{s}\right)\right] \end{aligned}$$

The maximum value of $E[G]$ occurs when its first derivative equals 0.

$$\begin{aligned} \frac{\partial E[G]}{\partial x} &= \left(1-\frac{1}{s}\right)^{x-1}\left[1+x\cdot\ln\left(1-\frac{1}{s}\right)\right] = 0 \\ 1 &= -x\cdot\ln\left(1-\frac{1}{s}\right) \\ 1 &= \ln\left(1-\frac{1}{s}\right)^{-x} \\ e &= \left[\left(1-\frac{1}{s}\right)^{-s}\right]^{\frac{x}{s}} \\ e &\approx (e)^{\frac{x}{s}} \Rightarrow x \approx s \approx n \end{aligned} \tag{23}$$

Obviously, the approximation in equation (23) is valid for large values of s. Moreover, equation (23) shows that the maximum value of $E[G]$ occurs when the number of bidders n is equal to the number of bidding slots s and that the maximum number of single-bidder slots expected is given by

$$E[G]_{max} = s \cdot (1 - \tfrac{1}{s})^{s-1}$$

For a task that requires a workforce of size w, the required number of single-bidder slots is also w. Based on this, we can determine the number of slots to use as follows,

$$\begin{aligned} G = w &= s\left(1 - \frac{1}{s}\right)^{s-1} \\ w &\approx \frac{s \cdot e^{-1}}{\left(1 - \frac{1}{s}\right)} \\ e \cdot w &\approx \frac{s^2}{s-1} \\ s^2 - e \cdot w \cdot s + e \cdot w &\approx 0 \\ s &\approx \frac{e \cdot w\left(1 + \sqrt{1 - \frac{4}{e \cdot w}}\right)}{2} \end{aligned} \tag{24}$$

For $w > 3$, $(1 + \sqrt{1 - \frac{4}{e \cdot w}}) \approx 2$, hence equation (24) can be simplified to

$$s \approx e \cdot w \tag{25}$$

Next, we turn our attention to the relation between n, the number of bidders, and c, the number of candidate sensors. Previously, we mentioned that each candidate sensor participates in bidding with probability that is proportional to the difference between its energy (i.e E_s) and the maximum energy among all candidate sensors (i.e E_{max}). In particular, a candidate sensor with energy level E_s should bid with probability $\frac{1}{1+E_{max}-E_s}$. Assuming that the energy of candidate sensors is uniformly distributed across l consecutive levels, the expected number of bidders can be related to the number of candidate sensors as follows,

$$\begin{aligned} E[n] &= \sum_{i=0}^{l-1} p_i \cdot c_i = \sum_{i=0}^{l-1} \frac{1}{1+i} \cdot \frac{c}{l} \\ &= \frac{c}{l}\left[1 + \frac{1}{2} + \frac{1}{3} + \dots \frac{1}{l}\right] \\ &= \frac{c \cdot H_l}{l} = \frac{c \cdot (\ln(l) + \gamma)}{l} \end{aligned}$$

Solving for c, yields

$$c = \frac{E[n] \cdot l}{\ln(l) + \gamma} \approx \frac{n \cdot l}{\ln(l) + \gamma} \tag{26}$$

where $\gamma \approx 0.57721$ is Euler's constant. The TIE uses equation (26) to estimate the required number of candidate sensors it has to collect through CTW messages in order to select the workforce for the next task. The parameter l in equation (26) reflects the average width of sensor energy spectrum at different points in network lifetime. Since the task management protocol is designed to balance energy expenditure among sensors by minimizing variations in their energy, it is expected that the value of l will remain small most of the time. Simulation results verified this expectation and showed that l did not exceed 6 levels in all our experiments. More details are presented in Section IV.4.

Recall that in Subsection IV.1.3, in the presence of an unknown value for E_{max}, the participation probability of sensors was taken to be $\frac{1}{2}$. Equation (26) can be used to justify our choice for this value as follows. Equation (26) expresses the ratio between the number of bidders to the number of candidate sensors as $\frac{\ln(l)+\gamma}{l}$. The average value of this ratio for small values of l (i.e ≤ 6) is $0.515 \approx 0.5$. Hence, at the very early stage of the network lifetime when the differences between sensor energy levels are minor (i.e l is small) and when E_{max} is unknown, if each of the candidate sensors participates in bidding with probability of 0.5, then the expected number of bidders will be close to n justifying our choice.

IV.2 DISTRIBUTED TASK ASSIGNMENT

A major problem with the centralized approach described in Section IV.1 is the excessive load it imposes on backbone sensors for coordinating workforce selection and data aggregation. Several techniques can be used to overcome this problem. For instance, we can use "Backbone Switching", in which several alternative backbones are constructed instead of having only a single backbone. Switching periodically between alternative backbones distributes the load on network sensors and gives exhausted sensors a chance to rest. Another way to solve this problem is to use "Task Management Delegation", in which the coordination needed for a specific task is treated like a task by itself and is delegated to one of the network sensors that is chosen to be with relatively high energy.

In addition to the two previous techniques, we propose a distributed version of our workforce selection protocol. In this version, we adopt the tasking model depicted in Figure 23.

The model is similar to the tasking model we used earlier in the centralized case

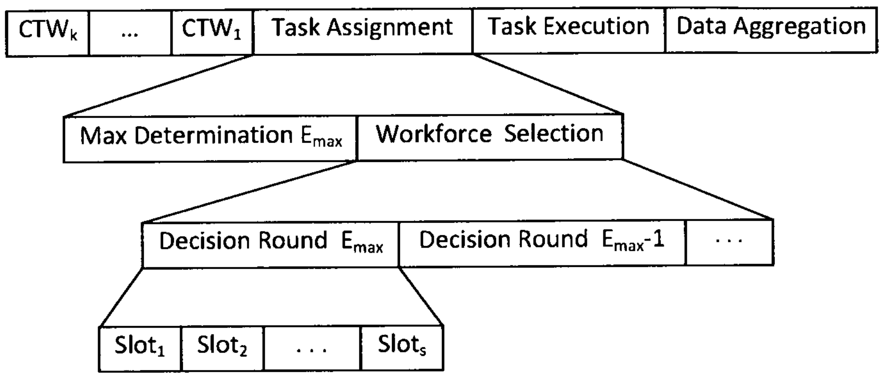


FIG. 23: *Tasking model for the distributed protocol*

with some differences in the task assignment stage and the way data is aggregated. In the distributed tasking model, task assignment consists of two phases. In the first phase, candidate sensors collected in the CTW stage run a distributed protocol to determine the maximum energy among themselves (see Subsection IV.2.1). In the second phase, each sensor decides whether or not to participate in the current task based on: (1) the difference between its current energy and the maximum energy determined in the first phase; (2) its distance to the position of the task center (see Subsection IV.2.2).

Our previous assumptions about the capabilities of sensors still hold. In the following subsections, we present the details of the different phases of the task assignment protocol.

IV.2.1 Phase 1: Estimating the Maximum Energy

The main goal of this phase is to run a fully distributed protocol in order to determine E_{max}, the maximum energy among collected candidate sensors. Assume that sensor energy E_s can be quantized into 2^n levels (i.e. E_s can be encoded in a string of n bits). The idea is to let candidate sensors transmit the strings that represent their energy levels bit by bit in a sequence of n short packets (n iterations). The time line is divided into n slots, and sensors start the encoding process immediately after the last CTW message [1]. Sensors start their transmission from the most significant bit to the least significant bit as follows: a value of 0 is not transmitted while a value

[1]We assume that CTW messages include information about the number of remaining CTW messages.

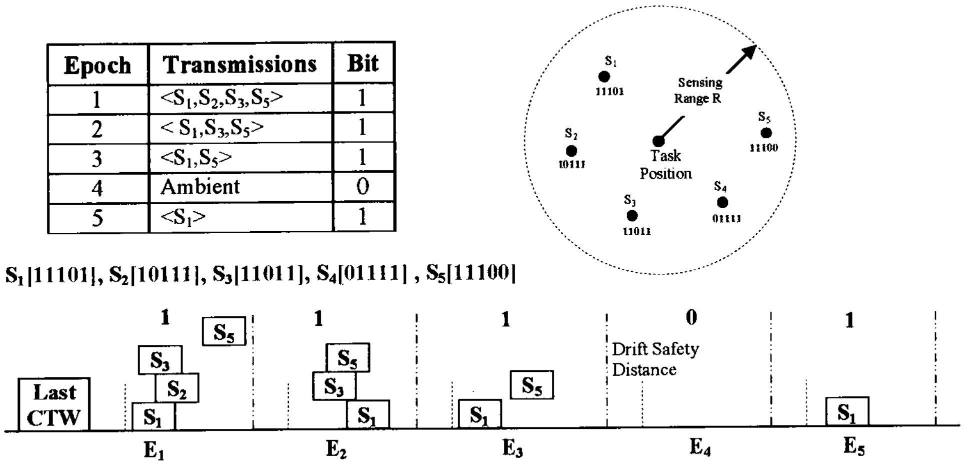
of 1 is transmitted. Sensors that pick up the values transmitted use the following disambiguation scheme:

- No packets received: 0 is recorded;
- A single packet received: 1 is recorded;
- Two or more packets received or a collision detected: 1 is recorded.

A sensor drops out if the binary representation of its energy has a 0 in its k^{th} most significant bit, and it detected a collision or received one or more packets in the k^{th} iteration (time slot). At the end, each sensor in the sensing area stores the maximum energy among all candidate sensors in the sensing area. Note also that there is no loss of information in the process of estimating the maximum.

The reader might argue that we cannot trust the synchronization achieved using the last CTW message because packets received by different sensors suffer from different propagation and processing delays. Although this seems to be true, we still can argue that the achieved level of synchronization is more than sufficient for our purpose especially when there is no actual payload (data) in the packets transmitted. Detecting a collision is equivalent in its interpretation to receiving a packet. Hence, if the iteration slot length is T, and the transmission time to send any of these small packets is T_t, the only way in which this protocol fails is when a sensor is delayed for a period longer than $T - T_t$. In this case its string is transmitted and interpreted shifted by one or more bits. However, since the slot time T can be chosen arbitrarily, we can choose T such that the probability that a sensor will be delayed longer than $T - T_t$ is very small, especially when the sensors are within the same sensing area.

Next, we show an example of how this protocol works. Referring to Figure 24, we assume a scenario where there are 5 sensors (i.e S_1, S_2, S_3, S_4 and S_5) that lie within the sensing area of a specific task. We assume that the respective energy levels of these sensors are 11101, 10111, 11011, 01111 and 11100. To determine the maximum, we need 5 iterations. In the first iteration, only sensors S_1, S_2, S_3 and S_5 transmit. A collision is recorded and all the sensors set the most significant bit of the perceived maximum to 1. Moreover, sensor S_4 realizes it should drop out since its most significant bit is a 0. In the second iteration, only sensors S_1, S_3, and S_5 transmit. Again a collision is recorded, and all the sensors set the second most significant bit to 1. Sensor S_2 drops out. In the third iteration, only sensors



Epoch	Transmissions	Bit
1	$<S_1,S_2,S_3,S_5>$	1
2	$< S_1,S_3,S_5>$	1
3	$<S_1,S_5>$	1
4	Ambient	0
5	$<S_1>$	1

FIG. 24: *Estimation of the maximum energy among candidate sensors*

S_1 and S_5 transmit. All the sensors set the third most significant bit to 0. Sensor S_3 remains inactive for the remaining iterations. In the fourth iteration, there is no transmissions, so 0 is recorded. Finally, in the fifth iteration, only sensor S_1 transmits. Sensors set the least significant bit of the perceived maximum to 1 and reach the consensus that the maximum energy is 11101.

It is worthwhile to mention that when sensors know the maximum energy within each sensing area, they can benefit from this in many different ways. For example, this knowledge allows each sensor to adjust its duty cycle (the ratio between its sleep and awake time) based on the difference between its remaining energy E_s and the maximum energy in its sensing area E_{max}. As a result, sensors with relatively low energy ($E_i < E_{max}$) can sleep for longer periods than sensors with relatively high-energy. More details about this feature is presented in Chapter V.

IV.2.2 Phase 2: To Participate or Not to Participate

After evaluating the maximum energy in the sensing area, each sensor has to decide whether or not it is going to participate in the task at hand. Each sensor makes this decision based on the difference between its energy E_s and the maximum energy E_{max} determined in the previous phase. Since our protocol is fully distributed and there is no central node to coordinate sensor participation, sensor decisions have to be taken independently of each other. Unfortunately, under these conditions we cannot

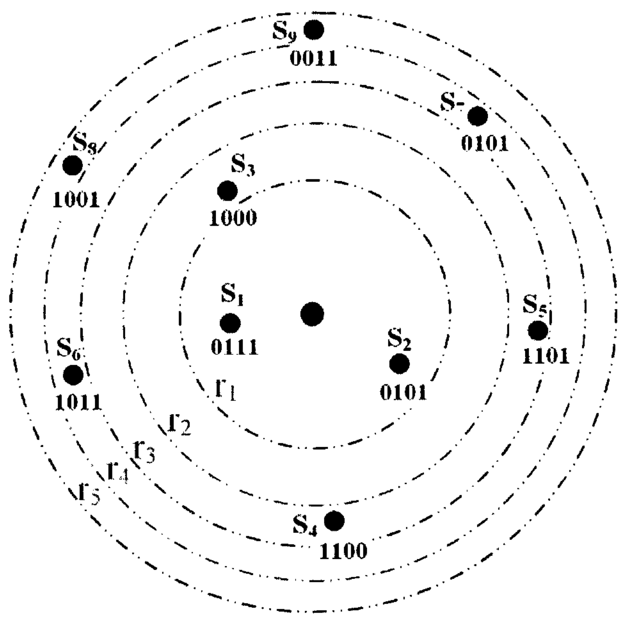


FIG. 25: *Dividing the sensing range into k disjoint subregions*

guarantee that the number of participating sensors matches the required workforce. The best we can do while keeping our protocol distributed is to keep the actual number of recruited sensors as close as possible to the required workforce.

To achieve this goal we divide the task sensing region into k disjoint subregions of equal size using concentric circles of radii $r_1 < r_2 < \cdots < r_k = R$. Based on the task center position and its own position, each sensor can determine its subregion (see Figure 25). Participation decisions are taken in decision rounds that run immediately after estimating the maximum energy in the sensing area. Each decision round is associated with an energy level that determines the set of sensors that can join the workforce.

Specifically, in the first decision round, only sensors with energy equals to E_{max} are allowed to transmit, in the second decision round, only sensors with energy equals to $E_{max} - 1$ are allowed to transmit and so on. The packets transmitted by sensors are very short and contain no payload, the payload is implicitly encoded in the transmission itself. Each decision round has k time slots corresponding to the k subregions described above. In slot i in decision round j, only sensors that are in subregion i with energy level equals j are allowed to transmit.

Each sensor initializes an internal counter to 0 and waits for the decision round corresponding to its energy and the time slot corresponding to its subregion. It is possible that the protocol collects the required workforce and terminates before the decision round and slot of a sensor comes. The idea of the protocol is to recruit sensors one by one based on their energy and using subregions to reduce the number of sensors that are being added in each step (only one sensor at each step if possible). The sensors that pick up the packets transmitted use the following scheme to update their counter:

- No packets received: no action is taken;
- A clear packet received: increment counter by 1;
- A collision recorded: increment counter by 2.

The protocol terminates when the internal counter of a sensor is greater than or equals the required workforce. Only sensors that had the chance to transmit during the participation phase join the workforce. To guarantee that the protocol terminates in a finite number of decision rounds, there should be a maximum number of decision rounds that the protocol allows. If the last decision round is reached, then all the sensors that are in the sensing area and have not already transmitted, transmit in their corresponding slot irrespective of their energy level.

Immediately after joining the workforce, sensors start the execution of their assigned tasks. When sensors complete the execution of their tasks, they send their results to the nearest backbone sensor in the same order they joined the workforce. After the nearest backbone sensor collects the data from all the sensors, it aggregates the collected results and forwards their aggregate to the TIE for further processing. Because more than one sensor can join the workforce at the same time, collisions between sensor transmissions during data aggregation might occur (i.e sensors will have the same transmission turn during data aggregation). In order to coordinate between sensor transmissions during data aggregation, we suggest that when their transmission turn comes, each of the competing sensors delays its transmission by a random amount of time uniformly distributed in the range $[0, Max]$. When its delay timer expires, a sensor transmits its result immediately. If the message is transmitted successfully, other sensors reset their delay timers to another random value and wait for their new timers to expire in order to be able to transmit. When a collision is detected, only colliding sensors reset their delay timers, non-colliding sensors should not reset their timers. If Max time units pass since the last successful transmission, then this indicates that all competing sensors have successfully transmitted their data to the backbone sensor. At this point of time, sensors in the next turn should start to send their data.

The radii that divide the sensing area into subregions are determined in such a way that the number of sensors in these subregions are as even as possible. The best way to do this under uniform distribution is to choose these radii such that the areas of the subregions are equal. Mathematically, this can be expressed as follows,

$$\begin{aligned} \pi \cdot (r_i^2 - r_{i-1}^2) &= \pi \cdot (r_{i-1}^2 - r_{i-2}^2) \\ r_i^2 &= 2r_{i-1}^2 - r_{i-2}^2 \end{aligned} \tag{27}$$

substituting in (27) for $r_{i-1}, r_{i-2} \cdots$

$$\begin{aligned} r_i^2 &= 3r_{i-2}^2 - 2r_{i-3}^2 \\ r_i^2 &= 4r_{i-3}^2 - 3r_{i-4}^2 \\ &\vdots \end{aligned}$$

A simple inductive argument shows that

$$\begin{aligned} r_i^2 &= i \cdot r_1^2 - (i-1)r_0^2 \\ r_i &= \sqrt{i} \cdot r_1 \end{aligned} \tag{28}$$

If the entire sensing area is divided into k subregions of equal area (i.e. $\frac{\pi R^2}{k}$), then

$$\begin{aligned} R = r_k &= \sqrt{k} \cdot r_1 \\ r_1 &= \frac{R}{\sqrt{k}} \end{aligned} \tag{29}$$

Finally, substituting the value of r_1 from equation (29) into equation (28) yields

$$r_i = \sqrt{\frac{i}{k}} R \tag{30}$$

IV.2.3 Average Over-Recruited Workforce

After determining the radii of the subregions, we estimate the average number of over-recruited workforce by evaluating the probabilistic distribution of the number of sensors in each subregion and show how it changes with the number of subregions

k. Assuming a uniform distribution of sensors in the sensing area, we can map the problem to the classical "balls and bins" problem in which n balls are distributed uniformly at random into k bins. In our scenario, we have n sensors that are deployed randomly in k subregions. The reader should note that here n refers to the number of sensors in the sensing area with energy level that matches the energy level determined by the decision round. Since sensors in the sensing area may have different energy levels, n only represents a fraction of the total number of candidate nodes collected in the sensing area.

The event that a given sensor will be deployed in a particular subregion is a Bernoulli trial with probability of success equal to the ratio between the region area to the whole sensing area (i.e. $\frac{1}{k}$ since the subregion areas are equal). Thus, the random variable X_i which represents the number of sensors in subregion i follows a Binomial distribution $B(n, \frac{1}{k})$.

$$Pr[\{X_i = j\}] = \binom{n}{j} \left(\frac{1}{k}\right)^j \left(\frac{k-1}{k}\right)^{n-j}$$

We are interested in evaluating the probability that more than two sensors fall in the same region because this may result in recruiting more sensors than the required size of the workforce. The sought probability is

$$\begin{aligned} Pr[\{X_i > 2\}] &= 1 - Pr[\{X_i \le 2\}] \\ &= 1 - \sum_{j=0}^{2} \binom{n}{j} \left(\frac{1}{k}\right)^j \left(\frac{k-1}{k}\right)^{n-j} \end{aligned} \tag{31}$$

To simplify our notation we define $\alpha_i(n, k)$ as follows

$$\begin{aligned} \alpha_i(n,k) &= \sum_{j=0}^{i} \binom{n}{j} \left(\frac{1}{k}\right)^j \left(\frac{k-1}{k}\right)^{n-j} \\ &= \frac{1}{k^n} \sum_{j=0}^{i} \binom{n}{j} (k-1)^{n-j} \\ \alpha_1(n,k) &= \frac{1}{k^n} \left[(k-1)^n + n(k-1)^{n-1}\right] && (32) \\ \alpha_2(n,k) &= \frac{(k-1)^{n-2}}{k^n} \left((k-1)^2 + n(k-1) + \frac{n(n-1)}{2}\right) && (33) \end{aligned}$$

By substituting (33) in (31) we obtain

$$Pr[\{X_i > 2\}] = 1 - \alpha_2(n,k) \tag{34}$$

We are no ready to evaluate the expected size of the extra workforce recruited for a given task. For this purpose, we define the random variable Y_i that represents the number of extra workforce in subregion i and also define $Y = \sum_{i=1}^{k} Y_i$, the size of extra workforce in the whole sensing area. Easy manipulations show that

$$\begin{aligned}
E[Y_i] &= \sum_{j=0}^{n-2} j \cdot Pr[\{Y_i = j\}] = \sum_{j=0}^{n-2} j \cdot Pr[\{X_i = j+2\}] \\
&= \sum_{j=1}^{n-2} j \cdot \binom{n}{j+2} \left(\frac{1}{k}\right)^{j+2} \left(\frac{k-1}{k}\right)^{n-(j+2)} \\
&= \sum_{l=3}^{n} (l-2) \cdot \binom{n}{l} \left(\frac{1}{k}\right)^{l} \left(\frac{k-1}{k}\right)^{n-l} \\
&= \frac{n}{k} \sum_{l-1=2}^{n-1} \binom{n-1}{l-1} \left(\frac{1}{k}\right)^{l-1} \left(\frac{k-1}{k}\right)^{(n-1)-(l-1)} - \\
&\quad 2\sum_{l=3}^{n} \binom{n}{l} \left(\frac{1}{k}\right)^{l} \left(\frac{k-1}{k}\right)^{n-l} \\
&= \frac{n}{k}\left[1 - \alpha_1(n-1,k)\right] - 2\left[1 - \alpha_2(n,k)\right]
\end{aligned}$$

By the linearity of expectation

$$\begin{aligned}
E[Y] &= \sum_{i=1}^{k} E[Y_i] = k \cdot E[Y_i] \\
&= n\left[1 - \alpha_1(n-1,k)\right] - 2k\left[1 - \alpha_2(n,k)\right] \qquad (35)
\end{aligned}$$

Figure (26) gives us more insight about how $E[Y]$ changes with changes in k and n. Obviously, as the number of subregions k increases the expected number of extra workforce recruited decreases to match the required workforce. It is also important to note that $E[Y]$ represents the expected number of extra workforce recruited assuming that the protocol continues to run in all the slots of the decision round. However this is not the case for many tasks in which the protocol terminates in the first few slots of the first decision round. So in fact $E[Y]$ is more like an upper bound on the extra workforce recruited.

IV.3 DATA AGGREGATION

As we mentioned earlier, the excessive load imposed on backbone sensors for coordinating workforce selection and data aggregation may result in depleting their energy

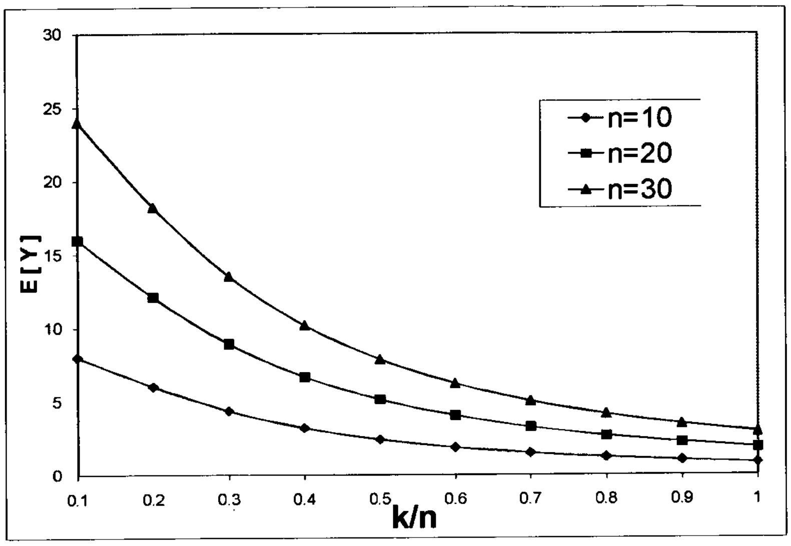


FIG. 26: $E[Y]$ *vs.* k/n

at a very high rate. In this section, we propose a set of distributed data aggregation techniques that can be used to aggregate sensory data collected by recruited sensors. In general, it is very difficult to devise efficient distributed protocols to evaluate a generic aggregate function. Fortunately, for certain functions like the MAX, MIN, and the Logical OR this can be done.

In section IV.2.1, we showed how the maximum can be obtained. By a simple trick we can use a similar approach to evaluate the minimum. For instance if X_i is the digitized reading of a sensor, instead of transmitting X_i directly, each sensor should transmit $V_i = N - X_i$, where N is a constant which is greater than the maximum possible value of the sensor reading X_i. After the last iteration, V_{max}, the maximum value transmitted should be known to all the sensors. Each sensor can evaluate the minimum as $X_{min} = N - V_{max}$.

A similar approach can be used to evaluate the logical OR of the values collected by different sensors in n iterations, where n is the number of bits used to encode these values. In this case, the disambiguation scheme should be as follows:

- Ambient noise: 0 is recorded;
- A single packet is received: 1 is recorded;
- Two or more packets are received or a collision is detected: 1 is recorded.

At the end, every recruited sensor in the sensing area stores the logical OR of the data collected by the recruited sensors.

Algorithm 2 Approximation of the most significant k bits of the average

```
1: $S = 0$
2: $N = 0$
3: for i=1 To $k$ do
4:     if (no packets are received) then
5:         continue
6:     end if
7:     if (a single packet is received) then
8:         $S = S + (i - 1)$
9:         $N = N + 1$
10:    end if
11:    if (collision is detected) then
12:        $S = S + 2 \cdot (i - 1)$
13:        $N = N + 2$
14:    end if
15: end for
16: if ($N = 0$) then
17:     $A_2 = 0$
18: else
19:     $A_2 = \lceil S/N \rceil$
20: end if
21: return $A_2$
```

Now, we show another protocol to approximate the average of the collected sensory data. Again, we assume that the sensory data can be encoded into a string of n bits. We further divide the n bits representation of each value into 2 parts, the first part consists of the least significant $n - k$ bits while the second part consists of the most significant k bits. Data aggregation occurs in 2 stages.

In the first stage, the logical OR of the least $n - k$ significant bits of each collected value is evaluated as an approximate of the least $n - k$ significant bits of the average function. We refer to the result of the logical OR operation as A_1 and this value can be evaluated in $n - k$ iterations as described above.

In the second stage, the most significant k bits of each collected value is written in the "Single-One representation" format. In this format, each value m, $\{0 \leq m < 2^k\}$ is represented by m zeros followed by a 1. Obviously, 2^k bits are needed to represent the first part. Table 1 shows the One-Bit representation of $0 \leq m < 2^3 = 8$.

After preparing the Single-One representation of the k most significant bits of

TABLE 1: Single-One Representations of $0 \leq m < 2^3 = 8$

m	Representation	m	Representation
0	00000001	1	00000010
2	00000100	3	00001000
4	00010000	5	00100000
6	01000000	7	10000000

each value (and in 2^k subsequent iterations from right to left), each sensor transmits a short packet only in the iteration corresponding to the position where it has 1 in its value. When sensors pick up the values transmitted, they implement Algorithm IV.3.

In Algorithm IV.3, A_2 represents an approximation of the most significant k bits of the average of the data collected by different sensors. The final value of the average approximation is $A_2 \cdot 2^{n-k} + A_1$ and can be evaluated in $n - k + 2^k$ iterations. Obviously, the accuracy of the obtained average depends on k which is chosen to be around 3 bits.

IV.4 SIMULATION RESULTS

Using C++, we built a WSN simulator that implements different workforce selection protocols. To the best of our knowledge we are the first to address workforce selection in WSN. Hence, we could not compare our protocols to other well-known protocols. Hence, we compare the performance of our proposed protocols to the performance achieved using ideal workforce selection in which the workforce is selected from sensors with the highest energy in the sensing range of the task being executed. In addition to this, we compare our protocols to energy-unaware or energy-neutral protocol which uses the same CTW and bidding rounds mechanisms described in our centralized protocol. The only difference is that the bidding decisions of candidate sensors are made irrespective of their remaining energy. This is different from our approach in which the estimate of the maximum energy among candidate sensors, E_{max}, is used to control the probability by which candidate sensors participate in bidding for the next task.

An important parameter which we need to consider when dealing with energy-neutral protocols is the "Participation Factor" (PF). PF defines the probability by which candidate sensors participate in bidding. In our simulation, we run several experiments assuming PF takes the values 0.5, 0.75, and 0.90.

Before presenting our results, we find it more appropriate to start by defining what we mean by the reliable-lifetime of the network. In a typical sensor network, sensors are deployed with a predetermined density ρ which is usually chosen in a way that satisfies QoS requirements expressed in terms of number of sensors participating in different sensing tasks. An appropriately chosen value of ρ can provide a reliable network performance by guaranteeing that enough sensors will be available to perform upcoming tasks with the required level of QoS. Unfortunately, sensors have limited and non-renewable energy budget, once the energy of a sensor is entirely depleted, it becomes non-operational and eventually the sensor density decreases. When sensor density goes below certain threshold, the number of sensors available may not be enough to satisfy QoS requirements of upcoming tasks and at this point the sensing results cannot be considered reliable anymore. Based on this, we define α-reliable lifetime of a network as the average number of tasks the network can perform till the sensor density goes below α of its initial value. For instance, the 0.1-reliable lifetime of a network with density 0.5 $sensors/m^2$, is the average number of tasks that can be performed on this network before the sensor density goes below 0.05 $sensors/m^2$.

We conducted several experiments to evaluate the performance of our proposed protocols. In our simulation, sensors were deployed uniformly at random in a square with side length $200m$. We used different sensor densities ranging from 0.3 to 1.5 sensors/m^2. The network has a single TIE which is placed at the origin $(0,0)$ and is responsible for tasking sensors across the deployment area. QoS requirements of generated tasks were expressed in terms of the minimum number of sensors needed to participate in each task. The required workforce for different tasks was selected randomly from the range $[1,20]$. We tried to balance tasking load on different spots of the network by selecting the positions of the task centers uniformly at random across the whole deployment area. Sensors were assumed to have a fixed sensing range of 10.0 meters, beyond this range sensor readings may not be reliable. Sensors sleep and wake up alternatively and asynchronously in a way that makes them active for only %10 of their lifetime. Initially, each sensor has exactly 30 units of energy (i.e. each sensor can at most participate in 30 tasks).

Figure 27 shows, for different network densities, the average number of CTW messages needed to get the attention of sufficient number of sensors in order to

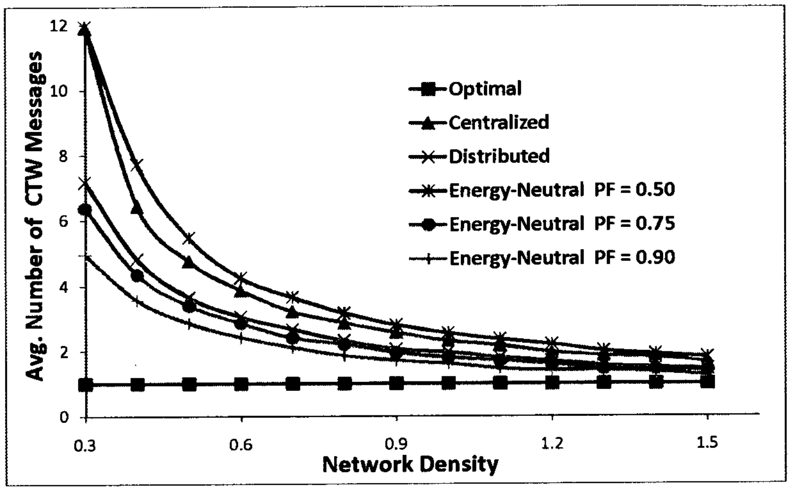


FIG. 27: *Average number of CTW messages*

execute the next upcoming task. The optimal protocol assumes that a single CTW message is enough. However, this is not the case for other protocols which substitute its specific estimate of the number of candidate sensors into equation (18) to evaluate the required number of CTW messages. Each of the shown protocols has its own way to estimate, c, the number of candidate sensors. The centralized protocol depends on equations (25) and (26) to estimate c. However, in the distributed protocol, c is evaluated by multiplying the workforce size, w, by some constant factor f_1 (in our simulation experiments, we assume $f_1 = 4$). The energy-neutral protocol uses equation (25) to estimate the number of bidders, n, from the workforce size. After that, it uses the participation factor to relate the number of bidders to the number of candidate sensors ($n = PF \cdot c$). From figure (27), we can see that the average number of CTW messages needed decreases as the network density increases with very minor differences between different protocols.

Figures 28 and 29, respectively, show the expected number of bidding rounds along with the expected number of bidding slots within each round. In the optimal scenario, the workforce selection protocol ends using a single bidding round which has a number of bidding slots that is equal to the size of the required workforce. However, for other protocols, typically more than one bidding round is needed to compensate for any empty or garbled slots that might arise due to bidding collisions (i.e when more than one sensor bid into the same slot). As the value of the participation

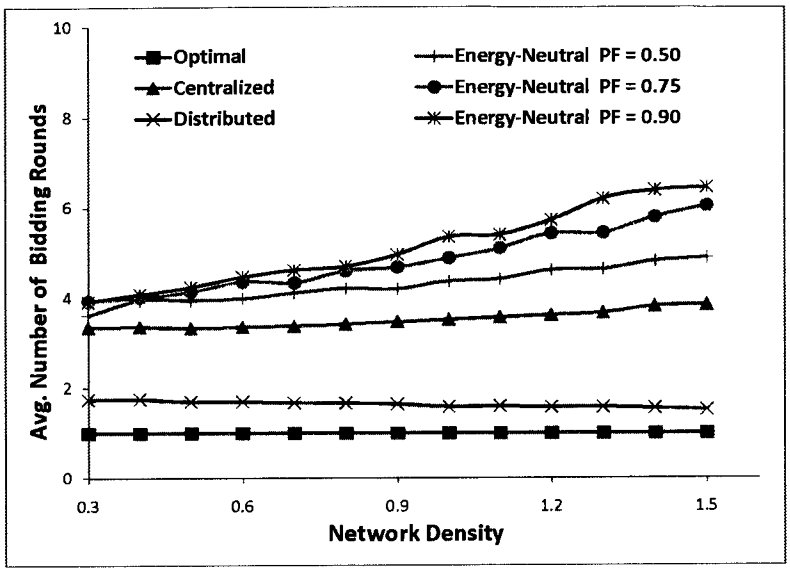


FIG. 28: *Average number of bidding rounds*

factor of energy-neutral protocols increases, the expected number of bidding rounds increases. This goes to the increase in the number of sensors willing to bid within the same round. Hence, the resulting number of single-bidder slots decreases and more rounds are needed to select the remaining workforce.

Figure 29 confirms that the average number of bidding slots used in the centralized protocol bidding rounds is very close to its counterpart in the energy-neutral protocol with very minor changes due to using different participation factors. It is also important to understand that the curve associated with the distributed protocol in figure 28 shows the number of decision rounds used in workforce selection since in this protocol there is no bidding. And for the same reason, the number of bidding slots for this protocol is always 0.

Figure 30 shows how our centralized and distributed protocols can preserve network density for longer periods by balancing the rate by which sensor energy is consumed. In particular, the figure compares the average width of sensor energy spectrum throughout the network 0.2-reliable lifetime for different network densities. To estimate the average width of sensor energy spectrum, we evaluated the width of

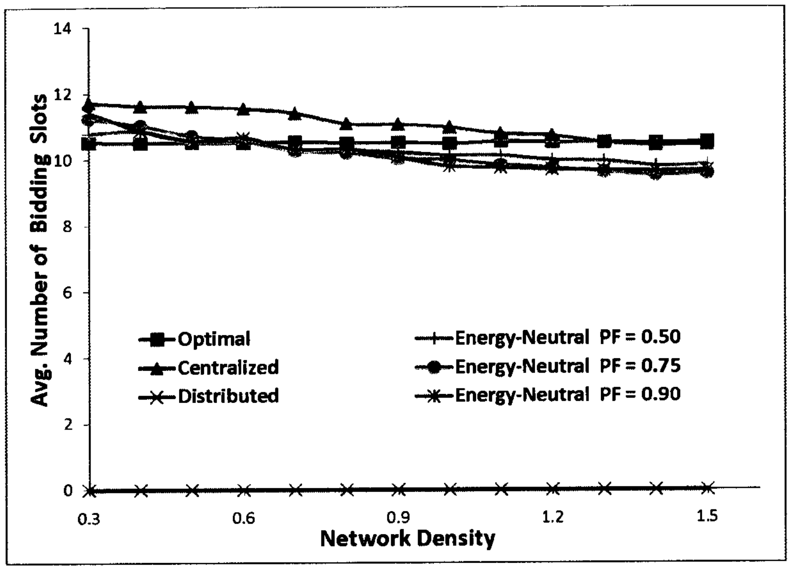


FIG. 29: *Average number of bidding slots in bidding rounds*

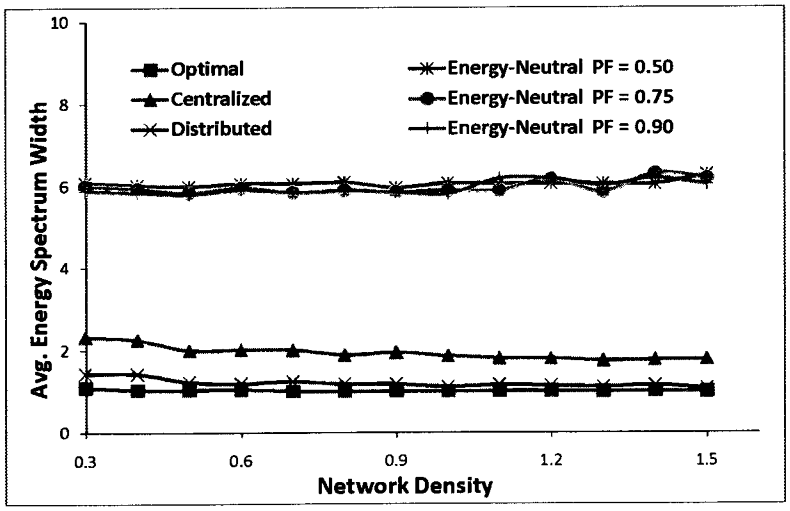


FIG. 30: *Average width of sensor energy spectrum*

the spectrum after the execution of every task using equation (36).

$$w_t = \begin{cases} \frac{1}{n_1}\sum_{s=1}^{n_1}\left(E_s - \overline{E}\right) & E_s \geq \overline{E} \\ \\ \frac{1}{n_2}\sum_{s=1}^{n_2}\left(\overline{E} - E_s\right) & E_s < \overline{E} \end{cases} \tag{36}$$

Where $\overline{E}$ is the average energy of sensors immediately after the execution of task t. n_1 is the number of sensors whose remaining energy is larger than or equal to $\overline{E}$. Similarly, n_2 is the number of sensors whose remaining energy is less than $\overline{E}$. Finally, the average spectrum width among all executed tasks was estimated as $\frac{1}{n}\sum_{t=1}^{n} w_t$, where n is the total number of tasks executed during the network lifetime.

From figure 30, we can see that the average width of sensor energy spectrum of the distributed protocol is much narrower than the width obtained when using any of the other protocols. Moreover, the average spectrum width is very close to its optimal value (i.e 1). The superior performance of the distributed protocol over other protocols can be attributed to: (1) The accurate estimation of the maximum energy among candidate sensors. (2) Selecting workforce using decisions rounds that give priority to sensors with higher energy over sensors with lower energy. Although, the average spectrum width of the centralized protocol is slightly wider than its counterpart in the distributed approach, it is much narrower than its value in other energy-neutral protocols. At this point, it is worthwhile to mention that in both of the centralized and the distributed protocols, the average spectrum width hardly changes with sensor initial energy. This is not the case for energy-neutral protocols, in which the average spectrum width increases when sensor initial energy increases as confirmed by Figure 31.

The large differences between the energy of sensors when using energy-neutral protocols makes one expect that many of the heavily loaded sensors would die out at an early stage of the network lifetime. Typically, when a large number of sensors which reside at some spot die out, the network density at this spot decreases. Figures 32-a, 32-b, and 32-c capture this phenomenon when using different workforce selection protocols with initial deployment densities of 0.3, 0.7, and 1.0 respectively.

As shown in figure 32, degrade in network density for energy-neutral protocols starts at relatively an earlier stage compared to other protocols. Table 2 lists the percentage of the network lifetime lived before the network density starts to degrade.

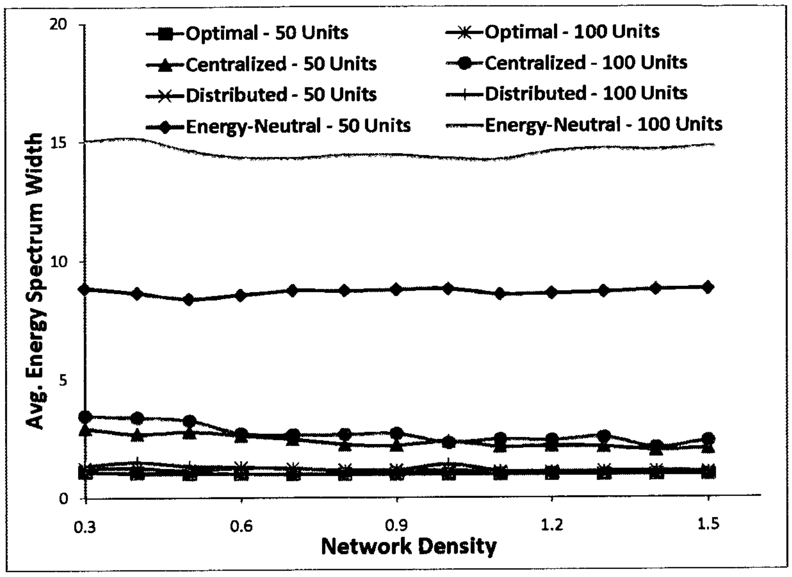
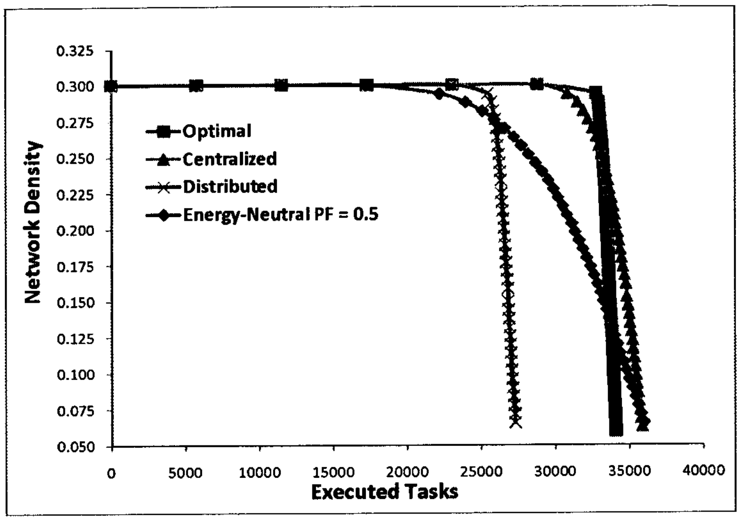


FIG. 31: *Avg. width of energy spectrum for different initial energy levels*

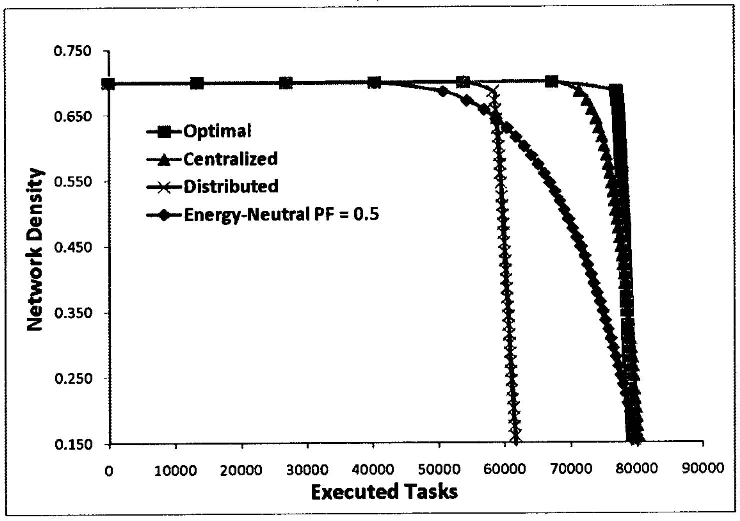
TABLE 2: Percentage of network lifetime lived before density degrades

Density	Optimal	Centralized	Distributed	Energy-Neutral
0.3	96.1	86.0	93.2	61.7
0.7	97.0	88.8	94.5	63.4
1.0	97.2	83.3	94.3	62.3
1.5	97.3	90.9	94.3	61.9

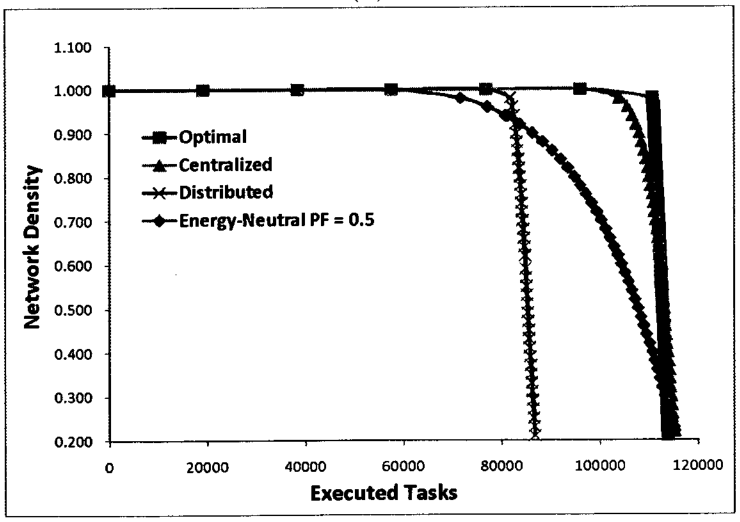
The continuous degrade in network density can eventually create energy holes. We conducted a set of experiments to capture the impact of using different workforce selection protocols on the rate by which holes grow up in the network. Figure 33 compares the growth rate of energy holes using the four protocols under different deployment densities. The initial deployment densities of figures 33-a,33-b, and 33-c are respectively 0.3, 0.7, and 1.0. The sharp slopes of the curves in figure 33 show that our centralized and distributed protocols can reduce the rate by which holes grow up in the network specially under small and medium network densities. Under dense deployments, our protocols tends to be less effective in reducing the growth rate of holes. In order to explain the reason behind this, we recall that energy holes grow up only when all the sensors within the hole are dead. Although the degrade



(a)



(b)

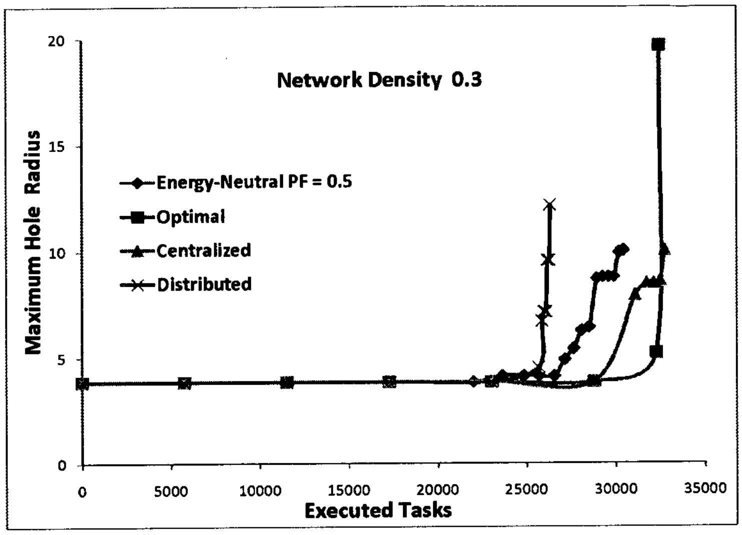


(c)

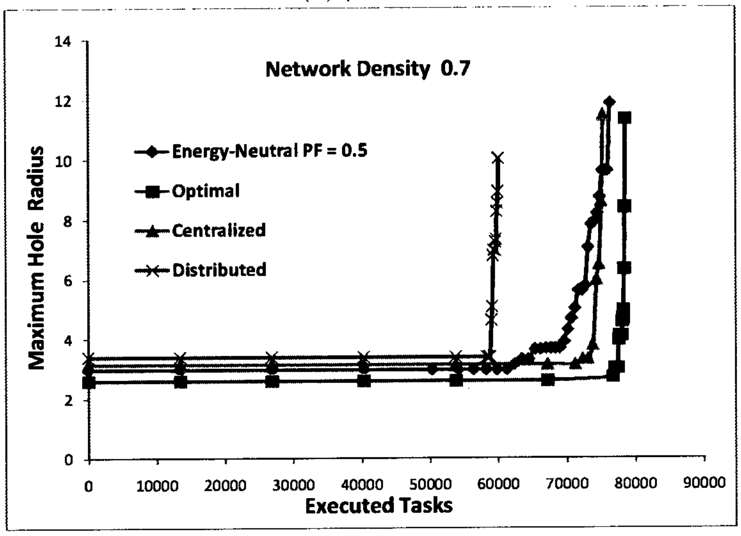
FIG. 32: *Average degrade in network density throughout its* 0.2*-reliable-lifetime using different protocols. (a)* $\rho = 0.3$*, (b)* $\rho = 0.7$*, (c)* $\rho = 1.0$

in network density when using energy-neutral protocols starts in an earlier stage, there is always a nonzero probability to find at least a single alive sensor that can restrict the growth of the energy holes. This probability gets higher under dense deployments which in turn delays the appearance and the growth of energy holes making our techniques less effective.

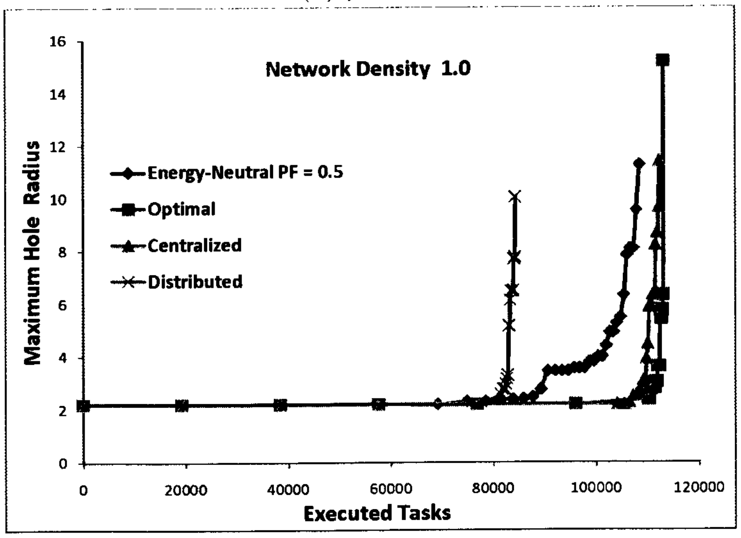
Figure 34-a compares the maximum number of tasks the network can execute throughout its 0.2-reliable-lifetime using the 4 different protocols. It is interesting to notice that some protocols are able to execute more tasks than the optimal protocol. To understand how this could happen, we recall our tasking model in which a sequence of CTW messages are transmitted by the TIE to attract the attention of sensors willing to participate in the execution of the next task. In some cases, especially at the late stages of the network lifetime, the number of collected sensors is less than the required size of the workforce as determined by QoS. Hence, those down-recruited tasks are executed using whatever was collected even if the collected number of sensors was less than what was specified in the CTW messages. For example, if at a late stage of the network lifetime all sensors within a certain spot died out except for a single sensor which has E units of energy remaining, then the network can assume falsely it can execute up to E additional tasks at this spot irrespective of the workforce size required by these tasks. Fortunately, the same scenario cannot happen using any of the optimal, centralized, or distributed approaches because of the minor differences between sensor energy. By the time the first sensor within a certain spot dies out, the remaining sensors within the same spot will be about to die out as well. Hence down-recruiting occurs for a small number of tasks. Our explanation is confirmed by the results we got in figure 34-b in which we show the number of tasks executed using the exact workforce size. From the figure, it is obvious that the optimal protocol has superior performance over other protocols. Although, the centralized approach is the closest protocol to optimal in performance, the distributed approach seems to perform poorly and even worse than energy-neutral protocols. The reason for this goes to over-recruiting. Because of the distributed nature of the protocol, sensors join the workforce independently, and as a result the number of recruited sensors may be larger than the required workforce size. Simulation results showed that for %38.3 of the total executed tasks, the size of the workforce recruited by the distributed protocol is larger than the required workforce size by around %17.4. We confirmed this by adding another curve to figure 34-b that shows



(a) $\rho = 0.3$



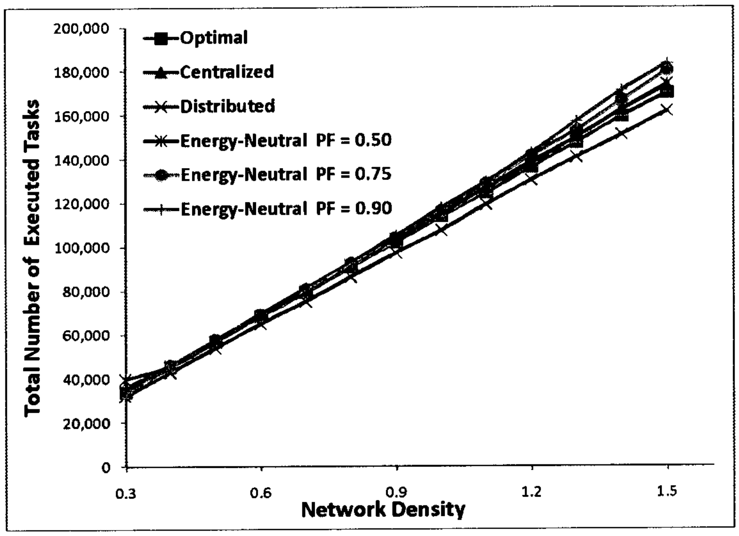
(b) $\rho = 0.7$



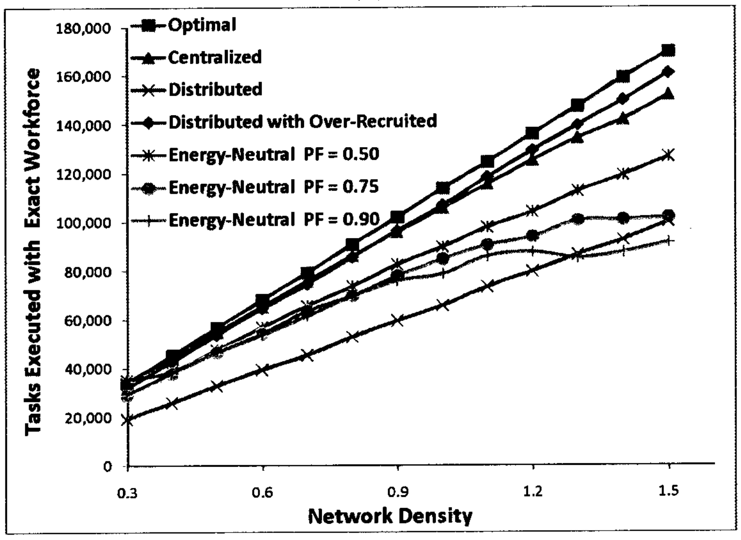
(c) $\rho = 1.0$

FIG. 33: *The growth rate of energy holes in the network* 0.2*-reliable-lifetime when using different workforce selection protocols and different network densities*

the total number of tasks executed using a workforce size that is at least as large as the required workforce size. Surprisingly, after adding over-recruited tasks, the performance of the distributed protocol has increased tremendously to the extent it became slightly better than the centralized protocol. On the average our centralized approach can increase the network 0.2-reliable lifetime by around %16.5 while our distributed approach can increase it by around %17.2.



(a)



(b)

FIG. 34: *(a) Total number of executed tasks (b) Total number of tasks executed using the exact workforce size*

CHAPTER V

SCHEDULING IN SENSOR NETWORKS

Sensors spend their entire lifetime switching between two modes: *sleep mode* where energy consumption is minimum and *awake mode* where energy consumption is relatively high. Due to their modest non-renewable energy, sensors spend most of their lifetime in sleep mode and wake up for short periods to participate in various tasks supportive of the overall mission of the network. Various deterministic and probabilistic schemes can be used to determine the schedule based on which sensors sleep and wake up. Each sleep/awake schedule has an impact on the *effective sensor density* (ESD), defined as the density of awake sensors. The ESD is an important network parameter because it is precisely the ESD that an application "sees" when it is launched. Consider for example, an intrusion event. The quality of the intrusion detection is a function of the number of awake sensors that witness the event. It is clear, therefore, that the network ESD is of fundamental importance to guarantee that sensing tasks are executed at their required level of QoS. In other words, it is necessary to have control over the ESD in order to satisfy QoS requirements expressed in terms of the minimum number of sensors needed to report monitored events.

Our main goal in this chapter is to show how our proposed backbone can help solving the scheduling problem in sensor networks. To achieve this goal, we start by conducting a rigorous mathematical analysis on ESD as seen from the perspective of the monitored events. We also provide design guidelines to determine deployment-time sensor density and an associated sleep schedule which probabilistically keeps the ESD at a level needed by QoS requirements. After that, we propose a backbone-guided fully distributed sleep schedule which adaptively adjusts the duty cycles of sensors within the same sensing area based on the relative difference in their remaining energy budget. To adjust its sleep schedule, each sensor must have access to up to date information about the minimum and the maximum sensor energy in its neighborhood. Through interaction with sensors in their neighborhood during workforce selection and data aggregation, backbone sensors get access to required information about sensors energy. After task completion and while forwarding the aggregated results to the sink node, backbone sensors attach the minimum and the maximum energy of sensors in its neighborhood as an extra field in the message payload. Sensors that receive the message, extract energy information and adjust their sleeping schedule accordingly. The main advantage of the proposed scheme is to balance energy consumption among sensors, thus promoting the functional longevity of the network, without changing the ESD.

We can summarize our contributions into the following:

(a) under the assumption that arrival of the events to be monitored is a Poisson process, we conduct a rigorous probabilistic analysis to prove that the well-known PASTA [44] property can be applied to the number of awake sensors (and, consequently, to ESD) as seen from the perspective of the monitored events. Specifically, we show that under a mild technical condition the limiting fraction of the events that find k awake sensors in a certain area is precisely the fraction of time that the area is under the surveillance of k awake sensors. This is a result of great importance as it justifies using the time-invariant probability distribution of k-coverage to analyze different scheduling schemes;

(b) we state the sleep-awake cycle of a sensor as a renewal process and, using the Key Renewal Theorem [45] derive a general expression for the limiting (i.e. time-independent) probability of a sensor to be awake at a given moment. To the best of our knowledge this is the first time such a result is obtained for sensor networks;

(c) we investigate different sleep scheduling schemes and their impact on ESD and the capability of the network to satisfy QoS requirements;

(d) we also look into how we can achieve better network lifetime by adjusting the sensor sleep schedule without reducing the ESD. In particular, in subsection V.3.3, we propose a sleep schedule scheme that prolongs the sleep time of sensors with relatively low energy and probabilistically compensates for their absence by shortening the sleep time of sensors with relatively high energy. An interesting by-product of our scheme is that the ESD, as perceived by the monitoring events, remains unchanged;

(e) finally and through simulation, we monitor sensor energy distribution during the network lifetime and show that using our energy aware scheduling approach can balance energy consumption among sensors. We also show that using the proposed scheme provides a substantial increase in reliable network lifetime when compared with static and dynamic scheduling schemes.

The remainder of this chapter is organized as follows: in Section V.1, we look into the number of awake sensors as a stochastic process then we prove that if the monitored events follow a Poisson process, then the PASTA property can be used to analyze the number of awake sensors as seen by the monitored events. In Section V.2 we cast the sleep/awake cycle of a sensor as a renewal process and derive a general expression for the limiting probability that a given sensor is awake at time t. Using the theoretical foundations laid down in Sections V.1 and V.2, in Section V.3, we evaluate the time independent probability distribution and the expected value of the number of awake sensors using different scheduling schemes including our proposed energy-aware scheduling scheme. In Section V.4, we show how our proposed backbone can be useful in solving the scheduling problem and propose several approaches by which sensors can have access to information needed to apply our scheduling scheme. Simulation results are summarized in Section V.5.

V.1 APPLICABILITY OF PASTA

Throughout this work, we take the view that the sensor network was deployed in support of detecting events of an unspecified nature. Moreover, we assume that QoS requirements stipulate that in the interest of *reliability* a minimum of k, $(k \geq 1)$, sensors must detect each event, where k is an application-specific parameter. Thus, in such a context, in order to meet the specifications of the QoS, a minimum of k sensors should be awake in each sensing neighborhood.[1] If an arriving event "finds" k awake sensors in its neighborhood, then it will be correctly detected - otherwise it will not. Thus, it is of great theoretical interest to look at the ESD of the sensor network as seen by the occurring events.

In this section, we show that if the events occur according to a Poisson process, then the PASTA [44] property (Poisson Arrivals See Time Averages) is applicable to the number of awake sensors available to detect the occurrence of these events. This result justifies analyzing the performance of the scheduling schemes presented in Section V.3 using the time-invariant probability distribution of the number of awake sensors in the sensing area centered at the location of the event.

[1]Some authors refer to this as k-coverage. To follow established terminology, we shall use the term k-coverage in the remainder of this work.

For a uniformly distributed random deployment of sensors where the sensing range of each sensor is r, we choose an arbitrary point p and define the *counting* process, $\{N(t), t \geq 0\}$, such that $N(t)$ represent number of awake sensors at time t in the disk D of radius r centered at p. We refer to $\{N(t) = k\}$ as "the process N is in state k at time t" whose semantic value is "disk D is k-covered at time t" where the maximality of k with this property is implied.

Assuming that the number of events arriving at the network is a Poisson process $\{A(t),\ t \geq 0\}$ with rate $\lambda > 0$, our objective is to compare the fraction of time the process $N(t)$ is in state k to the fraction of the events occurring in D that find the stochastic process $\{N(t), t \geq 0\}$ in state k. Put differently, we are interested in comparing the fraction of arriving events that occur while D is k-covered with the fraction of time disk D is k-covered.

For this purpose, define $I_k(t)$ as the indicator random variable of $\{N(t) = k\}$ at time t

$$I_k(t) = \begin{cases} 1 & \text{if } \{N(t) = k\} \\ 0 & \text{otherwise.} \end{cases}$$

In other words, $I_k(t)$ is 1 if and only if the number of awake sensors in D is k at time t. Recalling that we are interested in k-coverage (for some $k \geq 1$), to simplify the notation we shall drop the subscript k and write $I(t)$ instead of $I_k(t)$.

Define the random variable $F(t)$ as follows

$$F(t) = \frac{1}{t}\int_0^t I(s)\, ds. \tag{37}$$

The intention is for $F(t)$ to represent the fraction of time in $[0, t]$ that disk D is k-covered. Since $F(t)$ involves the Riemann integral of $I(t)$ on $[0, t]$, the definition of the integral, as a limit, allows us to approximate $F(t)$ for sufficiently large n by

$$F_n(t) = \frac{1}{t}\sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[\frac{(i+1)t}{n} - \frac{it}{n}\right] \tag{38}$$

with

$$\lim_{n\to\infty} F_n(t) = F(t). \tag{39}$$

For later reference we take note of the fact that $|F_n(t)|$ is bounded. Indeed,

$$\begin{aligned} |F_n(t)| &= F_n(t) \\ &\leq \frac{1}{t}\int_0^t ds \\ &= 1. \end{aligned} \tag{40}$$

An important consequence of (40) is that the Lebesgue Bounded Convergence Theorem applies to the sequence of random variables $\{F_n(t)\}$ and

$$\lim_{n\to\infty} E[F_n(t)] = E[F(t)]. \tag{41}$$

Similarly, let $a(t)$ be the random variable denoting the number of events occurring in disk D in the time interval $[0,t]$ that find the disk D k-covered. Formally, $a(t)$ is the Stieltjes integral of $I(t)$ with respect to $A(t)$. Thus, we can write

$$a(t) = \int_0^t I(s)\, dA(s).$$

By the definition of the Stieltjes integral, as a limit, it follows that $a(t)$ can be approximated for sufficiently large n by

$$a_n(t) = \sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[A\left(\frac{(i+1)t}{n}\right) - A\left(\frac{it}{n}\right)\right] \tag{42}$$

with

$$\lim_{n\to\infty} a_n(t) = a(t). \tag{43}$$

Next, we show that $|a_n(t)|$ is bounded. Indeed, we can write

$$\begin{aligned}
|a_n(t)| &= a_n(t) \\
&\leq a(t) \quad \text{[because } A(t) \text{ is monotone non-decreasing]} \\
&= \int_0^t I(s)\, dA(s) \\
&\leq \int_0^t dA(s) \quad \text{[because } I(s) \leq 1\text{]} \\
&= A(s)\Big|_0^t \\
&= A(t) - A(0) \\
&= A(t). \quad \text{[because } A(0) = 0\text{]}
\end{aligned} \tag{44}$$

Since it is well-known, and also very easy to prove from scratch, that the number $A(t)$ of Poisson arrival (i.e. events) in disk D in $[0,t]$ is bounded, the conclusion follows.

An important consequence of (44) is that the Lebesgue Bounded Convergence Theorem applies to the sequence of random variables $\{a_n(t)\}$ and, therefore,

$$\lim_{n\to\infty} E[a_n(t)] = E[a(t)]. \tag{45}$$

Before proceeding, we observe that the event arrival process A is independent of the number of awake sensors in disk D. Moreover, future increments of the events arrival process do not, in any way, depend on the current or any previous number of awake sensors in D. In other words, for each $0 \leq s \leq t, x \geq 0$

$$[A(t+x) - A(t)] \text{ and } I(s) \text{ are independent.} \tag{46}$$

This turns out to be a simple instance of the lack of anticipation assumption [44].

We are now ready to prove the first non-trivial result stating that in any finite interval $[0, t]$, the *expected* number of events that find the disk D k-covered equals the event arrival rate times the *expected* time during which D is k-covered.

Theorem V.1.1 *For all* $t \geq 0$

$$E[a(t)] = \lambda E\left[\int_0^t I(s)\, ds\right].$$

Proof 1 *We begin by evaluating* $E[a_n(t)]$. *First, by the linearity of expectation we have*

$$\begin{aligned} E[a_n(t)] &= E\left[\textstyle\sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[A\left(\frac{(i+1)t}{n}\right) - A\left(\frac{it}{n}\right)\right]\right] \\ &= \sum_{i=0}^{n-1} E\left[I\left(\frac{it}{n}\right)\left[A\left(\frac{(i+1)t}{n}\right) - A\left(\frac{it}{n}\right)\right]\right]. \end{aligned}$$

Since, by virtue of (46), the random variables $[A(t+x) - A(t)]$ *and* $I(s)$ *are independent for every choice of* $0 \leq s \leq t$ *and* $x \geq 0$, *we can write*

$$E[a_n(t)] = \sum_{i=0}^{n-1} E\left[I\left(\frac{it}{n}\right)\right] E\left[A\left(\frac{(i+1)t}{n}\right) - A\left(\frac{it}{n}\right)\right].$$

Recalling that Poisson processes have stationary increments, it is the case that

$$\begin{aligned} E\left[A\left(\frac{(i+1)t}{n}\right) - A\left(\frac{it}{n}\right)\right] &= E\left[A\left(\frac{t}{n}\right)\right] \\ &= \frac{\lambda t}{n} \end{aligned}$$

and, consequently, the expression of $E[a_n(t)]$ *becomes*

$$\begin{aligned} E[a_n(t)] &= \sum_{i=0}^{n-1} E\left[I\left(\frac{it}{n}\right)\right] \cdot E\left[A\left(\frac{t}{n}\right)\right] \\ &= \frac{\lambda t}{n} \sum_{i=0}^{n-1} E\left[I\left(\frac{it}{n}\right)\right] \\ &= \lambda E\left[\sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[\frac{(i+1)t}{n} - \frac{it}{n}\right]\right]. \end{aligned}$$

Further, by (41) and (45) combined, we can write

$$\begin{aligned}
E[a(t)] &= \lim_{n\to\infty} E\left[a_n(t)\right] \\
&= \lim_{n\to\infty} \lambda E\left[\sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[\frac{(i+1)t}{n} - \frac{it}{n}\right]\right] \\
&= \lambda E\left[\lim_{n\to\infty} \sum_{i=0}^{n-1} I\left(\frac{it}{n}\right)\left[\frac{(i+1)t}{n} - \frac{it}{n}\right]\right] \\
&= \lambda E\left[\int_0^t I(s)\, ds\right].
\end{aligned}$$

This completes the proof of the theorem.

Having proved Theorem V.1.1 it is helpful to put the result in perspective.

- We note that Theorem V.1.1 is part of the sensor network folklore and has been used in the literature without proof. To the best of our knowledge this is the first time it has received a formal proof for the case where the monitored events occur according to a Poisson process;
- We are, however, quick to point out that Theorem V.1.1 holds for other arrival processes; however, it would seem that the problem of characterizing the class of monitored events for which Theorem V.1.1 holds is still an open problem;
- In spite of its interest, Theorem V.1.1 only refers to the "average" case and does not provide a truly crisp view of the instantaneous status in which a given event occurring in disk D may find the process N.

We now turn our attention to the general case: our aim is to compare the fraction of time that disk D is k-covered with the fraction of the events occurring in D that find D k-covered. More precisely, define $P(t)$ as the fraction of the events in $[0, t]$ that find disk D k-covered

$$P(t) = \frac{a(t)}{A(t)}. \tag{47}$$

In this notation, we shall prove the following fundamental result, similar in spirit, to the well-known PASTA property proved in a different context [44].

Theorem V.1.2

$$\lim_{t\to\infty} P(t) = \lim_{t\to\infty} F(t).$$

The proof of Theorem V.1.2 is rather technical and relies on a number of results which are of independent interest. To begin, we define the stochastic process $\{R(t), t \geq 0\}$ where

$$R(t) = a(t) - \lambda t \cdot F(t). \tag{48}$$

The intention is for $R(t)$ to capture, for a given $t \geq 0$, the difference between the number $a(t)$ of events occurring in $[0, t]$ that find D k-covered and the arrival rate λ times the total time in $[0, t]$ during which D is k-covered. Since the arrival time λ multiplied by the total time in $[0, t]$ during which D is k-covered is the *expected* number of arriving events that find D is k-covered, it follows that $R(t)$ denotes the difference between the number of events occurring in $[0, t]$ that find D k-covered and the expected number of events that (upon arrival) find D k-covered.

Lemma V.1.3 *$R(t)$ is a continuous-time martingale.*

Proof 2 *In order to prove that $R(t)$ is a martingale we need to show that for $t > 0$; $h > 0$*

$$E[R(t+h) \mid R(s),\ 0 \leq s \leq t] = R(t).$$

We find it convenient to prove the following equivalent statement

$$E[R(t+h) - R(t) | R(s),\ 0 \leq s \leq t] = 0.$$

Recall from the definition of $R(t)$ that,

$$\begin{aligned} R(t) &= a(t) - \lambda t \cdot F(t) \\ R(t+h) &= a(t+h) - \lambda(t+h) \cdot F(t+h) \\ R(t+h) - R(t) &= [a(t+h) - a(t)] - [\lambda(t+h)F(t+h) - \lambda t F(t)] \end{aligned}$$

Applying the expectation operator on both sides of the previous equality and using elementary manipulations we write

$$\begin{aligned} E[R(t+h) - R(t)] &= E[a(t+h) - a(t)] - E[\lambda(t+h)F(t+h) - \lambda t F(t)] \\ &= E[a(t+h)] - E[a(t)] - E[\lambda(t+h)F(t+h) - \lambda t F(t)] \end{aligned} \tag{49}$$

Recall that by Theorem V.1.1

$$E[a(t)] = \lambda E\left[\int_0^t I(s)\, ds\right]$$

and

$$E[a(t+h)] = \lambda E\left[\int_0^{t+h} I(s)\,ds\right].$$

It follows that

$$E[a(t+h)] - E[a(t)] = \lambda E\left[\int_t^{t+h} I(s)\,ds\right]. \tag{50}$$

Further, recalling that

$$tF(t) = \int_0^t I(s)\,ds$$

and that

$$(t+h)F(t+h) = \int_0^{t+h} I(s)\,ds$$

we can write

$$\lambda(t+h)F(t+h) - \lambda tF(t) = \lambda \int_t^{t+h} I(s)\,ds$$

and so

$$E[\lambda(t+h)F(t+h) - \lambda tF(t)] = \lambda E\left[\int_t^{t+h} I(s)\,ds\right]. \tag{51}$$

Substituting (50) and (51) in (49), we obtain

$$\begin{aligned} E[R(t+h) - R(t)] &= \lambda E\left[\int_t^{t+h} I(s)\,ds\right] - \\ &\quad \lambda E\left[\int_t^{t+h} I(s)\,ds\right] \\ &= 0. \end{aligned}$$

We just proved that $R(t+h) - R(t)$ is independent of $R(s)$ for $0 \le s \le t$. This completes our proof of Lemma V.1.3.

For arbitrary $h > 0$ and positive integer n define the sequence of random variables

$$X_1,\ X_2,\ \cdots,\ X_n,\ \cdots$$

by writing

$$X_n = R(nh) - R((n-1)h). \tag{52}$$

Observe that (52) implies

$$\sum_{i=1}^{n} X_n = R(nh)$$

and that by linearity of expectation

$$\begin{aligned} E[X_n] &= E[R(nh) - R((n-1)h)] \\ &= E[R(nh)] - E[R((n-1)h)] \\ &= 0. \end{aligned}$$

Now, the previous two equalities together with the Law of Large Numbers allow us to write

$$\begin{aligned} \lim_{n\to\infty} \frac{R(nh)}{n} &= \lim_{n\to\infty} \frac{\sum_{i=1}^{n} X_n}{n} \\ &= E[X_n] \\ &= 0. \end{aligned} \qquad (53)$$

We now turn our attention to the proof of Theorem V.1.2. Observe that by dividing both sides of (48) by t, we can write

$$\begin{aligned} \frac{R(t)}{t} &= \frac{a(t) - \lambda t F(t)}{t} \\ &= \frac{a(t)}{t} - \lambda F(t) \\ &= \frac{a(t)}{A(t)} \cdot \frac{A(t)}{t} - \lambda F(t) \\ &= P(t)\frac{A(t)}{t} - \lambda F(t). \end{aligned} \qquad (54)$$

Observe that since $A(t)$ is Poisson distributed with parameter λ

$$\lim_{t\to\infty} \frac{A(t)}{t} = \lambda. \qquad (55)$$

Thus, taking limits in (54) as $t \to \infty$ we obtain

$$\begin{aligned} \lim_{t\to\infty} \frac{R(t)}{t} &= \lim_{t\to\infty} \left[P(t)\frac{A(t)}{t} - \lambda F(t)\right] \\ &= \lim_{t\to\infty} P(t) \lim_{t\to\infty} \frac{A(t)}{t} - \lim_{t\to\infty} \lambda F(t) \\ &= \lambda \lim_{t\to\infty} P(t) - \lambda \lim_{t\to\infty} F(t) \quad \text{[by (55)]} \\ &= \lambda \left[\lim_{t\to\infty} P(t) - \lim_{t\to\infty} F(t)\right] \end{aligned} \qquad (56)$$

Equation (56) makes it plain that in order to complete the proof of Theorem V.1.2 we need to show that $\lim_{t\to\infty} \frac{R(t)}{t} = 0$.

With this in mind we now provide the proof of that missing link. Specifically, we show that

Lemma V.1.4

$$\lim_{t\to\infty}\frac{R(t)}{t} = 0.$$

Proof 3 *Let $h > 0$ be arbitrary and write*

$$n = \left\lfloor \frac{t}{h} \right\rfloor . \tag{57}$$

By virtue of (57) we have the following double inequality

$$nh \le t < (n+1)h. \tag{58}$$

Now, recalling that $a(t)$ is monotone non-decreasing allows us to write

$$a(nh) \le a(t) \le a((n+1)h),$$

which, upon subtracting $\lambda tF(t)$, yields

$$a(nh) - \lambda tF(t) \le a(t) - \lambda tF(t) \le a((n+1)h) - \lambda tF(t).$$

Using (48), the above double inequality can be written as

$$\begin{aligned} &R(nh) + \lambda nhF(nh) - \lambda tF(t) \le R(t) \le \\ &R((n+1)h) + \lambda(n+1)hF((n+1)h) - \lambda tF(t) \end{aligned} \tag{59}$$

Further, using (37) and (58) it is straightforward to see that

$$\begin{aligned} (n+1)hF((n+1)h) - tF(t) &= \int_t^{(n+1)h} I(s)\,ds \\ &\le \int_{nh}^{(n+1)h} ds \\ &= h \end{aligned} \tag{60}$$

and, similarly, that

$$\begin{aligned} nhF(nh) - tF(t) &= \int_t^{nh} I(s)\,ds \\ &= -\int_{nh}^{t} I(s)\,ds \\ &\ge -\int_{nh}^{t} ds \\ &> -\int_{nh}^{(n+1)h} ds \\ &= -h \end{aligned} \tag{61}$$

Now, by replacing (61) and (60) in (59) we obtain

$$R(nh) - \lambda h \leq R(t) \leq R((n+1)h) + \lambda h. \tag{62}$$

Dividing (62) by t and taking limits as $t \to \infty$ we write

$$\lim_{t\to\infty} \frac{R(nh) - \lambda h}{t} \leq \lim_{t\to\infty} \frac{R(t)}{t} \leq \lim_{t\to\infty} \frac{R((n+1)h) + \lambda h}{t}. \tag{63}$$

Noticing that $\lim_{t\to\infty} \frac{\lambda h}{t} = 0$, *(63) can be written as*

$$\lim_{t\to\infty} \frac{R(nh)}{t} \leq \lim_{t\to\infty} \frac{R(t)}{t} \leq \lim_{t\to\infty} \frac{R((n+1)h)}{t}. \tag{64}$$

By (57), $t \to \infty$ implies that $n = \lfloor \frac{t}{h} \rfloor \to \infty$ and, moreover

$$\lim_{t\to\infty} \frac{n}{t} < \infty.$$

This observation allows us to write

$$\begin{aligned} \lim_{t\to\infty} \frac{R(nh)}{t} &= \lim_{n\to\infty} \frac{R(nh)}{n} \lim_{t\to\infty} \frac{n}{t} \\ &= 0. \quad \textit{[by (53)]} \end{aligned} \tag{65}$$

Similarly,

$$\begin{aligned} \lim_{t\to\infty} \frac{R((n+1)h)}{t} &= \lim_{n\to\infty} \frac{R((n+1)h)}{n+1} \lim_{t\to\infty} \frac{n+1}{t} \\ &= 0. \quad \textit{[by (53)]} \end{aligned} \tag{66}$$

Finally, replacing (65) and (66) in (64) yields

$$0 \leq \lim_{t\to\infty} \frac{R(t)}{t} \leq 0$$

completing the proof of Lemma V.1.4.

With this the proof of Theorem V.1.2 is complete.

Theorem V.1.2 tells us that for large intervals $[0, t]$, the fraction of events that occur while D is k-covered equals the fraction of time the disk D is k-covered. In other words, if the monitored events occur according to a Poisson process, the number of sensors that will be awake to report these events is a discrete random variable with probability distribution determined by the fraction of time the network spends in each state. Based on this, it is *probabilistically correct* to analyze sleep scheduling schemes using the *time-invariant* probability distribution of the number of awake sensors in a given sensing area, provided such a time-invariant probability distribution exists.

It is the goal of the next section to show that under very mild technical conditions, this is indeed the case.

V.2 REASONING ABOUT THE TIME-INDEPENDENT AWAKE PROBABILITY

In deployments populated by energy-constrained sensors it is of paramount importance to design energy-aware protocols that promote the functional longevity of the underlying network. This typically means that the sensors spend their lifetime alternating between two modes: in *sleep mode* the sensor turns off its radio interface clocks down its processor; in *awake* mode the sensor is fully functional, with its processor running at top speed and its radio interface turned on.

Let the sequence of random variables $A_1, A_2, \cdots, A_n, \cdots$ denote the consecutive *awake* times of a given sensor. It is quite natural to assume that these awake times are independent identical distributed (i.i.d.) with a common distribution function F_A and that A has *finite* expectation. Similarly, let the sequence of random variables $S_1, S_2, \cdots, S_n, \cdots$ denote the consecutive *sleep* times of a given sensor. We assume that the sleep times are i.i.d. with a common distribution function F_S. As before, we assume that S has *finite* expectation

To make a choice,[2] we assume that the sensor wakes up for the 0-th time at $t = 0$. Referring to Figure 35, we find it useful to model the lifetime of the given sensor as a renewal process $\{N(t),\ t \geq 0\}$, where the renewal points are the moments when the sensor wakes up (other than the 0-th wake-up at time $t = 0$). Let

$$T_0 = 0,\ T_1 = A_1 + S_1,\ T_2 = A_2 + S_2,\ \cdots,\ T_n = A_n + S_n,\ \cdots$$

be the inter-arrival times of the renewal process where for $n \geq 1$, T_n is the time interval between the $(n-1)$-th and the n-th renewals where, recall, $t = 0$ is taken as the 0-th renewal. It is clear that the random variables $T_1, T_2, \cdots, T_n, \cdots$ have a common distribution F_T, which is the convolution of F_A and F_S. Since, by assumption, both A and S have finite expectation, so does T. In fact, we can write

$$\mu = E[T] = E[A + S] = E[A] + E[S] < \infty. \tag{67}$$

We are interested in the limiting probability $h(t)$ that our sensor is awake at time t. To derive the integral equation satisfied by $h(t)$, we condition on the time of the

[2] As it turns out, this assumption is adopted for convenience only; as far as the limiting probability of the sensor to be awake at some arbitrary time t this assumption is irrelevant.

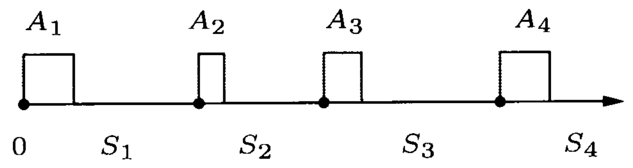


FIG. 35: *Illustrating the renewal process of wake-up times.*

first renewal, T_1. Indeed, we can write

$$\begin{aligned} h(t) &= \int_0^\infty \Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}] \Pr[\{T_1 = s\}] \\ &= \int_0^\infty \Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}]\, dF_{T_1}(s) \\ &= \int_0^t \Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}]\, dF_{T_1}(s) + \\ &\quad \int_t^\infty \Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}]\, dF_{T_1}(s) \end{aligned} \tag{68}$$

Observe that for $0 \leq s \leq t$, we can write

$$\Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}] = h(t-s). \tag{69}$$

On the other hand, for $t > s$ we have

$$\begin{aligned} &\int_t^\infty \Pr[\{\text{awake at time } t\} \mid \{T_1 = s\}]\, dF_{T_1}(s) \\ &= \Pr[\{A_1 > t\} \cap \{T_1 > t\}] \\ &= \Pr[\{A_1 > t\}] \;\; [\text{since } \{A_1 > t\} \subseteq \{T_1 > t\}] \\ &= 1 - F_A(t). \end{aligned} \tag{70}$$

By virtue of (69) and (70), combined, (68) becomes

$$h(t) = [1 - F_A(t)] + \int_0^t h(t-s)\, dF_T(s) \tag{71}$$

Writing

$$Q(t) = 1 - F_A(t)$$

it is easy to see that $h(t)$ satisfies the renewal integral equation

$$h(t) = Q(t) + \int_0^t h(t-s)\, dF_T(s) \tag{72}$$

and that $Q(t)$ has the following properties (q1) $Q(t) \geq 0$;

(q2) $Q(t)$ is non-increasing for all $t \geq 0$;

(q3) $\int_0^\infty Q(t)\, dt = \int_0^\infty [1 - F_A(t)]\, dt = E[A] < \infty$.

Assuming that T is non-lattice,[3] the Key Renewal Theorem [45] guarantees that

$$\begin{aligned}
\lim_{t\to\infty} h(t) &= \lim_{t\to\infty} [1 - F_A(t)] + \lim_{t\to\infty} \frac{1}{\mu} \int_0^\infty Q(s)\, ds \\
&= \lim_{t\to\infty} \frac{1}{\mu} \int_0^\infty Q(s)\, ds \\
&= \frac{1}{\mu} E[A] \quad \text{[by property (q3) above]} \\
&= \frac{E[A]}{E[A] + E[S]} \quad \text{[by (67)]} \qquad (73)
\end{aligned}$$

To summarize our findings, we have proved the following result.

Theorem V.2.1 *Assuming that A has finite expectation and that $T = A + S$ is non-lattice with finite expectation, the limiting probability that a given sensor is awake at time t equals*

$$\frac{E[A]}{E[A] + E[S]}.$$

Having proved Theorem V.2.1, it is important to put this result in perspective.

- Since the conclusion of Theorem V.2.1 is intuitively satisfying, the result itself has been a part of the sensor network folklore and has been used without proof. To the best of our knowledge this is the first time the result is formally proved;

- Theorem V.2.1 has an unmistakable PASTA "flavor": for, consider an observer external to the network that is watching and noting the behavior of our sensor. The observer will note that the long-term probability of the sensor to be awake is $\frac{E[A]}{E[A]+E[S]}$. On the other hand, Theorem V.2.1 tells us that far away from the origin (i.e. $t = 0$), the probability that the sensor is awake at an arbitrary time t is also $\frac{E[A]}{E[A]+E[S]}$;

- Theorem V.2.1 is a very convenient tool since it allows one to use the time-invariant probability of a sensor being awake in lieu of the instantaneous awake probability.

[3] A discrete random variable is said to be *lattice* if all the values it can assume with positive probability are of the form nh for some $h > 0$ and integer n.

We put Theorem V.2.1 to work in the Section V.3, where it will be used to reason about various scheduling schemes. Before that, in the next subsection, we offer an empirical validation of Theorem V.2.1.

V.2.1 Empirical validation

In order to validate empirically the results of Theorem V.2.1, we conducted the following experiment to compare the probability that a sensor is awake (i.e. the expected value of $\frac{A}{A+S}$) to the value obtained from Theorem V.2.1 (i.e $\frac{E[A]}{E[A]+[S]}$). It is straightforward, albeit tedious, to show that if A is uniformly distributed at random in $[a, b]$ and S is uniformly distributed in $[c, d]$ then, with $\Delta = 2(b-a)(d-c)$, the expectation $E\left[\frac{A}{A+S}\right]$ of the random variable $\frac{A}{A+S}$ reads

$$\begin{aligned} E\left[\frac{A}{A+S}\right] &= \frac{1}{2} + \frac{c^2-b^2}{\Delta}\ln\left(\frac{b+c}{b+d}\right) + \\ & \frac{d^2-c^2}{\Delta}\ln\left(\frac{a+c}{b+d}\right) + \frac{d^2-a^2}{\Delta}\ln\left(\frac{a+d}{a+c}\right). \end{aligned}$$

The interested reader can find a detailed discussion of the probability distribution function of $\frac{A}{A+S}$ as well as the derivation of $E[\frac{A}{A+S}]$ in Appendix A.

Our goal was to measure the difference between the exact and the approximated values of the awake probability assuming different ranges for $[a, b]$ and $[c, d]$. Without loss of generality we can always use shifting and scaling to map one of the two ranges into the range $[1, 2]$ and express the other range using the same mapping. Figure 36 shows a plot of the simulated, theoretical, and approximated values of sensor awake probability. In Figure 36, we assumed that the range $[a, b]$ is mapped into the range $[1, 2]$ and we tried different ranges for $[c, d]$ based on the ratio $\frac{d-c}{b-a}$. Specifically, we tried the following ratios: $\frac{1}{128}$, $\frac{1}{64}$, $\frac{1}{32}$, $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$, $\frac{1}{2}$, 1, 2, 4, 8, 16, 32, 64, and 128. As shown in Figure 36, for all the ranges, the differences between the theoretical, simulated, and approximated values of the probability are minor and can be ignored in almost all practical applications.

V.3 SCHEDULING SCHEMES

In a sensor network with *nominal* (i.e. deployment-time) density ρ where, in order to promote system longevity, some sensors will be in sleep mode, of interest is the effective sensor density (ESD) ρ_e defined as the density of the awake sensors in the

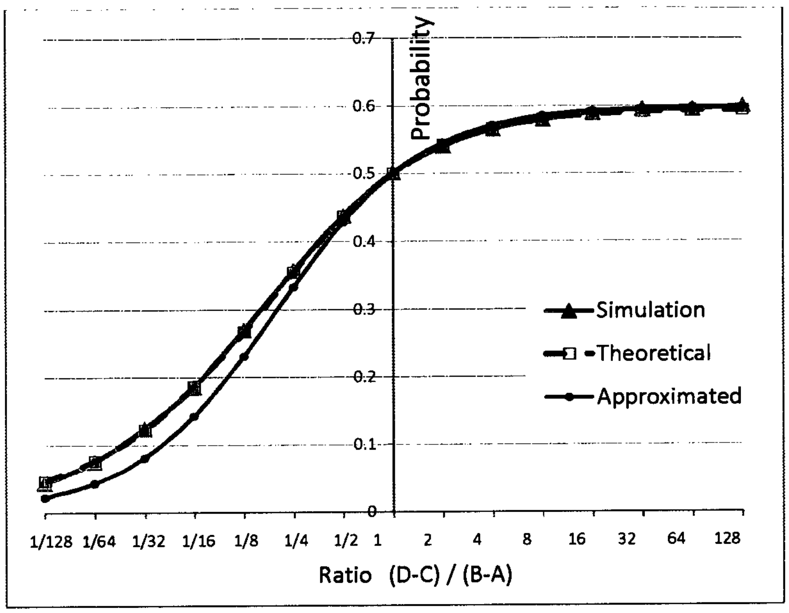


FIG. 36: *Empirical validation of Theorem V.2.1*

network at any point of time. If a time-invariant probability P_{awake} of individual sensors to be awake exists, then ρ_e is related to ρ by

$$\rho_e = P_{awake}\rho.$$

Switching between the sleep and awake modes is controlled by a *scheduling scheme* used in the network. Different deterministic and probabilistic scheduling schemes can be invented and implemented; each such scheme makes decisions about how much the sensors sleep and, as such, impacts the ρ_e and the ESD.

We begin this section by introducing two relatively simple scheduling schemes: our first such scheme is static, the second is dynamic. We also study the implications of each scheme on the ESD. Finally, we propose a third, energy-aware scheduling scheme designed to balance energy consumption among the sensors in the same sensing area.

V.3.1 Static Scheduling

The main advantage and unmistakable appeal of static scheduling schemes are their simplicity. Before deployment, each sensor is given two values T_s and T_a, which represent the number of time units each sensor stays in sleep mode and awake mode,

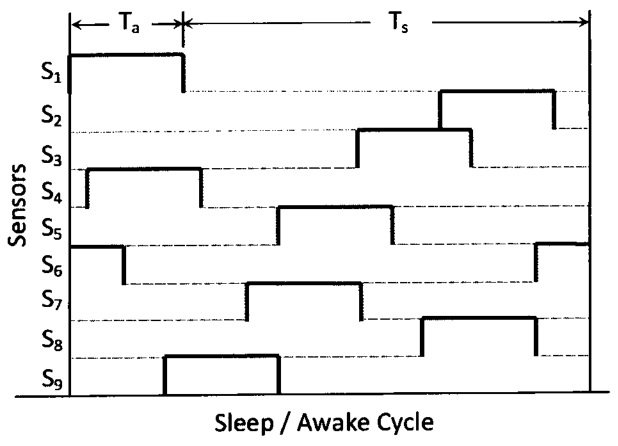


FIG. 37: *Sleep and awake cycles in static scheduling*

respectively. In general T_s and T_a, are chosen such that $T_s >> T_a$. This choice allows the sensors to sleep more than they stay awake helping them to save their energy budget and to extend their lifetime. Moreover, it is highly recommended to give sensors common values for T_s, and T_a, otherwise the energy consumption rate of sensors with longer awake times will be much higher than the energy consumption rate of other sensors. This, in turn, would adversely impact network longevity and reliability and could, eventually, lead to the creation of energy holes that would partition the network into disconnected islands [10, 46].

Because the sensors are asynchronous, the sleep time of a sensor usually overlaps with the awake time of other sensors. Hence, using an appropriate deployment density we can ensure there is always, with any desired probability, a sufficient number of awake sensors to attend to any required network task. Figure 37 shows an example where the overlapping of the awake times of sensors S_1 through S_9 totally cover the scheduling cycle.

In the static scheduling scheme the total length of the sleep and awake cycle is fixed to $T_a + T_s$ time units. Each sensor stays awake for T_a time units and sleeps for T_s time units. It is clear that, in this case, the time-independent probability of a sensor to be awake is

$$\frac{T_a}{T_a + T_s}.$$

Given a nominal network density ρ and a sensor sensing range of r, the *expected* number of sensors in the sensing area can be evaluated as $n = \pi r^2 \rho$. Under these assumption, the probability distribution of the number X of awake sensors is a *binomial* random variable and for all k, $(0 \leq k \leq n)$,

$$\begin{aligned} P[\{X = k\}] &= \binom{n}{k} \left(\frac{T_a}{T_a + T_s}\right)^k \left(\frac{T_s}{T_a + T_s}\right)^{n-k} \\ &= \binom{n}{k} \frac{(T_a)^k (T_s)^{n-k}}{(T_a + T_s)^n}. \end{aligned} \tag{74}$$

Since the schedule is static, it makes sense to define for an arbitrary sensor the *awake ratio* A as

$$A = \frac{T_a}{T_a + T_s}.$$

In this notation, (74) becomes

$$P[\{X = k\}] = \binom{n}{k} A^k (1 - A)^{n-k}. \tag{75}$$

Now, (75) implies, unsurprisingly, that the ESD, $E[X]$, corresponding to this scheme is

$$E[X] = nA = \pi r^2 \rho A. \tag{76}$$

Equation (76) can be used to estimate the required nominal density, given QoS requirements expressed in terms of number of sensors needed per task.

V.3.2 Dynamic Scheduling

In this scheme, we assume that a sensor sleeps for a random amount of time selected uniformly at random in the interval $[T_s, T_S]$, and is awake for another random amount of time that is selected uniformly in the interval $[T_a, T_A]$. Define the random variables A_i and S_i representing, respectively, the awake and sleep time of a generic sensor in an arbitrary cycle i. The expected values of sleep time and awake time in cycle i are given by

$$E[S_i] = \frac{T_S + T_s}{2} \tag{77}$$

$$E[A_i] = \frac{T_A + T_a}{2} \tag{78}$$

Our choice of the distribution of A_i and S_i along with (77) and (78) ensure that the random variable $T_i = A_i + S_i$ is non-lattice and has finite expectation. By Theorem

V.2.1, the time-invariant probability of a sensor to be awake is

$$\begin{aligned} P_A &= \frac{E[A]}{E[A]+E[S]} \\ &= \frac{T_a+T_A}{T_a+T_A+T_s+T_S} \end{aligned} \tag{79}$$

Define X as the random variable that denotes the number of awake sensors when the next event occurs. To evaluate the probability distribution of A, we model the problem as a sequence of n Bernoulli trials, where the probability of success is P_A. According to this model, the random variable X follows a binomial distribution with parameters n and P_A.

$$\begin{aligned} P[\{X=k\}] &= \binom{n}{k} P_A^k (1-P_A)^{n-k} \\ &= \binom{n}{k} \frac{(T_a+T_A)^k (T_s+T_S)^{n-k}}{(T_a+T_A+T_s+T_S)^n} \end{aligned}$$

and, similarly,

$$\begin{aligned} E[X] &= \frac{n(T_a+T_A)}{(T_a+T_A+T_s+T_S)} \\ &= \frac{\pi r^2 \rho (T_a+T_A)}{(T_a+T_A+T_s+T_S)} \end{aligned} \tag{80}$$

As in the case of static scheduling, equation (80) can be used to estimate required deployment density to satisfy QoS requirements when the network uses dynamic scheduling.

It is also worth noting that our dynamic scheduling scheme presented in this work follows the "standard" (and, perhaps, simplistic view that the awake and sleep periods are *uniform* random variables. There is no need for this; in fact, we could have taken any distribution for the awake times and for the sleep times that satisfies the conditions of Theorem V.2.1.

Observe that by Theorem V.2.1, for any dynamic scheme it is always possible to construct a probabilistically equivalent static scheme that draws its sleep and awake times from general probability distribution functions. To do this, in the newly created static scheme the sensors sleep and awake times are adjusted at the expected values of the sleep and awake distribution functions of the original dynamic scheme.

V.3.3 Energy-Aware Scheduling

Finally, we propose a dynamic scheduling scheme [47, 48] that contributes to balancing energy consumption among sensors by adjusting their sleep time based on their remaining energy budget. The main idea behind the proposed scheme is to probabilistically prolong the sleep time of sensors with relatively low energy and shorten the sleep time of sensors with relatively high energy, without affecting the ESD of the network.

As in Subsection V.3.2, we assume that a sensor sleeps for a random amount of time *uniformly* distributed in $[T_s, T_S]$ and is awake for a random amount of time *uniformly* distributed in $[T_a, T_A]$. Either periodically or after each task, the sensors adjust the upper bounds of the ranges from which they select their sleep and awake times. This adjustment is based on the relative difference between sensor energy e_i, the minimum energy e_{min} and the maximum energy e_{max} among the neighboring sensors. In Section V.4 we propose several approaches by which sensors can have access to energy information of the sensors in their neighborhood.

Several adjustment formulas can be used; one natural choice is

$$T_{S_{new}} = T_s + c\left(\frac{e_{max} - e_i}{e_{max} - e_{min}}\right)(T_S - T_s) \tag{81}$$

$$T_{A_{new}} = T_a + c\left(\frac{e_i - e_{min}}{e_{max} - e_{min}}\right)(T_A - T_a) \tag{82}$$

where c is a constant to be determined. Using equations (81) and (82), each sensor can adjust its sleep and awake time by selecting them, uniformly at random, from the ranges $[T_s, T_{S_{new}}]$ and $[T_a, T_{A_{new}}]$, respectively. The constant c is determined such that the expected value of ESD remains unchanged.

If the sleep time and the awake time upper bounds in cycle i are $T_{S_{new}}$ and $T_{A_{new}}$ respectively, then the expected sleep time and awake time can be expressed, respectively, as

$$\frac{T_{S_{new}} + T_s}{2}$$

and

$$\frac{T_{A_{new}} + T_a}{2}$$

Unfortunately, both $T_{A_{new}}$ and $T_{S_{new}}$ change over time. However, because these changes do not occur abruptly, it is safe to assume that $T_{S_{new}}$ and $T_{A_{new}}$ will remain fixed for the upcoming m cycles until the next adjustment. Based on this, A_i, S_i for the upcoming m cycles are drawn uniformly from the ranges $[T_a, T_{A_{new}}]$ and $[T_s, T_{S_{new}}]$

respectively. Hence, we can express the expected value of A and S as follows,

$$\begin{aligned} E[A] &= E\left[\sum_{i=1}^{m} A_i\right] = \sum_{i=1}^{m} E[A_i] \\ &= \sum_{i=1}^{m} E\left[\frac{T_{A_{new}} + T_a}{2}\right] \\ &= \frac{mT_a}{2} + \frac{1}{2}\sum_{i=1}^{m} E[T_{A_{new}}] \end{aligned} \tag{83}$$

Similarly, $E[T_{A_{new}}]$ can be evaluated as

$$\begin{aligned} E[T_{A_{new}}] &= T_a + c\left(\frac{E[e_i] - e_{min}}{e_{max} - e_{min}}\right)(T_A - T_a) \\ E[e_i] &= \frac{e_{max} + e_{min}}{2} \\ E[T_{A_{new}}] &= T_a + c\left(\frac{\frac{e_{max}+e_{min}}{2} - e_{min}}{e_{max} - e_{min}}\right)(T_A - T_a) \\ &= T_a + \frac{c}{2}(T_A - T_a) \end{aligned} \tag{84}$$

Substituting the expression for $E[T_{A_{new}}]$ from (84) into (83) yields

$$\begin{aligned} E[A] &= \frac{mT_a}{2} + \frac{1}{2}\sum_{i=1}^{m} T_a + \frac{c}{2}(T_A - T_a) \\ &= mT_a + \frac{mc(T_A - T_a)}{4} \\ &= \frac{m(cT_A + (4-c)T_a)}{4} \end{aligned} \tag{85}$$

Using the same approach, we can evaluate $E[S]$ as,

$$\begin{aligned}
E[S] &= E\left[\sum_{i=1}^{m} S_i\right] = \sum_{i=1}^{m} E\left[\frac{T_{S_{new}} + T_s}{2}\right] \\
&= \frac{mT_s}{2} + \frac{1}{2}\sum_{i=1}^{m} E[T_{S_{new}}] \\
E[T_{S_{new}}] &= T_s + c\left(\frac{e_{max} - E[e_i]}{e_{max} - e_{min}}\right)(T_S - T_s) \\
&= T_s + c\left(\frac{e_{max} - \frac{e_{max}+e_{min}}{2}}{e_{max} - e_{min}}\right)(T_S - T_s) \\
&= T_s + \frac{c}{2}(T_S - T_s) \\
E[S] &= \frac{mT_s}{2} + \frac{1}{2}\sum_{i=1}^{m} T_s + \frac{c}{2}(T_S - T_s) \\
&= mT_s + \frac{mc(T_S - T_s)}{4} \\
E[S] &= \frac{m(cT_S + (4-c)T_s)}{4} \qquad (86)
\end{aligned}$$

Now, using equations (85) and (86), Theorem V.2.1 tells us that the probability a sensor is awake is

$$\begin{aligned}
P_A &= \frac{\frac{m(cT_A+(4-c)T_a)}{4}}{\frac{m(cT_A+(4-c)T_a)}{4} + \frac{m(cT_A+(4-c)T_a)}{4}} \\
&= \frac{cT_A + (4-c)T_a}{cT_A + (4-c)T_a + cT_S + (4-c)T_s} \qquad (87)
\end{aligned}$$

Solving equations (79) and (87) for c, yields $c = 2$; with this, the equations (81) and (82) can be rewritten as

$$\begin{aligned}
T_{S_{new}} &= T_s + 2\left(\frac{e_{max} - e_i}{e_{max} - e_{min}}\right)(T_S - T_s) \qquad (88) \\
T_{A_{new}} &= T_a + 2\left(\frac{e_i - e_{min}}{e_{max} - e_{min}}\right)(T_A - T_a). \qquad (89)
\end{aligned}$$

After determining c, the equations (88) and (89) are used by individual sensors to adjust their sleep/awake schedule. As we noted at the end of the previous subsection, we have chosen the uniform distribution of sleep/awake times for convenience only; in fact, any distribution satisfying the conditions of Theorem V.2.1 could have been employed just as well. We expect that, ultimately, the choice of the distribution to be application-dependent.

TABLE 3: *Mica2 Power Requirements*

Component	Power	Component	Power
CPU Active	24.0 mW	Rx	21.0 mW
CPU Sleep	225.0 μw	Tx(-20 dBm)	11.1 mW
Sensor ADC	3.0 mW	Tx(-15 dBm)	16.2 mW
Sensor board	2.1 mW	Tx(- 8 dBm)	19.5 mW
EEPROM Read	18.6mW	Tx(0 dBm)	25.5 mW
EEPROM Write	55.2mW	Tx(+ 4 dBm)	34.8 mW

V.4 ENERGY ESTIMATION

The energy-aware scheduling scheme proposed in Subsection V.3.3 assumes that each sensor has information about the minimum and the maximum energy in its neighborhood. In this section we propose several approaches that can be used to provide such information to sensors. Particularly, we provide answers to two questions: (1) How can a sensor estimate its own energy? (2) How can a sensor get access to information about the energy of other sensors in its neighborhood?

Since current technology enables battery-powered electronic devices to measure their remaining energy [49], this might be taken as an easy answer to the first question regarding how sensors can estimate their remaining energy. However, from a practical prospective, this technology may not be sufficiently mature to be applicable at a reasonable price for our tiny inexpensive sensors. Fortunately, it is possible to find an alternative software solution to this problem. To be able to apply this solution, the manufacturer of each sensor should provide an energy consumption data-sheet that specifies the energy consumed per unit time for different sensor features. Table 3 shows the power consumption for the Mica2 platform for different sensor components and different operating modes. For instance, the CC1000 transceiver in the Mica2 mote [50] features a transmission range proportional to the radiated output power, which can be selected from -20dBm to $+5$dBm. Table 3 shows the energy consumption when a Mica2 mote transmits at power -20dBm, -15dBm, -8dBm,0dBm, and +4dBm. Also the table shows the CPU energy consumption when the sensor is in awake or sleep mode.

We can use the energy consumption information provided in the data-sheet to estimate the remaining sensor energy budget as follows:

- Each sensor should have an internal counter initialized to zero $e_c = 0$.
- Upon changes in the functionality of the sensor, the counter value should be updated based on the manufacturer data-sheet to reflect changes in energy consumption. For example, if a Mica2 mote is to switch to sleep mode for 2 seconds, then before the CPU switches to the sleep mode, it should increment the counter as $e_c \leftarrow e_c + 0.45$ to reflect changes in consumed energy.
- $e_c(t)$, the value stored in the counter at time t is a measure of the total energy consumed up to this point in time. If e_I is the initial number of energy units a sensor has, then $e(t)$, the remaining energy of a sensor at time t, can be obtained as $e(t) = e_I - e_c(t)$.

We have just outlined two different ways in which a sensor can estimate its remaining energy at any point in its lifetime. However, this might not be sufficient since the energy-aware scheduling scheme described in Subsection V.3.3 not only requires each sensor to be aware of its remaining energy but also requires each sensor to be aware of the minimum and the maximum energy among the sensors in its neighborhood (i.e. within the same sensing area). Next, we propose two different strategies that the sensors can use to estimate the minimum and the maximum energy of sensors in their neighborhood.

The first strategy to solve this problem is to let each sensor include the value of its remaining energy $e(t)$ in the header of each packet transmitted. Basically, each sensor uses two internal variables to keep track of its estimation of the minimum and the maximum energy among the sensors in its sensing area. Each sensor should initialize these variables to the current value of its own energy. When a sensor receives a packet from another sensor, it extracts the value of the current energy of the transmitting sensor from the packet header. After that it uses this value to update its estimate of the minimum and the maximum energy in its sensing area. It is worthwhile to mention that a sensor should not use all received packets to update its internal variables as some of the received packets may be transmitted from sensors outside its sensing area. Based on the strength of the received signal a sensor can determine whether the transmitting sensor is within its sensing area or not, hence it can decide whether to update the variables or to ignore the packet completely. Also, when the energy of a sensor changes, a sensor may need to update its estimation of the minimum and the maximum energy in its sensing area. This way a sensor has access to up to date information about the minimum and the maximum energy of sensors in its sensing area with minimum overhead, hence it can apply our proposed scheme to adjust its sleeping schedule.

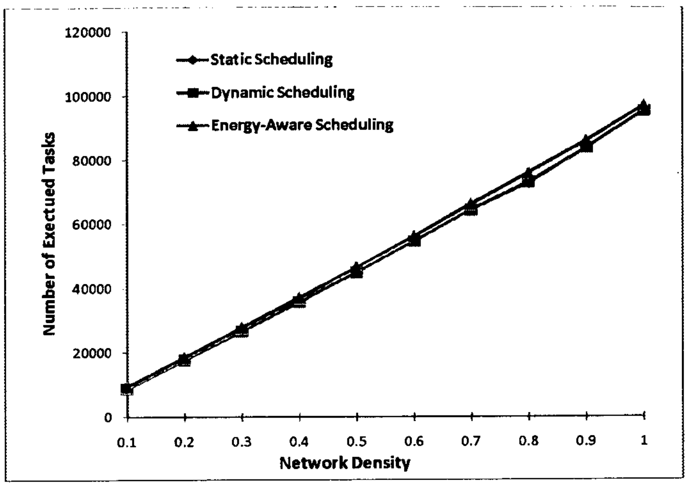
Our second strategy to allow sensors estimate the minimum and the maximum energy in their neighborhood is through integrating sleep scheduling with task management. In particular, information about the minimum and the maximum energy in the sensing area is obtained during the time the workforce for a given task is selected [40, 39, 42].

In Section IV.1, we proposed a bidding-based centralized approach for workforce selection in which backbone sensors (i.e task coordinators) would have access to up to date information about the remaining energy of sensors in their neighborhood. After task completion and while forwarding aggregated results toward the sink, a backbone sensor can always attach the minimum and the maximum energy among sensors in its neighborhood. Sensors that receive this message should extract these values and adjust their sleep schedule accordingly.

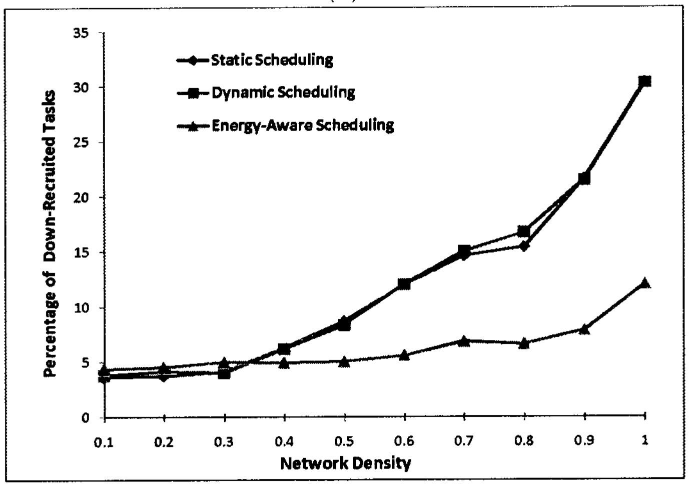
Also, in Section IV.2, we proposed a distributed workforce selection protocol that considers sensor energy while selecting the required workforce. The proposed protocol simply consisted of two phases where in the first phase, the maximum energy among sensors in the area of interest is determined (please refer to subsection IV.2.1 for details). Also in Section IV.3, we showed how the minimum energy among awake sensors within the same sensing area can be evaluated using a similar approach. Once the maximum and the minimum energy are known, sensors can adjust their sleep schedule as described in Subsection V.3.3.

V.5 PERFORMANCE EVALUATION

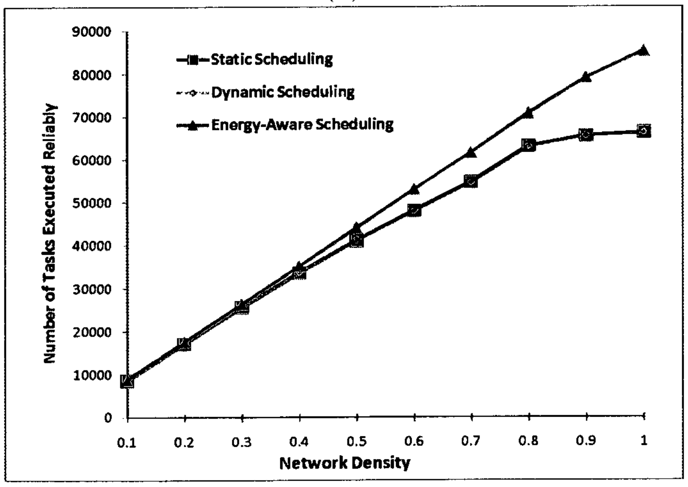
To verify our analytical results and to evaluate the performance of the scheduling scheme proposed in Subsection V.3.3, we built a simulator of a wireless sensor network that can be configured to implement static, dynamic and energy-aware scheduling schemes. In our simulation, we assumed that sensors are equipped with necessary hardware to estimate their remaining energy, also we assumed that the header of each packet transmitted by a sensor contains the current remaining energy of the transmitting sensor. For static scheduling, awake time T_a was set to 10 time units and sleep time T_s was set to 90 time units (a sensor is awake 10% of the time). In dynamic scheduling, awake and sleep times was selected uniformly from the ranges



(a)



(b)



(c)

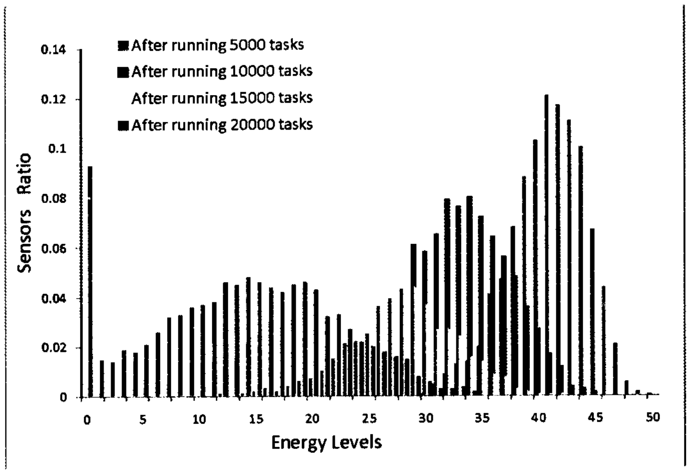
FIG. 38: *(a) Average number of tasks, (b) Percentage of under-recruited tasks, (c) Average number of reliable tasks, executed in the network throughout its* 0.2*-reliable-lifetime using different scheduling schemes.*

$[1,9]$ and $[1,89]$ respectively (on the average a sensor is awake 10% of the time). We used the same settings of dynamic scheduling in our energy-aware scheduling scheme.

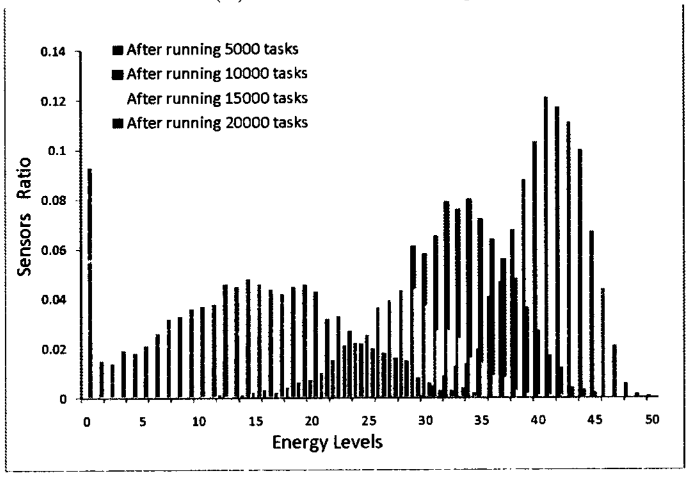
For the simulation configuration, we assumed a square deployment area of size 100m $\times$ 100m. We adopted a reactive tasking model which assumes the existence of a powerful sink node placed at the origin and responsible for assigning tasks to sensors according to a Poisson point process. A task is defined using its position (x, y) and QoS expressed in terms of required workforce w. Each task is assumed to consume one unit of sensor energy. Sensors had a transmission range of 25m and a sensing range of 15m. Unless mentioned otherwise, we assumed that each sensor starts with 50 units of energy.

Recalling the definition of the α-reliable lifetime of the network given in Chapter IV, to show the impact of using different scheduling protocols on network reliable-lifetime, we ran our simulator using the three scheduling schemes presented in Section V.3 and measured the 0.20 reliable-lifetime of the network. Figure 38(a) compares the average number of tasks the network can execute throughout its 0.2-reliable-lifetime using different scheduling protocols. The figure assumes different network densities ranging from 0.1 to 1.0 $sensors/m^2$. Interestingly, the number of tasks executed when using our energy-aware scheduling scheme is hardly higher than its counterpart values when using static or dynamic scheduling schemes. Although, this might imply that our energy-aware protocol is not very useful in extending network lifetime, however a deeper insight showed that this conclusion is incorrect.

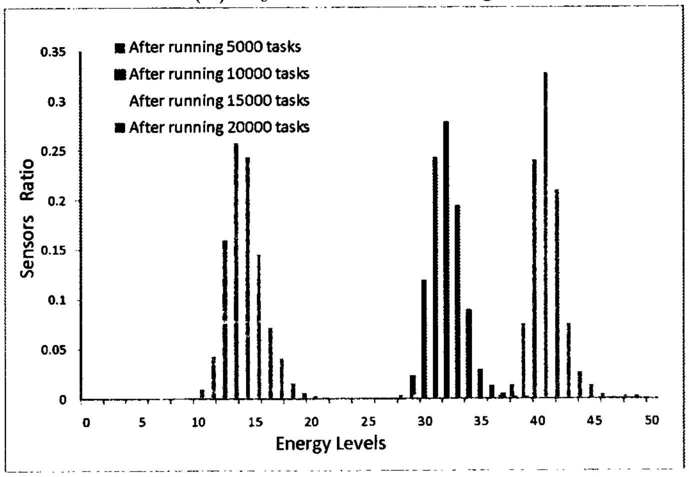
While recruiting the workforce for some tasks especially in later stages of the network lifetime, the number of sensors available at the task position may be less than the minimum size of the workforce specified in QoS requirements. Hence, those tasks are under-recruited and executed using whatever is available even if that number is much lower than what was specified in QoS requirements. This situation gets worse when the variations between sensor energy increase. For instance, if at a late stage of the network lifetime all sensors within a certain area have expired except for a single sensor which has e units of energy remaining, then the network can assume incorrectly that it can execute up to e additional tasks, irrespective of the workforce size required by these tasks. Fortunately, using techniques that consume sensor energy evenly reduces the probability that such a scenario can happen. By the time the first sensor within a certain area is about to expire, the remaining sensors within the same area will be about to expire as well. Hence under-recruiting occurs



(a) Static Scheduling



(b) Dynamic Scheduling



(c) Energy-Aware Scheduling

FIG. 39: *Sensor energy distribution in subsequent stages of the network lifetime for different scheduling protocols*

for a small number of tasks and at the very late stages in the life of the network. Our explanation is confirmed by Figure 38(b) that shows for different scheduling protocols the percentage of the executed tasks that are under-recruited. On the average, only about 5% of the tasks executed on networks running Energy-Aware scheduling are under-recruited compared to around 25% of the tasks executed on networks implementing static or dynamic scheduling.

Figure 38(c) shows the number of reliable tasks executed by different protocols. By reliable tasks, we mean tasks that were executed while satisfying their required level of QoS (i.e. number of sensors participating in each task is greater than or equal to the required workforce specified in the task QoS requirements). As shown in Figure 38(c), using Energy-Aware scheduling increases reliable-network lifetime for different network sizes by around 18.1%. Before leaving this point, it is instructive to note that the average number of executed tasks during the network 0.2 reliable-lifetime when using static scheduling and dynamic scheduling is almost the same. This result is consistent with the theoretical results, as for any dynamic scheduling scheme one can always construct an equivalent static scheme where the sleep and awake times of sensors are fixed at the averages of the dynamic scheme.

Figure 39(a) shows the distribution of sensor energy during subsequent stages of the network lifetime when using static scheduling. For clarity of presentation, we represented different stages by different colors. Figures 39(b), and 39(c) respectively show sensor energy distribution in case of dynamic scheduling and energy-aware scheduling. Apparently, when using static and dynamic scheduling as time passes and network ages, the energy of sensors spans a broad range of the whole energy spectrum which implies severely unbalanced consumption of sensor energy. On the other hand, using energy-aware scheduling can bound variations in sensor energy within around 4 energy levels in different stages of the network lifetime which implies balanced consumption of sensor energy.

Figure 40 summarizes sensor energy distribution patterns for different scheduling schemes and different network densities. In particular, the figure compares the average width of sensor energy spectrum throughout the network 0.2-reliable lifetime for different network densities ranging from 0.1 to 1.0 $sensors/m^2$. To estimate the average width of sensor energy spectrum, we used the same formula we used earlier in Chapter IV (i.e. equation (36)).

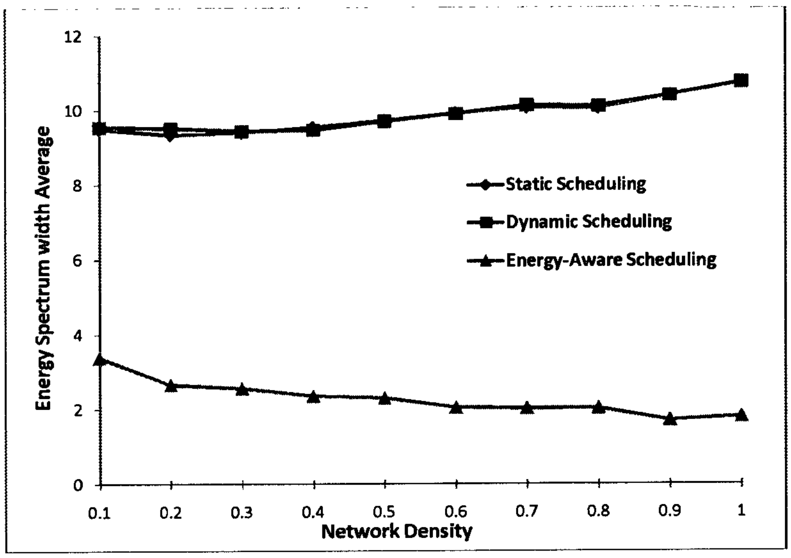


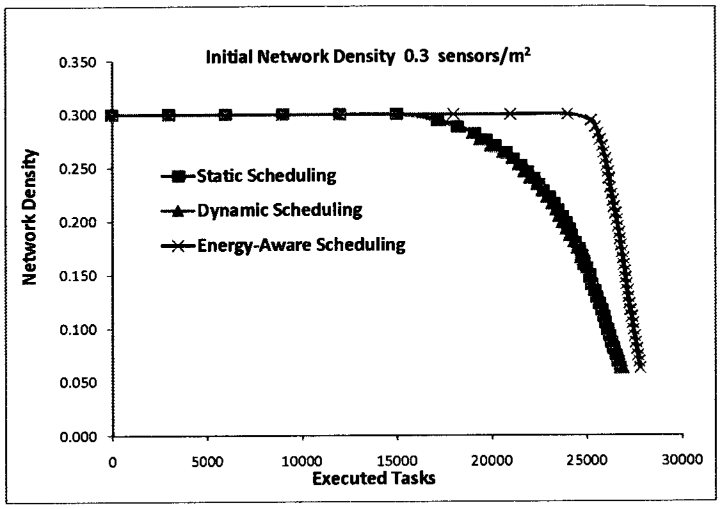
FIG. 40: *The average width of sensor energy spectrum*

Figure 40 reveals that the average width of sensor energy spectrum of energy-aware scheduling is much narrower than the width obtained when using other schemes. Moreover, although it is not shown here due to limited space, we found that the spectrum width for static and dynamic scheduling schemes gets even larger when sensor initial energy was set to larger values (e.g 70 or 100 units).

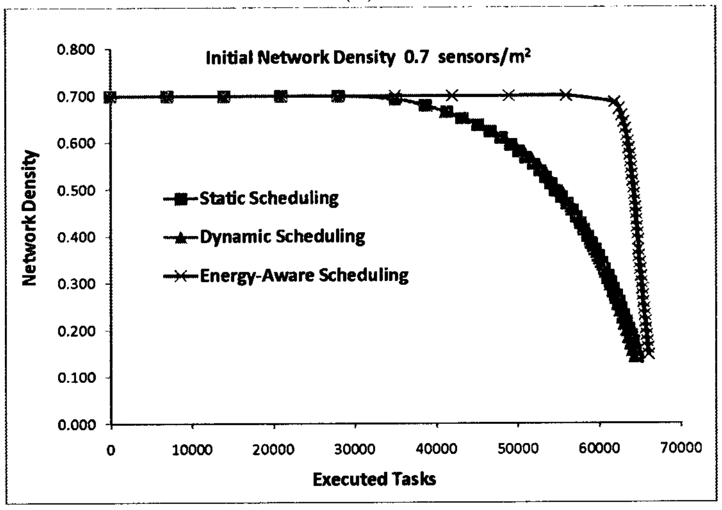
The large variations in sensor energy when using static or dynamic scheduling protocols makes one expect that many of the heavily loaded sensors would die out at an early stage of the network lifetime. Typically, when a large number of sensors which reside at some spot die out, the network density at this spot decreases. Figures 41-(a), 41-(b), and 41-(c) capture this phenomenon when using different workforce selection protocols with initial deployment densities of 0.3, 0.7, and 1.0 respectively.

As shown in Figure 41, the degradation in network density for static and dynamic scheduling starts at relatively an earlier stage compared to energy-aware scheduling. Table 4 lists the percentage of the network lifetime lived before the network density starts to degrade for different schemes.

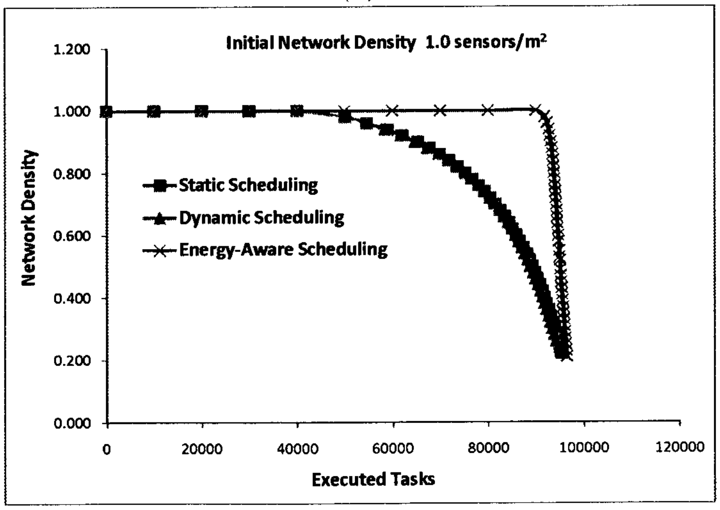
The degradation in network density can eventually lead to the creation of energy holes. We conducted a set of experiments to capture the impact of different scheduling protocols on the rate at which holes appear in the network. Figure 42 compares the



(a)



(b)



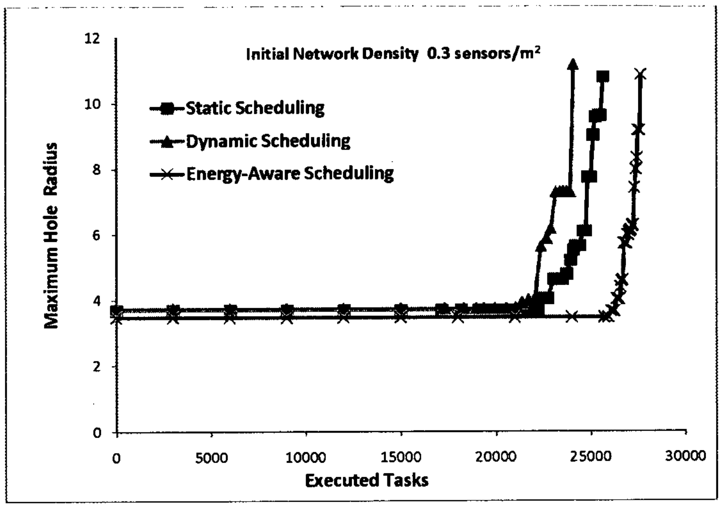
(c)

FIG. 41: *Average degrade in network density throughout its* 0.2*-reliable-lifetime using different scheduling protocols. (a)* $\rho = 0.3$*, (b)* $\rho = 0.7$*, (c)* $\rho = 1.0$

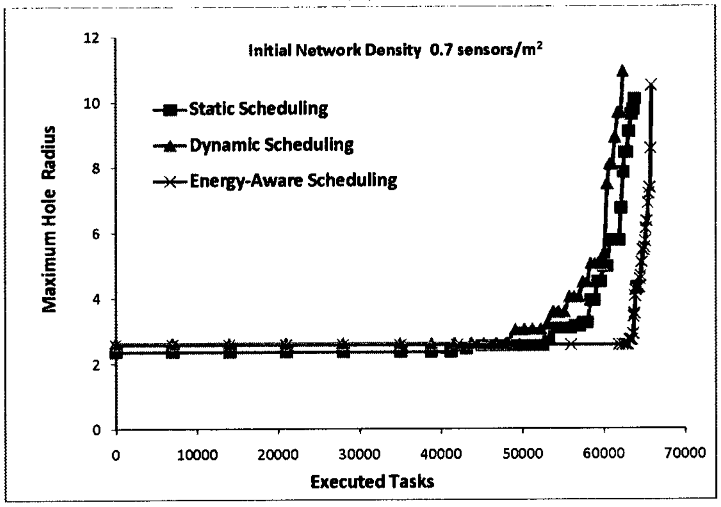
TABLE 4: Percentage of network lifetime lived before density degrades

Density	Static	Dynamic	Energy-Aware
0.3	64.0	64.3	90.8
0.7	54.2	54.4	93.4
1.0	52.4	52.7	93.3

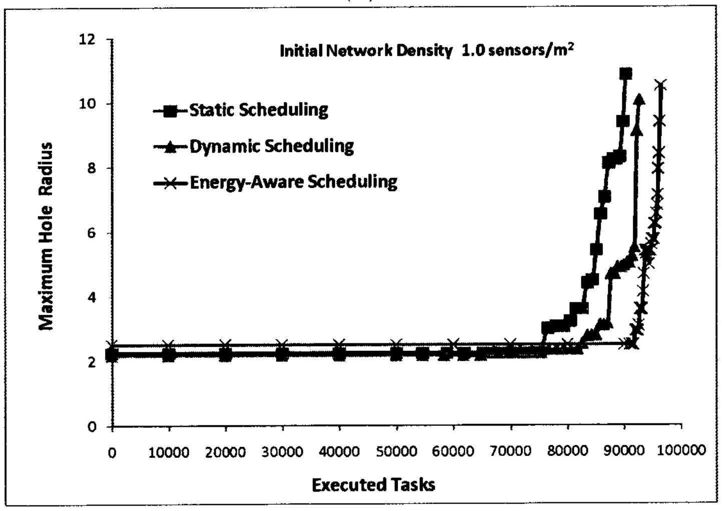
growth rate of energy holes using static, dynamic and energy-aware scheduling under different deployment densities. The initial deployment densities of Figures 42-(a),42-(b), and 42-(f) are respectively 0.3, 0.7, and 1.0. The steep slope of the curve of the energy-aware scheduling protocol shows that it can reduce the rate at which holes appear in the network under different network densities.



(a)



(b)



(c)

FIG. 42: *A comparison between the growth rate of energy holes throughout* 0.2-*reliable-lifetime of the network using different scheduling protocols for different network densities. (a)* $\rho = 0.3$*, (b)* $\rho = 0.7$*, (c)* $\rho = 1.0$

CHAPTER VI
CONCLUSIONS

This work has explored the construction and the advantages of having a virtual backbone for WSNs. Specifically, we proposed a new backbone construction protocol for WSNs where, in addition to the tiny sensors, several more powerful devices known as sinks have been deployed. The protocol works equally well with static or mobile sinks as long as they are capable of omnidirectional as well as directional transmission. Initially, the deployment area around a given sink is tiled using identical regular hexagons. Backbone sensors are selected to be the closest sensors to the centers of the hexagons they represent. Collectively, all the backbone sensors in a certain disk around the sink is referred to as the "Network Backbone" relative to that sink. We also introduced the concept of "Backbone Switching", that can be used to create alternative backbones by rotating the tiling hexagons for different arbitrary angles. When several alternative backbones are available, the network can periodically switch between them to balance energy consumption among sensors.

The main advantage of the constructed backbone is to help mitigate many of the challenges inherent to sensor networks. In addition to sensor localization, the proposed protocol provides an implicit clustering mechanism in which each hexagon represents a cluster and the backbone sensor around the center of the hexagon is the cluster head. Moreover, we showed how the proposed backbone can simplify many of network management tasks including data aggregation and geographic routing. We also point to how using mobile sinks on top of our backbone can reduce the growth rate of energy holes within the network.

In addition to that, we showed the usefulness of the proposed backbone in energy aware task management and workforce selection. We started by demonstrating how tasking sensors improperly can affect the reliability and the durability of the network by reducing network density, creating energy holes, and isolating the network into disconnected islands. After that, we proposed one centralized and another distributed workforce selection protocols for maximizing network lifetime by balancing task load among sensors within the same sensing area. The centralized approach depends on running a contention-based bidding rounds to select required workforce for any task based on sensor remaining energy. Backbone sensors play a crucial rule in coordinating the bidding process and in aggregating and forwarding results to the sink node after task completion. On the other hand, the distributed approach works in two phases. In the first phase, sensors within the task sensing range runs a distributed protocol to estimate the maximum energy among themselves. After that, and in the second phase sensors join the workforce in decision rounds based on their distance to the task center and the difference between their current energy and the maximum energy determined in the first phase. Again, after task completion, sensory results are aggregated and forwarded toward the sink node through backbone sensors.

Due to the importance of using an appropriate sleep scheduling scheme on the coverage capability of the network, we dedicated a separate chapter to discuss different scheduling schemes and to demonstrate how our backbone can be useful for this purpose. In Chapter V, We showed that when the monitored events in an area occur according to a Poisson process, the PASTA property can be applied to the effective sensor density (ESD) (i.e. density of awake sensors) as seen from the perspective of the monitored events. Specifically, we showed that under a mild technical condition the limiting fraction of the events that find k awake sensors in a certain area equals the fraction of time that this area is under the surveillance of k awake sensors. This result has a great importance as it justifies using the time-invariant probability distribution of k-coverage to analyze different scheduling schemes. We also modeled the sleep-awake cycle of a sensor as a renewal process and derived a general expression for the limiting (i.e. time-independent) probability of a sensor to be awake at a given moment. To the best of our knowledge this is the first time such a result is obtained for sensor networks.

We have developed a rigorous mathematical analysis on several scheduling schemes including static, dynamic, and energy-aware scheduling. We also briefly discussed the problems inherent in each of these scheme. We proposed a backbone-guided energy-aware scheduling scheme designed to extend network reliable lifetime by balancing energy consumption among sensor nodes. The main idea of the proposed scheme is to continuously and probabilistically adjust sleep and awake times of sensors based on differences in their remaining energy. In particular, the proposed scheme tries to prolong the sleeping periods of sensors with relatively low energy and compensate for their absence by shortening sleeping periods of sensors with relatively high energy. The main challenge is to do this without reducing the effective density of the network. To achieve this goal, we conducted a probabilistic analysis to determine necessary parameters needed for the adjustment process.

We have conducted extensive simulation experiments to simulate the performance of our backbone under different network sizes and network densities. We also considered different deployments in noisy and noise-free environments. Simulation results showed that our backbone construction protocol can construct strongly connected backbones that is well distributed across the whole network. The results also proved that our backbone can be very useful in mitigating many of the typical challenges inherent to sensor network design. For instance, when we compared the localization accuracy achieved by our backbone against other localization techniques known in the literature, simulation results showed that using the same number of sink nodes (anchors), the localization accuracy achieved by our backbone is higher than those achieved by (e.g. DV-HOP and APIT). Simulation results also demonstrated that backbone-guided task management and scheduling can increase the reliable-lifetime of the network by evenly consuming sensors energy and reducing energy differences between sensors within the same sensing area.

VI.1 FUTURE RESEARCH DIRECTIONS

Despite the encouraging results, many important challenges and research questions remained unanswered. In this section, we list several future research directions through which this work can be extended.

- In the construction protocol, it would be useful to reduce the amount of inaccuracy due to RSS; one way around this difficulty would be to estimate the initial distance through several readings and to continuously update this estimate form different messages received throughout the network lifetime.

- Given the limited on-board energy budget available to sensors, it would be of interest to see how far can one streamline the computational requirements of the construction protocol. In other words, in the current version of the construction protocol, backbone sensors in different columns are selected simultaneously, can we also parallelize the selection of backbone sensors in different rows? The answer to this question is yes, one way to do this is to let the sink node estimate the positions of all the hexagon centers and to broadcast them to all the sensors. After that, sensors in different hexagons can simultaneously estimate the closest sensor to the center of each hexagon.

- Sensor Synchronization is one of the most challenging problems in sensor networks. It would be of interest to show, if possible, how our proposed backbone can be useful in simplifying the synchronization problem.
- Network security is something that we completely overlooked in our presented work. Revisiting the proposed protocols form a security point of view would definitely open a new dimension of research challenges.

This all promise to be exciting research directions for future work.

BIBLIOGRAPHY

[1] T. Bokareva, W. Hu, S. Kanhere, B. Ristic, N. Gordon, T. Bessell, M. Rutten, and S. Jha. Wireless sensor networks for battlefield surveillance. In *Proceedings of The Land Warfare Conference (LWC)*, Brisbane, Queensland, Australia, 2006.

[2] Andrew S. Tanenbaum, Chandana Gamage, and Bruno Crispo. Taking sensor networks from the lab to the jungle. *Computer*, 39(8):98–100, 2006.

[3] Ricardo Tiago Luis Bernardo, Rodolfo Oliveira and Paulo Pinto. A fire monitoring application for scattered wireless sensor networks: A peer-to-peer cross-layering approach. In *WINSYS '07: Proceedings of the 2nd International Conference on Wireless Information Networks and Systems*, 2007.

[4] Thomas Haenselmann, Thomas King, Marcel Busse, Wolfgang Effelsberg, and Markus Fuchs. Scriptable sensor network based home-automation. In *EUC Workshops*, volume 4809 of *Lecture Notes in Computer Science*, pages 579–591. Springer, 2007.

[5] Wenjie Chen, Lifeng Chen, Zhang long Chen, and Shi liang Tu. A realtime dynamic traffic control system based on wireless sensor network. In *ICPP Workshops*, pages 258–264. IEEE Computer Society, 2005.

[6] Chris R. Baker et al. Wireless sensor networks for home health care. In *AINAW '07: Proceedings of the 21st International Conference on Advanced Information Networking and Applications Workshops*, pages 832–837, Washington, DC, USA, 2007. IEEE Computer Society.

[7] Huaming Li and Jindong Tan. Heartbeat driven medium access control for body sensor networks. In *HealthNet '07: Proceedings of the 1st ACM SIGMOBILE international workshop on Systems and networking support for healthcare and assisted living environments*, pages 25–30, New York, NY, USA, 2007. ACM.

[8] E. Elnahrawy, Xiaoyan Li, and R.P. Martin. The limits of localization using signal strength: a comparative study. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 406–414, Oct. 2004.

[9] A. Wadaa, S. Olariu, L. Wilson, M. Eltoweissy, and K. Jones. Training a wireless sensor network. *Mobile Networks and Applications*, 10:151–168, 2005.

[10] S. Olariu and I. Stojmenovic. Design guidelines for maximizing lifetime and avoiding energy holes in sensor networks with uniform distribution and uniform reporting. In *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pages 1–12, April 2006.

[11] Alan A. Bertossi, Stephan Olariu, and Cristina M. Pinotti. Efficient corona training protocols for sensor networks. *Theor. Comput. Sci.*, 402(1):2–15, 2008.

[12] Anand Srinivas, Gil Zussman, and Eytan Modiano. Construction and maintenance of wireless mobile backbone networks. *IEEE/ACM Trans. Netw.*, 17(1):239–252, 2009.

[13] Hannes Frey and Daniel Gorgen. Geographical cluster-based routing in sensing-covered networks. *IEEE Trans. Parallel Distrib. Syst.*, 17(9):899–911, 2006.

[14] N. Bulusu, J. Heidemann, and D. Estrin. Gps-less low-cost outdoor localization for very small devices. *Personal Communications, IEEE Wireless Communications*, 7(5):28–34, Oct 2000.

[15] S. Lederer, Yue Wang, and Jie Gao. Connectivity-based localization of large scale sensor networks with complex shape. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 789–797, April 2008.

[16] Tian He, Chengdu Huang, Brian M. Blum, John A. Stankovic, and Tarek Abdelzaher. Range-free localization schemes for large scale sensor networks. In *MobiCom '03: Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 81–95, New York, NY, USA, 2003. ACM.

[17] D. Niculescu and Badri Nath. Ad hoc positioning system (aps) using aoa. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, pages 1734–1743, 30 March-3 April 2003.

[18] Radhika Nagpal, Howard E. Shrobe, and Jonathan Bachrach. Organizing a global coordinate system from local information on an ad hoc sensor network. In *IPSN*, pages 333–348, 2003.

[19] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43, New York, NY, USA, 2000. ACM.

[20] Andreas Savvides, Chih-Chieh Han, and Mani B. Strivastava. Dynamic fine-grained localization in ad-hoc networks of sensors. In *MobiCom '01: Proceedings of the 7th annual international conference on Mobile computing and networking*, pages 166–179, New York, NY, USA, 2001. ACM.

[21] Kay Römer. The lighthouse location system for smart dust. In *MobiSys '03: Proceedings of the 1st international conference on Mobile systems, applications and services*, pages 15–30, New York, NY, USA, 2003. ACM.

[22] Chunyi Peng, Guobin Shen, Yongguang Zhang, Yanlin Li, and Kun Tan. Beep-beep: a high accuracy acoustic ranging system using cots mobile devices. In *SenSys '07: Proceedings of the 5th international conference on Embedded networked sensor systems*, pages 1–14, New York, NY, USA, 2007. ACM.

[23] Zhong Zhou, Jun-Hong Cui, and A. Bagtzoglou. Scalable localization with mobility prediction for underwater sensor networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 2198–2206, April 2008.

[24] Wei Cheng, A.Y. Teymorian, Liran Ma, Xiuzhen Cheng, Xicheng Lu, and Zexin Lu. Underwater localization in sparse 3d acoustic sensor networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 236–240, April 2008.

[25] Jaehoon Jeong, Shuo Guo, Tian He, and D. Du. Apl: Autonomous passive localization for wireless sensors deployed in road networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages 583–591, April 2008.

[26] Qing Cao, Tarek Abdelzaher, Tian He, and John Stankovic. Towards optimal sleep scheduling in sensor networks for rare-event detection. In *IPSN '05: Proceedings of the 4th international symposium on Information processing in sensor networks*, page 4, Piscataway, NJ, USA, 2005. IEEE Press.

[27] G. Lu, N. Sadagopan, B. Krishnamachari, and A. Goel. Delay efficient sleep scheduling in wireless sensor networks. In *INFOCOM 2005. Proceedings of 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, pages 2470–2481, March 2005.

[28] Ramanan Subramanian and Faramarz Fekri. Sleep scheduling and lifetime maximization in sensor networks: fundamental limits and optimal solutions. In *IPSN '06: Proceedings of the 5th international conference on Information processing in sensor networks*, pages 218–225, New York, NY, USA, 2006. ACM.

[29] Jing Deng, Yunghsiang S. Han, W.B. Heinzelman, and P.K. Varshney. Balanced-energy sleep scheduling scheme for high density cluster-based sensor networks. In *The 4th workshop on Applications and Services in Wireless Networks*, pages 99–108, August 2004.

[30] Zhenning Kong and Edmund M. Yeh. Distributed energy management algorithm for large-scale wireless sensor networks. In *MobiHoc '07: Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 209–218, New York, NY, USA, 2007. ACM.

[31] Hady S. AbdelSalam and Stephan Olariu. Bees: Bio-inspired backbone selection in wireless sensor networks. *Transactions on Parallel and Distributed Systems*, (to appear).

[32] Hady S. AbdelSalam and Stephan Olariu. A lightweight skeleton construction algorithm for self-organizing sensor networks. In *Proceedings of IEEE International Conference on Communications 2009 (ICC'09)*, Dresden, Germany, June 2009.

[33] Fabian Garcia Nocetti, Ivan Stojmenovic, and Jingyuan Zhang. Addressing and routing in hexagonal networks with applications for tracking mobile users and connection rerouting in cellular networks. *IEEE Trans. Parallel Distrib. Syst.*, 13(9):963–971, 2002.

[34] Ivan Stojmenovic. Honeycomb networks: Topological properties and communication algorithms. *IEEE Transactions on Parallel and Distributed Systems*, 8:1036–1042, 1997.

[35] Constantine A. Balanis. *Antenna Theory Analysis and Design*, pages 31–32. John Wiley & Sons, Inc., 2nd edition, 1997.

[36] D.E. Knuth. *The Art of Computer Programming: Seminumerical Algorithms*, volume 2, pages 467–469. Addison-Wesley, Reading, MA, 3rd edition, 1998.

[37] Hady S. AbdelSalam and Stephan Olariu. Hexnet: Hexagon-based localization technique for wireless sensor networks. In *PERCOMW '09: Proceedings of the Seventh IEEE International Conference on Pervasive Computing and Communications Workshops*, Galveston, Texas, March 2009. IEEE Computer Society.

[38] Hady S. AbdelSalam and Stephan Olariu. Towards efficient task management in wireless sensor networks. *Transactions on Computer*, (to appear).

[39] Hady S. AbdelSalam, Sayed R. Rizvi, Scott Ainsworth, and Stephan Olariu. A durable sensor enabled lifeline support for firefighters. In *INFOCOM Workshops (Mission Critical Networks), IEEE*, pages 1 -6, Phoenix, AZ, April 2008.

[40] Hady S. AbdelSalam and Syed R. Rizvi. Energy efficient workforce selection in special-purpose wireless sensor networks. In *INFOCOM Students Workshop, IEEE*, pages 1–4, Phoenix, AZ, April 2008.

[41] Scott Ainsworth, Hady AbdelSalam, and Syed Rizvi. Towards maximum longevity of energy limited sanets. In *SANET '07: Proceedings of the First ACM workshop on Sensor and actor networks*, pages 61–62, New York, NY, USA, 2007. ACM.

[42] Hady S. AbdelSalam and Stephan Olariu. Energy-based task load balancing in wireless sensor networks. In *THASN'08, First Workshop on Theory of Ad Hoc and Sensor Networks held in conjunction with MASS 2008*, pages 778–783, 29 2008-Oct. 2 2008.

[43] Hady S. AbdelSalam, Syed R. Rizvi, and Stephan Olariu. Energy-aware task assignment and data aggregation protocols in wireless sensor networks. In *The 6th IEEE Consumer Communications and Networking Conference, 2009. CCNC 2009.*, pages 1–5, Jan. 2009.

[44] Ronald W. Wolff. Poisson arrivals see time averages. *Operations Research*, 30(2):223–231, 1982.

[45] Emanuel Parzen. *Stochastic Processes*, pages 182–183. Holden-Day, Inc., 1962.

[46] Xiaobing Wu, Guihai Chen, and Sajal K. Das. On the energy hole problem of nonuniform node distribution in wireless sensor networks. *Mobile Adhoc and Sensor Systems (MASS), 2006 IEEE International Conference on*, pages 180–187, Oct. 2006.

[47] Hady S. AbdelSalam and Stephan Olariu. Towards adaptive sleep schedules for balancing energy consumption in wireless sensor networks. *Transactions on Computer*, (submitted).

[48] Hady S. AbdelSalam and Stephan Olariu. On prolonging network lifetime by adjusting sleep/awake cycles in wireless sensor networks. In *INSS'09: Proceedings of the 6th international conference on Networked sensing systems*, pages 10–15, Piscataway, NJ, USA, 2009. IEEE Press.

[49] Roberto Casas and Oscar Casas. Battery sensing for energy-aware system design. *Computer*, 38(11):48–54, 2005.

[50] Victor Shnayder, Mark Hempstead, Bor-rong Chen, Geoff Werner Allen, and Matt Welsh. Simulating the power consumption of large-scale sensor network applications. In *SenSys '04: Proceedings of the 2nd international conference on Embedded networked sensor systems*, pages 188–200, New York, NY, USA, 2004. ACM.

APPENDIX A

DISTRIBUTION AND EXPECTATION OF $\frac{X}{X+Y}$

Given the uniformly distributed random variables X and Y defined as shown below,

$$X \in U[a,b], \qquad 0 \le a \le b$$

$$Y \in U[c,d], \qquad 0 \le c \le d$$

We define the random variable Z as,

$$Z = \frac{X}{X+Y}; \qquad 0 < a+c$$

We are interested in $F_Z(Z)$, $\forall Z \in R$ as well as $E[Z]$.

Since X and Y are positive, so is Z. Moreover, It is straightforward to realize that,

$$\frac{a}{a+d} \le Z \le \frac{b}{b+c}$$

Consequently,

$$F_Z(Z) = 0 \qquad \forall Z \le \frac{a}{a+d}$$

and

$$F_Z(Z) = 1 \qquad \forall Z \ge \frac{b}{b+d}$$

Hence, from now on, we assume that

$$Z \in \left[\frac{a}{a+d}, \frac{b}{b+c}\right]$$

Initially, we observe that,

$$\left\{\frac{X}{X+Y} \le z\right\} \Leftrightarrow \left\{\frac{X}{z} \le X+Y\right\} \Leftrightarrow \left\{X \cdot \frac{1-z}{z} \le Y\right\}$$

Now consider the planar domain

$$D = \left\{(u,v) | a \le u \le b; c \le v \le d; \frac{1-z}{z}u \le v\right\}$$

In this notation, $\forall z \in \left[\frac{a}{a+d}, \frac{b}{b+c}\right]$,

$$F_Z(z) = P[(u,v) \in D] = \frac{|D|}{(d-c)(b-a)}$$

Consider the line δ defined by the following equation

$$v = \frac{1-z}{z}u$$

Each value of z determines the slope of the line δ which in turn determines the boundary of the area of interest D.

Now, we distinguish between the following two cases.

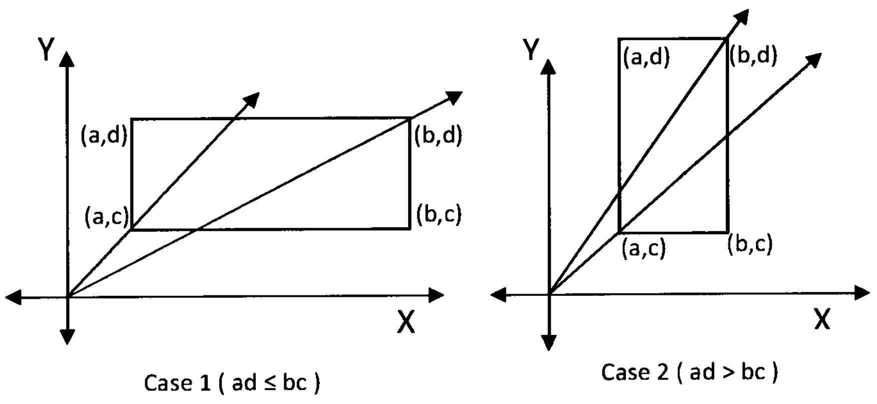


FIG. 43: *An illustration of different possible cases or different ranges of the random variables X and Y*

A.0.1 Case 1 ($ad \leq bc$)

This is equivalent to $\frac{d}{b} \leq \frac{c}{a}$. In other words, the slope of the line passes through $(0,0)$ and (b,d) does not exceed the slope of the line passes through $(0,0)$ and (a,c). As shown in Figure 44, depending on where δ intersects with the line of equation $v = c$, we have the following subcases:

- Subcase 1.1: $\frac{a}{a+d} \leq z < \frac{a}{a+c}$, the intersection points of the line $v = \frac{1-z}{z}u$ with the rectangle are $(a, \frac{(1-z)a}{z})$ and $(\frac{zd}{1-z}, d)$. It follows that our area of interest D is the triangle whose vertices are the points (a,d), $(a, \frac{(1-z)a}{z})$, and $(\frac{zd}{1-z}, d)$ which can be evaluated as,

$$\begin{aligned} |D| &= \frac{1}{2}\left(\frac{zd}{1-z} - a\right)\left(d - \frac{(1-z)a}{z}\right) \\ &= \frac{[(a+d)z - a]^2}{2z(1-z)} \end{aligned}$$

  Hence, the required probability is evaluated by,

$$P[\{Z \leq z\}] = \frac{[(a+d)z - a]^2}{\Delta z(1-z)} \tag{90}$$

- Subcase 1.2: $\frac{a}{a+c} \leq z < \frac{b}{b+d}$, the intersection points of the line $v = \frac{1-z}{z}u$ with the rectangle are $(\frac{zc}{1-z}, c)$ and $(\frac{zd}{1-z}, d)$. In this subcase, D is the trapezoid whose vertices are the points $(\frac{zc}{1-z}, c)$, (a,c), (a,d),and $(\frac{zd}{1-z}, d)$. Hence,

$$\begin{aligned} |D| &= \frac{(d-c)}{2}\left[\left(\frac{zc}{1-z}-a\right)+\left(\frac{zd}{1-z}-a\right)\right] \\ &= \frac{(d-c)}{2(1-z)}[(2a+d+c)z-2a] \\ P[\{Z \leq z\}] &= \frac{(d-c)[(2a+d+c)z-2a]}{2(1-z)(d-c)(b-a)} \\ &= \frac{(d-c)[(2a+d+c)z-2a]}{\Delta(1-z)} \end{aligned} \tag{91}$$

- Subcase 1.3: $\frac{b}{b+d} \leq z \leq \frac{b}{b+c}$, the intersection points are $(\frac{zc}{1-z}, c)$ and $(b, \frac{(1-z)b}{z})$. In this subcase, apparently, it is more appealing to start by evaluating the area of $\bar{D}$ which is a triangle whose vertices are the points $(\frac{zc}{1-z}, c)$ (b,c), and $(b, \frac{(1-z)b}{z})$. The area of $\bar{D}$ can be evaluated as,

$$\begin{aligned} |\bar{D}| &= \frac{1}{2}\left(b-\frac{zc}{1-z}\right)\left(\frac{(1-z)b}{z}-c\right) \\ &= \frac{[b-(b+c)z]^2}{2z(1-z)} \end{aligned}$$

Consequently,

$$\begin{aligned} P[\{Z \leq z\}] &= \frac{|D|}{(d-c)(b-a)} = 1-\frac{|\bar{D}|}{(d-c)(b-a)} \\ &= 1-\frac{[b-(b+c)z]^2}{\Delta z(1-z)} \end{aligned} \tag{92}$$

From equations (90),(91), and (92), it follows that in the case $ad \leq bc$, $F_Z(z)$ can be defined as,

$$F_Z(z) = \begin{cases} 0 & z < \frac{a}{a+d} \\ \frac{((a+d)z-a)^2}{\Delta z(1-z)} & \frac{a}{a+d} \leq z < \frac{a}{a+c} \\ \frac{(d-c)[(2a+d+c)z-2a]}{\Delta(1-z)} & \frac{a}{a+c} \leq z < \frac{b}{b+d} \\ 1-\frac{(b-(b+c)z)^2}{\Delta z(1-z)} & \frac{b}{b+d} \leq z < \frac{b}{b+c} \\ 1 & \frac{b}{b+c} \leq z \end{cases} \tag{93}$$

A.0.2 Case 2 $(ad > bc)$

The case condition is equivalent to $\frac{d}{b} > \frac{c}{a}$. In other wards, the slope of the line passes through $(0,0)$ and (b,d) exceeds the slope of the line passes through $(0,0)$ and (a,c). In general, case 2 is very similar to case 1 and can be handled in a similar

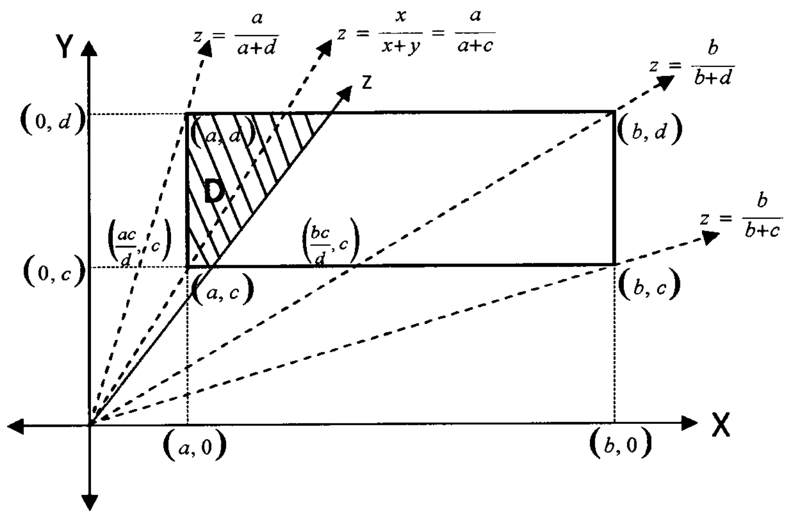


FIG. 44: *An illustration of the different subcases of Case 1*

way. Again, and as shown in Figure 45, depending on where δ intersects with the line of equation $v = c$, we have the following subcases:

- Subcase 2.1: $\frac{a}{a+d} \leq z < \frac{b}{b+d}$, the intersection points of the line $v = \frac{1-z}{z}u$ with the rectangle are $(a, \frac{(1-z)a}{z})$ and $(\frac{zd}{1-z}, d)$. It follows that our area of interest, D, is the triangle whose vertices are the points (a, d), $(a, \frac{(1-z)a}{z})$, and $(\frac{zd}{1-z}, d)$ which can be evaluated as,

$$|D| = \frac{[(a+d)z - a]^2}{2z(1-z)}$$

Hence, the required probability is evaluated by,

$$P[\{Z \leq z\}] = \frac{[(a+d)z - a]^2}{\Delta z(1-z)} \qquad (94)$$

- Subcase 2.2: $\frac{b}{b+d} \leq z < \frac{a}{a+c}$, the intersection points of the line $v = \frac{1-z}{z}u$ with the rectangle are $(a, \frac{(1-z)a}{z})$ and $(b, \frac{(1-z)b}{z})$. In this subcase, D is the trapezoid

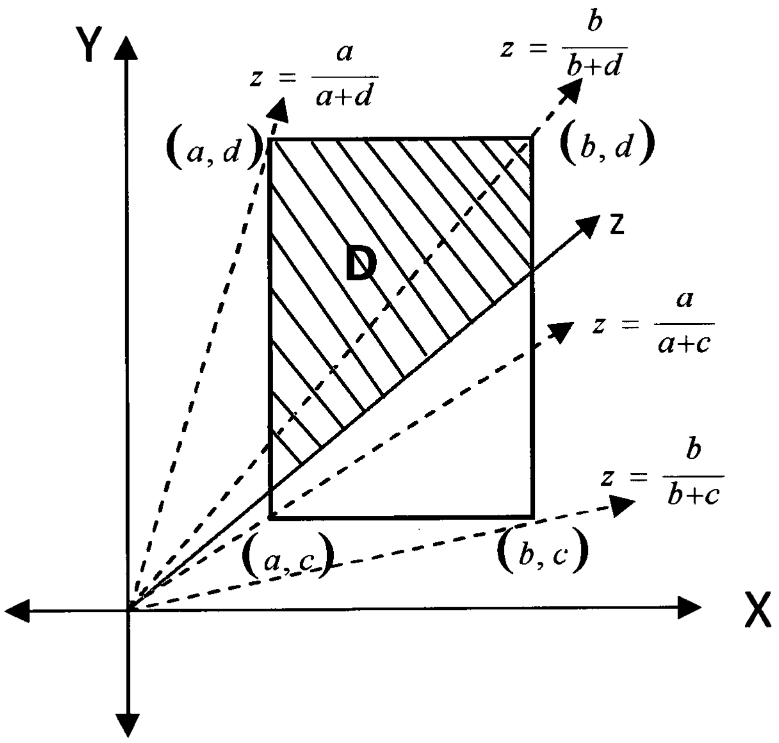


FIG. 45: *An illustration of the different subcases of Case 2*

whose vertices are the points $(a, \frac{(1-z)a}{z})$, (a, d), (b, d), and $(b, \frac{(1-z)b}{z})$. Hence,

$$\begin{aligned} |D| &= \frac{(b-a)}{2}\left[\left(d - \frac{(1-z)a}{z}\right) + \left(d - \frac{(1-z)b}{z}\right)\right] \\ &= \frac{(b-a)}{2z}[(2d+a+b)z - a - b] \\ P[\{Z \le z\}] &= \frac{(b-a)[(2d+a+b)z-a-b]}{2z(d-c)(b-a)} \\ &= \frac{(b-a)[(2d+a+b)z-a-b]}{\Delta z} \end{aligned} \tag{95}$$

- Subcase 2.3: $\frac{a}{a+c} \le z \le \frac{b}{b+c}$, the intersection points are $(\frac{zc}{1-z}, c)$ and $(b, \frac{(1-z)b}{z})$. Again, it is more appealing to start by evaluating the area of $\bar{D}$ which is a triangle whose vertices are the points $(\frac{zc}{1-z}, c)$ (b, c), and $(b, \frac{(1-z)b}{z})$. The area of

$\bar{D}$ can be evaluated as,

$$\begin{aligned}|\bar{D}| &= \frac{1}{2}\left(b - \frac{zc}{1-z}\right)\left(\frac{(1-z)b}{z} - c\right) \\ &= \frac{[b-(b+c)z]^2}{2z(1-z)}\end{aligned}$$

Consequently,

$$\begin{aligned}P[\{Z \le z\}] &= \frac{|D|}{(d-c)(b-a)} = 1 - \frac{|\bar{D}|}{(d-c)(b-a)} \\ &= 1 - \frac{[b-(b+c)z]^2}{\Delta z(1-z)} \qquad (96)\end{aligned}$$

From equations (94),(95), and (96), it follows that in the case $ad > bc$, $F_Z(z)$ can be defined as follows,

$$F_Z(z) = \begin{cases} 0 & z < \frac{a}{a+d} \\ \frac{((a+d)z-a)^2}{\Delta z(1-z)} & \frac{a}{a+d} \le z < \frac{b}{b+d} \\ \frac{(b-a)[(2d+a+b)z-a-b]}{\Delta z} & \frac{b}{b+d} \le z < \frac{a}{a+c} \\ 1 - \frac{(b-(b+c)z)^2}{\Delta z(1-z)} & \frac{a}{a+c} \le z < \frac{b}{b+c} \\ 1 & \frac{b}{b+c} \le z \end{cases} \qquad (97)$$

A.1 EVALUATING $E[Z]$

Now, we turn our attention to the evaluation of $E[Z]$. In particular, we show that,

$$\begin{aligned}E\left[\frac{X}{X+Y}\right] &= \frac{1}{2} + \frac{c^2-b^2}{\Delta}\ln\left(\frac{b+c}{b+d}\right) + \frac{d^2-c^2}{\Delta}\ln\left(\frac{a+c}{b+d}\right) + \frac{d^2-a^2}{\Delta}\ln\left(\frac{a+d}{a+c}\right) \\ &= \frac{1}{2} + \frac{c^2}{\Delta}\ln\left(1+\frac{b-a}{a+c}\right) + \frac{b^2}{\Delta}\ln\left(1+\frac{d-c}{b+c}\right) - \\ &\quad \frac{d^2}{\Delta}\ln\left(1+\frac{b-a}{a+d}\right) - \frac{a^2}{\Delta}\ln\left(1+\frac{d-c}{a+c}\right)\end{aligned}$$

where $\Delta = 2(b-a)(d-c)$.

Because the distribution function is different, we again have the same two cases we pointed out earlier.

A.1.1 Case 1 ($ad \leq bc$):

To prove this, we find it more appropriate to start the evaluation of $E[Z]$ using the following formula,

$$\begin{aligned} E[Z] &= \int_0^\infty [1 - F_Z(z)]\, dz = \int_0^{\frac{a}{a+d}} dz + \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} [1 - F_Z(z)]\, dz + \\ & \int_{\frac{a}{a+c}}^{\frac{b}{b+d}} [1 - F_Z(z)]\, dz + \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} [1 - F_Z(z)]\, dz \end{aligned} \tag{98}$$

$$\int_0^{\frac{a}{a+d}} dz = \frac{a}{a+d} \tag{99}$$

$$\begin{aligned} \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} [1 - F_Z(z)]\, dz &= \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} 1 - \frac{((a+d)z - a)^2}{\Delta z(1-z)} dz \\ &= \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} 1 - \frac{(a+d)^2 z^2 - 2a(a+d)z + a^2}{\Delta z(1-z)} dz \\ &= \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} 1 + \frac{(a+d)^2}{\Delta} \cdot \left(\frac{1 - z - 1}{1 - z} \right) + \frac{1}{\Delta} \cdot \frac{2a(a+d)z - a^2}{z(1-z)} dz \\ &= \left(1 + \frac{(a+d)^2}{\Delta}\right) \frac{a(d-c)}{(a+d)(a+c)} + \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{-a^2 z - d^2 z + 2a^2 z - a^2}{\Delta z(1-z)} dz \end{aligned} \tag{100}$$

However,

$$\begin{aligned} \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{a^2 z - d^2 z - a^2}{\Delta z(1-z)} dz &= \frac{1}{\Delta} \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{(a^2 - d^2)}{(1-z)} - \frac{a^2}{z(1-z)} dz \\ &= \frac{1}{\Delta} \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{(a^2 - d^2)}{(1-z)} - \frac{a^2}{z} - \frac{a^2}{(1-z)} dz \\ &= \frac{1}{\Delta} \int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{-d^2}{(1-z)} - \frac{a^2}{z} dz \\ &= \frac{d^2}{\Delta} \left(\ln \frac{\frac{a+c-a}{a+c}}{\frac{a+d-a}{a+d}} \right) - \frac{a^2}{\Delta} \left(\ln \frac{\frac{a}{a+c}}{\frac{a}{a+d}} \right) \\ &= \frac{d^2}{\Delta} \left(\ln \frac{a+d}{a+c} + \ln \frac{c}{d} \right) - \frac{a^2}{\Delta} \left(\ln \frac{a+d}{a+c} \right) \end{aligned} \tag{101}$$

Substituting equation (101) into equation (100) yields,

$$\int_{\frac{a}{a+d}}^{\frac{a}{a+c}} [1 - F_Z(z)]\, dz = \frac{d^2 - a^2}{\Delta} \ln \frac{a+d}{a+c} - \frac{d^2}{\Delta} \ln \frac{d}{c} + \frac{a(d-c)}{(a+c)(a+d)} \left[1 + \frac{(a+d)^2}{\Delta}\right] \tag{102}$$

And similarly,

$$\begin{aligned}
\int_{\frac{a}{a+c}}^{\frac{b}{b+d}} [1 - F_Z(z)]\, dz &= \int_{\frac{a}{a+c}}^{\frac{b}{b+d}} 1 - \frac{(d-c)[(2a+d+c)z - 2a]}{\Delta(1-z)}\, dz \\
&= \int_{\frac{a}{a+c}}^{\frac{b}{b+d}} 1 + \frac{(d-c)(2a+d+c)(1-z-1)}{\Delta(1-z)} + \frac{2a(d-c)}{\Delta(1-z)}\, dz \\
&= \int_{\frac{a}{a+c}}^{\frac{b}{b+d}} 1 + \frac{(d-c)(2a+d+c)}{\Delta} + \tfrac{2a(d-c)-(d-c)(2a+d+c)}{\Delta(1-z)}\, dz \\
&= \int_{\frac{a}{a+c}}^{\frac{b}{b+d}} \frac{\Delta + (d-c)(2a+d+c)}{\Delta} - \frac{(d-c)(d+c)}{\Delta(1-z)}\, dz \\
&= \tfrac{(d-c)(2(b-a)+(2a+d+c))}{\Delta} \cdot \tfrac{(bc-ad)}{(a+c)(b+d)} + \tfrac{(d^2-c^2)}{\Delta} \ln \tfrac{1-\frac{b}{b+d}}{1-\frac{a}{a+c}} \\
&= \tfrac{d^2-c^2}{\Delta} \ln \tfrac{a+c}{b+d} + \tfrac{d^2-c^2}{\Delta} \ln \tfrac{d}{c} + \tfrac{(bc-ad)(d-c)(2b+d+c)}{\Delta(a+c)(b+d)} \qquad (103)
\end{aligned}$$

And also,

$$\begin{aligned}
\int_{\frac{b}{b+d}}^{\frac{b}{b+c}} [1 - F_Z(z)]\, dz &= \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} \frac{(b-(b+c)z)^2}{\Delta z(1-z)}\, dz \\
&= \frac{1}{\Delta} \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} \frac{b2 - 2(b+c)bz + (b+c)^2 z^2}{z(1-z)}\, dz \\
&= \frac{1}{\Delta} \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} \tfrac{b2}{z} + \tfrac{b2}{1-z} - \tfrac{2(b+c)b}{1-z} - \tfrac{(b+c)^2(1-z-1)}{1-z}\, dz \\
&= \frac{1}{\Delta} \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} -(b+c)^2 + \frac{b2}{z} - \frac{b^2 + 2bc - (b+c)^2}{1-z}\, dz \\
&= \frac{(b+c)^2}{\Delta} \frac{b(c-d)}{(b+d)(b+c)} + \frac{1}{\Delta} \int_{\frac{b}{b+d}}^{\frac{b}{b+c}} \frac{b2}{z} + \frac{c^2}{1-z}\, dz \\
&= -\frac{b(b+c)}{2(b-a)(b+d)} + \frac{1}{\Delta} \left(b^2 \ln \frac{\frac{b}{b+c}}{\frac{b}{b+d}} - c^2 \ln \frac{\frac{c}{b+c}}{\frac{d}{b+d}} \right) \\
&= -\tfrac{b(b+c)}{2(b-a)(b+d)} + \tfrac{1}{\Delta} \left(-b^2 \ln \tfrac{b+c}{b+d} - c^2 \ln \tfrac{c}{d} + c^2 \ln \tfrac{b+c}{b+d} \right) \\
&= \frac{c^2}{\Delta} \ln \frac{d}{c} + \frac{c^2 - b^2}{\Delta} \ln \frac{b+c}{b+d} - \frac{b(b+c)}{2(b-a)(b+d)} \qquad (104)
\end{aligned}$$

Now, gathering all the pieces together,

$$\begin{aligned}
E[Z] &= \tfrac{a}{a+d} + \left[\tfrac{-d^2}{\Delta} + \tfrac{d^2-c^2}{\Delta} + \tfrac{c^2}{\Delta} \right] \ln \tfrac{d}{c} + \tfrac{d^2-a^2}{\Delta} \ln \tfrac{a+d}{a+c} + \tfrac{d^2-c^2}{\Delta} \ln \tfrac{a+c}{b+d} + \tfrac{c^2-b^2}{\Delta} \ln \tfrac{b+c}{b+d} \\
&\quad + \tfrac{a(d-c)}{(a+c)(a+d)} + \tfrac{a(d-c)(a+d)^2}{(a+c)(a+d)\Delta} + \tfrac{(bc-ad)(d-c)(2b+d+c)}{\Delta(a+c)(b+d)} - \tfrac{b(b+c)}{2(b-a)(b+d)} \qquad (105)
\end{aligned}$$

Interestingly, the non-logarithmic terms can be simplified as follows,

$$\begin{aligned}
I_1 &= \frac{(bc-ad)(d-c)(2b+d+c)}{\Delta(a+c)(b+d)} - \frac{b(b+c)}{2(b-a)(b+d)} \\
&= \frac{(bc-ad)((b+c)+(b+d)) - b(b+c)(a+c)}{2(b-a)(a+c)(b+d)} \\
&= \frac{bc(b+c)+bc(b+d)-ad(b+c)-ad(b+d)-ba(b+c)-bc(b+c)}{2(b-a)(a+c)(b+d)} \\
&= \frac{bc(b+d)-ad(b+c)-ad(b+d)-ba(b+c)}{2(b-a)(a+c)(b+d)} \\
&= \frac{bc(b+d)-ad(b+d)-a(b+c)(b+d)}{2(b-a)(a+c)(b+d)} \\
&= \frac{bc-ab-ac-ad}{2(b-a)(a+c)} \\
&= \frac{abc-a^2b-a^2c-a^2d+bcd-abd-acd-ad^2}{2(b-a)(a+c)(a+d)} \\
I_2 &= \frac{a(d-c)}{(a+c)(a+d)} + \frac{a(d-c)(a+d)^2}{(a+c)(a+d)\Delta} \\
&= \frac{2a(d-c)(b-a)+a(a+d)^2}{2(a+c)(a+d)(b-a)} \\
&= \frac{2a(d-c)(b-a)+a(a+d)^2}{2(a+c)(a+d)(b-a)} \\
&= \frac{a(2bd-2bc+2ac+a^2+d^2)}{2(a+c)(a+d)(b-a)}
\end{aligned}$$

$$\begin{aligned}
I_1+I_2 &= \tfrac{abc-a^2b-a^2c-a^2d+bcd-abd-acd-ad^2+2abd-2abc+2a^2c+a^3+ad^2}{2(b-a)(a+c)(a+d)} \\
&= \tfrac{abc-a^2b-a^2c-a^2d+bcd-abd-acd-ad^2+2abd-2abc+2a^2c+a^3+ad^2}{2(b-a)(a+c)(a+d)} \\
&= \frac{-ad(a+c)+(a+c)bd-ab(a+c)+a^2(c+a)}{2(b-a)(a+c)(a+d)} \\
&= \frac{-ad+bd-ab+a^2}{2(b-a)(a+d)} \\
&= \frac{d(b-a)-a(b-a)}{2(b-a)(a+d)} \\
&= \frac{d-a}{2(a+d)} \qquad (106)
\end{aligned}$$

Substituting the value of equation (106) into equation (105), $E[Z]$ can be written as,

$$\begin{aligned}
E[Z] &= \frac{a}{a+d} + \frac{d-a}{2(a+d)} + \frac{d^2-a^2}{\Delta}\ln\frac{a+d}{a+c} + \frac{d^2-c^2}{\Delta}\ln\frac{a+c}{b+d} + \frac{c^2-b^2}{\Delta}\ln\frac{b+c}{b+d} \\
&= \frac{1}{2} + \frac{c^2-b^2}{\Delta}\ln\frac{b+c}{b+d} + \frac{d^2-c^2}{\Delta}\ln\frac{a+c}{b+d} + \frac{d^2-a^2}{\Delta}\ln\frac{a+d}{a+c} \qquad (107)
\end{aligned}$$

Which can be further simplified and rewritten in any of the following formats,

$$\begin{aligned}
E[Z] &= \frac{1}{2} + \frac{c^2 - b^2}{\Delta}\ln(b+c) - \frac{c^2 - b^2}{\Delta}\ln(b+d) + \frac{d^2 - c^2}{\Delta}\ln(a+c) - \\
&\quad \frac{d^2 - c^2}{\Delta}\ln(b+d) + \frac{d^2 - a^2}{\Delta}\ln(a+d) - \frac{d^2 - a^2}{\Delta}\ln(a+c) \\
&= \frac{1}{2} + \frac{c^2 - b^2}{\Delta}(\ln(b+c) - \ln(b+d)) + \frac{d^2 - c^2}{\Delta}(\ln(a+c) - \ln(b+d)) + \\
&\quad \frac{d^2 - a^2}{\Delta}(\ln(a+d) - \ln(a+c)) \\
&= \frac{1}{2} + \frac{c^2 - b^2}{\Delta}\ln(b+c) + \frac{b^2 - d^2}{\Delta}\ln(b+d) + \\
&\quad \frac{a^2 - c^2}{\Delta}\ln(a+c) + \frac{d^2 - a^2}{\Delta}\ln(a+d) \qquad (108)
\end{aligned}$$

and,

$$E[Z] = \frac{1}{2} + \frac{c^2}{\Delta}\ln\frac{b+c}{a+c} + \frac{b^2}{\Delta}\ln\frac{b+d}{b+c} - \frac{d^2}{\Delta}\ln\frac{b+d}{a+d} - \frac{a^2}{\Delta}\ln\frac{a+d}{a+c} \qquad (109)$$

and also,

$$\begin{aligned}
E[Z] &= \frac{1}{2} + \frac{c^2}{\Delta}\ln\left(1 + \frac{b-a}{a+c}\right) + \frac{b^2}{\Delta}\ln\left(1 + \frac{d-c}{b+c}\right) - \\
&\quad \frac{d^2}{\Delta}\ln\left(1 + \frac{b-a}{a+d}\right) - \frac{a^2}{\Delta}\ln\left(1 + \frac{d-c}{a+c}\right) \qquad (110)
\end{aligned}$$

A.1.2 Case 2 ($ad > bc$):

Again, we start by writing $E[Z]$ as follows,

$$\begin{aligned}
E[Z] &= \int_0^\infty [1 - F_Z(z)]\,dz = \int_0^{\frac{a}{a+d}} dz + \int_{\frac{a}{a+d}}^{\frac{b}{b+d}} [1 - F_Z(z)]\,dz + \\
&\quad \int_{\frac{b}{b+d}}^{\frac{a}{a+c}} [1 - F_Z(z)]\,dz + \int_{\frac{a}{a+c}}^{\frac{b}{b+c}} [1 - F_Z(z)]\,dz \qquad (111)
\end{aligned}$$

Now, looking at the terms of equation (111) separately yields,

$$\begin{aligned}
\int_{\frac{a}{a+d}}^{\frac{b}{b+d}} [1 - F_Z(z)]\,dz &= \int_{\frac{a}{a+d}}^{\frac{b}{b+d}} 1 - \frac{((a+d)z - a)^2}{\Delta z(1-z)}\,dz \\
&= \int_{\frac{a}{a+d}}^{\frac{b}{b+d}} 1 - \frac{(a+d)^2 z^2 - 2a(a+d)z + a^2}{\Delta z(1-z)}\,dz \\
&= \int_{\frac{a}{a+d}}^{\frac{b}{b+d}} 1 + \frac{(a+d)^2}{\Delta}\cdot\left(\frac{1-z-1}{1-z}\right) + \frac{1}{\Delta}\cdot\frac{2a(a+d)z - a^2}{z(1-z)}\,dz \\
&= \left(1 + \tfrac{(a+d)^2}{\Delta}\right)\tfrac{d(b-a)}{(a+d)(b+d)} + \int_{\frac{a}{a+d}}^{\frac{b}{b+d}} \tfrac{-a^2z - d^2z + 2a^2z - a^2}{\Delta z(1-z)}\,dz \qquad (112)
\end{aligned}$$

However,

$$\begin{aligned}
\int_{\frac{a}{a+d}}^{\frac{b}{b+d}} \frac{a^2 z - d^2 z - a^2}{\Delta z(1-z)}\, dz &= \frac{1}{\Delta}\int_{\frac{a}{a+d}}^{\frac{a}{a+c}} \frac{(a^2-d^2)}{(1-z)} - \frac{a^2}{z(1-z)}\, dz \\
&= \frac{1}{\Delta}\int_{\frac{a}{a+d}}^{\frac{b}{b+d}} \frac{(a^2-d^2)}{(1-z)} - \frac{a^2}{z} - \frac{a^2}{(1-z)}\, dz \\
&= \frac{1}{\Delta}\int_{\frac{a}{a+d}}^{\frac{b}{b+d}} \frac{-d^2}{(1-z)} - \frac{a^2}{z}\, dz \\
&= \frac{d^2}{\Delta}\left(\ln \frac{\frac{b+d-b}{b+d}}{\frac{a+d-a}{a+d}}\right) - \frac{a^2}{\Delta}\left(\ln \frac{\frac{b}{b+d}}{\frac{a}{a+d}}\right) \\
&= \frac{d^2-a^2}{\Delta}\ln\frac{a+d}{b+d} - \frac{a^2}{\Delta}\ln\frac{b}{a} \qquad (113)
\end{aligned}$$

Substituting equation (113) into equation (112) yields,

$$\int_{\frac{a}{a+d}}^{\frac{b}{b+d}} [1 - F_Z(z)]\, dz = \tfrac{d^2-a^2}{\Delta}\ln\tfrac{a+d}{b+d} - \tfrac{a^2}{\Delta}\ln\tfrac{b}{a} + \tfrac{d(b-a)}{(a+d)(b+d)}\left[1 + \tfrac{(a+d)^2}{\Delta}\right] \qquad (114)$$

And similarly,

$$\begin{aligned}
\int_{\frac{b}{b+d}}^{\frac{a}{a+c}} [1 - F_Z(z)]\, dz &= \int_{\frac{b}{b+d}}^{\frac{a}{a+c}} 1 - \frac{(b-a)[(2d+a+b)z - a - b]}{\Delta z}\, dz \\
&= \int_{\frac{b}{b+d}}^{\frac{a}{a+c}} \frac{\Delta - (b-a)(2d+a+b)}{\Delta} + \frac{b^2-a^2}{\Delta z}\, dz \\
&= \frac{(a-b)(2c+a+b)}{\Delta}\left(\frac{a}{a+c} - \frac{b}{b+d}\right) + \frac{b^2-a^2}{\Delta}\ln\frac{\frac{a}{a+c}}{\frac{b}{b+d}} \\
&= \tfrac{(a-b)(2c+a+b)(ad-bc)}{\Delta(a+c)(b+d)} + \tfrac{b^2-a^2}{\Delta}\ln\tfrac{b+d}{a+c} + \tfrac{b^2-a^2}{\Delta}\ln\tfrac{a}{b} \qquad (115)
\end{aligned}$$

And also,

$$\begin{aligned}
\int_{\frac{a}{a+c}}^{\frac{b}{b+c}} [1 - F_Z(z)]\, dz &= \int_{\frac{a}{a+c}}^{\frac{b}{b+c}} \frac{(b-(b+c)z)^2}{\Delta z(1-z)}\, dz \\
&= \frac{1}{\Delta}\int_{\frac{a}{a+c}}^{\frac{b}{b+c}} \frac{b^2 - 2(b+c)bz + (b+c)^2z^2}{z(1-z)}\, dz \\
&= \frac{1}{\Delta}\int_{\frac{a}{a+c}}^{\frac{b}{b+c}} \frac{b^2}{z} + \frac{b2}{1-z} - \frac{2(b+c)b}{1-z} - \frac{(b+c)^2(1-z-1)}{1-z}\, dz \\
&= \frac{1}{\Delta}\int_{\frac{a}{a+c}}^{\frac{b}{b+c}} -(b+c)^2 + \frac{b^2}{z} - \frac{b^2+2bc-(b+c)^2}{1-z}\, dz \\
&= \frac{(b+c)^2}{\Delta}\frac{c(a-b)}{(a+c)(b+c)} + \frac{1}{\Delta}\int_{\frac{a}{a+c}}^{\frac{b}{b+c}} \frac{b^2}{z} + \frac{c^2}{1-z}\, dz \\
&= \frac{c(b+c)}{2(a+c)(c-d)} + \frac{1}{\Delta}\left(b^2 \ln \frac{\frac{b}{b+c}}{\frac{a}{a+c}} - c^2 \ln \frac{\frac{b+c-b}{b+c}}{\frac{a+c-a}{a+c}}\right) \\
&= \frac{c(b+c)}{2(a+c)(c-d)} + \frac{b^2-c^2}{\Delta}\ln\frac{a+c}{b+c} + \frac{b^2}{\Delta}\ln\frac{b}{a} \qquad (116)
\end{aligned}$$

Now, gathering all the pieces together,

$$\begin{aligned}
E[Z] = {}& \frac{a}{a+d} + \frac{d(b-a)}{(a+d)(b+d)}\left[1 + \frac{(a+d)^2}{\Delta}\right] + \frac{(a-b)(2c+a+b)(ad-bc)}{\Delta(a+c)(b+d)} + \\
& \frac{c(b+c)}{2(a+c)(c-d)} + \frac{d^2-a^2}{\Delta}\ln\frac{a+d}{b+d} - \frac{a^2}{\Delta}\ln\frac{b}{a} + \frac{b^2-a^2}{\Delta}\ln\frac{b+d}{a+c} + \\
& \frac{b^2-a^2}{\Delta}\ln\frac{a}{b} + \frac{b^2-c^2}{\Delta}\ln\frac{a+c}{b+c} + \frac{b^2}{\Delta}\ln\frac{b}{a} \qquad (117)
\end{aligned}$$

Again, the non-logarithmic terms can be simplified as follows,

$$\begin{aligned}
I_3 &= \frac{(a-b)(2c+a+b)(ad-bc)}{\Delta(a+c)(b+d)} + \frac{c(b+c)}{2(a+c)(c-d)} \\
&= \frac{(2c+a+b)(bc-ad) - c(b+c)(b+d)}{2(b+d)(a+c)(d-c)} \\
&= \frac{(2bc^2 - 2acd + abc - a^2d + b^2c - abd) - (b^2c + bcd + bc^2 + c^2d)}{2(b+d)(a+c)(d-c)} \\
&= \frac{bc^2 - 2acd + abc - a^2d - abd - bcd - c^2d}{2(b+d)(a+c)(d-c)} \\
&= \frac{(abc^2 - 2a^2cd + a^2bc - a^3d - a^2bd - ac^2d) + (bc^2d - 2acd^2 - a^2d^2 - abd^2 - bcd^2 - c^2d^2)}{2(b+d)(a+c)(a+d)(d-c)}
\end{aligned}$$

$$
\begin{aligned}
I_4 &= \frac{d(b-a)}{(a+d)(b+d)}\left[1+\frac{(a+d)^2}{\Delta}\right] \\
&= \frac{d(b-a)}{(a+d)(b+d)}\cdot\frac{(2bd-2ad-2bc+2ac)+(a^2+2ad+d^2)}{2(b-a)(d-c)} \\
&= \frac{2bd^2-2bcd+2acd+a^2d+d^3}{2(a+d)(b+d)(d-c)} \\
&= \frac{(2abd^2-2abcd+2a^2cd+a^3d+ad^3)+(2bcd^2-2bc^2d+2ac^2d+a^2cd+cd^3)}{2(a+d)(b+d)(a+c)(d-c)}
\end{aligned}
$$

$$
\begin{aligned}
I_3+I_4 &= \tfrac{(abc^2-2a^2cd+a^2bc-a^3d-a^2bd-abcd-ac^2d)+(bc^2d-2acd^2+abcd-a^2d^2-abd^2-bcd^2-c^2d^2)}{2(b+d)(a+c)(a+d)(d-c)}+ \\
&\quad \tfrac{(2abd^2-2abcd+2a^2cd+a^3d+ad^3)+(2bcd^2-2bc^2d+2ac^2d+a^2cd+cd^3)}{2(a+d)(b+d)(a+c)(d-c)}
\end{aligned}
$$

Rearranging terms,

$$
\begin{aligned}
I_3+I_4 &= \frac{(abd^2-abcd+bcd^2-bc^2d+ad^3-acd^2+cd^3-c^2d^2)}{2(a+d)(b+d)(a+c)(d-c)}+ \\
&\quad \frac{(-a^2bd+a^2bc-abcd+abc^2-a^2d^2+a^2cd-acd^2+ac^2d)}{2(a+d)(b+d)(a+c)(d-c)} \\
&= \tfrac{d(abd-abc+bcd-bc^2+ad^2-acd+cd^2-c^2d)-a(abd-abc+bcd-bc^2+ad^2-acd+cd^2-c^2d)}{2(a+d)(b+d)(a+c)(d-c)} \\
&= \frac{(d-a)((abd-abc+bcd-bc^2)+(ad^2-acd+cd^2-c^2d))}{2(a+d)(b+d)(a+c)(d-c)} \\
&= \frac{(d-a)(b(ad-ac+cd-c^2)+d(ad-ac+cd-c^2))}{2(a+d)(b+d)(a+c)(d-c)} \\
&= \frac{(d-a)(b+d)(ad-ac+cd-c^2)}{2(a+d)(b+d)(a+c)(d-c)} \\
&= \frac{(d-a)(a(d-c)+c(d-c))}{2(a+d)(a+c)(d-c)} \\
&= \frac{(d-a)(a+c)(d-c)}{2(a+d)(a+c)(d-c)} \\
&= \frac{d-a}{2(a+d)} \qquad (118)
\end{aligned}
$$

By substituting the value of equation (118) into equation (117), yields $E[Z]$ as,

$$
\begin{aligned}
E[Z] &= \tfrac{a}{a+d}+\tfrac{d-a}{2(a+d)}+\tfrac{d^2-a^2}{\Delta}\ln\tfrac{a+d}{b+d}+\tfrac{b^2-a^2}{\Delta}\ln\tfrac{b+d}{a+c}+\tfrac{b^2-c^2}{\Delta}\ln\tfrac{a+c}{b+c} \\
E[Z] &= \frac{1}{2}+\frac{d^2-a^2}{\Delta}\ln\frac{a+d}{b+d}+\frac{b^2-a^2}{\Delta}\ln\frac{b+d}{a+c}+\frac{b^2-c^2}{\Delta}\ln\frac{a+c}{b+c} \qquad (119)
\end{aligned}
$$

We can relate equation (107) to equation (119) by the following nice argument,

$$
\begin{aligned}
E[Z] &= E\left[\frac{X}{X+Y}\right]=E\left[\frac{X+Y-Y}{X+Y}\right] \\
&= E\left[1-\frac{Y}{X+Y}\right]=1-E\left[\frac{Y}{Y+X}\right]=1-E[Z']
\end{aligned}
$$

By exchanging the random variables $X \Leftrightarrow Y$, and their corresponding ranges $a \Leftrightarrow c$, and $b \Leftrightarrow d$, one can convert from case 1 to case 2 and vice versa. More interestingly, evaluating the difference between equation (107) and equation(119) yields,

$$\begin{aligned}
\text{Difference} &= \frac{1}{2} + \frac{c^2 - b^2}{\Delta} \ln \frac{b+c}{b+d} + \frac{d^2 - c^2}{\Delta} \ln \frac{a+c}{b+d} + \frac{d^2 - a^2}{\Delta} \ln \frac{a+d}{a+c} \\
&\quad - \frac{1}{2} - \frac{d^2 - a^2}{\Delta} \ln \frac{a+d}{b+d} - \frac{b^2 - a^2}{\Delta} \ln \frac{b+d}{a+c} - \frac{b^2 - c^2}{\Delta} \ln \frac{a+c}{b+c} \\
&= \frac{1}{\Delta} \big((c^2 - b^2 + b^2 - c^2) \ln(b+c) + \\
&\quad (b^2 - c^2 + c^2 - d^2 - a^2 + d^2 - b^2 + a^2) \ln(b+d) + \\
&\quad (d^2 - c^2 + a^2 - d^2 + b^2 - a^2 - b^2 + c^2) \ln(a+c) + \\
&\quad (d^2 - a^2 - d^2 + a^2) \ln(a+d) \big) = 0 \qquad (120)
\end{aligned}$$

Which implies that equation (107) and equation(119) are equal, completing the proof.

VITA

Hady S. Abdel Salam
Department of Computer Science
Old Dominion University
Norfolk, VA 23529

Hady received his B.Sc. and M.Sc. degrees in Computer Science from Alexandria University in 1999 and 2004, respectively. His passionate toward computer science theory has encouraged him to have his master thesis on the verification of minimum-redundancy prefix codes and Huffman trees. He joined the PhD program in the Computer Science department at Old Dominion University in Spring 2006. Initially, Hady joined the Policy research group to work in several research projects with IBM and OCCS. Hady's work during this period was reported in several publications in different autonomic management related conferences and journals. In Fall 2007, Hady decided to pursue his dissertation under the supervision of Prof. Stephan Olariu in the area of Wireless Sensor Networks. Hady's main research was focused on backbone construction protocols and on developing energy efficient protocols for workforce selection and data aggregation in sensor networks. He also worked on terrain modeling using advanced 3D localization techniques and on developing adaptive sleep scheduling protocols for sensors. Hady's PhD work flourished into more than 10 publications in different IEEE and ACM conferences/workshops and 3 IEEE transactions manuscripts. In Fall 2010, Hady successfully defended his thesis to join the StreamInsight team in Microsoft Corporation.

Typeset using $\LaTeX$.