Exploring the Tight Asymptotic Bounds of the Trade-off Between Query Anonymity & Communication Cost in Wireless Sensor Network

by

Alireza Mortezaei

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering
Waterloo, Ontario, Canada, 2014

Abstract

We address query-anonymity in the context of wireless sensor networks. Query-anonymity is the property that the destination of a client's query is indistinguishable from other potential destinations. Prior work has established that this is an important issue, and has also pointed out that there appears to be a natural trade-off between query-anonymity and communication-cost. We explore what we call the limits of this trade-off: what is the communication-cost that is sufficient to achieve a certain query-anonymity, and what is the communication-cost that we must necessarily incur to achieve a certain query-anonymity? We adopt an unconditional notion of query-anonymity that we argue has intuitive appeal. We then establish the limits of the trade-off. In particular, we show that in wireless sensor networks which are source-routed, the necessary and sufficient communication-cost for query-anonymity asymptotically smaller than the diameter of the network d is a function of d only, and the necessary and sufficient communication-cost for query-anonymity larger than d is a function of the desired query-anonymity only. Our result applies to any network topology that is an arbitrary connected undirected graph. We validate our analytical insights empirically, via simulations. In summary, our work establishes sound and interesting theoretical results for query-anonymity in wireless sensor networks, and validates them empirically.

Acknowledgements

Foremost, I would like to express my sincere gratitude to my co-supervisor Dr. Mahesh Tripunitara for the continuous support of my master's study and research, as well as his mentorship, motivation and vast knowledge. He provided me with practical insight and patiently guided me through theoretical pitfalls. More so, he greatly helped me in writing of this thesis. I would, also, like to voice my utmost appreciation to my other co-supervisor Dr. Gordon B. Agnew for his consistent moral and financial support. I could not have imagined a better experience throughout my entire masters program without either of my co-supervisors.

Besides my supervisors, I would like to thank Dr. Bill Cowan and Dr. Alejandro López-Ortiz for their kindness and assistance in helping me better understand the properties of random graphs.

Contents

List of Figures

List of Algorithms

Chapter 1

Introduction

We address query-anonymity in the context of wireless sensor networks. A wireless sensor network is a collection of network elements with limited capabilities (e.g., related to computation or power), each of which is typically dedicated to a particular task, such as recording the temperature or barometric pressure. Anonymity is the property that a principal is indistinguishable from others.

The query-anonymity problem arises when a client, that is outside the wireless sensor network, wants to occasionally query a sensor or node in the network for its data. However, the client wants the particular node it queries to be anonymous to passive eavesdroppers.

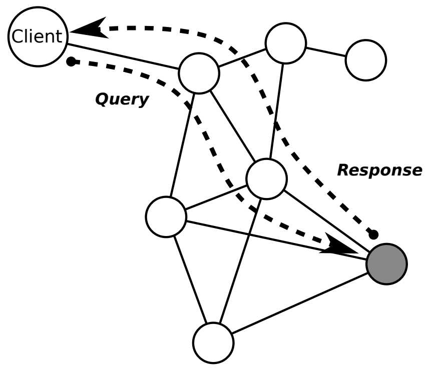

Figure 1.1: *An example of WSN Query. The client routes a query packet to the target of the query displayed as a grey circle. The target, then, routes a response packet back to the client.*

This broad scenario is depicted in Figure 1.1. Query-anonymity has a number of applications, as prior work [3] discusses. For example, a company that creates and maintains sensor networks may provide access to such a network to another company, e.g., one that is interested in searching for oil. The latter may not want the former to know in what regions it is interested in. If the former is able to profile the sensors in which the latter shows interest, this may leak information on where oil is likely to be found.

As Figure 1.1 depicts, the client routes its queries to nodes in the network. We premise that the client is able to directly communicate with one or more of the nodes. There is then some strategy for routing via which the client is able to communicate a query-message to, and a response-message from the node it seeks to query.

Communication comes at a cost. This cost is determined by several choices and limitations. For example, the nodes in the wireless sensor network may be too limited to maintain routing tables. In such circumstances, the client is assumed to be aware of the topology of the network, and specifies a route in messages it sends to or through the network. This is called source-routing, and is a common assumption in the context of such networks.

As one may expect, the length of the source-route directly impacts the size of the network data that is sent. This issue of communication cost is relevant when we consider query-anonymity. A natural way of providing anonymity is by "hiding in a crowd" [12], or k-anonymity [15]. As we discuss in Chapter 2, when the nodes in the wireless sensor network may not be trustworthy, for example because they are owned by a potentially malicious party, there seems to be no option for the client other than to redundantly query nodes to hide the true destination of its query. This is akin to hiding in a crowd, and k-anonymity.

1.1 Motivation and the Problem Statement

Prior work [3] has observed, that, in the context of wireless sensor networks, there seems to be a natural trade-off between the query-anonymity that can be achieved, and the communication-cost that must be incurred. Our intent in this work is to study what we call the limits of this trade-off. Specifically, we seek assertions of the following two forms:

– To achieve query-anonymity a, it is sufficient that we incur a communication-cost of c.

– To achieve query-anonymity a, it is necessary that we incur a communication-cost of c.

To our knowledge, while the existence of the trade-off has been observed in prior work [3, 7], it has not been studied in the manner we mention above.

One of the first issues that arises when we seek to study this trade-off is: what is a meaningful, sufficiently precise, quantitative notion of query-anonymity that we can adopt?

1.2 Contributions

In our paper [1], we have proposed an unconditional notion of query-anonymity. We will discuss this notion in more depth in Section 3.1. In this body of work, we heavily employ this notion to meaningfully investigate the limits of the trade-off between query-anonymity and communication-cost.

The main contribution of this paper is establishing the asymptotic boundaries of this trade-off when our notion of query-anonymity is applied to any wireless sensor network that (a) has an arbitrary connected undirected graph topology and (b) uses source routing as their main routing techniques.

We hypothesize the asymptotic boundaries of this trade-off and proceed to prove that these boundaries are both sufficient and necessary to achieve any desired level of query-anonymity a. We observe that there exists a simple protocol that achieves query-anonymity a (where a is any value from 1 to n and n is the number of nodes in the network) at a communication cost that is consistent with our proposed asymptotic boundaries. This proves sufficiency by the way of construction. We also show that no protocol exists that can achieve some query-anonymity given a lower communication-cost than the cost described by our proposed asymptotic boundaries. This proves necessity by the way of contradiction.

We validate our assertions on sufficiency via an empirical evaluation (see Chapter 5). For this, we have built a simulation using Tossim which is a de facto standard tool for simulating wireless sensor networks [11].

Chapter 2

Background and Terminology

Before we start our analytical assessment of the trade-off between communication-cost and query-anonymity in wireless sensor networks, we need to introduce certain notions and concepts upon which we establish our theretical approach. The following sections provide the required background with respect to the context of our work.

2.1 Topology

We model the topology as a graph. A node in the graph corresponds to a region in the wireless sensor network. We make the same assumptions as prior work regarding sensors within a region [3]. A sensor is in exactly one region, and a region contains one or more sensors. One of the sensors in a region plays the role of region-head, and it disseminates queries and aggregates responses for the region. Thus, it is meaningful to model a region as a node; this node can be seen as the region-head for the region displayed as red circles in Figure 2.1.

An edge in the graph corresponds to whether a node is able to communicate directly with another node. By direct communication, we mean that if node u has an edge to node v, then when node u transmits something, node v receives it. Given that a node represents a region, some routing of messages within a region is needed for a node to communicate with another node. This level of detail is not relevant to our work, and therefore, as in prior work [3] on query-anonymity, we do not model it.

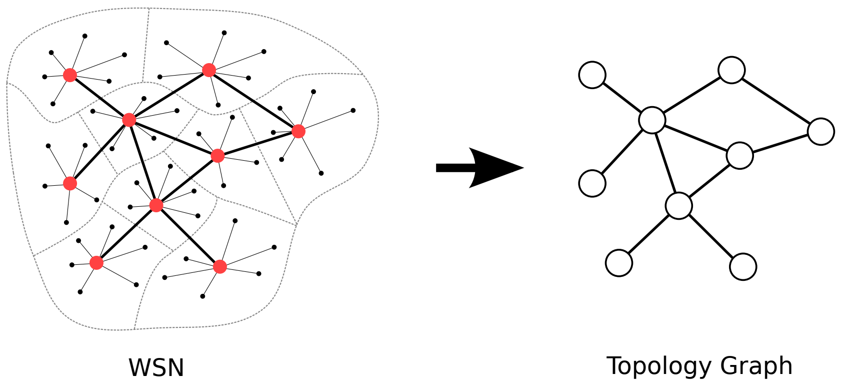

Figure 2.1: *Nodes within each region communicate only with the region-heads (shown in red). Region-heads, on the other hand, communicate with one another to route queries through the network. We model each region (region-head along with other regional nodes) as a graph node and the communication channels between the region-heads as graph edges.*

For entirety of this document, we consider undirected edges only — that is, node u is able to communicate directly to v if and only if v is able to communicate directly to u.

2.2 Protocol, query and response

A *protocol*, in the context of this work, comprises messages for sequences of query-response exchanges between the client and one or more nodes. The client sends out a *query packet*. A query packet contains a *query*, and some control information (e.g., source-routing). A query identifies one or more *destinations*, each of which is a node. We assume that the nodes are uniquely identified (e.g. a 16-bit node ID).

Each node that receives a query packet with a query addressed to it sends a *response packet* to the client. A response packet contains some constant-size information per node and some control information.

Prior work [3] proposes notions of query-anonymity that are characterized in the context of sequences of queries, rather than a single query only. To place our work in the same context, we say that a protocol may allow for sequences of queries rather than a single query only. We model this in a straightforward way — a query packet from the client may comprise a sequence of query packets, each of which contains a query and control information as characterized above. In Chapter 4, we introduce and describe our *Disjoint Anonymity Sets*(DAS) protocol which is a wireless sensor network protocol that uses source routing and can achieve any level of query-anonymity desired by the user.

Capabilities of a node

As our focus is wireless sensor networks, in which nodes are constrained, we need to clarify the capability that we associate with each node. Various choices affect the capabilities a node is assumed to have. For example, in a source-routed wireless sensor network, the only things a node needs to be able to do are:

1. Receive and transmit packets of bounded but variable size.
2. Read and manipulate the content of packets it receives i.e. inspect the packet for control information and strip/attach data from/to the packet.
3. Perform simple algorithms that can be expressed by a general-purpose programming language such as C and be executed on a general purpose microprocessor.

2.3 Threat Model

Our attacker is a passive eavesdropper who is able to observe every protocol exchange, including the contents of packets. This is the customary honest-but-curious threat model [9, 10]. We adapted this threat model as follows:

The client trusts itself only from the standpoint of disclosure. However, the attacker is aware of any algorithms that the client uses to achieve query-anonymity.

We discuss this further in Section 2.4. The attacker may "own" the wireless sensor network such that she can view everything that occurs within it along with any communication. The attacker may perform traffic- or other pattern-analysis with her observations. However, the attacker does not disrupt the functioning of the wireless sensor network, or any communication.

This is a strong eavesdropper in the context of query-anonymity, and certainly subsumes other threat models, such as ones in which components of the wireless sensor network are trusted, that some prior work adopts [14]. We emphasize, in the context of our threat model, that our interest is in query-anonymity only, and not, for example, the integrity of responses or availability of the network. It is likely that for such additional security requirements, we need to trust components of the wireless sensor network.

There is also the question as to whether our results apply to an active attacker, that is, an attacker that modifies a client's query-packet after it leaves the client, or modifies a response-packet on its way back to the client. We point out that our interest is in analyzing the trade-off between query-anonymity and communication-cost, and not in proposing novel approaches for achieving query-anonymity. The main issue that arises with an active attacker is whether a certain level of query-anonymity is achieved for the client in the presence of such an attacker, or even what a meaningful notion of query-anonymity is in such scenarios.

These are certainly interesting and challenging technical questions, and it is likely that we can adapt ideas from the work on mix networks [4] for these purposes. However, these issues are beyond the scope of this paper.

On the futility of encryption

Given our threat model above, approaches that use encryption in a manner that components of the wireless sensor network must be trusted to keep a key secret do not work. Some information must be revealed to the nodes in the network so they can be queried. For example, when a node receives a query packet, it must be able to determine from the packet whether it is a destination of a query that is in the packet. This applies to every node, and every collection of nodes.

Furthermore, it is worth mentioning that in the context of this literature, the attacker is mainly interested in the target of the query and not necessarily its content. Encryption only hides the content of a packet but since WSNs use a shared medium communication channel the destination of a packet is not strictly a property of its content. The attacker can simply observe the order in which participating nodes broadcast the packet along its path and make a calculated guess about its ultimate destination. The act of broadcasting can be observed by a radio receiver and does not require any cryptanalysis.

2.4 Quantitative Anonymity

Given our discussions on "thread model" in the previous Section 2.3, the only query-anonymity we can expect is from requiring the client to query nodes redundantly, in addition to those that it intends to query. Thus, we expect that every query comprises one or more *true* destinations, and zero or more *bogus* destinations. This is not surprising of course, and underlies approaches that achieve k-anonymity and its variants in the context of data stores [15]. In our work, however, we only focus on queries that contain only one *true* destination and zero or more *bogus* destinations.

As we characterize towards the start of this chapter above, a query comprises a set of destination nodes. An attacker can observe a query and the destination nodes that comprise it. The attacker is also aware of any algorithms the client may use to choose bogus destinations. For example, the client may choose bogus destinations uniformly at random from those nodes that are not true destinations [3]. However, the attacker does not know which destinations are bogus, and which are true in a query (If she did, no query-anonymity can be achieved).

Consider the case that there is only one true destination in a query. Then, the maximum query-anonymity corresponds to the case that the true destination is indistinguishable from all of the other $n - 1$ nodes. The minimum corresponds to the case that it is uniquely distinguishable. Given this, it seems natural to perceive query-anonymity as a probability in the interval $[1/n, 1]$. This is consistent with prior work on query-anonymity in wireless sensor networks [3], and broader prior work, on anonymity [6, 13].

In our characterization of query-anonymity (see Definition 1 in Section 3.1), we simply use the inverse of this probability as the measure. That is, given an

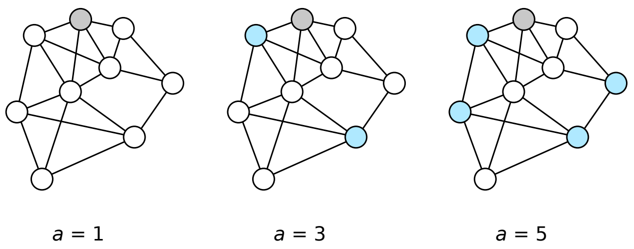

Figure 2.2: *An example demonstrating various degrees of query anonymity with true destinations as grey and bogus destinations as blue. Note that true and bogus destination are not necessarily adjacent.*

n-node wireless sensor network, query-anonymity lies in $[1, n]$. The semantics of 1 is that we have no query-anonymity (the true destination is uniquely distinguishable). The semantics of n is that we have the maximum query-anonymity (the true destination is indistinguishable from any of the other $n - 1$ nodes). Throughout this document, we use the symbol a to represent the degree of query anonymity.

Chapter 3

Theoretical Approach

Throughout this chapter, we will introduce, explore and prove various aspects of the theory behind our assertions regarding the asymptotic boundaries that describes the nature of the trade-off between communication cost and query anonymity. This includes assessment of previous works and defining and describing various notions that are required by our theoretical assessments of the problem and ultimately proving our claims.

3.1 Query-a-anonymity

Cabunar et al. in their work [3], propose two notions of query-anonymity: **temporal** and **spatial** anonymity. In work that is currently under review [1], we point out that these notions are deficient and fall short of describing the essence of query-anonymity in the context of wireless sensor networks. In this body of work, we adopt a definition of query-anonymity [1] which is an unconditional notion of query-anonymity that we call query-a-anonymity. This notion can be seen as an extension of the work of Serjantov and Danezis [13]. We discuss the differences in Chapter 7 on related work.

Definition 1 (Query-a-anonymity)**.** *We say that a query-response protocol achieves query-a-anonymity, or query-anonymity of a, if for every $a \in [1, n]$, for each $d_i \in D = \{d_1, \ldots, d_n\}$, there exists $D_i \subseteq D$ where $|D_i| \geq a$ and $d_i \in D_i$, such* *that an attacker, who is characterized as an algorithm $\mathcal{A}$, is unable to win the following game.*

1. *The attacker chooses destinations $X \subseteq D_i$, for some $i \in [1, n]$ such that $|X| = k \in [1, a]$. The attacker sends X to the client.*
2. *The client chooses some $d_l \in X$ uniformly at random and constructs a query q such that $d_l \in d(q)$. The client then runs an instance of the protocol with the query q.*
3. *The attacker must now identify which amongst the members of X the destination d_l chosen by the client in Step (2) above was.*

The client wins if the attacker's probability of correctly identifying d_l in Step (3) above is $\leq 1/k$ for all $k \in [1, a]$. Otherwise, the attacker wins.

Our notion of anonymity above is unconditional. We do not constrain the attacker $\mathcal{A}$ in any way than to require that it is an algorithm. In particular, the attacker may have access to any information he wants before he starts the game. This includes, for example, any algorithms the client may adopt to generate the set D_i and the choice d_l in Step (2), and the d_l choices in Step (2) that the client has made in prior runs of the protocol.

3.2 Communication Cost

We characterize communication-cost in a natural way as follows. Transportation of a constant (i.e., $\Theta(1)$) amount of data over one hop of the wireless sensor network is considered constant communication-cost. Consider the example illustrated in Figure3.1. Suppose the client communicates directly with m_1 only, and seeks to send a packet to the node m_d over a shortest-path. With source-routing, the route must be specified by the client as control information in the outgoing packet. The nodes along the path are labelled $m_1, m_2, ..., m_i, ...m_{d-1}, m_d$ in the order that the packet visits them.

Assuming each entry in the source-route is of constant size, and a node strips the

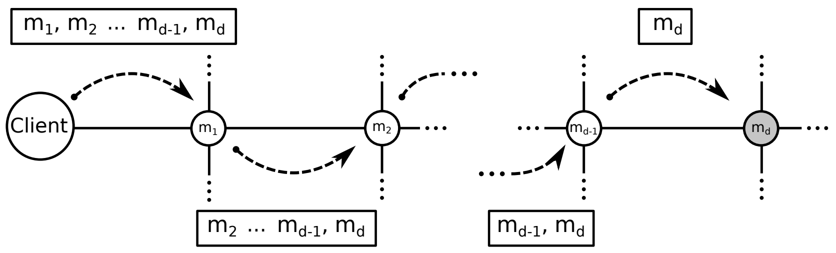

Figure 3.1: *An example of a source-routed query traveling from the client to the target node m_d. At each hop, the size of the query decreases by one unit of routing information.*

entry that corresponds to it from the source-route when it receives it, over the first hop, we transmit a packet of size $\Theta(d)$. Over the second hop, we transmit a packet of size $\Theta(d-1)$ and so on, until a constant-sized packet arrives at node m_d . Thus, the total cost of sending the packet is $\Theta(d) + \Theta(d-1) + ... + \Theta(1) = \Theta(d^2)$.

Note that whether a node strips its entry from the source-route or not, the communication-cost remains $\Theta(d) + ... + \Theta(d) = \Theta(d^2)$. We point out also that the asymptotic cost does not change based on other, mundane, aspects we may change with how the query- response protocol works. For example, consider the following two alternatives to how a node that is queried communicates a response back to the client. It may send the response as a separate packet, or it may piggy-back the response in the query packet, which presumably contains a source-route back to the client. In section 4.2.2, we will further elaborate on why these two options are equivalent with respect to the asymptotic communication cost. This is the case even if we query a set of nodes, rather than a single node.

3.3 Proof Strategy

As we mention in Section 1, our assertions are of two types:

1. communication-cost c is sufficient to achieve query-anonymity of a
2. communication-cost c is necessary to achieve query-anonymity of a.

We will further expand on these two types of assertions.

Sufficiency

Our approach to proving an assertion of type (1) above is by construction. That is, we provide a protocol for which we show analytically that it achieves anonymity a, and incurs a cost of only c. We now briefly discuss the protocol, which we call *the protocol with disjoint anonymity sets* (the DAS protocol), and establish that it achieves Query anonymity a. The detailed description of DAS protocol is provided in chapter 4.

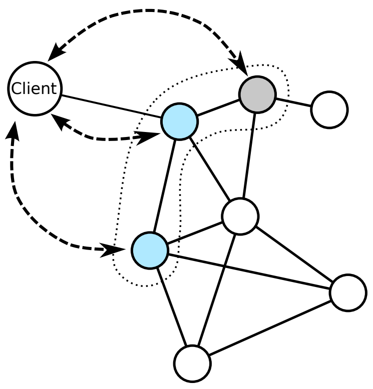

Figure 3.2: *A simplified example of the DAS protocol. We show the true destination of the client's query (grey), and two other nodes (blue) that the client also queries so an attacker is unable to distinguish the true destination from them.*

The DAS protocol is simple, and therefore has intuitive appeal. It is, however, somewhat surprisingly powerful, as we show in the remainder of this paper. It achieves Query anonymity a, and does so with optimal cost. The DAS protocol works as follows:

- The client partitions the n WSN nodes (each of which is a possible destination of a query) into $\lfloor n/a \rfloor$ disjoint sets $S_1, \ldots, S_{\lfloor n/a \rfloor}$ each of size $\in [a, 2a-1]$. We call each set S_m an *anonymity set*.
- When the client wants to query node d_i, it identifies the set S_m to which d_i belongs, and constructs a query q with destination set $d(q) = S_m$. It issues the query q and receives the response.

We show this pictorially in Figure 3.2. In the figure, the node shaded grey is the client's true destination. Presumably, the client has placed that node in the same partition as the two nodes colored blue. Therefore, the client queries all three nodes when it intends to query the node shaded black only. We now make an assertion regarding the query-anonymity that the DAS protocol achieves.

Theorem 1. *The DAS protocol achieves query-anonymity a.*

Proof. In Definition 1, we set D_i to the S_m from the DAS protocol which contains d_i. For any $X \subseteq D_i = S_m$ of size $k \in [1, a]$ that the attacker chooses in Step (1) of Definition 1, the client, under the DAS protocol, simply queries all the nodes in S_m in Step (2) of Definition 1. Note that we query S_m for every choice of $d_l \in X \subseteq S_m$ that the client chooses in Step (2). The attacker's probability of identifying the $d_l \in X$ that the client chooses in Step (2) is $1/|X| = 1/k$, notwithstanding any prior knowledge he has, provided the client chooses d_l randomly. □

An observation we can make from the proof of Theorem 1 above is that the manner in which we construct the disjoint anonymity sets is inconsequential to the Query anonymity that the DAS protocol achieves. This property is important in future sections in which we consider the trade-off between cost and Query anonymity. If we have a query Q that contains two nodes that are far apart in the WSN topology, then it can be expensive to route packets in order to query both nodes. Consequently, we need to construct anonymity sets in a manner that does not compromise the Query anonymity that is achieved, but optimizes cost. We make our observation precise via the following lemma.

Lemma 1. *If the anonymity sets are disjoint and of size at least a each, the DAS protocol achieves Query anonymity a.*

Necessity

Our approach to proving that cost c is necessary to achieve Query anonymity of a, (i.e., an assertion of type (2) above), is by contradiction. We assume that there exists a protocol that achieves Query anonymity a with a cost less than c. We then derive a contradiction. Usually, the contradiction we derive is that with such a low cost, the client is unable to even communicate with every node in the WSN, let alone do so while simultaneously achieving the desired Query anonymity.

The following observation for a lower-bound for achieving Query-anonymity of a is often useful in our proofs for necessity.

Theorem 2. *In Step (2) of Definition 1, to achieve Query-a-anonymity, it must be true that $|d(q)| \geq k$.*

Proof. By contradiction: assume otherwise. If the attacker guesses randomly that one of the destinations in $d(q)$ is d_l, the attacker's probability of success is $>1/k$. $\square$

3.4 Cost vs. Anonymity

We now consider the trade-off between query-anonymity and communication-cost in an n-node WSN with any arbitrary topology and with source-routing as its routing method. As we mention in Section 2.1, a topology, to us, is an undirected, connected graph.

For the cost that is necessary to achieve query-anonymity a, as we mention in Section 3.3, we use contradiction. For the cost that is sufficient for query-anonymity of a, we rely on the DAS protocol, with the anonymity sets in it constructed a particular way. As we observe in the previous section in the context of Theorem 1, this does not compromise the level of query-anonymity we achieve. Therefore, in this section, we will also explain the particular manner in which we construct anonymity sets for DAS protocol.

We adopt the following terminology that is customary in computing [5]. A tree is an acyclic, undirected, connected graph. A spanning tree of a connected undirected graph G is a subgraph of G that is a tree of all of G's nodes. A rooted tree is a tree in which one of its nodes is perceived as the root, and the other nodes as its descendants. Henceforth, when we use the term tree, we mean a rooted tree. A leaf is a node in a tree that has no children. The height of a node in a tree is the distance (number of edges) from the node to its farthest leaf. The height of a tree is the height of its root. And finally, if h is the height of a tree and h_u is the height of node u in it, the depth of the node u is $h - h_u$.

We first seek to establish Theorem 3 below, regarding the existence of a particular kind of path through nodes for an arbitrary topology. Before we get to Theorem 3, we establish the following lemma that helps prove that theorem.

Lemma 2. *Given a connected undirected graph $G = \langle V, E \rangle$ with $|V| = n$, there exists a node $u \in V$ and a path $u \rightsquigarrow u$ that traverses every node in V at least once and is of length $O(n)$.*

Proof. We first construct a spanning tree, T of G, and adopt some $r \in V$ as its root. This r serves as the node u in the statement of our lemma. We can now traverse T the following way: starting at r, we visit a node, then recursively each of its children, and then return to the node. Such a traversal constructs a path in which every node is visited at least once, and every edge is traversed at most twice, once in the downward direction and once back upwards. As the number of edges in the tree is $O(n)$, the path is of length $O(n)$. $\square$

Theorem 3. *Given a connected undirected graph $G = \langle V, E\rangle$ of n vertices, there exists a partition of V into disjoint sets $S_1, \ldots, S_{\lfloor n/a \rfloor}$ each of size $\in [a, 2a-1]$ for $a \in [1, n]$, such that given any such set S_i, there exists $u \in S_i$ such that there is cycle $u \rightsquigarrow u$ of length $\Theta(|S_i|)$ that traverses every vertex in S_i.*

Algorithm 1 Procedures for constructing anonymity sets in the DAS protocol

```
1: procedure ConstructAllS($T$,$a$)
2:     $n \leftarrow |V[T]|$
3:     for all $i \in [1, \lfloor n/a \rfloor]$ do
4:         $|S_i| \leftarrow \begin{cases} a & \text{if } i \in [1, \lfloor n/a \rfloor - 1] \\ n - a(\lfloor n/a \rfloor - 1) & \text{otherwise} \end{cases}$
5:         $T_i \leftarrow T_{i-1}$ with nodes from $S_{i-1}$ removed ( special case: $T_0 = T$).
6:         MakeLeftHeavy($T_i$)
7:         $S_i \leftarrow$ ConstructOneS($T_i$, $|S_i|$)
8: procedure MakeLeftHeavy($T'$)
9:     Upon return, $T'$ has the following property: given any
10:    node in $T'$, suppose its children from left to right
11:    are $c_1, \ldots, c_k$. Then, height($c_1$) $\geq \ldots \geq$ height($c_k$).
12: procedure ConstructOneS($T'$, $n'$)
13:    Post-order traverse $T'$
14:    Return the first $n'$ nodes we encounter
```

The proof for the above theorem relies on the procedure *ConstructAllS* in Algorithm 1, which invokes as subroutines the procedures *MakeLeftHeavy* and *ConstructOneS*. We first describe the procedure, and then establish properties for them that help us with the proof for Theorem 3.

The procedure *ConstructAllS* takes as inputs a tree T of n nodes and an integer $a \in [1, n]$. In Line 2, we set up an iterator, i, which takes on values 1 through

$\lfloor n/a \rfloor$ in turn. We first specify the size $|S_i|$ for the set S_i we intend to construct in the i^{th} iteration. In Line 4, we first specify T_i to be T, but with all the nodes from $S_1 \cup \ldots \cup S_{i-1}$ that were identified in prior iterations removed. As Lemma 4 below asserts, T_i is a tree of at least a nodes. We then invoke *MakeLeftHeavy* on T_i in Line 5, and then *ConstructOneS* with second argument the size of S_i we wish to construct.

In *MakeLeftHeavy*, we reorder the children of every node so that they are ordered left to right in non-increasing height. That is, the leftmost child has the maximum height amongst all the node's children, the next child to the right has height at most the height of its left sibling and so on. A simple algorithm to do this is the following. We maintain an integer for each node, that is initialized to 0. We first walk up the tree starting at the leaves and update each node's height. We then walk down the tree starting at the root and reorder the children of each node based on their height. The algorithm runs in time linear in the size of the tree. It has a space-complexity of $O(n \log n)$, where n is the number of nodes in the tree, because the height of the tree is at most $n-1$, and therefore at each node, we need to maintain at most $\log n$ bits to record its height.

Finally, in *ConstructOneS*, we are given a tree T' and a positive integer n', where n' is at most the number of nodes in T'. We perform a post-order traversal of T' and return the first n' nodes we encounter. In a post-order traversal, we recursively visit the children of a node from left to right, and then visit the node. Figure 3.3 demonstrates how this process divides the network into disjoint anonymity sets.

We now first present lemmas for properties of our three algorithms, and then prove Theorem 3.

Lemma 3. *Suppose T' is a tree, and n' is an integer that is at most the number of nodes in T'. Then, removing the n' nodes returned by ConstructOneS(T', n') from T' results in a tree.*

Proof. In a post-order traversal, we visit all the descendants of a node before the node itself. Therefore, a node is in the set of n' nodes only if all of its descendants are also in that set. □

Lemma 4. *In Line 4 of ConstructAllS, for all i, T_i is a tree of at least $|S_i|$ nodes, for $|S_i|$ as specified in Line 3.*

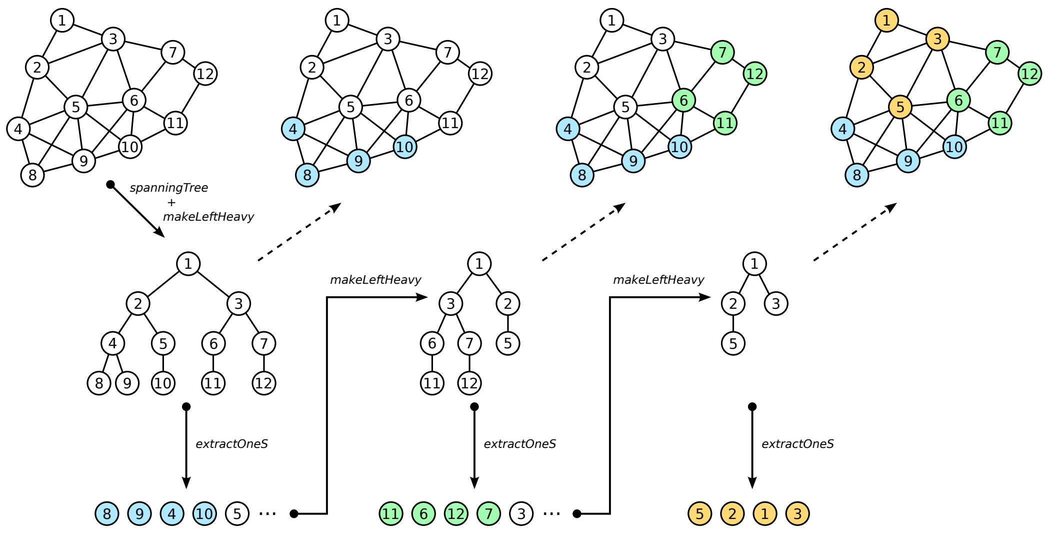

Figure 3.3: *An example of workings of ConstructAllS and its subroutines MakeLeft-Heavy and ConstructOneS. We adopt $n = 12$ and $a = 3$ which results in three anonymity sets of blue , green and yellow with sizes 4, 4 and 4 respectively. We construct graph's rooted BFS tree and repeatedly apply MakeLeftHeavy and ConstructOneS in that order until all the nodes are grouped into desired anonymity sets.*

Proof. By induction on i. For the base case, $i = 1$, and as $a \leq n$, $T_1 = T$ has at least $|S_1|$ nodes and is a tree. For the step, we assume that the assertion is true for all $i \in [1, k-1]$. We first show, for $i = k$, that T_i has at least $|S_i|$ nodes. We have two cases. (1) $k < \lfloor n/a \rfloor$. As $|S_1| = \ldots = |S_{k-1}| = a$ and $k \leq \lfloor n/a \rfloor$, $a(k-1) \leq n - a$, and T_k has at least a nodes. (2) $k = \lfloor n/a \rfloor$. As $|S_1| = \ldots = |S_{k-1}| = a$, we remove $a\left(\lfloor n/a \rfloor - 1\right)$ nodes from T in Line 4 of *ConstructAllS* to get T_k, and therefore T_k has $|S_k| = n - a\left(\lfloor n/a \rfloor - 1\right)$ nodes as desired.

We are left with establishing that T_k is a tree. For this, we first observe that from the induction assumption, T_{k-1} is a tree. And in Line 4 of *ConstructAllS*, when $i = k$, T_k is T_{k-1} but with the nodes from S_{k-1} removed. S_{k-1} is the result of the invocation of *ConstructOneS*$(T_{k-1}, |S_{k-1}|)$, and therefore by Lemma 3 above, T_k is a tree. $\square$

Lemma 5. *Construct the graph $t_i = \langle V_i, E_i \rangle$ as follows, for a set S_i output by ConstructAllS(T, a). Start with t_i as the smallest subtree of T that contains all the nodes in S_i. Recursively remove all leaves from t_i that are not in S_i. Then, the following are true:*

(a) t_i *is a tree*

(b) $|V_i| = O(|S_i|)$.

Proof. To establish (a), we merely observe that we start with t_i being a tree, and recursively remove leaves only, and therefore the resultant t_i is also a tree.

To establish (b), we first observe that t_i is a subtree of T_i in the algorithm *ConstructAllS*. We consider t_i as a subtree of T_i immediately after Line 5 is executed in *ConstructAllS*. Let r be the root of t_i. If r has no children or has only one child, then all the nodes of t_i are in S_i are we are done. Otherwise, that is, if r has more than one child, then, only the rightmost branch of r, which is a simple path, can contain any nodes not in S_i. This is a consequence of the post-order traversal we conduct in *ConstructOneS* to determine S_i. Let $n_{\neg S_i}$ be the number of nodes in t_i that are not in S_i. Then $n_{\neg S_i} \leq |S_i|/2$ because t_i is "left heavy," that is, the descendants of its root r are ordered in non-increasing height as we go left to right. Thus, $|V_i| \leq 2|S_i|$, and therefore $|V_i| = O(|S_i|)$. □

Figure 3.4 can help us better understand the proof for part (b) of Lemma 5.

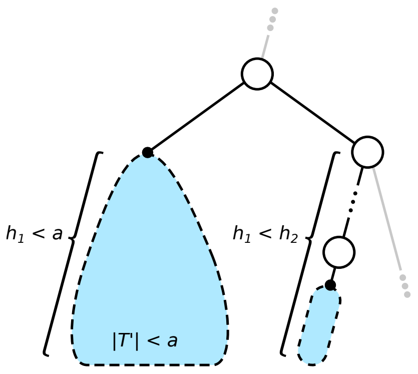

Figure 3.4: *An example of a spanning subtree returned after applying on instance of ConstructOneS which represents some anonymity set S_i of size a.*

Proof (Theorem 3) . We first construct a spanning tree T of G, and invoke *ConstructAllS*(T, a). For each S_i, we have a t_i as in Lemma 5 above. As t_i is a tree, it is connected, and therefore by Lemma 2, we have the desired proof. □

We are now able to make assertions regarding the limits of the trade-off between anonymity and cost for any WSN of arbitrary topology with source-routing. In the following, the diameter of a graph is the longest amongst the shortest distances (number of edges) across all pairs of nodes. That is, if $G = \langle V, E \rangle$ is a connected undirected graph with $V = \{1, \ldots, n\}$, and $d_{i,j}$ is the shortest distance between i and j, then G's diameter $d = \max_{i,j \in V}\{d_{i,j}\}$.

Theorem 4. *To achieve query-anonymity $a \in [1, \Theta(d)]$ in a source-routed WSN whose diameter is d, a worst-case cost of:*

- $O(d^2)$ *is sufficient, and,*
- $\Omega(d^2)$ *is necessary.*

Proof. Sufficiency: consider the worst-case cost in the DAS protocol of achieving query-anonymity $a \in [1, \Theta(d)]$. The worst-case node to be queried is of distance d from the client, and belongs to some anonymity-set S_i. This set is associated with a node u as in Theorem 3. Consider the route as being split into 3 pieces: (1) from the client to u, (2) from u through all the nodes in S_i back to u, and, (3) from u to the client. (1) and (3) are each of length $O(d)$. By Theorem 3, (2) is of length $O(|S_i|) = O(d)$. Therefore, the total route-length is $O(d)$. And thus, the worst-case cost is $O(d^2)$.

Necessity: assume otherwise for the purpose of contradiction. That is, assume that there is a protocol that achieves query-anonymity of a with cost $o(d)$.

Consider the worst-case cost of sending 1-bit to a node in the WSN in terms of communications-cost. A node that actualizes such a worst-case scenario is a node that is of distance d from the client, and therefore the length of the source-route from the client to that node is initially $\Omega(d)$. Thus, to transport the bit costs $\Omega(d^2)$ thereby contradicting the assumption. $\square$

Theorem 5. *To achieve query-anonymity* $a \in [\Theta(d), n]$ *in a source-routed WSN whose diameter is* d*, a worst-case cost of:*

- $O(a^2)$ *is sufficient, and,*
- $\Omega(a^2)$ *is necessary.*

Proof. Sufficiency: as in the proof for Theorem 4 above, in the DAS protocol, we split the route into three pieces: (1) a route from the client to the node u from Theorem 3 associated with an anonymity set S_i, (2) a route of length $\Theta(|S_i|)$ through the members of S_i, from u to u, and, (3) the route back from u to the client. The total route-length is $O(2d + |S_i|) = O(2d + a) = O(a)$ because $|S_i| \in [a, 2a)$, and $d = O(a)$. Thus, the total cost is $O(a^2)$.

Necessity: From Theorem 2, we know that the client must choose $d(q)$ of size $\geq a$ in Step (2) of Definition 1. That is, the source-route the client specifies must include at least a distinct hops. Consider the communication cost of transporting such a source-route. We have "at least" 1 hop across which a packet of size $\Theta(a)$ traverses, "at least" 1 hop across which a packet of size $\Theta(a-1)$ traverses, and so on. Thus, the total cost is $\Theta(a) + \Theta(a-1) + \ldots + \Theta(1) = \Theta(a^2)$. Thus, a cost of $\Omega(a^2)$ is necessary. $\square$

The special case of square-grid topology

Square grid topology is one of the widely used classes of WSN topologies. An interesting property of these topologies, as the direct result of applying the Theorem 4 to such networks, is that the communication cost of achieving anonymity a, when $a \in [1, \Theta(\sqrt{n})]$, depends only on n where n is the number of nodes in the network. The reason behind this characteristic is that the diameter of a network with square grid topology is a function of the size of the network as shown in Figure 3.5. The longest shortest path resides between the two nodes on the opposing corners of the grid and is laid along any two adjacent sides of the square. If the size of the grid is n, each side can be traversed in $\sqrt{n} - 1$ hops. Therefore, the diameter d is exactly $2(\sqrt{n} - 1)$. Thus, we proceed to establish the following corollary:

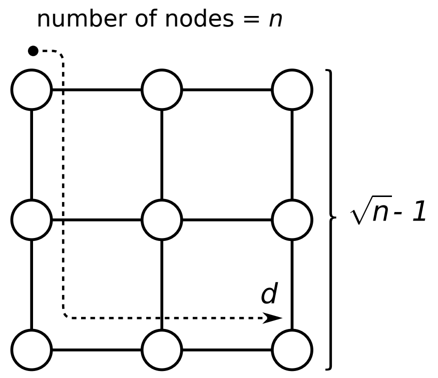

Figure 3.5: *An example of a WSN of size n with square grid topology. Each side of the square grid can be traversed in $\sqrt{n} - 1$ hops.*

Corollary 1. *To achieve query-anonymity $a \in [1, \Theta(\sqrt{n})]$ in a source-routed, square-grid WSN of size n, a worst-case cost of:*

- *$O(n)$ is sufficient, and,*
- *$\Omega(n)$ is necessary.*

Proof. The proof is very straightforward. We know that the diameter of a square grid WSN is $2(\sqrt{n} - 1)$ which is $\Theta(\sqrt{n})$. We simply substitute d in Theorem 4 for this value. We get $O(d^2) = O([2(\sqrt{n} - 1)]^2) = O(4n - 8\sqrt{n} - 4) = O(n)$. We can apply the same logic for the "necessary" property $\Omega(\cdot)$. $\square$

As for $a \in [\Theta(\sqrt{n}), n]$, the asymptotic cost is exactly the same as it would be in any arbitray topology i.e. $\Theta(a^2)$. However, this result is less interesting than

Corollary 1. Our work that is currently under review [1] presents a more in depth discussion of this special-case.

Chapter 4

Disjoint Anonymity Sets (DAS) Protocol

In this chapter, we describe various aspects of DAS protocol that has not been discussed so far and paint a broader picture of DAS's behaviour in a WSN. This chapter is a precursor to Chapter5 and helps us eliminate any uncertainty that could arise when assessing the experimental results.

4.1 Anonymity Set Arrangement

In Section 3.4, we described how we apply *ConstructAllS* in order to divide the WSN into disjoint sets. However, we made no assertions about the size of individual sets other than for each set S_i, $|S_i| \in [a, 2a-1]$ where a is the desired anonymity. If there are to be k disjoint sets in the network, *ConstructAllS* naively allocates a nodes for each of the first $k-1$ sets and assigns the remaining nodes to the k^{th} set. However, we wish to distribute WSN nodes as evenly as possible among the anonymity sets in cases where the size of the network n is not divisible by query-a-anonymity a (i.e. $a \nmid n$). We do this by calculating the size of each individual anonymity set before we divide the network. Once we establish the size of each individual anonymity set, only then we advance to divide the network accordingly.

In order to divide the network evenly, we initially set the size of each anonymity set S_i to a. Doing so, we assure that each set provides the anonymity of at least a. Next, we calculate the number of unallocated nodes $rem = n(\mod k)$. Finally, we increment the size of $S_1, S_2 \ldots, S_k$ in round-robin manner for rem repetitions. This procedure is demonstrated in Algorithm2.

Algorithm 2 Uniform distribution of nodes among anonymity sets

```
1: procedure arragneAnonSets($(a, n)$)
2:     $k \leftarrow \lfloor n/a \rfloor$                          ▷ $k$ is the number of anonymity sets
3:     $\forall S_i$ where $i \in [1, k] : |S_i| \leftarrow a$
4:     $rem \leftarrow n(\mod ak)$
5:     for $i \in [1, rem]$ do
6:         $idx \leftarrow i(\mod k)$                    ▷ Adjusting the index for round-robin
7:         $|S_{idx}| \leftarrow |S_{idx}| + 1$
```

By dividing the network in this manner, the difference between the size of any two anonymity sets is at most 1. Therefore, we do not compromise the cost of one query for the benefit of another. Doing do, we avoid the situations in which the largest set resides furthest from the client which can result in significant communication overhead.

Anonymity set arrangement for square-grid topology

In WSNs with square grid topologies, there is a more efficient way of arranging the anonymity sets which results in a smaller cost for each query. For square grid networks, we can always arrange the anonymity sets so that each set S_i can be traversed in exactly $|S_i|$ "consecutive" hops. To form such an arrangement, first, we calculate the size of each anonymity set S_i as described above. Instead of applying *ConstructAllS* to the network graph, we order the graph nodes in the fashion which we refer to as a *snake-walk* throughout the graph as shown in Figure 4.1. We, then, assign the first $|S_1|$ nodes of this ordering to the set S_1, the next $|S_2|$ to the set S_2 and so on.

We traverse each S_i in the same ordering provided by its corresponding portion of our *snake-walk* which guarantees exactly $|S_i|$ hops. This is a better arrangement compared to the arrangement provided *ConstructAllS* which might result in a walk as large as $2|S_i|$ through S_i as it was explained by the proof of Lemma 5. This

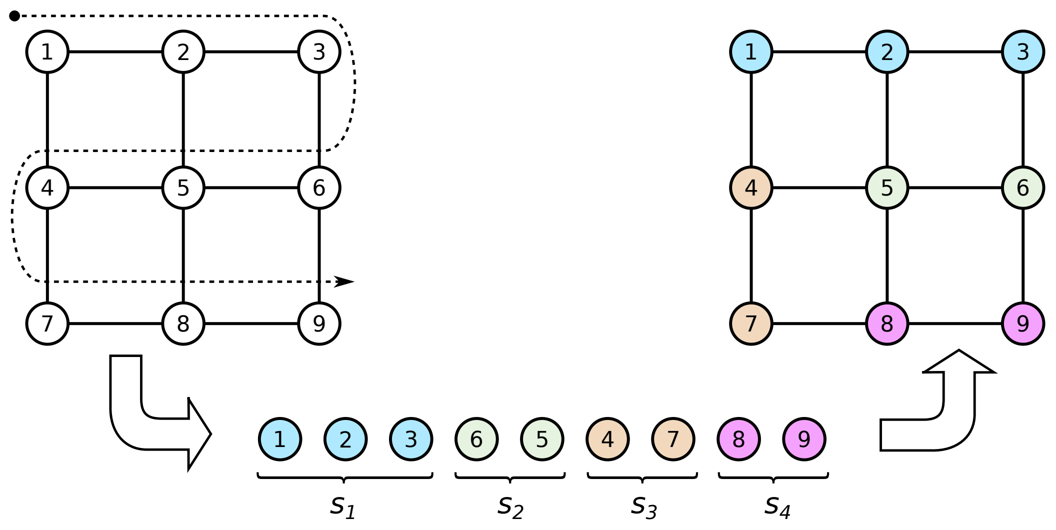

Figure 4.1: *In this example the size of the network is $n = 9$ and the anonymity is $a = 2$ which results in anonymity sets of sizes 3, 2, 2 and 2. The snake-walk through the graph gives us the ordering shown above from which we allocate nodes to each set according to their sizes.*

reduces the overall length of the path that a query takes which means less routing information required per packet. This eventually means smaller cost per query. We use this arrangement for square grid WSNs thought out our experimentation.

4.2 Communication Protocol

In this section, we accurately describe how DAS protocol operates in order to guide the query through the network and collect the required data.

4.2.1 DAS Packet Format

Our implementation of DAS uses the packet format illustrated in Figure 4.2. The first field of the packet is the header. This field consists of two segments. The first segment is 16 bits which indicates the ID of the target node. By the "target node", we mean the target of the inter-node transmission at the link layer and not the ultimate destination of the query. This segment gets updated for each hop throughout the packet's route. More so, even though the node IDs are 15 bits, we used 16 bits since it is much more convenient to access this memory arrangement and also it makes it easier to avoid segmentation errors at the hardware level. The second segment is also 16 bits which dictates what action the target node should perform alongside the parameter/s required for that action. The 2 most significant bits in this segment represent the action and the remaining 14 bits are reserved for the parameter/flags. The most important action in this context is the "routing" action for which the parameter is the size of the payload in bytes (i.e. the combined length "route" and the "data" portions of the payload in bytes). There are other actions that are used in the initial phase of the simulation but the details of these actions are outside of the scope of this paper's discussion.

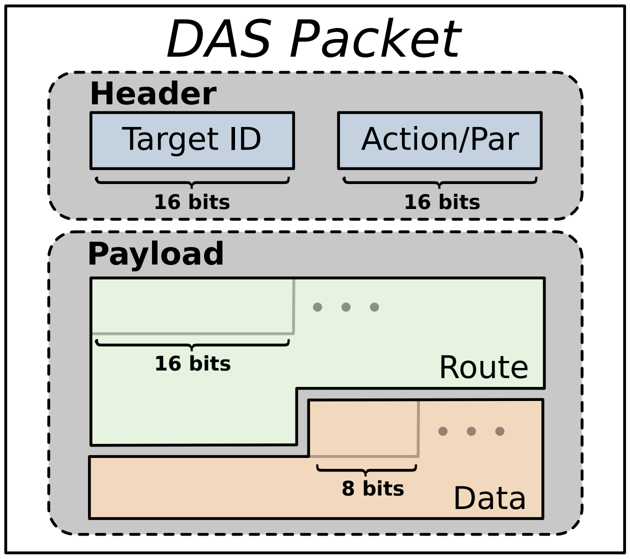

Figure 4.2: *Header field of the packet (blue) contains one 16-bit segment to indicated the target and another 16-bit segment reserved for instructions and required parameters. The payload field of a DAS packet consists of route_buff (green) which is a16-bit array that contains routing information and finally the data array (orange) which is a 8-bit buffer that contains the piggy-backed data*

The second field of the packet is the payload. Payload is a variable length 8-bit buffer. The payload buffer is partitioned into two portions. The first portion of the payload buffer, called *route_buff*, contains the information required by the source routing process in order to guide the query throughout the network. We treat this portion of the payload as a variable length array of 16-bit segments representing an ordered sequence of node IDs along the route. The most significant bit, called **CA** flag, in each segment is used as a flag which indicates whether the corresponding node should attach its sensor data to the packet before forwarding the packet. The remaining 15 bits represent the node ID.

The second portion of the payload buffer contains the carry-on sensor data. This portion of the payload immediately follows is treated as a variable length array of 8-bit segments. If the **CA** flag is set when a query arrives at a node, the node will append its sensor data to this array before forwarding the packet to the next node on the route.

4.2.2 DAS Routing

The query's route is constructed by the client and is put in the DAS packet before the query is released into the network. If the target of a query belongs some anonymity set S, the route of that query consists of (1) the shortest path to some node m_0 within S, (2) a walk that starts from m_0 and visits every node in S by taking the shortest path between any two of such nodes and finally (3) the shortest path from the last node visited in S back to the client. There exist a route which results in the least query cost based on which ordering of nodes in S we choose to route the packet through. However, trying to achieve this optimal route deems impractical since it is equivalent to solving the *traveling salesman problem* which is NP-complete [5]. Thus, we adopt the following simple approach of constructing our routes for WSNs with arbitrary topology. We visit the members of any anonymity set S in the order that was returned by the procedure *ConstructOneS*. As for grid topology, we visit the members of S in the order that was return by the *snake-walk* described in Section 4.1

Algorithm 3 An instace of DAS protocol that runs at each relay node

```
1:  procedure receivePacket($P$)
2:      if $P.target \neq node\_ID$ then
3:          drop $P$
4:      else if $P.action = Routing$ then
5:          create a new pakect $Q$
6:          $Q.action \leftarrow Routing$
7:          $Q.target \xleftarrow{16-bits} P.payload[0].node\_id$
8:          $new\_lenght \leftarrow P.payload\_length - 1$
9:          for $i \in [1, new\_length]$ do
10:             $Q.payload[i-1] \xleftarrow{8-bits} P.payload[i]$
11:         if $P.payload[0].CA$ then
12:             $Q.payload[new\_length] \xleftarrow{8-bits} sensor\_data$
13:             $Q.payload\_length \leftarrow new\_length + 1$
14:         else
15:             $Q.payload\_length \leftarrow new\_length$
16:         $broadcast(Q, Q.payload\_length + 4)$
```

After the construction of the initial query packet, the client injects the query into the network. Upon DAS packets' arrival, each node in the network performs the steps describe in Algorithm3. This protocol is consistent with the "capabilities of a node" describe in Section 2.2. In order to explain how this algorithm governs the routing mechanism in the WSN, we provide the following story.

A day in the life of a DAS packet

In this example, we work closely with Figure 4.3. More so, we refer to a node whose ID is x as m_x.

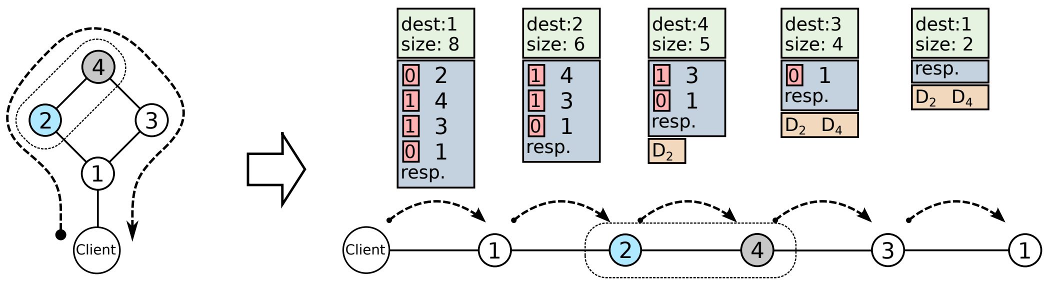

Figure 4.3: *In this example, the client wants to query m_4 which belong to the anonymity set $\{m_2, m_4\}$. At each hop, a node strips a segment from the head of the route_buff and updates packet's header accordingly. Moreover, nodes m_2 and m_4 piggy-back their data onto the packet.*

As illustrated above, the client wishes to query $m_4 \in S$ where $S = \{m_2, m_4\}$ in that specific order. Thus, the client constructs the route m_1, m_2, m_4, m_3, m_1. Next, the client creates a DAS packet and configures its header (shown as green in the figure) by setting the destination ID of the packet to m_1 and puts the rest of route along with the appropriate **CA** flags (shown as small red boxes) in the *route_buff* (shown as blue in the figure). Each ID occupies two bytes which results in size of $4 \times 2 = 8$ bytes for *route_buff*. This value is also put into the header as the $payload\_size$. At this point, the client broadcast the packet. m_1 receives the packet and inspects its destination field realising that this packet is in fact intended for it. m_1 detaches the first segment of the *route_buff* and extracts the ID of the next node on the path which is m_2. m_1 then creates a new packet and set its destination to m_2. Then, m_1 copies the next 6 bytes of the *route_buf* into the new packets *route_buff* and sets its $payload\_size$ to 6 accordingly. m_1 checks the **CA** flag of the stripped segment and learns that is not being queried. Therefore, m_1 proceeds to broadcast this new packet. Both m_2 and m_3 receive this packet, however, m_3 drops the packet since the packet destination does not match its ID. We avoid mentioning the action of dropping an unwanted packet through the rest of the story as is happens quite so often. Like m_1, m_2 strips the first segment of the *route_buff*, creates a new packet, sets its destination to m_4 and copies the remaining 4-bytes of the routing data into the new packet. However, by checking the **CA** flag of the stripped segment, m_2 notices that it has to attach its response to the end of the *route_buff* which is does (shown as orange in the figure). The size of the payload is now 5. m_2 adjusts $payload\_size$ of the new packet and broadcasts. m_4 takes the same steps as m_2. The new packet is targeted for m_3 and has a payload of size 4 with m_4's response data attached and the end of the payload. it is easy at this point to see how DAS modifies the query packet throughout its journey.

Piggy-backed vs. direct-routed response

In DAS protocol, we piggy-back responses onto the query as it travels through the network. However, we justify our decision by proving that the query cost is oblivious to the specific method we choose to route the the query-response back to the client. When considering an individual node in the network, it is impossible for that node to route the response back to the client using source route since it does not have a complete map of the network. However, an individual node might not require to know the entire route back to the client. The client can establish a broadcast tree, i.e. a minimum spanning tree, which reaches to each individual node and informs it about its parent along the tree. In this manner, all the node need to do is to pass the response to its parent. Each subsequent parent, in turn, hands the response to its own parent until the response reaches the client. We call this method direct-routing. Direct-routing is a very limited version of table driven routing. We show that even if direct-route method is used for query-response, the communication cost associated with the query-response belongs to the same asymptotic class that the query-response would have if the piggy-backed method was used.

Claim 1. *Piggy-backed and directly routed response incur the same asymptotic communication cost on the network.*

Proof. We can completely separate the communication cost incurred by the transfer of the routing information from the cost incurred by the transfer of response data. Since the two options of routing only differ in their delivery of response data, we solely focus on the communication cost due to the transfer of response data.

Let the size of a single response segment be D (D as in Data). We can assert that size of each individual direct-routed response packet is constant $O(D) = O(1)$. We also know that in the case direct-routed response, for each query, all the nodes in the anonymity set S have to send their data to the client. This means that for query anonymity a, at least a separate response packets would have to be send to the client. The maximum number of hops for each of these packets is d the diameter of the network. Therefore, we can conclude that in case of direct-routed response, the communication cost imposed by the response is at most $O(ad)$.

As with piggy-packed response, each node in the anonymity set appends its data to the ongoing query until the query reaches the last node in the anonymity set at which point a response data has been accumulated onto the packet. The last packet sends these a response data back to the client which is at most d hops away. Thus the cost imposed by the query response is also $O(ad)$ in the case of piggy-backed response.

We can see that for both options, the cost incurred by the response data belong to the same asymptotic class of $O(ad)$.

□

The only minor difference between the two options, with respect to the query-response cost, is that the piggy-backed option incurs some extra. This cost occurs while the response data are being collected and continues until the query reaches the last node in the anonymity set. From Theorem 3, we know that the anonymity set S can be traversed in $\Theta(a)$. Thus, we know that at least one of the responses would be transmitted over $\Theta(a)$ hop, at least one other response would be transmitted over $\Theta(a-1)$ hop and so on. The communication cost incurred during this period is $\Theta(a^2)$ or more accurately $\Theta(a^2D)$. However, this cost gets dominated by the cost imposed by the routing portion of the query $\Theta(a^2R)$ where R is the size of each route information. $\Theta(a^2R)$ cost also incur for the direct route method so it is safe to ignore this extra $\Theta(a^2D)$ caused by the piggy-back method unless $R \ll D$ which is not the case for majority of WSN routing protocols. In particular, for DAS, D is 8-bits where R is 16-bits .

Communitation cost revisited

Here, we refine our notion of the communication cost introduced in Section 3.2 in order to adapt it to our implementation of DAS protocol as we, gradually, shift our focus from the theory to experimentation. In the following , We define the communication cost of a query Q_k that targets some node m_k. Let R_{Q_K} represent the ordered set of nodes that reside along the Q_k route. The overall cost of Q_k is:

$$Cost(Q_k) = \sum_{m_i \in R_{Q_K}} DAS_{bc}(m_i, Q_k^i) \tag{4.1}$$

where $DAS_{bc}(m_i, Q_k^i)$ represent the number of bytes that get broadcasted (at line 16 of Algorithm 3) after applying DAS protocol at the m_i upon Q_k^i's arrival. Q_k^i is the transformed version of the original Q_k after $i - 1$ iterations of DAS protocol.

4.3 DAS tight Upper-bound

We have established a function for the tight upper-bound of the query cost in WSNs that employ DAS. The first purpose of this function is to show that (1) it is congruent with the asymptotic boundary of sufficiency property of the theorems in Section 3.4 based on the values that a assumes and (2) to demonstrate that the collected data from the simulation does in fact fall below this upper-bound. This function will be used in Chapter 5 to verify the experimental result.

Figure 4.4 helps us achieve this function. First, we wish to calculate the maximum number of hops a DAS packet goes through in order to query the node m_k

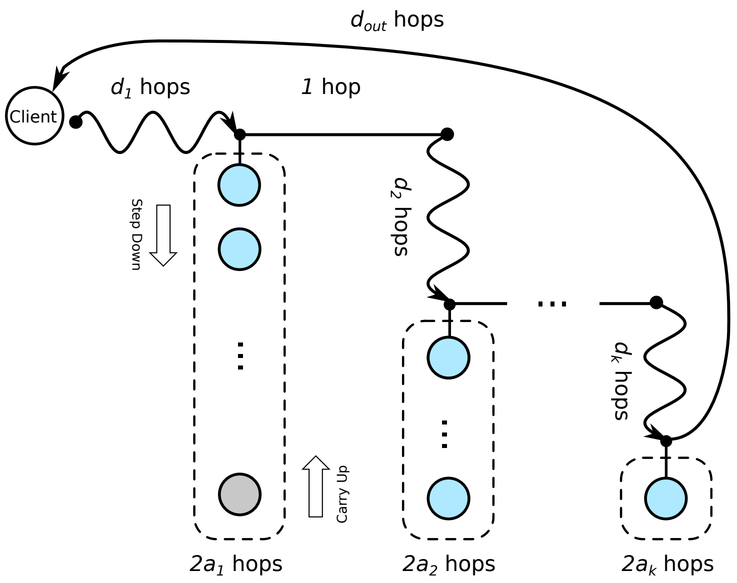

Figure 4.4: *A depiction of the worst case DAS query route in a WSN with an arbitrary topology. On its way into the network, the query alternates between route-in and data-collection phases. Data-collection phases are displayed within dashed regions. The query travels down each region and carries the response data on its way up. Once all the data has been collected, the query make its way to the client i.e. the route-out phase.*

which is furthest from the server. Throughout the following we assume that the size of the anonymity set which holds m_k is exactly a. We can divide the query's journey into three separate phase: route-in phase, data-collection phase, route-out phase. During the route-in phase, the query travels through nodes that are not members of the anonymity set. This phase continues until all the response data are collected. During the data-collection phase, the query travels through members of the anonymity set and collects the required response data. This phase, also, continues until all the response data are collected. The route-out phase start after all the response data are collected at which point the query make its way back to the client. Due to the way the anonymity sets are arranged in arbitrary topology, the anonymity set might be broken into k disjoint chunks of nodes with sizes $a_1, a_2, \ldots, a_k$ such that $a = a_1 + a_2 + ... + a_k$. This causes the query to frequently alternate between the route-in and data-collection phases as opposed to square grid topology where it completes the route-in phase and then moves on to the data-collection phase. Therefore, the number of hops the query take during its route-in phase can be broken into k parts $d_1, d_2, \ldots, d_k$. Let's' say that the query takes d_{in} hops through its combined route-in phases where:

$$d_{in} = (d_1 + 1) + (d_2 + 1) + \cdots + d_k$$

Claim 2. *$d_{in} \leq d$ where d is the diameter of the network.*

Proof. For the purposes of this proof, we can safely assume that the path that a packet takes to reach any node in the network is a subset of the same spanning tree T used by *ConstructAllS* procedure to create the anonymity sets. The reason that we can safely make such an assumption is that we know that at least this spanning three does exist, otherwise we wouldn't be able to arrange the anonymity sets. We can't make any assumptions regarding the existence any other edges that the network graph offers besides the edges in T although they might actually exist. Furthermore, these paths are the shortest path from the server to every other node in the network since T is a minimum spanning tree. This leads to the conclusion that $d_{in} \leq d$ since d_{in} is in fact the shortest from the server to the last member of the anonymity set and by definition it cannot be longer than the diameter of the network. $\square$

During the i^{th} partial data-collection phase, the query travels downward on the branch that contains a_i nodes for a_i consecutive hops and starts collecting data while traveling upward for another a_i hops. Therefore, the number of hops that query takes through its combined data-collection phase is:

$$2(a_1 + a_2 + ... + 2a_k) = 2a$$

Finally, the maximum number of hops taken during the route-out phase is d_{out} which bounded by the diameter of the network. Therefore the maximum number of hops that a query takes in the network in the worst case scenario is:

$$l = d_{in} + 2a + d_{out} \leq 2d + 2a$$

We can calculate the cost incurred by the three sections of a DAS packet: header, route and data, separately throughout query's journey. For simplicity, we consider the size of the header to be H, the size of individual routing segments to be R and the size of individual data segments to be D anonymity set. The header information is transmitted for every hop:

$$\textcircled{A} \leftarrow Hl = 4(2d + 2a) = 8d + 8a$$

The routing information transmitted for the first hop is l and for each subsequent hop its size is decrease by one segment until there are no more routing information:

$$\textcircled{B} \leftarrow \sum_{i=1}^{l} Ri = \sum_{i=1}^{2d+a} 2i = 4d^2 + 2d + 8ad + 2a + 4a^2$$

The sensor data is collected over the span of $2a$ hops. For each single data segment, the earlier it is collected the more overhead it imposes on the network since it has to be carried for a longer distance. Thus in the worst case, all a data segments are collected in the first a hops during which the size of the data continuously increments by one segment per each hop and then all those a data segments are carried for another $a + d$ remaining hops:

$$\textcircled{C} = (\sum_{i=1}^{a} Di) + a(a + d)D = 1.5a^2 + 0.5a + ad$$

Thus the maximum total number of bytes transferred is:

$$\textcircled{A} + \textcircled{B} + \textcircled{C} = 4d^2 + 10d + 9ad + 10.5a + 5.5a^2$$

We achieve the final form of upper-bound for query cost in arbitrary topologies:

$$ub_a(d, a) = 4d^2 + 10d + 9ad + 10.5a + 5.5a^2 \tag{4.2}$$

Tight upper-bound for square grid

We thrive to construct a similar tight upper-bound for WSNs with square grid topology. Similar to our approach in dealing with arbitrary graph topology, we can divide query's journey into three phases of route-in, data-collection and phase-out. There are two main difference between square grid and arbitrary topology. In square grid topology:

1. The DAS query completes the entire route-in phase without interruption through d_1 consecutive hops where $d_1{<}d$
2. The DAS query completes the entire data-collection phase without interruption through a consecutive hops

This leaves us with the following maximum number of hops in the worst case scenario:

$$l = d_{in} + a + d_{out} \leq 2d + a$$

Figure 4.5 demonstrates these aspects of query's journey through square grid topology. Again, we can assess the costs incur by each of the fields of the packet, header Ⓐ, route Ⓑ and data Ⓒ separately :

$$\textcircled{A} \leftarrow Hl = 4(2d + a) = 8d + 4a$$

$$\textcircled{B} \leftarrow \sum_{i=1}^{l} Ri = \sum_{i=1}^{2d+a} 2i = 4d^2 + 2d + 4ad + a + a^2$$

$$\textcircled{C} \leftarrow [\sum_{i=1}^{a} Di] + adD = 0.5a^2 + 0.5a + ad$$

Thus the maximum total number of bytes transferred is:

$$\textcircled{A} + \textcircled{B} + \textcircled{C} = 4d^2 + 10d + 5ad + 5.5a + 1.5a^2$$

We achieve the final form of upper-bound for query cost in square grid topologies:

$$ub_g(d, a) = 4d^2 + 10d + 5ad + 5.5a + 1.5a^2 \tag{4.3}$$

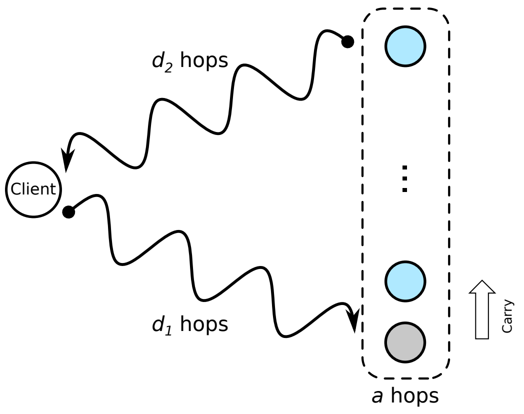

Figure 4.5: *A depiction of the worst case DAS query route in a WSN with square grid topology. During its uninterrupted route-in phase, the query travels to the first member of the anonymity set (based on its respective snake-walk ordering). During its data-collection, the query visits the members of the anonymity set through a consecutive hops. Finally, the query make its way to the client during its route-out phase.*

Chapter 5

Experimental Methods

We have conducted a simulation-based empirical evaluation in support of our analytical results from the previous sections. Our particular intent with the empirical evaluation is to validate the assertions on sufficiency in Theorems 4 and 5 and corollary 1. That is, we simulate the DAS protocol, measure the communication-cost it incurs in various settings, and observe that it is indeed upper-bounded as the analytical assertions tell us.

In Section 5.1 we explain how we generated the WSN topologies on which we run the simulation for various scenarios. In Section 5.2, we discuss our simulation setup and the scenarios that we have simulated. Finally, in Section 5.3, we present the simulation results and explain our observations.

5.1 Random Network Generation

Our experimentation required a large number of randomly generated WSN topologies. Some scenarios required hundreds of random topologies that share certain properties such as size n or diameter d or both. We use random geometric graphs to construct random WSNs topologies required for our experimentation. A random geometric graph is a random undirected graph drawn on a bounded region, e.g. the unit square $[0, 1)^2$. The x and y coordinates of each node are chosen uniformly at random from $[0, 1)$. Two nodes u and v are connected if and only if their Euclidean is less than or equal to some designated value r:

$$||u - v||_2 \leq r$$

However, we have to enforce the following three properties onto the generated network:

1. The network is connected
2. The size of the network is n
3. The diameter of the network is d

To enforce the first two property of the network, we adopt the following simple strategy. We keep adding new node to the graph and merging those components that become connected at each step until the size of the giant component is larger than n. We call this phase the *bulk-up* phase. Figure 5.1 can help us better understand this procedure.

Next we remove all other components of the graph except for the giant component. This will be our main graph. Finally, we continuously remove safely removable nodes from the graph until we reach the desired size of n. We call this phase the *trim-down* phase. A safely removable node is a node that will not undermine the connectivity of the graph if removed from the graph. At each step, we find the set of safely removable nodes S_{rmv}, choose a random node from this set and remove that node from the main graph. S_{rmv} can be obtain by subtracting the set of articulation nodes S_{art} from the set of all graph nodes V:

$$S_{rmv} = V / S_{art}$$

Figure 5.2 shows some examples of articulation points.

The size of the diameter of the network is calculated using the Floyd-Warshal algorithm[5]. There is no deterministic approach of enforcing the size of the network diameter. However, we can increase our chances on achieving the desired

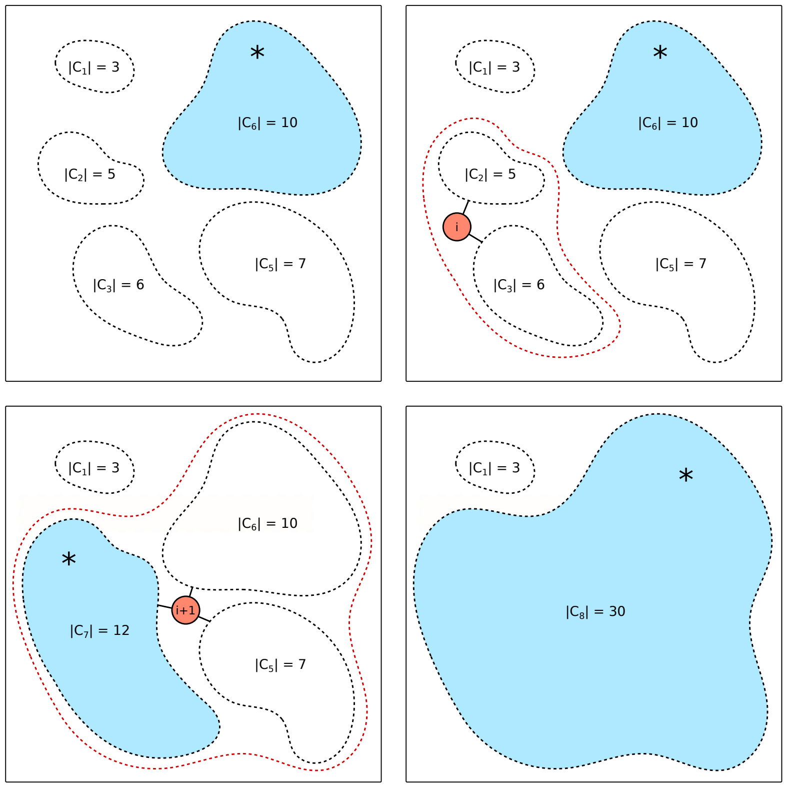

Figure 5.1: *This is an illustration of bulk-up procedure. The desired network size is 25. At step i of the process, C_6 is the giant component with size 10. We add the i^{th} new node which connects C_2 and C_3 and forms the new giant component C_7 of size 12. We add the $(i+1)^{th}$ new node connecting C_5, C_6 and C_7 and forming the new giant component C_8 of size 30 >25. At this point we are done and C_8 is the candidate graph.*

network diameter by adjusting the value of r recursively. Smaller r results in larger diameter d and vice versa. For each network, we set the initial value of r to:

$$r = \frac{1}{\sqrt{n} - 1}$$

This is a good starting point. The rational behind choosing this value is that if n nodes where uniformly arranged in a unit square, each two adjacent nodes would lay $\frac{1}{\sqrt{n}-1}$ Euclidean distance away from one another. In each iteration, we

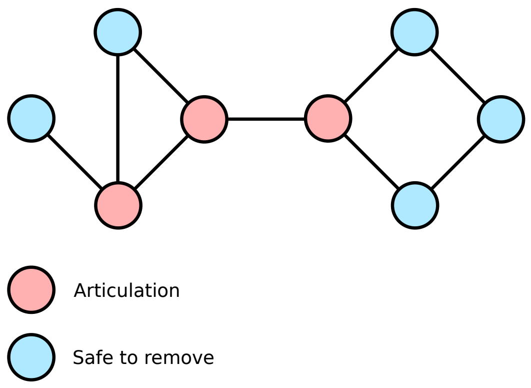

Figure 5.2: *Example of articulation points in a graph. Removing any of the articulation nodes (red) renders the graph disconnected.*

create a new random network and calculate its diameter $\acute{d}$. If $\acute{d}<d$, we have to decrease r. We decrease r by a random amount chosen with normal distribution of mean 0 and standard deviation of:

$$\sigma = |\frac{d-\acute{d}}{d}|$$

If $\acute{d}>d$, we have to increase r. We increase r by the same value described above. Note that the closer we get to d, the smaller the change in the value of r; Therefore, we greedily move towards the optimal value of r which maximizes our chances of finding the desired diameter. Friedrich et al. in their work [8] offer an estimation for the value of d given n, the size of a random geometric graph, and the value of r. However, their approach is not applicable in the context of our intentions. They estimate the diameter of the giant component of a random geometric graph of size n where we ensure that the giant component itself is at least as large as n and then we trim it down. More so, Friedrich requires that r is larger than the connectivity threshold. In our case, for relatively large values of d, it is necessary that r fall below the connectivity threshold.

Another important aspect of our approach is that we do not discard the undesirable networks created during the continuous runs of this process. We save and archive these networks. In certain cases, it takes tens of tries before we achieve a specific network with some desired n and d. By saving all those unwanted networks throughout the process, over time, we accumulate a massive database of networks with a variety n and d combinations which are reusable for future scenarios. This greatly improves the overall efficiency of random network generation process. The details of process described above is provided in the Algorithm 4.

Algorithm 4 Random network Generation

```
 1: procedure constructRandomWSN($n$, $d$)
 2:     $G \leftarrow$ searchNetworkDatabase$(n, d)$
 3:     if $G = \varnothing$ then
 4:         $\acute{r} \leftarrow r \leftarrow$ adjustR$(n, d)$
 5:         repeat
 6:             $G \leftarrow$ bulkUp$(n, \acute{r})$
 7:             trimDown$(G, n)$
 8:             saveNetworkToDatabase$(G)$
 9:             $\acute{d} \leftarrow$ getNetworkDiameter$(G)$
10:             $dr \leftarrow \frac{d-\acute{d}}{d}$
11:             $s \leftarrow |dr|/dr$
12:             $\acute{r} \leftarrow r - s *$ randomNormalDstr$(0, |dr|)$
13:         until $\acute{d} = d$
14:     return $G$ as $N$
15: procedure bulkUp$(n, r)$
16:     $C_* \leftarrow \varnothing$                         ▷ $C_*$ is the current giant component
17:     while $|C_*| < n$ do
18:         choose random $x, y \in [0, 1]$
19:         $m \leftarrow$ createNewNode$(x, y)$
20:         $S \leftarrow$ neighboursOf$(m, r) \cup m$
21:         $C \leftarrow \bigcup_{m_i \in S} C(m_i)$        ▷ $C(m_i)$ is the component containing $m_i$
22:         if $|C_*| < |C|$ then
23:             $C_* \leftarrow C$
24:         return $C_*$ as $G$
25: procedure trimDown$(G, n)$
26:     while $|G| \neq n$ do
27:         $S_{atr} \leftarrow$ findSafeToRemoveNodes$(G)$
28:         choose random $m \in S_{atr}$ and remove $m$ from $G$
```

5.2 Simulation Environment, Setup and Scenario

We use Tossim[11] running on TinyOS 2.1.2 framework to simulate DAS. Networks are generated in a homogenous manner using MICAz nodes. Each node is assigned a unique 15-bits ID. TinyOS's *Active Message*[2] interface is used for data transmission between the nodes. All nodes receive their neighbors' transmission at -10 dBm. Also, we use *Casino Lab* noise trace [1] as the background noise model. This noise model provides a quite environment for wireless communication. Furthermore, since we run the simulation for each query separately, there is at most one active instance of transmission at any moment. This arrangement prevents any packet collision. This combination of configurations provides an strong enough signal from the receiver node's perspective that the possibility of packet loss is almost non-existent. We would like to minimized the possibility of packet drop in order to focus solely on DAS behavior rather than communication quality of the network. Throughout all the scenarios described below, we use the notion of query cost described the Equation . Tossim provides us with facilities that enable us to accurately measure the number of byte that each node broadcasts.

In a WSN with arbitrary topology, there are three acting variables that determine how DAS behaves with respect to trade-off between query-anonymity and communication-cost. These three variables are n, the size of the network, a, the size of the anonymity set and, d the diameter of the. We isolate the effects each of these factors have on the query cost into a separate scenarios. In all scenario, we calculate the cost of the most expensive query in the network based on the notion of cost defined in Equation 5.2. The most expensive query in the context of arbitrary topologies is the node that its distance from the server is exactly the size of the diameter of the network.

1. *Variable anonymity set size a*:

   In this scenario, we apply DAS to four different networks that share the same values of $n = 100$ and $d = 20$. We simulate and record the cost of the most expensive query in each of the four networks as we increase a the size of the anonymity set. Finally for each a, we average the costs of

[1]There are two noise traces provided by Tossim suit: Meyer library and Casino lab. Among the two, Casino lab offers a much higher SNR.

these four individual cases and show that it falls under the asymptotic upper-bound. The result of this experiment is shown in Figure 5.3 and discussed in Section 5.3.

2. *Variable diameter size d*:

   In this scenario, for each value of $d \in [5, 10, 15, ..., 50]$, we generate four different networks that share the same values of $a = 25$ and $n = 100$. For each value of d, we simulate and record the cost of the most expensive query in each of the four networks. Finally for each d, we average the costs of these four individual cases and show that it falls under the asymptotic upper-bound. The result of this experiment is shown in Figure 5.4 and discussed in Section 5.3.

3. *Variable network size n*:

   In this scenario, for each value of $n \in [100, 110, 120, ..., 270]$, we create four different network that share the same values for $a = 25$ and $d = 100$. For each value of n, we simulate and record the cost of the most expensive query in each of the four networks. Finally for each n, we average the costs of these four individual cases and show that it falls under the asymptotic upper-bound. The result of this experiment is shown in Figure 5.5 and discussed in Section 5.3.

In the case of square grid, there are only two acting variables that dictate DAS's behaviour with respect to trade-off between query anonymity and communication cost. These two variables are n, the size of the network, and a, the size of the anonymity set. We explore three values of $n = 100, 225$ and 576, which correspond to $10 \times 10, 15 \times 15$ and 24×24 square grids. For each values of n we steadily increase the size of anonymity set a. We wish to observe that DAS displays a consistent behavior across various network sizes. For square grid topology, we simulate DAS under only one scenarios. In this scenario, we simulates the most expensive query. The most expensive query is the query that targets the node that is furthest from the server in terms of Manhattan distance. We position the server and the target of the query on the opposite corners of the square grid. This arrangement gives us the maximum distance between the server and the target of the query. The experimental results after running this scenario is shown in Figure 5.6 and will be discussed in Section 5.3.

Limits of the simulation tool and other considerations

TinyOS's radio modules imposes a maximum size on the packet size during any one instance of transmission. TinyOS documents does not specify an exact value for this maximum. However, through trial and error, we found this maximum to be about $215 \sim 225$ bytes . If a packet is larger than this maximum, it must be broken into smaller fragments which would then be transmitted over multiple instances of packet transmission. We decided not to fragment DAS packets in our simulation. A ramification of such a decision is that the length of the route of any simulated DAS query would be bounded by ~ 110 hops. We have taken this into consideration for the simulated topologies.

Another important consideration is that, when constructing anonymity sets, consecutive values of a in bounded intervals result in the exact same anonymity set arrangement. This circumstance is the direct result of the way that we obtain k, the number of the anonymity sets, where $k = \left\lfloor \frac{n}{a} \right\rfloor$. This fact in combination with our method of constructing anonymity sets (described in Algorithm 2) will result in the same sets arrangement for all values of a within each interval. For example, for $n = 100$, for all values of $a \in [21, 25]$ we get $k = 4$ and therefore the same set arrangement. Similarly, for all values of $a \in [26, 33]$ we get $k = 3$, for all values of $a \in [34, 50]$ we get $k = 2$ and so on. Therefore, in the cases of "*Variable anonymity set size a*" and "*Square Grid*" scenarios, there is no point in simulating all values of a that belong to the same interval. We only choose the largest a from each interval since it is the closest value to the actual size of the anonymity sets.

5.3 Simulation Results

,

We can see that the function ub_g shown in Equation 4.2 is strictly increasing with respect to both variables a and d. Hence, if we keep the variable $d = d_*$

fixed, for $a_1 \leq a_2 \Rightarrow ub_a(a_1, d_*) \leq ub_g(a_2, d_*)$. Similarly, if $a = a_*$ is fixed, for $d_1 \leq d_2 \Rightarrow ub_a(a_*, d_1) \leq ub_g(a_*, d_2)$. Having these properties, we can infer that for $a \leq d$, the worst case cost c_{wst}:

$$c_{wst} \leq ub_a(a, d) \leq ub_a(d, d) = 18.5d^2 + 20.5d \tag{5.1}$$

We can conclude, in WSNs with arbitrary topology, that for $a \in [1, \Theta(d)]$ we have $c_{wst} \in O(d^2)$ which is aligned with sufficient property of Theorem 4. The results of the simulation of the "*Variable anonymity set size a*" scenario is shown in Figure 5.3. We can see that the collected data for any $a \leq 20$ falls below the proposed tight upper-bound shown by the flat line in the plot. Similarly, for $a{>}d$, the worst case cost c_{wst} is:

$$c_{wst} \leq ub_a(a, d) \leq ub_a(d, d) = 18.5a^2 + 20.5a \tag{5.2}$$

Therefore, when $a \in [\Theta(d), n]$ we have $c_{wst} \in O(a^2)$ which is aligned with sufficient property of Theorem 5. Again, we can see that in Figure 5.3 the collected data for any $a{>}20$ also falls below the proposed tight upper-bound.

Furthermore, by exploring the results from simulating the "*Variable diameter size d*" scenario, we can show that changing the network diameter d impacts DAS's behavior in the manner predicted by the Theorems4 and 5. If we fix the value of the anonymity set size a, we would expect the cost for $d \in [1, \Theta(a)]$ to be asymptotically bounded by $O(a^2)$. Using Equation 4.2 and the same logical reasoning we used for the "variable a" scenario to achieve the relation 5.1 we can assert that this is in fact the case and that the cost is asymptotically bounded by $O(a^2)$. Equivalently, for $d \in [\Theta(a), n]$, we would expect the cost to be asymptotically bounded by $O(d^2)$. Relation confirms this our prediction which further verifies the validity of Theorems4 and 5. Figure 5.4 illustrates the results after running the simulation for the "*Variable diameter size d*" scenariofor $n = 150$ and $a = 25$. We can observe that the collected data in all cases falls below the predicted asymptotic boundaries.

Finally, Figure 5.5 present the simulation results for the "*Variable network size n*" scenario for $a = 10$ and $D = 20$. We can see that increasing the size of

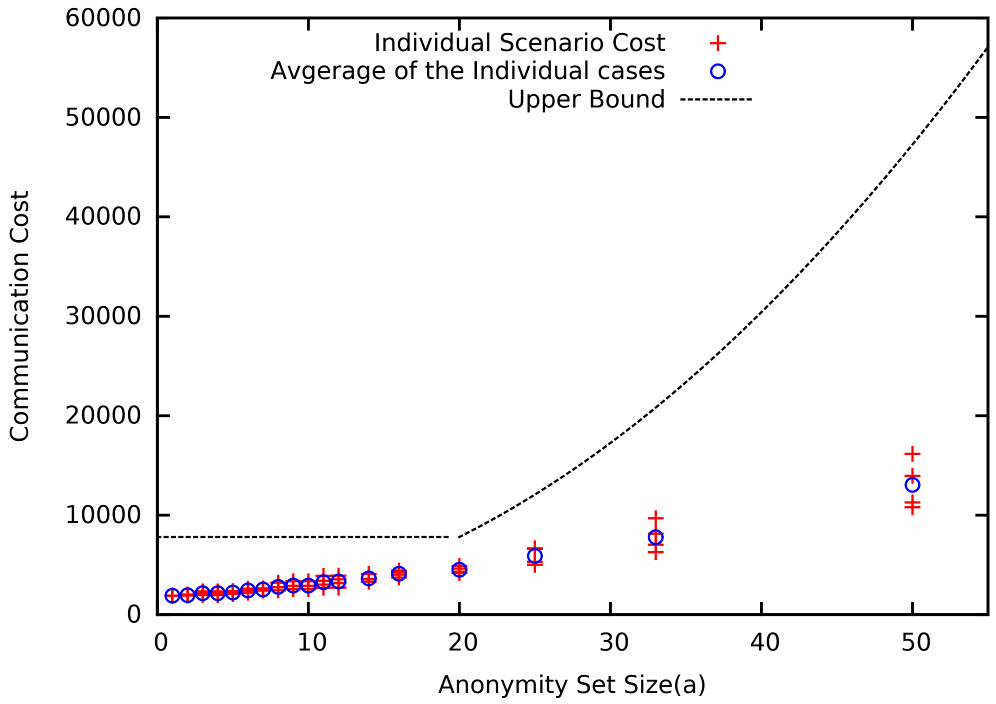

Figure 5.3: *Our empirical results for worst case query cost in arbitrary topologies with varying anonymity, a but fixed d, n. Red crosses on each point of the "Anonymity Set Size (a)" axis correspond to the worst case query costs for four different networks and their average value is represented by a blue circle. All values of a use the same four network since the value of a affect the set arrangements and not the topology of the network graph.*

the WSN, i.e. the number of nodes n does not impact the communication cost as long as as the diameter d and the anonymity set size a are the same. Thus, we can safely assert that in arbitrary topologies, the communication cost is independent of network size and only depends on the degree of the query anonymity and the diameter of the network.

A closer inspection of the function ub_a reveals yet another, more implicit, aspect of the trade-off between communication cost and query anonymity. If we fix the value of a, ub_a becomes a function of the network diameter d. We notice that the WSN designer can adjust the communication cost by modifying d in much the same way that she would through tweaking a. This is a valuable characteristic of this tradeoff. If the network designer is designing a certain WSN with

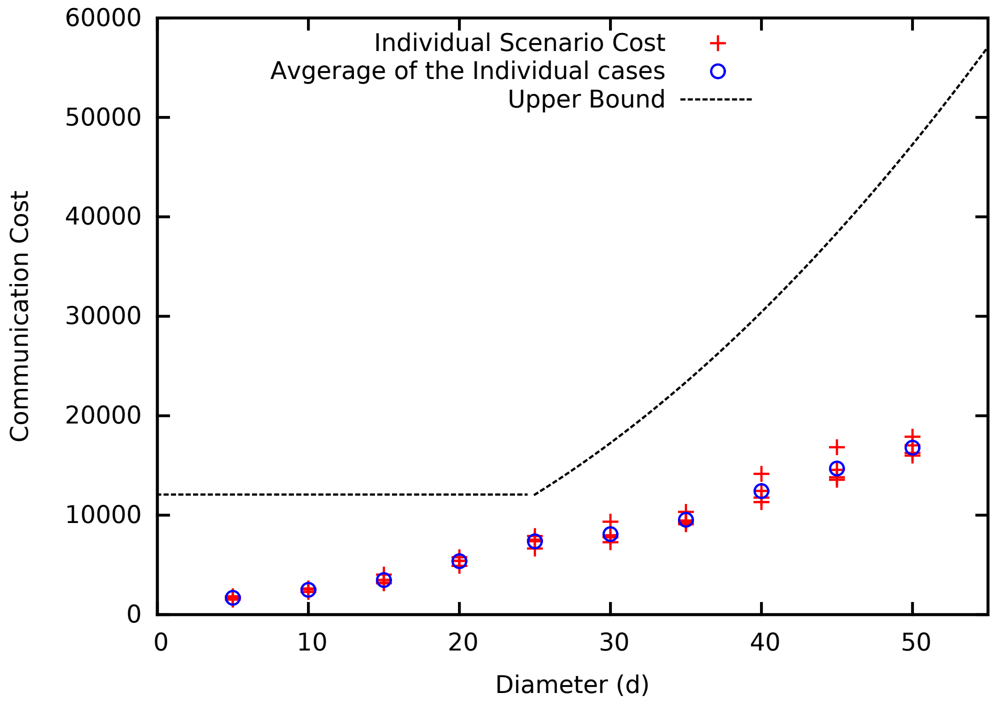

Figure 5.4: *Our empirical results for worst case query cost in arbitrary topologies with varying diameter, d but fixed a, n. Like Figure 5.3 red crosses on the "Diameter (d)" corresponds to the worst case query costs for four different networks and the blue circle is their average. However, unlike Figure 5.3, for each d there is a different set of four arbitrary network since d affects the topology of the networks.*

an specific constraint on communication cost, she can maximize the anonymity by minimizing the network diameter. In simple words, denser networks result in smaller diameter which allows for higher anonymity and sparser network result in larger diameters and penalize the design for desired anonymity requirements. This becomes specially useful when we realize that DAS is scalable with respect to network size. In other words, for larger values of a cost is independent of the number of nodes in WSN. We can add new nodes to the network without worrying about cost increase as long as the diameter of network does not increase.

In the case of square grid topology, we can see that the function ub_g shown in Equation 4.3 is also strictly increasing with respect to both variables a and d. Thus similar to our approach for arbitrary topology, we can infer that for $a \leq d$,

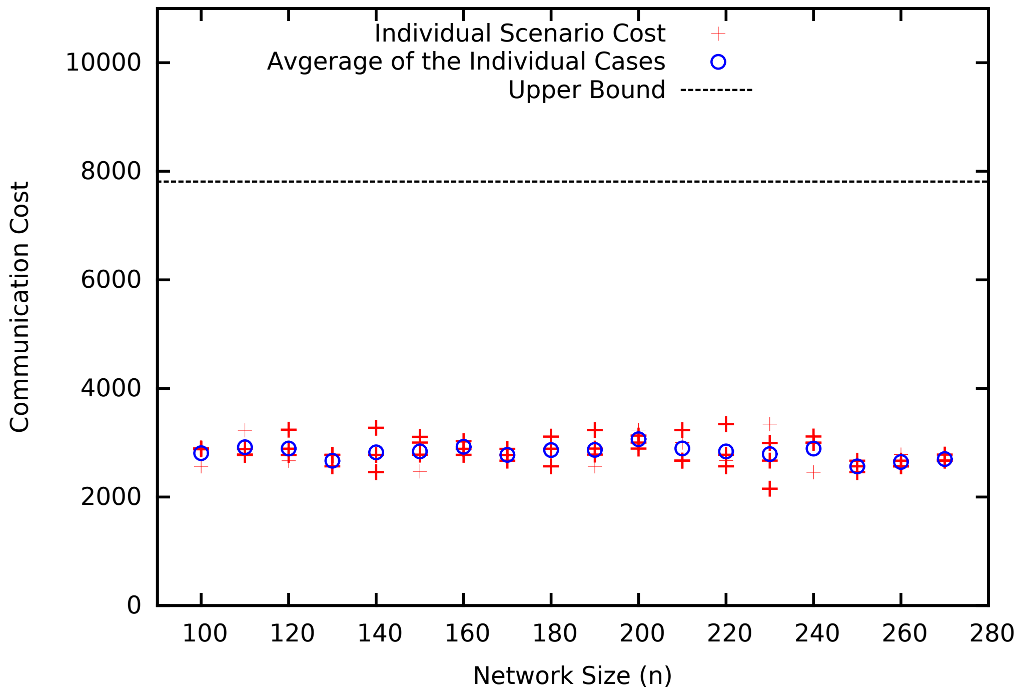

Figure 5.5: *Our empirical results for worst case query cost for four networks with different arbitrary topologies with varying network size, n but fixed a, d. Similar to Figure 5.4, for each n there is a different set of four arbitrary network since n affects the topology of the networks.*

the worst case cost c_{wst}:

$$c_{wst} \leq ub_g(a, d) \leq ub_g(d, d) = 10.5d^2 + 15.5d$$

However, in square grid topology, the value of d is directly tied to the size of the network n through the following relationship:

$$d = 2(\sqrt{n} - 1)$$

Therefore, we rephrase the statements above by substituting $2(\sqrt{n} - 1)$ for d. For $a \leq 2(\sqrt{n} - 1)$, the worst case cost c_{wst}:

$$c_{wst} \leq 42n - 53\sqrt{n} + 11 \tag{5.3}$$

This means for $a \in [1, \Theta(\sqrt{n})]$ we have $c_{wst} \in O(n)$ which is aligned with sufficiency property of Corollary 1 . After simulating the worst-case scenario with three different WSN networks of sizes of $n_1 = 100$, $n_2 = 225$ and $n_3 = 576$ (i.e 10×10, 15×15 and 24×24), the collected results verifies that the worst case query cost for all three networks falls under the calculated upper-bound. The upper-bound valuse for the thee network size are $ub_1 = 3681$, $ub_2 = 8666$ and $ub_3 = 22931$. These upper-bounds are shown by flat-lines in Figure 5.6. We can see that in all cases, when $a \leq 2(\sqrt{n}-1)$, i.e. when $a \in [1, \Theta(\sqrt{n})]$, the collected data falls below the calculated upper-bound.

We use the same logic for when $a>d$ and conclude:

$$c_{wst} \leq ub_g(a, d) \leq ub_g(a, a) = 10.5a^2 + 15.5a \tag{5.4}$$

This means for $a \in [\Theta(\sqrt{n}), n]$ we have $c_{wst} \in O(a^2)$ which is aligned with sufficiency property of Corollary 1. Figure 5.6 illustrates that for all three network sizes, worst query cost falls under the calculated upper-bound for values of $a>2(\sqrt{N} - 1)$.

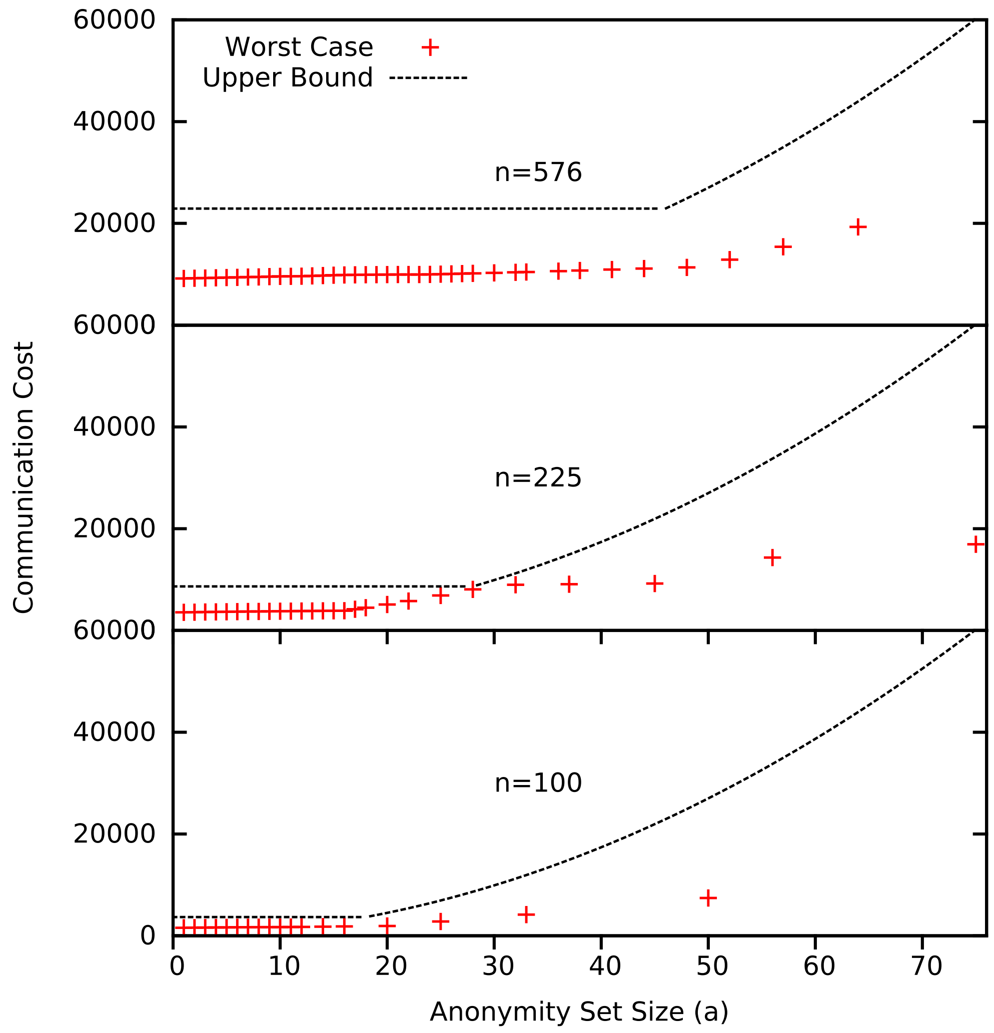

Figure 5.6: *Our empirical results for worst and average case query cost in square grid topology. Three networks of sizes* $n = 100, 225$ *and* 576. *Their respective diameters are 18, 28 and 46*

Chapter 6

Conclusion

We have analysed the trade-off between query-anonymity and communication-cost in the context of wireless sensor networks. For an unconditional notion of query-anonymity [1], we have established what we call the limits of this trade-off. We began with source-routed connected undirected graphs. In such settings, the necessary and sufficient communication-cost for achieving query-anonymity $a \in [1, d]$ is $\Omega(d^2)$ and $O(d^2)$ respectively, where d is the diameter (longest shortest-path) of the graph. For $a \in [d, n]$, we have shown that the corresponding costs are $\Omega(a^2)$ and $O(a^2)$ respectively. We have also conducted a simulation using Tossim, which provides empirical validation for our assertions on sufficiency.

There is considerable scope for future work. One is to explore table-based routing which is an alternative to source-routing. That is, the client does not provide a route; rather, the nodes in the wireless sensor network are capable of routing by themselves. In this case, the trade-off between communication-cost and query-anonymity does change in particular cases. Another involves incorporating other parameters into the problem that are relevant. For example, it can be argued that grouping adjacent wireless sensor nodes into the same anonymity set may not provide sufficient query-anonymity. This may be because simply the attacker knowing that a set of adjacent nodes are the targets of queries may reveal information we seek to protect, in certain applications.

Consequently, we may want to incorporate location-sensitivity in our notion of query-anonymity. This may require, for example, that nodes that are grouped together in an anonymity set are necessarily far apart in the topology of the network. It will then be interesting to ask the same questions that we ask in this work: what is the communication-cost that is necessary and sufficient to achieve a certain query-anonymity?

Chapter 7

Related Work

We analyse the trade-off between communication-cost and query-anonymity in the context of wirless sensor networks. To our knowledge, this trade-off was first pointed out in the work of Carbunar et al. [6]. That also happens to be the work to which ours is related most closely. That work proposes notions of query-anonymity, two of which we have addressed in this work and argued to be deficient. We have proposed an unconditional notion in lieu. Also, though the work of Carbunar et al. [6] point out the trade-off, they do not analyze it. Specifically, they do not point out the necessary and sufficient communication-cost as we do to achieve a certain query-anonymity. The work of De Cristofaro et al. [11] also is related to our work in that that work also considers query-anonymity. However, the threat model is weaker than ours. Also, the trade-off between communication- cost and query-anonymity is not analyzed as we do. Our notion of query-a-anonymity is related to the work such as that of Serjantov and Danezis [24], and Pfitzmann and Hansen [21], which propose quantitative notions of anonymity. The former work points out that work prior to it on quantifying anonymity based on anonymity sets, for example, [10], is deficient. We point out that our notion of an anonymity set in the context of the DAS protocol does not have those deficiencies. Our work generalizes the work of Serjantov and Danezis [24] in that we allow a, the query-anonymity, to take on any value in [1, n]. However, our work is more limited in that we address query-anonymity in the context of wireless sensor networks only. The work of Serjantov and Danezis [24] also does not consider the trade-off as we do. The work of Pfitzmann and Hansen [21] mentions a variant of what we call the DAS protocol. They also propose notions of anonymity, but precisely as we do, and also do not consider the trade-off issue as we do. Certainly, our notion of query anonymity has commonalities with other prior work, for example, [3, 4, 12, 15, 16], in that it follows analogous approaches to what is used in the formalization of semantically secure encryption [14] to define the receiver anonymity notion. We see this as a validation for the intuitive appeal of our notion. As we mention earlier, we claim our notion as novel only in the context of query-anonymity. Furthermore, our focus is beyond characterizing a notion of anonymity; we seek to study the trade-off that is the focus of this paper. Of course, our work is also related to work on privacy and anonymity in the context of wireless sensor networks, such as [25, 28]. The work of Carbunar et al. [6] provides an excellent overview of the work in that area. The work of Shao et al. [25] addresses a weaker attacker than we do in that the wireless sensor network is assumed to be trusted from the standpoint of disclosure. Also, the trade-off between communication-cost and query-anonymity is not their focus, as in our paper. This is also the case with the work of Yang et al. [28], whose focus, rather, is the placement of what they call proxies, which generate dummy traffic to provide anonymity for sensor nodes.

Bibliography

[1] Anonymous authors. Anonymized title. In *Proceedings of Fifth ACM Conference on Data and Application Security and Privacy, CODASPY 2015*, San Antonio, TX, USA, Under anonymous review. ACM.

[2] P. Buonadonna, J. Hill, and D. Culler. Active Message Communication for Tiny Networked Sensors, 2001. Available from `http://www.tinyos.net/papers/ammote.pdf`.

[3] B. Carbunar, Y. Yu, W. Shi, M. Pearce, and V. Vasudevan. Query privacy in wireless sensor networks. *ACM Trans. Sen. Netw.*, 6(2):14:1–14:34, Mar. 2010.

[4] D. L. Chaum. Untraceable electronic mail, return addresses, and digital pseudonyms. *Commun. ACM*, 24(2):84–90, Feb. 1981.

[5] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms (3. ed.)*. MIT Press, 2009.

[6] C. Díaz, S. Seys, J. Claessens, and B. Preneel. Towards measuring anonymity. In *Proceedings of the 2nd international conference on Privacy enhancing technologies*, PET'02, pages 54–68, Berlin, Heidelberg, 2003. Springer-Verlag.

[7] E. De Cristofaro, X. Ding, and G. Tsudik. Privacy-Preserving Querying in Sensor Networks. In *Proceedings of 18th International Conference on Computer Communications and Networks (ICCCN 2009)*, pages 1–6, aug. 2009.

[8] T. Friedrich, T. Sauerwald, and A. Stauffer. Diameter and Broadcast Time of Random Geometric Graphs in Arbitrary Dimensions. *Algorithmica*, 67(1):65–88, 2013.

[9] O. Goldreich. *Foundations of Cryptography: Volume 2, Basic Applications*. Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[10] L. Kissner. *Privacy-Preserving Distributed Information Sharing*. PhD thesis, Carnegie Mellon University, July 2006. Available from `http://www.cs.cmu.edu/~leak/papers/thesis.pdf`.

[11] P. Levis, N. Lee, M. Welsh, and D. Culler. TOSSIM: Accurate and Scalable Simulation of Entire TinyOS Applications. In *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, SenSys '03, pages 126–137, New York, NY, USA, 2003. ACM.

[12] M. K. Reiter and A. D. Rubin. Crowds: Anonymity for Web Transactions. *ACM Trans. Inf. Syst. Secur.*, 1(1):66–92, Nov. 1998.

[13] A. Serjantov and G. Danezis. Towards an information theoretic metric for anonymity. In *Proceedings of the 2nd international conference on Privacy enhancing technologies*, PET'02, pages 41–53, Berlin, Heidelberg, 2003. Springer-Verlag.

[14] M. Shao, Y. Yang, S. Zhu, and G. Cao. Towards Statistically Strong Source Anonymity for Sensor Networks. In *INFOCOM 2008. The 27th Conference on Computer Communications. IEEE*, pages –, April 2008.

[15] L. Sweeney. K-anonymity: A Model for Protecting Privacy. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 10(5):557–570, Oct. 2002.