LEVERAGING STRUCTURE FOR COMMUNICATION IN HUMAN-CENTRIC DTNS

BY

SAMUEL C. NELSON V

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2011

Urbana, Illinois

Doctoral Committee:

Associate Professor Robin Kravets, Chair, Director of Research
Associate Professor Tarek Abdelzaher
Assistant Professor Matthew Caesar
Assistant Professor Yih-Chun Hu
Ram Ramanathan, Ph.D., Raytheon BBN Technologies

Abstract

Current delay- and disruption-tolerant networks are *human-centric* in nature, in that mobility and communication tend to follow human-based characteristics, such as certain high-level mobility patterns, the development of groups, clustering, and variance in popularity. These networks do not rely on infrastructure and hence are critical to supporting many environments including emergency response, pocket-switched, vehicular, military, and community networks. Unfortunately, there is currently a lack of communication protocols that work well in these environments, where resources can be heavily constrained and malicious nodes may be present. Therefore, the goal of our work is to allow effective, efficient, and robust communication in human-centric DTNs, which we show possible by taking advantage of inherent *structure* found in these networks. To accomplish this goal, and progress the state-of-the-art, we identify two categories of research that must be enhanced: (1) supporting tools and (2) routing protocols.

Supporting tools are used for the understanding, development, and support of routing protocols, and we consider two components for this category. First, in order to design routing protocols for human-centric DTNs, it is critical to properly understand the environment and challenges posed by these networks. Therefore we develop a high-level mobility model for a typical human-centric DTN: a disaster recovery network. This model integrates the concept of different classes of nodes, referred to as roles, reacting differently to external events. This highly-parametrized model allows exploration of many different graph-theoretic properties of human-centric DTNs and provides a better understanding of how DTN routing protocols can best transmit data throughout the network. Furthermore, it acts as a useful mobility tool for protocol simulation. Second, in order to utilize the potential of group-based communication, we develop a local and robust group-management protocol, called MembersOnly. This protocol allows nodes to quickly and accurately transmit group membership information throughout the network, and is robust to malicious nodes attempting to disrupt the process. Through analysis and simulation, it is shown that MembersOnly can withstand multiple types of attacks, with only very limited periods of vulnerability.

The second category, routing protocols, directly allows applications to transmit data throughout the network. Here, we consider three components. Like most networks, unicast is a fundamental and neces-

sary form of communication for human-centric DTNs. Therefore, we develop a highly efficient and effective unicast protocol, called Encounter-based Routing (EBR). EBR is an intentionally resource-friendly protocol that excels in the resource-constrained environments of human-centric DTNs through intelligent replication based on inherent network structure. We show that EBR can achieve up to a 40% improvement in message delivery over current state-of-the-art, while achieving up to a 145% increase in goodput. Next, due to structure inherent in many human-centric DTNs, group-based communication can be a natural and powerful form of communication. Therefore, to support group-based communication, we present an overarching, protocol-independent way to enhance current unicast protocols, giving them the ability to perform anycast. This enhancement can be done in a thin shim beneath the routing layer, allowing the unicast protocols to run unmodified. Through evaluation, we show how different parameters and network conditions affect anycast performance, and how these differ from unicast. Finally, to allow highly flexible group-based communication, we explore manycast in DTNs. First, we thoroughly analyze the difficulty of varying manycast requests, deepening our understanding of how replication should change as the request becomes more or less difficult. Second, we show via simulation that an effective manycast protocol must dynamically change its replication strategy on a per-message basis. Third, we present a manycast meta-protocol, which dynamically selects an appropriate low-level protocol based on the current request and network conditions.

To my wonderful wife Rachel.

Acknowledgments

This thesis would not have been possible without the support of many people. I thank my family, particularly my wife Rachel, my mom and dad, and my sister Michelle, for their encouragement and support throughout my life. Additionally, I thank my mother- and father-in-law for their support. I also extend thanks to my research group members: Albert Harris, Mehedi Bakht, Farhana Ashraf, Riccardo Crepaldi, and Nat Thompson. I extend a special thanks to my gracious advisor Robin Kravets, who worked tirelessly to give me help and guidance. Furthermore, I thank my committee members for their guidance: Tarek Abdelzaher, Matthew Caesar, Yih-Chun Hu, and Ram Ramanathan. Finally, I thank God for His numerous blessings on my life.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The ubiquitous availability of wireless communication has pushed researchers from looking at relatively connected ad-hoc networks to frequently disconnected delay- and disruption-tolerant networks (DTNs). These networks, characterized by intermittent connectivity and heavy partitioning, generally cannot rely on supporting infrastructure, leaving them at the mercy of the individual ad-hoc nodes to transmit data [26]. The key benefit to these networks is their ability to enable communication without having to assume the presence of infrastructure. For this reason, DTNs are critical to supporting emergency response networks [34], pocket-switched social networks [13], vehicular networks [10], military or battlefield networks, third-world development networks [26], and community networks (which include a composition of vehicular networks, pedestrian networks, and potentially stationary access-points) [16]. There are many reasons why infrastructure should not be assumed in these environments. First, infrastructure many not be available at all, as is often the case in emergency response or battlefield networks, and hence the only way to communicate is to utilize opportunistic contacts between nodes in the network. Second, even if infrastructure is available, it may be either too costly or too inconvenient. For instance, user-created mobile networks may be a free alternative to costly cellular networks. Furthermore, DTNs may be a much faster way to communicate with friends, since many 3G cellular networks are frequently overloaded, severely degrading performance.

An interesting characteristic for many DTNs, including the ones listed above, is that their mobility and communication patterns tend to follow human-based characteristics, such as certain high-level mobility patterns, the development of groups, clustering, and variance in popularity [13, 16, 49]. These *human-centric DTNs* are particularly interesting, since wireless-enabled mobile devices are becoming pervasive, and advances in wireless vehicular technology are increasing. However, there are currently many technological issues involving communication that must be addressed before these networks can experience wide-spread use. In particular, we believe there are two fundamental issues that are hindering the use of these networks. First, there is a lack of effective unicast routing protocols that work well *even in resource-constrained environments*. Since many DTN environments are created from mobile devices, resources such

as battery life, bandwidth, buffer space, and contact duration cannot be assumed to be plentiful, as many current DTN unicast protocols do [32, 4, 10, 55]. Unfortunately, the ones designed for resource-constrained environments are not as effective as they should be [51]. Second, since current DTNs are human-centric in nature, group-based communication is natural and should be enabled. The ability to form, propagate, and reconcile group information, even in the generally untrustworthy DTN environment, is necessary for critical forms of communication. This includes *anycast*, where the goal is to reach at least one member of a particular group, and *manycast*, where the goal is to reach at least k members of a particular group, where k can vary between requests. Group-based communication has not been practically considered for DTNs, which is hindering their wide-spread use.

Of the many potential environments suitable for DTNs, we choose two examples to illustrate the importance of our stated goals. First, consider an emergency response scenario where primary infrastructure, such as centralized cellular services, are either down (e.g., destroyed) or unusable (e.g., overloaded) [35]. Since it is likely that many of the people will be carrying wireless devices, an impromptu DTN fits this scenario perfectly, and offers the best, and probably only, solution for communication. Unicast is clearly important, since there will be many messages inquiring and responding to inquires about the well-being of specific people, as well as fine-grained communication between emergency responders. In addition, group-based communication will be critical, since people are naturally divided into role-based groups. A civilian is likely to attempt to enlist the help of *any* emergency responder, as opposed to a specific one. Similarly, police officers are likely to request the service of any ambulance, or any k ambulances, as opposed to specific ones. Note that this scenario presents numerous resource-related challenges, such as limited battery life, buffer space, bandwidth, and contact duration, as well as the potential presence of untrustworthy nodes.

As a second example, consider a community DTN network consisting of pedestrians, cars (e.g., a vehicular network), buses, stationary access points, etc [16]. In many areas, a DTN is preferred to cellular services, for numerous reasons. First, it will likely be free, as opposed to cellular networks. Second, current cellular networks are heavily overloaded resulting in degraded performance. A free alternative for relatively short-range communication presents an attractive option. Third, many areas around the world are not covered by cellular networks at all. DTNs working in all of these environments must support unicast, since most current applications require it and the desire for direct communication between two specific nodes is not likely to vanish in the near future. However, there are many useful scenarios for group-based anycast. As buses become equipped with Internet-able gateways, individual cars on the road would have incentive to contact *any* bus or stationary access point, instead of a specific one, for its gateway capabil-

ity. Furthermore, pedestrians may be more interested in using the network to call any cab as opposed to a specific one. Like before, many resources will be highly constrained, posing challenges to routing protocols. Hand-held devices may be constrained by battery life and buffer space, whereas cars may be constrained by contact duration.

Our key observation is that, while communication is more difficult in DTNs than traditional ad-hoc networks, many scenarios suitable for DTNs, particularly human-centric DTNs, exhibit a high degree of both mobility, social, and communication structure. Communication protocols should be aware of and work cooperatively with this structure, instead of being oblivious to it. We therefore leverage this structure and work with it to allow communication to be *effective*, in that there are high-delivery ratios and low end-to-end delays, *efficient*, in that they are resource-friendly and perform well in resource-constrained environments, and *robust*, in that they are, to some level, resistant to attacks in untrustworthy or malicious environments.

1.1 Necessary Components and Research Contributions

The goal of this thesis is to allow effective, efficient, and robust communication in DTNs, including unicast and anycast. We particularly focus on human-centric DTNs, where structure can be leveraged. In order to accomplish this goal, we identify two categories that must be enhanced: (1) supporting tools and (2) routing protocols. Supporting tools are used for the understanding, development, and support of routing protocols, and we consider two components that fall under this category. First, it is necessary to understand the graph-theoretic characteristics of human-centric DTNs, as well as the challenges they pose, in order to understand how communication protocols react to and can be effective in these environments. Along these lines, it is necessary to have human-centric DTN mobility models for which to evaluate routing protocols. Second, in order to enable the communication potential provided by inherent group structure in these networks, local and robust group management must be enabled. This allows group information to quickly propagate throughout the network, and be accurate even if many nodes are untrustworthy, and opens the door for group-based communication. The second category is the development of actual routing protocols, which are the protocols that directly interact with the application. Here, we consider three components. First, unicast routing must be enabled for resource-constrained DTNs, since the vast majority of DTNs are, to some degree lacking resources. Unicast is a fundamental routing paradigm that many applications rely on. Second, building off of the group management, group-based communication, particularly anycast, must be developed. Finally, in order to generalize group-based communication, manycast in a DTN set-

ting is explored. This includes understanding how the fundamental difficulty of a k of m manycast request changes as k changes, and how replication rate must adjust. While some of these components have been briefly studied in literature, none has been studied to the extent required for wide-spread use.

1.1.1 Understanding the Environment: Mobility in Structured DTNs

As a prerequisite to developing communication protocols, it is necessary to fully understand the environment in which these protocols operate. Therefore, developing high-level mobility models for structured DTNs that are generic enough to explore a large mobility space is critical to both better understanding as well as having better tools to evaluate proposed routing solutions. First, understanding how movement occurs in DTN environments, such as disaster recovery networks, helps guide routing decisions such as how often replication should occur, when replication should occur, and to whom replication should occur. This guidance comes from graph-theoretic properties of these networks, such as how clustering coefficient and average node density vary throughout the different locations in the network. Second, high-level, generic mobility models act as an invaluable tool for evaluating proposed routing protocols in their target environments. Since real-life experimentation is often time-consuming, costly, and difficult to reproduce, realistic DTN mobility models provide an excellent simulation-based environment in which to evaluate communication protocols.

To respond to this challenge, we have developed a high-level mobility model for disaster recovery networks, a popular scenario exhibiting DTN behavior. Taking a highly parametrized approach, the model is capable of simulating a wide range of movement, where nodes assume roles, such as civilian, police officer, and ambulance, and, depending on the role, react differently to different global events in the network. Different roles cluster and move in an intuitive fashion, giving insight into how communication protocols should best route information. This model, along with other mobility models for different DTN scenarios, acts as a tool for us to explore communication protocols and analyze their effectiveness.

1.1.2 Local and Robust Group Management

Although many current applications rely heavily on unicast, an equally if not more important paradigm is group-based communication. Groups are naturally found in many environments, and are especially common in human-centric DTNs where much of the communication and mobility is based on human interaction [13, 23, 49]. In these environments, the composition of groups may be based on many different abstractions, including roles [35], social networks [13], or geographical closeness. Unfortunately, due to the intermittently connected nature of DTNs, maintaining and disseminating such group information is chal-

lenging, especially in the presence of malicious attackers. The successful enhancement of DTN communication through the use of group information requires that nodes throughout the network be aware of the membership lists for all groups. While such group membership management is not difficult in connected environments, heavy partitioning and the lack of readily available end-to-end paths in DTNs break centralized membership services, just like they break traditional routing [26].

In response to this challenge, we developed a robust group information management protocol called MembersOnly. MembersOnly allows groups to accurately distribute membership lists to nodes throughout the network without the use of cryptography, and is robust to multiple malicious nodes attempting to alter these lists. We show that it can protect against many types of membership list altering attacks. In addition, we show that even the most basic anycast routing protocols can gain a significant advantage in hostile environments by using MembersOnly.

1.1.3 Unicast Communication

The staple of communication in current networks is unicast, where one node attempts to communicate with another specific node. It is therefore of the utmost importance that efficient unicast protocols be developed for human-centric DTNs before wide-spread adoption can occur. Many current DTN routing protocols are resource-heavy since they attempt to overcome the intermittent connectivity with large amounts of message replication, in hopes that at least one message copy will reach the destination [4, 10, 32, 55]. Unfortunately, in many human-centric DTNs, resources such as battery life, contact duration, bandwidth, and even storage space are limited and these flooding-based protocols quickly overwhelm them, having a severe negative impact on performance metrics. The few protocols that attempt to be resource friendly generally suffer in either message delivery ratio or end-to-end latency due to missing important contact opportunities [51]. It is necessary for unicast protocols to be both effective in terms of performance metrics, while being efficient in terms of resources.

To meet this challenge, we developed a highly efficient and effective unicast routing protocol for DTNs that excels in resource-constrained environments, called Encounter-Based Routing, or EBR. By taking advantage of inherent network structure, it is able to achieve a significant performance advantage over existing protocols, particularly in resource-constrained environments.

1.1.4 Anycast Communication

As previously mentioned, group-based communication is an important paradigm for human-centric DTNs, and enabling group-based communication is critical towards progressing their wide-spread use. In these

environments, group-based communication is central and hence a natural and useful routing paradigm is *anycast*, where a node attempts to communicate with at least one member of a particular group. Anycast is useful in DTNs both as a stand alone paradigm [20], when contacting any member of a particular group is sufficient, as well as enabling smarter unicast paradigms [25]. In connected environments, basic anycast routing techniques are relatively straightforward, as messages can be unicast to a particular node in the group that has the lowest cost. This technique, however, does not work in disconnected environments, since it is extremely difficult to predict which of the nodes of the group would even get the message, let alone get it the quickest. The disconnected and unpredictable environment indicates that anycast must instead be smarter and attempt to truly reach any node in the group. Unfortunately, there are very few anycast solutions for disconnected environments, and of these almost all are based on single-copy routing or work only in highly constrained mobility patterns.

To address this pressing need, we have developed a protocol-independent technique to enhance many existing unicast protocols and allow them to perform anycast. This enhancement can be done in a thin shim inserted beneath the routing layer, and hence unicast protocols can run unmodified above it. Through an exhaustive set of simulations, we evaluate how different parameters and network conditions affect the performance of these newly transformed anycast protocols.

1.1.5 Manycast Communication

In order to capture the full power of group-based communication, we must introduce and explore the most general paradigm - manycast. The goal of manycast is to reach at least k members of a group of size m. This paradigm includes anycast (when $k = 1$) as well as multicast (when $k = m$). Fundamentally understanding the difficulty of a manycast request, as well as developing a routing framework to handle it, is beneficial for a few reasons. First, it deepens our theoretical knowledge of the relative difficulty of anycast, unicast, and multicast, which can help guide replication decisions. Second, it allows applications a high degree of flexibility to express their wishes. Group-based applications where anycast, unicast, and multicast are not sufficient and/or efficient may include reaching a statistically significant sample of sensor networks, requesting at least k emergency response vehicles in a disaster zone, and contacting k local friends for social multiplayer gaming.

In response to this need, we have conducted a thorough exploration of manycast in DTN environments. To conduct this exploration, we took a three-pronged approach. First, the relative difficulty of manycast requests was quantified via analysis. Second, extensive simulations were performed to understand how existing classes of protocols handle varying manycast requests. This has shown that effective

DTN manycast protocols must dynamically react on a per-message basis by drastically changing their routing approach. Finally, we developed a manycast meta-protocol capable of dynamically switching between low-level protocols based on the current request and network conditions.

1.1.6 Thesis Structure

In Chapter 2, we describe our role-based, event-driven mobility model for disaster recovery networks. Chapter 3 presents our group-management protocol that helps open the door for group-based communication. Chapter 4 presents our highly efficient unicast routing protocol. In Chapter 5, we describe an overarching technique to enhance current unicast protocols, giving them the ability to perform anycast. Chapter 6 describes our exploration of manycast and presents a dynamic manycast meta-protocol. Finally, we conclude and present future directions in Chapter 7.

Chapter 2

Event–driven, Role–based Mobility

One of the most important tools in understanding the complex characteristics of DTNs is simulation. While many mobility models exist for simulating ad-hoc networks, they do not realistically capture the behavior of objects in disaster scenarios. In this chapter, we propose a high level event- & role-based mobility paradigm in which objects' movement patterns are caused by environmental events. The introduction of roles allows different objects to uniquely and realistically react to events. For instance some roles, such as civilian, may flee from events, whereas other roles, such as police, may be attracted to events. Furthermore, to incorporate reaction from multiple events in a realistic fashion, we propose a low-level gravity-based mobility model in which events apply forces to objects. Simulation results show that our disaster mobility paradigm coupled with our gravitational mobility model creates a network topology that differs from the popular Random Walk mobility model. This new disaster mobility model opens up the door for more realistic simulation of communication and routing protocols for disaster recovery networks.

2.1 Disaster Recovery Networks

Much of the recent interest in delay tolerant networks has come from the need to support communication for organizations like police and fire departments, as well as other first responders. The behavior of most of these organizations is driven by the need to respond to events and participate in those events based on the particular role of the organization. Since this type of behavior is very specific to emergency and disaster response scenarios, understanding communication patterns in such networks is critical to understanding how to improve the current state of emergency response. However, current mobility models for wireless networks do not capture the complexity of either the behavior of the different components of such networks or the specifics of the expected communication patterns. Realistic mobility and communication models can enable more effective evaluation via simulation and eventually lead to more effective solutions.

One of the biggest challenges faced by communication in disaster recovery networks comes from the high expectation of network partitions and dynamic object behavior. The high potential for object fail-

ure further complicates matters. Intuition says that, as objects react differently to different events, the variance in density of the communication graph will change and cause strain on current routing protocols. Due to this unique behavior, disaster recovery networks present challenging environments.

One of the fundamental aspects of simulating disaster recovery networks is realistically modeling the movement patterns of the mobile objects. Modeling mobility enables testing the effectiveness of current routing protocols as well as provides insight into how routing protocols can be improved. In a disaster environment, it presents unique challenges in that environmental events and roles directly affect a node's movement patterns. Intuitively, events act as stimuli for mobile nodes in the network, causing them to react in ways according to the predefined roles they take on. Many roles in disaster networks must react to multiple events by fleeing or approaching in a realistic fashion.

Many ad-hoc network mobility models have been developed and analyzed [11, 5, 15, 27, 31, 43]. Models based on random movement are particularly popular and heavily studied [6, 7, 47, 21]. These models, while adequate to study environments for which they were designed, do not allow for groups of objects to react differently to environmental events. Therefore, a higher-level, more general mobility model is needed to incorporate the different roles objects play in disaster scenarios.

In this chapter, we present an event- & role-based mobility paradigm that effectively characterizes the movement patterns of objects in a disaster scenario. Different sets of these patterns are embedded into different object roles. Attaching actions to roles and not directly to objects has the advantage that movement patterns are organized and objects can quickly shift from one pattern to another by assuming different roles. To the best of our knowledge, this is the first disaster mobility paradigm that is reactive, in a role-based fashion, to environmental events and their associated parameters.

The main contribution of our research in this chapter is the classification of a generic event- & role-based mobility paradigm that completely defines movement patterns given a series of environmental events for a set of characteristic roles. Additionally, we present a low-level physics-based gravitational mobility model that "plugs in" to our event-driven, role-based paradigm allowing objects to react to the presence of numerous disaster events based on the particular role of the node. This allows objects to flee from or approach multiple, unrelated events. To evaluate the effect of our comprehensive model on communication patterns in disaster recovery networks, we discuss a new set of relevant metrics that help characterize the changes in topology as disaster events unfold. Finally, we have developed a set of tools to realistically construct a mobility scenario and trace files of our disaster mobility model for the ns2 network simulator [1].

2.2 Modeling Object Behavior

Modeling the movement behavior of mobile objects has been heavily studied. Due to its simplicity and effectiveness, the Random Walk [21] model is widely used to model such object behavior. In this model, a node randomly chooses a direction in $[0, 2\pi)$ along with a speed, and moves according to those choices for a set amount of time. After this time has expired, the node repeats the process. A recent model by Jardosh *et al.* [27] takes polygon-shaped objects into account by using Voronoi diagrams to build walks. While many of these models are adequate for their particular environments, all objects generally act in the same way and, therefore, do not give the flexibility required for modeling disaster scenarios. This is because real people and vehicles take on roles, allowing them to react to events in a distinct fashion.

Mobility in disaster recovery scenarios is fundamentally driven by environmental events. These events act as stimuli towards objects and directly cause them to change their movement patterns. While some current models, including [27], could be extended to react to external stimuli, truly capturing the complex interactions between more than one environmental event requires building new mobility models with event-driven actions as the primitive concept. Furthermore, while all objects react to relevant events, different classes of objects react in different ways. In other words, object behavior changes over time and is not uniform across all nodes. Objects also must react to multiple events in a realistic and smooth fashion. There is currently no adequate mobility model that takes into account these observations.

To illustrate this idea with an example, consider an apartment fire in a populated neighborhood. There are, intuitively, at least three different classes of behavior that objects can assume: (1) fleeing from the event, as is the case with civilians, (2) approaching the event with the intent of staying, as is the case with police and firefighters, and (3) oscillating from event to a predetermined location, as is the case with ambulances. These high-level behavior classes, which we refer to as roles, help give general but clear mobility patterns, which are realistic and relatively easy to simulate.

Roles, however, need not be limited to specifying movement patterns during a disaster scenario. It is easy to extend the concept of a role to cause an object to react differently during different stages of an event. For example, cleanup crews may want to react to disaster events by approaching them long after the event has occurred, whereas police may instead want to approach the event immediately.

In response to the need for unique behavior modeling in disaster scenarios, we have developed a high-level, role-based, event-driven mobility paradigm in which different roles take on different mobility patterns in reaction to specific events. Our paradigm is high level in the sense that an object may take on multiple mobility patterns over the course of some time period. For instance, a civilian may first be modeled by Random Walk. After some time, an event, such as a fire, may trigger the civilian to change its

model to one of fleeing from the fire. By modeling mobility as a series of event-driven, role-based actions, we can properly select which specific mobility model to use for a given object and situation. While the specific rules for reaction should be based on observational studies, our paradigm is sufficiently general enough to allow future, accurate movement patterns to be used.

Our paradigm allows for different objects to react to events in unique ways by attaching a mobility pattern for each possible (event, role) combination. These lower level mobility patterns can be any of the previously developed mobility models, including Random Walk. To further illustrate the behavior of certain roles in a disaster scenario, we have developed a low level physics-based gravitational model that allows objects to flee or approach disaster events in an intuitively realistic fashion. It is important to note that our high level, event- & role-based paradigm is not tied to the gravitational model in any way, and can support any of the previously defined low level mobility models.

Our gravitational model captures the effect-distance relationship between events and objects, allowing events to act on objects via forces. Objects closer to events are affected more than objects farther away. Additionally, each event has an *event horizon*, defined as the maximum distance at which an event affects objects, which allows events to have a defined radius of effect. Finally, there is a *communication threshold*, which defines the time until emergency vehicles are notified of an event. Until this time, emergency vehicles outside the event horizon do not respond to the event; however, after the threshold time has passed, those emergency vehicles begin to converge on the event. It is possible that multiple events could take place in a single scenario. Additionally, these events may or may not be simultaneous. One of the major benefits of the gravitational model is that it easily captures the interactions between multiple events.

The classification of behavior into roles can also play a part in establishing realistic communication patterns. For instance, civilians are most likely in contact only with police and other civilians, police are in contact with all roles, firefighters are in contact with police and other firefighters, and ambulances are in contact with police. These communication patterns, along with the mobility, can simplistically model an entire disaster scenario. For this chapter, however, we concentrate solely on modeling mobility,

2.3 Disaster Mobility Paradigm

In this section, we formally describe our high-level mobility paradigm that incorporates external, environmental events along with role-based reaction. By partitioning objects into roles, which define their reactions to events, our paradigm can realistically obtain a set of (role, event, action) triples that define the overall movement patterns of objects in disaster recovery scenarios. A single triple can be read as follows:

"Role r reacts to event e by taking action a". Then, by instantiating the triples with the characteristics for different agents operating in a disaster scenario, a mobility pattern can be generated for the scene.

Three entities, and their relationships to one another, help define our high level mobility paradigm: objects, roles, and events. *Objects* are nodes in the system that provide movement and communication. Each object assumes a *role*, or set of roles, that indicate what movement pattern the object should assume in response to external stimuli. The specific areas of interest that provide external stimuli to objects are referred to as *events*. The event-based response a role dictates to an object is an *action*, which is generally a low-level mobility model, such as Random Walk or the gravitational model presented in Section 2.4.1.

We now elaborate on these three entities and their relationships to one another.

Objects are the critical components of the scenario, including people, buildings, and vehicles, and are defined by the following parameters:

- Location: The (x,y,z) location of the object.
- Role: The role, or set of roles, associated with the object.
- Velocity: The current velocity of the object (in vector form).

Roles dictate how objects react to events. We define four main categories of roles, although it is possible to define any number of them. First, the *repelling* category causes objects to be repelled from events. The low-level mobility models that support this role should allow objects to move away from events in a realistic and easy-to-use fashion. The most common use of this role is to model normal civilians in a disaster scenario. An attribute of this role is *curiosity*, which dictates how likely it is for an object to stop at the event horizon, simulating curious on-lookers. Second, the *attraction* role causes objects to converge on events. Low-level mobility models that support this role should cause objects to quickly approach an event or events. Common uses of this role are to model police and firefighters. Third, the *oscillating* role models objects that first approach an event and then, upon reaching the event, travel immediately to a predefined location. This movement pattern is then repeated continuously. Low-level mobility models that support this role should allow this action to be as realistic as possible. One use of this role is to model an emergency response system in which ambulances oscillate between the event and a hospital. Finally, the *immobile* role models any object that remains stationary for the duration of the simulation or until the object takes on a new role. This role can be supported by the lack of a low-level mobility model, since it does not perform any movement. This role is useful to model both naturally static objects, such as hospitals, as well as event-caused immobility.

We anticipate that the default action dictated by many roles is Random Walk, since it simply models motion when movement patterns are unknown or seem random. It is important to note, however, that any mobility pattern, such as one that accounts for navigating around buildings or objects, can be used.

Events act as the stimuli for mobility changes in the scenario. In a real disaster scenario, an object's proximity to an event is a major factor in how it reacts. Therefore, it is important to clearly mark distinct areas of an event. Our paradigm captures this type of behavior by defining the *Disaster Radius*, the *Event Horizon*, and the *Relevant Radius*. Reaction to an event is also dependent on time, which is modeled by the *Start Time* and the *Radio Contact Time*. The following defines the full set of parameters for an event.

- Location: The (x,y,z) location of the event.
- Start Time: Time when the event occurs.
- End Time: Time when the event ends.
- Radio Contact Time: Time when radio contact outside of the event horizon occurs.
- Disaster Radius: Area inside which all objects become immobile.
- Event Horizon: Area inside which all objects react to the event, even before radio contact occurs.
- Relevant Radius: Area inside which objects, based on their role, react to the event after radio contact occurs, assuming they are not already reacting. Some roles, such as the civilian role, may not react when inside this radius but outside the event horizon.
- Intensity: A numeric representation of the event's current intensity.

2.4 A Disaster Mobility Model

With the high-level disaster mobility paradigm formalized, we now describe a disaster mobility model that we believe intuitively models simple disaster scenarios. For implementation purposes, we have made some simplifying assumptions. First, we assume events are stationary, have a constant intensity, and, after their start, persist for the remainder of the simulation. Furthermore, objects assume a single initial role and do not change it for the duration of the simulation, unless changing to the immobile role. Finally, all roles except the Immobile role default to the Random Walk action. At the end of this section, we discuss how to capture events that are mobile and change intensity, as well as events that are not best represented by a single point.

2.4.1 Gravitational Model

Intuitively, many roles react to disaster events by either fleeing from or approaching them. To model these actions in the presence of multiple events, we use a physics-based gravitational model to define the "Flee" and "Approach" actions. Gravitation has been used to model group mobility dynamics, particularly in [48], but not event-based mobility. We designed this model based on the observation that objects, in general, either gravitate towards or away from disaster events.

Physics states that the gravitational force between two objects with masses m_1 and m_2 at a distance d from each other is:

$$F = \frac{G \cdot m_1 \cdot m_2}{d^2}$$

where G is the gravitational constant. The total resulting vector force on an object in the vicinity of multiple objects is calculated by the vector sum of all forces on the object. The resulting force directly affects the object's acceleration.

We borrow the concepts of gravity and force from physics to model the *flee* and *approach* actions. By letting m_2 be the "mass" of a given event, and assuming m_1 is negligible, the force on an object by that event can be described as $F = I/d^2$, where I, the intensity, encompasses G and m_2. Mobile objects can then be repelled or attracted to events (or multiple events) by assigning a particular intensity to every event.

To calculate the motion trajectories of objects as a result of multiple forces, we sway from physics slightly to allow for a more intuitively realistic movement pattern. Physics states that forces directly affect an object's acceleration. However, humans are more concerned with maintaining a particular velocity at a given time than maintaining a particular acceleration. We generally do not maintain a constant acceleration, but rather accelerate quickly to a desired velocity and then hold an acceleration of zero. Therefore, it is intuitively more correct to say that humans will adjust their *speed*, not their *acceleration*, according to how far they are from a disaster event (of course, they will adjust their acceleration to obtain that speed, but only for a short period of time). Therefore, in our gravitational model, forces act directly on velocity, not acceleration, to account for this phenomenon. The benefit of taking a gravitational-based approach to model mobility is that it allows the reaction of objects to be intuitive and easy to compute for multiple, unsynchonized, dynamic events.

2.4.2 Disaster Model

Let M be the set of (role, event, action) triples that define our mobility model. M is populated by adding triples to cover all components or the desired scenario. All mobile nodes initially start using the Random Walk Model to either walk or drive, with speeds appropriately bounded. Formally, there are a set of initial (role, event, action) triples for each role as follows:

$\{$*(r, No Event, Random Walk)* : $r \in R\}$,

where R is the set of all possible roles. The "No Event" event is simply the default event every role assumes when there are no relevant events in the network.

If a disaster event occurs, two areas are immediately formed. The first area is ground-zero, as defined by the disaster area parameter, within which objects are immediately immobilized. We model this by simply immobilizing all nodes within a set radius of the disaster event. We formally model this by the inclusion of the following triples into M:

$\{$*(r, DE1 - At Ground-Zero, Switch to Immobile)* : $r \in R\}$

The "DE1 - At Ground-Zero" event is a disaster event (with the label "DE1" representing disaster event #1) that has occurred when the object was within the disaster radius of the event. Once an object is immobile, it stays immobile for the remainder of the simulation. To accomplish this, the action "Switch to Immobile" instructs the object to immediately switch roles to the "Immobile" role. This role is defined in M as follows:

(Immobile, No Event, Stay Still)

It is important to note that this should be the only entry for the immobile role, since it should always default to staying still. Static objects, such as a hospital, are also assigned the immobile role.

The second area is defined by the *event horizon*. All objects within the event horizon react to the event by either gravitating towards it or fleeing from it, at a speed dependent on the object's proximity to the event. The inclusion of a set of triples into M formally models this phenomenon. For instance, the following triples define the area within the event horizon for a simple disaster scenario:

(Civilian, DE1 - In Event Horizon, Flee)

(Police, DE1 - In Event Horizon, Approach)

(Firefighter, DE1 - In Event Horizon, Approach)

(Ambulance, DE1 - In Event Horizon, Oscillate)

The event "DE1 - In Event Horizon" event refers to the situation that the object is within the event horizon radius of a disaster event. Notice that while all the events beginning with "DE1" are technically the same event, to incorporate proximity into the action taken by a role, we break the event into multiple areas (or regions), in which roles reacting to the same event may respond differently based on which area of the event they are in.

After the radio contact time of the event expires, the relevant radius of the event is formed. Roles with radio contact in this region, but outside the event horizon, should react to the event. Continuing from the previous example, the following tuples in M formally define this area:

(Police, DE1 - In Relevant Radius, Approach)

(Firefighter, DE1 - In Relevant Radius, Approach)

(Ambulance, DE1 - In Relevant Radius, Oscillate)

An object may have multiple applicable triples at any given instance. This would occur, for example, if a civilian were in the radii of two different events. Our gravitational model easily accommodates scenarios of this type.

It is possible to extend our model to account for events of different shapes and sizes, as well as mobile events. Currently, events provide forces from a central point within the event, and have different radii that allow for a circular (or spherical) shape. Elongated event shapes, such as floods, can be simulated by placing multiple events close to each other at varying intensities. Furthermore, there is nothing prohibiting the changing of intensity or location of an event, as forces can be quickly recomputed at every object's location based only on current information. The natural memoryless computation of forces allows for highly dynamic events.

2.5 Analyzing Mobility Models

The benefits of an effective mobility model come from its ability to capture and expose the characteristics of the network and the behavior of nodes in the network. This information can then be used by network designers to understand how to design protocols that are suitable for the specific scenarios. In this section, we first discuss the metrics necessary to describe network behavior in disaster scenarios. Although most evaluations focus on simple network characteristics, such as node density and path length, the unique behavior of nodes in a disaster scenario results in more interesting network conditions that require us to look at more complex parameters, such as average node density and partitioning. We then present a set

of tools that we implemented to generate ns2 mobility scenario files. In the next section, we present our evaluation of a number of these metrics using our tools.

2.5.1 Metrics

When discussing mobility models, it is important to understand how a model affects different topological network metrics. Two standard metrics are average node density and average path length. Average node density, as defined by the average number of neighbors per node, can be used to help characterize the potential connectivity of a network, since a network with low density will likely be partitioned. Average path length, as defined by the average number of hops from source to destination, captures the distance between sources and destinations. However, due to the highly dynamic nature of networks under disaster scenarios, it is important to not only consider these metrics but also those that show how the structure of the network progresses over time. For an event-driven, role-based mobility model, the following network parameters highlight how the network is changing over time.

- Partitioning over Time: The average path length metric is meaningless when the graph is partitioned, which is likely in many disaster scenarios. Therefore, tracking whether or not the graph is partitioned is critical to understanding the flux in network topology.
- Clustering Coefficient over Time: The clustering coefficient of a particular node is the number of that node's neighbors that are connected to each other divided by the total possible links between them [58]. The higher the value of this metric, the more clustered the graph is.
- Average Node Density over Time: Although node density is an important metric, in an event-driven mobility model, it is important to capture how density changes in *reaction* to different events, which indicates churn.
- Maximum Node Density over Time: Maximum node density gives insight into the potential cluster sizes, which can provide insight into potential bottlenecks. Although it would be useful to track actual cluster sizes, maximum node density provides a much cheaper, though quite effective, heuristic.
- Variance of Node Density over Time: Since some parts of the network may be more stable than others, the variance in node density gives insight into the amount of variance in cluster sizes as a result of different events.

2.5.2 Tools

To evaluate the impact of our model on the metrics described above, we have implemented two tools for the ns2 simulator. The first tool is a parameters file generator that creates a properly formatted parameters file appropriately choosing random values when necessary. This tool prompts the user for the following input: size of the network (in terms of meters squared), number of civilians, number of ambulances, and number of police. Since both the police and firefighter roles are similar, we have chosen to omit firefighters and simply add more police to simulate firefighters. However, it is quite simple to include firefighters, or other responders, and give them appropriate behaviors. The output parameters file contains the following information:

- Grid size and simulation runtime
- Randomized coordinates for all objects and events, and coordinates for four hospitals located at the corners of the grid
- Minimum and maximum speeds for objects
- Percentage of curious civilians
- Random Walk parameters
- Randomized trigger times and radio contact times for four events
- Randomized intensities for events, which determine radii for event horizons and damage zones

The specific parameters generated for the experiments in this chapter are detailed in Section 2.6.1. For any given input, this tool produces unique output since many parameters are randomly chosen. Usage for this tool is as follows:

Usage: paramGen > paramFile.

The second tool is our main event-driven simulator. This tool accepts as input the parameters file generated by the first tool and runs a complete simulation with knowledge of Random Walk and our physics-based gravitational model. It is important to note, however, that any mobility model can be plugged into the tool in place of Random Walk and/or the gravitational model. The output of this tool is an ns2-compatible mobility trace file that gives the current velocity and destination of every object at every second in the simulation. All of the mobility model logic is performed in this tool. For any given input, the output of this tool again produces unique output, since Random Walks performed by objects not reacting to events may differ from one simulation to the next. Usage for this tool is as follows:

Usage: disasterSim [-d] < paramFile > nsMobilityTrace.

The *-d* flag, when passed, displays each individual step of the simulation via "ASCII art" to the console. This is printed to standard error, so it is not written to the *nsMobilityTrace* file.

2.6 Simulation Results & Analysis

To analyze the difference in network topology changes generated by our disaster recovery mobility model, we generated numerous topologies and ran simulations with them using ns2. Using the same initial setups, in terms of node placement and numbers, we ran the simulations using the Random Walk model for comparison. In this section, we present results from 10 different sets of simulations, each with its own group of both deterministically and randomly chosen parameters. Each of the metrics presented in Section 2.5.1 are evaluated for each of the resulting sets of trace files. Sufficient numbers of experiments were run to minimize the 95% confidence interval.

For this chapter, we were only interested in the mobility patterns of objects in a disaster scenario and the topological affects the patterns have on the network graph. Therefore, we did not simulate communication between nodes. However, we did specify the communication range to be 150 meters to obtain information about links in the network.

2.6.1 Simulation Parameters

The simulation time runs from 0 to 1500 seconds with a grid size of 1000 m^2. The communication range of every node is 150 m to simulate an urban environment. There are 75 civilians, 10 ambulances, and 15 police each randomly located. A total of 90% of civilians, randomly chosen, are considered curious, meaning they stop at event horizons to look at the event. Furthermore, we have chosen a minimum and maximum speed of 1 m/s and 4 m/s respectively for civilians, and 17 m/s and 20 m/s for ambulances and police. All Random Walks are done for 30 seconds.

Four events are randomly placed on the grid. The first event occurs randomly between 100 and 200 seconds, the second between 125 and 225 seconds, the third between 150 and 250 seconds, and the fourth between 175 and 275 seconds. Radio contact for a specific event occurs randomly between 40 to 80 seconds after the event has occurred. The intensity of events are randomly chosen between 10000 m^3/s and 20000 m^3/s. The event horizon for each event is 2% of the intensity and the damage radius is 0.1% of the intensity.

We choose high intensity events to easily illuminate the differences between the topology of our model

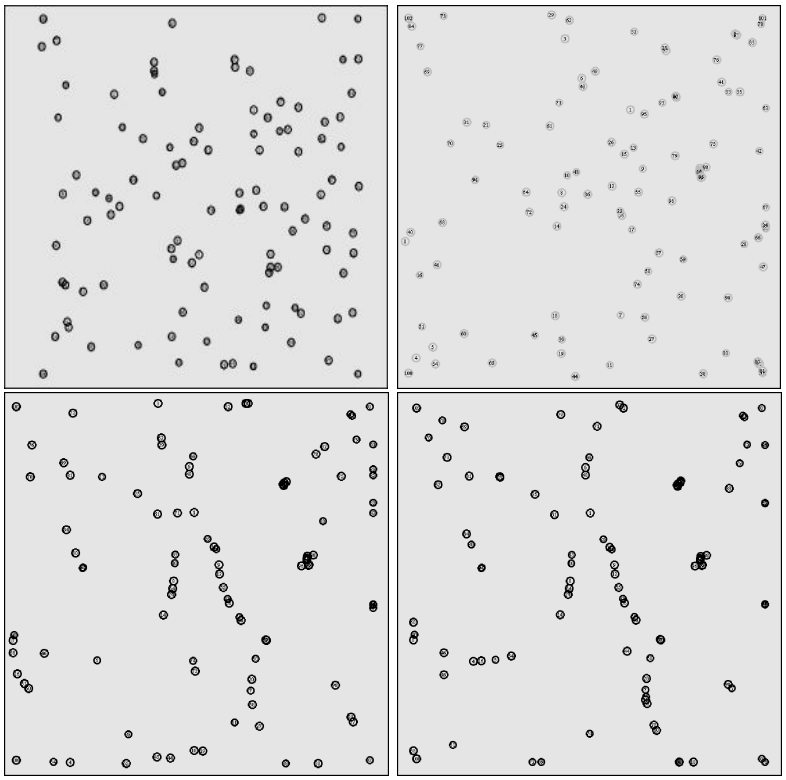

Figure 2.1: Network snapshots at time 0, 200, 400, and 1400 seconds

versus the topology of the Random Walk model. We also choose to include hospitals as stationary objects, since they will most likely participate in communication with other objects (particularly ambulances and police). Therefore, there are a total of 104 objects in the system, 4 being immobile from the beginning. Furthermore, we assumed that radio contact for all events reached ambulance and police regardless of where they were. All responders are in either CB or cellular radio contact at all times, meaning that the relevant radius for each event is set large enough to encompass the entire grid.

2.6.2 Snapshot of Topology Change

Figure 2.1 shows a series of four snapshots during one simulation run of our disaster mobility model. The first box shows the state of the network at the start of the simulation. Object location at this point is random, except for the 4 hospitals located at each corner of the grid. The second box shows the state of the network 200 seconds into the simulation. At this time, some events have triggered but radio contact for many has not occurred. Only objects within the event horizon have reacted at this time. The third box shows the state of the network 400 seconds into the simulation. By this time, all events have been triggered and radio-contact has been made. All emergency response objects (police and ambulances) and

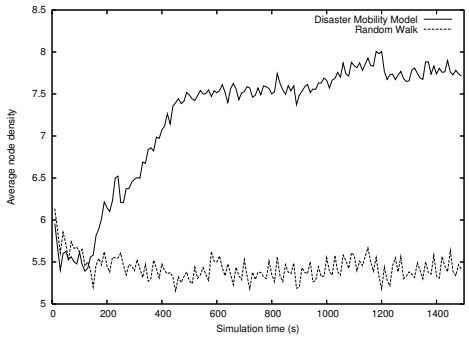

Figure 2.2: Average node density

civilians within the event horizon have reacted to the event. It is now possible to see some of the different roles active in the system, simply by visually observing their locations. The fourth box shows the state of the network 1400 seconds into the simulation. By this time, most metrics have come close to convergence and mobility is noticeable only by ambulances and civilians who have not approached the event horizon. Civilians who are not curious and have left the event horizon are mobile again.

A clear topological difference between the disaster mobility model and Random Walk is primarily due to the clustering of objects around the event horizon. This separates the graph into three primary areas: (1) areas inside event horizons, (2) areas at or near event horizons, (3) areas outside of event horizons. The first area is very sparse since all civilians able to move leave the scene. Almost all of the concentration is in the "damage radius", since emergency response workers immediately move towards that zone. The only interaction between that zone and the event horizon are the oscillators when they pass through the event horizon. The second area is very dense since all of the civilians inside the event horizon gravitate towards its edge and civilians that happen to stumble into the event horizon stay there. The third area contains objects who are performing random walks and have not been notified of the event. As the simulation continuously runs, the third area should slowly lose objects to the second area since they randomly hit the event horizon.

This series of snapshots clearly shows the location of events and the formation of crowds of people around the event horizon. It also illustrates the behavior of ambulances going to and from events and hospitals. We would expect very similar results in a real disaster scenario, further confirming our implementation.

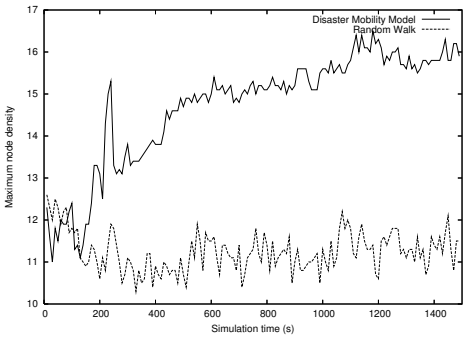

Figure 2.3: Maximum node density

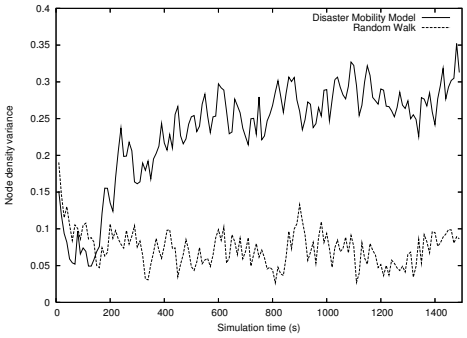

Figure 2.4: Variance of node density

2.6.3 Metric Evaluation

Figures 2.2 through 2.6 clearly show the event-driven response of the metrics around the time of the events. Between 0 and 100 seconds, the data sets are similar for all metrics, as expected. Between 100 and 355 seconds, during the triggering of events and radio contact time, a clear divergence between the disaster mobility model and Random Walk model is readily seen as the topology of the event-driven simulation starts to take form.

Figure 2.2 shows the average node density of the network as time progresses. The average node density in the disaster mobility model increases in response to events. This is due to the gathering of nodes around the event horizon, forcing them into a smaller area than before. An interesting observation is that the size and frequency of the oscillations in average node density become both smaller and less frequent in the disaster mobility model as time progresses. This is due to the topological convergence of the disaster mobility model that does not occur in Random Walk. The difference in average node density between the disaster mobility model and Random Walk is important because it gives overall information as to how many neighbors a node can expect to have at a given time and so provides hints about network connectivity.

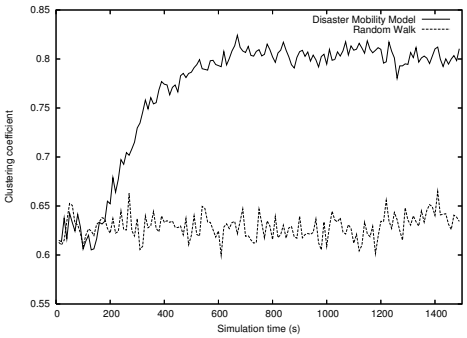

Figure 2.5: Clustering coefficient

Figure 2.3 shows the maximum node density between the two data sets as time progresses. The maximum node density quickly increases in response to the events. There are two highly-clustered areas for each event in the system, the area inside of the damage radius and the event horizon. The jump in maximum node density is due to the quick response by the police to the event, increasing the node density of nodes at the damage radius. After this, the maximum node density remains relatively constant from around 500 to 900 seconds, as the maximum density around the event horizon starts to catch up to the maximum density around the damage radius. At around 900 seconds, the maximum density starts to increase as the maximum density of the event horizon increases. The diminishing of high-frequency oscillations as time progresses is, again, due to the convergence of the disaster mobility model not found in Random Walk.

Figure 2.4 shows the variance of node density as the simulation progresses in time. The variance of node density clearly increases as events are triggered. This is because many nodes have either a fairly small node density (if they are being partitioned or close to being partitioned in the graph), or a high density (if they are clustered at the event horizon or damage radius). It is interesting to note that the high-frequency oscillations do not seem to diminish as time progresses. This is likely due to both the Random Walk civilians and the ambulances oscillating between hospitals and events.

Figure 2.5 shows the clustering coefficient in the network as the simulation progresses in time. The clustering coefficient of a node i is defined as:

$$C_i = \frac{2\,|\{e_{jk}\}|}{k_i(k_i-1)} : v_j, v_k \in N_i, e_{jk} \in E,$$

where N_i are the neighbors of i, E is the set of edges in the graph, and k_i is the degree of node i [58] (this definition assumes an undirected graph). This gives a general indication of how well a node's neighbors know each other, which in turn gives insight into how clustered the network is. We define the clustering

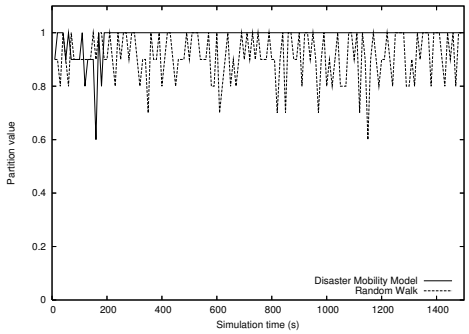

Figure 2.6: Network partitioning

coefficient of a node with degree less than 2 to be 0. The average graph clustering coefficient increases sharply in response to the events. This is again due to clustering around the event horizon and damage radius for each event. As before, the diminishing of high-frequency oscillations is apparent.

At any given time, a graph is either partitioned or not. If it is, we say it has a "partition value" of 1. If not, it has a "partition value" of 0. For each data point, we have averaged the partition value of each of the 10 simulations. Figure 2.6 shows that the average partition value in the disaster mobility model increases as a result of the events. In fact, due to the high intensity of the events, after around 200 seconds the network is always partitioned in the disaster mobility model. This is because the nodes at the events are partitioned from the rest of the network, since the event horizons are generally out of their communications range. The disaster mobility model consistently has a partition value higher or equal to that of Random Walk, indicating a more fragile network.

These results show that our mobility model produces a topology much different than that of the popular Random Walk model. The vast difference between the topologies indicate that it is not sufficient to use Random Walk as a mobility model for disaster recovery networks.

2.7 Conclusions and Future Directions

This chapter presents a generic event- & role-based mobility paradigm used to characterize movement patterns of objects in response to environmental events. We specifically concentrate on applying this to disaster recovery scenarios, where current mobility models fail to realistically represent objects. To accurately characterize the movement of objects in response to one or more disaster events, we have developed a gravity-based model in which events emit forces that attract or repel objects depending on the object's role. Using simplified laws of physics, it is straightforward to calculate the velocity vector of an object, even in the presence of multiple events.

Our disaster mobility model has been fully implemented in simulation form and was used to generate ns2 mobility trace files. The resulting topology of our disaster mobility model had a higher average node density, maximum node density, variance of node density, and clustering coefficient. This is due to the grouping of nodes at or near the event horizons and near the damage radius of events. Furthermore, the partitioned value was consistently higher with our disaster mobility model, indicating the network was partitioned more often.

Our event- & role-based disaster mobility paradigm realistically captures objects' responses to disaster events. Furthermore, our simulation results show that the topological characteristics of the network drastically differ from that of Random Walk. As future work, one could perform studies on actual disaster scenarios to develop a rich set of role-based rules and further refine our low-level gravitational model. Furthermore, it would be interesting to use our disaster mobility model to understand the effects it has on routing protocols such as AODV [44] and DSR [28], as well as explore security concerns with a role-based system. This will most likely lead to the development of new DTN-style disaster routing protocols specifically tuned for disaster recovery networks.

Chapter 3

Robust Group Management with MembersOnly

Effectively utilizing groups in DTNs can both improve the throughput of unicast routing protocols and open the door for a wide range of group-based paradigms, such as anycast and manycast. Unfortunately, in DTN environments, there is no centralized entity that can quickly and reliably transmit group membership lists, and hence group information must be disseminated through unreliable and potentially malicious nodes. In this chapter, we propose a local and robust group information dissemination and consolidation protocol, called *MembersOnly*, that both quickly and accurately transmits group membership information to all nodes in the network, even if multiple malicious nodes attempt to disrupt the process. We show via analysis and simulations that MembersOnly is able to withstand multiple types of attacks, with only very limited periods of vulnerability that disappear relatively quickly. This is in contrast to current techniques that cannot withstand many of these attacks, resulting in quick and thorough corruption of group membership lists. In addition, we show via simulation that even the most basic routing protocols can gain a performance advantage when using MembersOnly.

3.1 Group Management in DTNs

Groups are naturally found in many environments. Such grouping is especially common in delay and disruption tolerant networks, where much of the communication and mobility is based on human interaction [13, 49]. In these environments, the composition of groups may be based on many different abstractions, including roles (i.e., firefighters or police officers [35]), social networks (i.e., friends communicating using wireless mobile devices [13]), or geographical closeness (i.e., coworkers who meet every day for meetings). Knowledge of groups can greatly enhance many aspects of communication in DTNs, including unicast [25], anycast and multicast routing, information access control, and privacy. In particular, it has been shown that contacting a node's group or affiliation is an effective and efficient first step towards contacting the node itself [25]. However, due to the intermittently connected nature of DTNs, maintaining and disseminating such group information is challenging, especially in the presence of malicious attackers.

The enhancement of DTN communication through the use of group information requires that nodes throughout the network be aware of the membership lists for all groups. While such group membership management is not difficult in connected environments, heavy partitioning and the lack of readily available end-to-end paths in DTNs break centralized membership services, just like they break traditional routing [32, 51, 34]. Instead, group information must be disseminated through the network using DTN-specific mechanisms that leverage contacts between nodes that meet. Unfortunately, the reliance on contact-based dissemination and the presence of malicious or faulty nodes may result in inaccurate knowledge of group membership. While cryptographic techniques (i.e., PKI [33]) could provide proof of group membership, such techniques are not currently feasible in the intermittently connected environment of DTNs due to the lack of availability of a trust anchor. The main challenge then lies in ensuring accuracy of disseminated group membership lists.

Group membership management in DTNs can be broken into four components: group creation, group information propagation, group information consolidation, and group-assisted routing. During *group creation*, members of a group learn about their membership in the group. At this point, nodes outside of the group are not aware of the group's membership, or perhaps even of the group itself. Once the group has been formed, group members *propagate* the group's membership list throughout the network. Simultaneously, faulty and malicious nodes may be propagating inaccurate membership lists. Non-member nodes collect and *consolidate* all group membership lists as they receive them, locally resolving any conflicting information about a group's membership list. Finally, the resolved group membership list can be used to support *routing* and other services in the DTN. While there exists some work on group creation and group routing, current approaches ignore propagation and consolidation issues and do not consider the presence of malicious nodes tampering with group information.

Current approaches to disseminating group membership information in DTNs epidemically [55] propagate the information and, working under the assumption of a completely trustworthy environment, do not have any mechanism to handle conflicting information [24, 25]. Instead of resolving any conflicting information, these approaches typically default to selecting the newest information as truth. Unfortunately, this simplistic approach is inappropriate in many DTN environments, especially in the presence of malicious nodes. Essentially, such approaches form inaccurate group membership information, and so, result in ineffective routing. We show that these approaches actually break down in the presence of even a small number of malicious nodes. The main problem with these approaches stems from the lack of quick and robust propagation and consolidation.

The main contribution of our work comes from the design, analysis and evaluation of *MembersOnly*,

our group membership management protocol for DTNs that enables accurate group membership dissemination, even in the presence of multiple malicious nodes, through effective distribution and consolidation of group membership lists. The main strength of MembersOnly comes from the lack of reliance on any type of cryptographic techniques, making it an appropriate system for networks where groups are quickly created and destroyed and where centralized authorities do not exist. Given a set of partially conflicting membership lists, MembersOnly builds off of techniques from data mining to establish a local view of group membership. Compared to current methods, MembersOnly provides more accurate local views during convergence and quickly converges to very accurate views. Given that the maintenance of group membership lists is susceptible to various types of attacks on the consistency of a group's membership list, we show both analytically and through simulations that MembersOnly provides accurate results, despite the presence of moderate levels of malicious nodes. Finally, we demonstrate the impact of accurate group membership information through the evaluation of a simple group-based routing protocol, showing that MembersOnly can improve routing performance.

3.2 Four Components to Group-Management

Group-based communication is an important paradigm for DTNs. To understand how to enable and manage groups, we break down the problem into four components: group creation, group information propagation, group information consolidation, and group-assisted routing.

3.2.1 Group Creation

The goal of group creation is to form groups and allow all nodes of the group to know they are a part of it. Furthermore, group creation also informs group members of other nodes that are part of their group. In a DTN, groups can be formed based on some common feature of the individual nodes, including geography, node roles, and social affiliations. In some cases, groups can be created in advance; in other cases, they can be rather spontaneous.

The specific algorithms used for group creation depend on the type of group. For example, geographic groups can be created using centralized algorithms for community detection [40] or distributed versions of centralized algorithms [25]. In geographic groups, group centrality is important, and approaches such as weighted network analysis [38] can be implemented in a distributed fashion [25]. On the other hand, some dynamic groups rely on physical contact and verbal agreement, and so group creation is mostly out-of-band.

Although group creation is an interesting problem, we focus primarily on role-based or geographic-based groups where nodes initially know which groups they are in and group membership does not change. However, our algorithms do support dynamic groups where members might not initially know all other members of the group and where group membership changes over time. As a first step towards understanding how to support dynamic groups, we evaluate our algorithms during convergence, which gives us insight into how these algorithms behave in the face of dynamic groups.

3.2.2 Group Information Propagation

Once nodes know which groups they are in and who else is in these groups, they can start to propagate group membership information throughout the network. Such propagation enables nodes outside of a group to gather information about the membership list of that group. Propagation is a key component of group membership management, since the consolidation algorithms discussed next calculate their membership lists based on the information collected during propagation. There are two types of information that can be transmitted: group names for groups a node is a member of, and entire group membership lists. Effective propagation faces two challenges: convergence speed and malicious nodes.

The quickest way to propagate group membership information is for all nodes to epidemically disseminate all group information they are aware of. While optimal in terms of propagation speed, it is extremely vulnerable to attacks since malicious nodes can spread unrestricted amounts of information about any group. In doing so, they can convince non-malicious nodes to send false information, even if they are not part of the group itself, making consolidation too difficult. On the other hand, nodes could be more conservative and only disseminate a list of the groups to which they belong [24, 25]. Unfortunately, these approaches are very slow at propagating information.

To balance speed and security, we propose to limit nodes to only sending information about groups that they are members of. This limits the spread of false information while keeping propagation speed fast, providing a good basis for our consolidation protocol to achieve a high level of protection against malicious nodes.

3.2.3 Group Information Consolidation

Since group membership is propagated through the network, nodes collect multiple, potentially conflicting, pieces of information about each group. The nodes must consolidate this information into a single local view of each group's membership list. This local view can then be utilized by the routing protocol for use with routing decisions. In DTN environments, there is very little applicable work on consolidating conflict-

ing information. Therefore, we turn to the field of data mining and analysis. The TruthFinder system [61] solves a similar problem by first obtaining a set of "facts" about "objects" from multiple online servers, and then attempting to consolidate these potentially conflicting facts to determine the truth about the object.

Unfortunately, TruthFinder cannot be easily adapted to fit a DTN environment. It assumes that all of the servers, or fact providers, are always available and can be readily contacted, and hence gathers all facts before running the consolidation step; an impossible assumption in DTNs. Even if TruthFinder could be adapted to work in a disconnected environment, the algorithms in TruthFinder are designed to never omit any information from the final result, and hence result in a relatively high false positive rate. While acceptable for TruthFinder's purposes, a malicious node should not be able to easily append itself to a membership list.

To support group information consolidation in DTN environments, we present an on-line algorithm that collects group membership information from each node it meets and consolidates it on-the-fly without requiring contact with any centralized server. Our algorithm continually refines its decisions as more information becomes available, quickly converging to very accurate decisions about group membership. The combination of this algorithm with our membership-based propagation approach enables successful defense against many types of attacks, even in the presence of a large number of malicious nodes.

3.2.4 Group-Assisted Routing

Accurate group membership information can be used for many services, but the most obvious is group-assisted routing. For example, BubbleRap [25] utilizes group membership information to improve standard unicast routing. In essence, BubbleRap attempts to transmit a message to nodes that are part of the message destination's group, assuming that members of the same group have a high probability of contacting one another. Group information can also be used as a foundation for building anycast routing systems, where the goal is to reach at least one member of a particular group [20]. In the presences of faults and malicious users, the accuracy of the group membership information can have a large impact on the performance of any routing protocol.

Although we do not focus on routing protocols in this chapter, we evaluate the effect of the accuracy of group membership information on a simple anycast routing protocol, which can be used as a building block for more advanced protocols. Essentially, more accurate membership information, even during convergence, significantly improves even a simple anycast routing protocol. As part of our future research, we are investigating the use of group membership information in unicast, multicast and anycast routing pro-

tocols, as well as other group-based services.

3.3 MembersOnly

The ability to quickly and accurately distribute group information opens up the door for many group-based services. In DTN environments, both unreliable links and malicious nodes make this a challenging problem. In this section, we present MembersOnly, a local and robust group propagation and consolidation protocol that provides accurate views of groups even in the presence of malicious nodes.

3.3.1 Membership Dissemination

Effective propagation of membership information requires quick distribution while using mechanisms that support high integrity of group membership information. To help achieve this goal, MembersOnly takes a membership-based approach: nodes propagate group membership lists only for groups of which they are members. In other words, a set of membership lists, one for each group a node is a member of, is transmitted to every contact that node makes. This approach has two major benefits. First, it provides enough information for quick propagation, as opposed to transmitting only a list of groups the node is a member of. Second, it does not transmit everything (namely, information about groups the node is not a part of), hindering attackers' abilities to quickly spread false information. In other words, information transmission is not transitive.

Duplication is expected in DTNs, and hence nodes may receive multiple membership lists from the same node. Since membership lists are constantly evolving, the newest list should be used. Furthermore, if a membership list is not replaced with a newer one from the same node for some time, it is possible to assign a weighting factor to account for aging, which we will consider in future work.

3.3.2 Consolidation of Membership Lists

While MembersOnly enables fast propagation of membership lists, its true power lies in its ability to filter out malicious information. At any given time, a node has a set of, potentially old, membership lists for some or all groups in the network. MembersOnly leverages the availability of these multiple lists to build confidence levels for each potential member of a group, enabling well-informed consolidation that filters suggestions from malicious nodes. The goal of the consolidation component is to locally create a single high-confidence membership list for each group in the network. This high-confidence list contains members that the node believes are really part of the group. There can be at most $n \cdot g$ membership lists stored at a

node, where n is the number of nodes in the network and g is the number of groups. We note that storage space for group members should not be a major concern for the vast majority of modern mobile devices, particular those that are capable of storing videos, music, and pictures.

During consolidation, MembersOnly looks at all recommendations about membership for each potential member of a group and computes a confidence score about that member. This score is based on *positive* evidence extracted from all membership lists that claim that the node is part of the group and *negative* evidence extracted from all membership lists that do not have that node listed as part of the group. To reconcile these differences of opinion, MembersOnly calculates a function of the difference between the strength of the positive evidence and the strength of the negative evidence. If the result from this function is greater than a threshold, the node is placed on the high-confidence list for that group. Since malicious nodes may wrongfully inflate both the positive and the negative evidence, this function must be designed to filter out malicious evidence.

The MembersOnly consolidation component achieves high-confidence membership lists as follows. Each node n collects a set of membership lists, L, about a group G. Each entry on a membership list looks like "node m is a member of group G". Node n then computes a list, H, that contains, from n's point of view, all high confidence members of group G. To determine the confidence of a node m's membership in G, MembersOnly creates two subsets of L, L_m (lists containing m) and $L_{\bar{m}}$ (lists not containing m). $X_m = |L_m|$, the total number of lists m is found on, provides the positive evidence for believing that m is a member of G and $X_{\bar{m}} = |L_{\bar{m}}|$, the total number of lists m is not found on, represents the negative evidence.

Node n can now combine this positive and negative evidence to determine a confidence value about m's membership in G. Intuitively, the confidence can be captured by the difference between functions of the positive and negative evidence. To capture this, we build off of techniques used in data mining [61], where confidence should start low and quickly rise only after enough supporting evidence is obtained. This resulting S-shaped curve can be generalized by the popular Sigmoid Function [56] defined as

$$Y = \frac{1}{1 + e^{-X}}.$$

Applying the Sigmoid function, the strength of the positive and negative evidence for a node m is computed, respectively, as:

$$\frac{1}{1 + e^{-(X_m - \alpha)}} \text{ and } \frac{1}{1 + e^{-(\tau \cdot X_{\bar{m}} - \alpha)}}.$$

To support flexibility in our model, we have added the parameter α to generalize the standard Sigmoid

function. Changing α shifts the function along the x-axis, such that $Y = 0.5$ when $X = \alpha$. Furthermore, we add a weighted factor, τ, to the negative evidence. The effect of these parameters on the resulting confidence levels are described in detail shortly.

The total confidence about a node m's membership in group G can be found by taking the difference between the positive and negative evidence. Since it is unclear what negative confidence means, we ensure that the confidence does not fall below 0.

$$C(m) = max\left\{\frac{1}{1+e^{-(X_m-\alpha)}} - \frac{1}{1+e^{-(\tau \cdot X_{\bar{m}}-\alpha)}}, 0\right\}$$

$C(m)$ gives an indication of how confident node n should be in node m's membership. After node n computes this value for all possible members of a group, it selects the high-confident nodes to be part of the consolidated list H:

$$H = \{m | C(m) \geq \gamma\},$$

where γ is a system defined parameter that determines an accuracy threshold for the system. Essentially, if the difference between the strength of the positive evidence and the strength of the negative evidence is greater than the threshold γ, the membership in question is accepted. This process is repeated for all groups the node is aware of, and results in a consolidated list H for each group. The final set of H values can then be passed to the routing protocol.

Note that the parameter $\tau \in [0, 1]$ is used as a weighting factor for negative evidence, and the larger it is, the less negative evidence is needed to doubt an entry. In Section 3.4, we show how τ should be set to counter various attacks. Another important parameter is α. The smaller α is, the faster the propagation speed is when there are no attackers in the network, because a smaller amount of positive evidence is needed to reach γ. However, as α gets smaller, it has a negative impact on the appropriate setting of τ, which is detrimental to the system in regards to attacks. The discussion of these parameters is continued in Section 3.4.

3.4 Model Analysis with Attackers

The primary goal of MembersOnly is to provide quick and accurate group information to all nodes in the network, even in the presence of multiple malicious nodes. In this section, we present two high level classes of attacks and show, via mathematical analysis, how MembersOnly can be configured to defend against them.

3.4.1 Potential Attacks

We now briefly describe two generic attacks, an addition attack and a deletion attack, that give good insight into how attackers can affect and exploit different systems. We assume malicious nodes have similar abilities to normal nodes, in that they can send and receive any information they want during a contact.

The goal of the *addition attack* is to convince as many nodes as possible that the attacking node is part of some or all groups in the network. Therefore, attackers must convince normal nodes that they are valid entries on the local membership lists for those groups. If successful, attackers will be members of many groups and will be considered good intermediate nodes for routing to those groups. This positions the attackers to launch powerful black hole attacks, or other more sophisticated attacks. To demonstrate the effect of this attack, in Section 3.5, we instantiate a version of the addition attack where each attacker appends itself to all membership lists, and transmits this new information during contacts.

The goal of the *deletion attack* is to convince as many nodes as possible that valid members of a group are in fact not members. Attackers must therefore provide enough negative evidence about a node to cast doubt about it's membership in a group. Deleting members from membership lists can severely hinder routing performance. Essentially, a denial-of-service attack is launched, since nodes hold data until they meet a member of a particular group. To demonstrate the effect of this attack, in Section 3.5, we instantiate a "worst-case" version of the deletion attack where attackers simply broadcast membership lists containing only themselves for all groups they are currently aware of. This essentially attempts to delete all true nodes from the group.

3.4.2 Model Analysis with Attackers

Given that malicious nodes have the ability to perform both addition and deletion attacks, we now perform an analysis of our model to determine how best to defend against these attacks. Recall that if

$$\frac{1}{1+e^{-(X_m-\alpha)}} - \frac{1}{1+e^{-(\tau\cdot X_{\bar{m}}-\alpha)}} \geq \gamma,$$

then node n is confident to add m to the high confidence list H for group G. Also recall that if malicious nodes perform a deletion attack, their goal is to inflate the negative evidence against m enough to drop the confidence value below γ. Hence, to protect against this attack, τ, the negative evidence weighting factor, should be decreased. This gives less weight to the false negative evidence the attackers are providing. In contrast, if malicious nodes perform an addition attack, their goal is to inflate the positive evidence for their own false information, to bring the confidence value above γ. To protect against this, τ should be

increased, so true group members can provide the necessary negative evidence against the false positive evidence.

If a system only wishes to defend against deletion attacks, τ should be 0. Similarly, if a system only wishes to defend against addition attacks, τ should be 1. However, it is possible to set τ to defend against both addition and deletion attacks simultaneously by keeping it within a valid range. Larger ranges of τ are best, since this gives more flexibility in the actual choice for τ for a given system. To find the outer limits of this range, we analyze the steady state case, when every node has met every other node, to ensure that both types of attacks are, in the long run, completely defeated. We assume, for simplicity, that attackers cannot convince actual group members of changes in their own groups. While in practice this may not be true, particularly if nodes can join and leave groups without informing all group members of the action, it provides a good approximation.

In the MembersOnly group information propagation component, the only way a node can obtain information about a group is to meet a member of that group (or, at least a node that claims to be a member of that group). Given M, the total number of true members of a particular group, and A, the total number of attackers in the network attacking that group, the total amount of possible evidence for or against a node's membership is $M + A$.

For a deletion attack in the long run, nodes outside of the group obtain A recommendations *against* and M recommendations *for* the node in question. To protect against this attack, the confidence value computed by the non-member node should be greater than γ. In other words,

$$\frac{1}{1+e^{-(M-\alpha)}} - \frac{1}{1+e^{-(\tau \cdot A-\alpha)}} \geq \gamma.$$

Solving for τ, we find that to protect against the deletion attack,

$$\tau \leq \frac{1}{A} \cdot \left[\alpha - ln\left(\frac{-(\gamma + e^{\alpha-M} + \gamma \cdot e^{\alpha-M})}{\gamma + \gamma \cdot e^{\alpha-M} - 1}\right)\right].$$

Now consider attackers launching an addition attack against a particular group, where the goal is to get non-members to believe that the malicious nodes are actually part of the group. In the long run, the nodes outside of the group will have A recommendations for the entries and M recommendations against them. This is analogous to the previous inequality, and hence to protect against the addition attack,

$$\tau \geq \frac{1}{M} \cdot \left[\alpha - ln\left(\frac{-(\gamma + e^{\alpha-A} + \gamma \cdot e^{\alpha-A})}{\gamma + \gamma \cdot e^{\alpha-A} - 1}\right)\right].$$

It is immediately clear that if the number of attackers is greater than the number of nodes in the group, MembersOnly cannot defend against both types of attacks simultaneously. In this case, the user would have to choose which attack they were most concerned about, and adjust τ accordingly.

To clarify, consider the following example, which we also use for our evaluations. Assume a group of size of $M = 45$ and $\gamma = 0.75$.

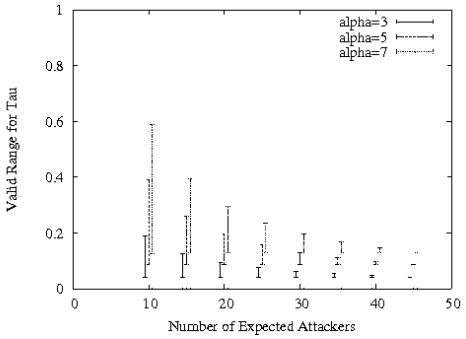

Figure 3.1: Valid ranges for τ

When choosing τ, it is important to choose a value that is within the valid range for the maximum expected number of malicious nodes in the network, to ensure that the system is within the valid τ range at all times. Figure 3.1 shows the valid ranges of τ as the number of attackers varies from 10 to 45. This figure shows that MembersOnly is able to defend against both addition and deletion attacks at the same time, as long as the number of attackers is less than the group size of the group in question. However, as the number of malicious nodes increases, the valid range of τ shrinks. It is also interesting to note that as α decreases, the ranges become smaller, which is undesirable. However, as α decreases, the propagation speed increases, which is desirable. We recommend setting α to be around the square root of the group size, since this allows for both quick propagation speeds and large ranges of τ. This model currently requires a weak estimate of the group size as well as an upper bound on the number of attackers in the network. While obtaining weak estimates like this is often not too difficult in practice, we are currently looking at ways to alleviate this requirement.

3.5 Evaluation

The goal of our evaluation is two-fold. First, we evaluate the propagation speed and attack resistance of MembersOnly in comparison to existing approaches and show that MembersOnly enables fast propagation and is extremely resistant to attacks, even in the presence of multiple malicious nodes. Second, we evaluate the effect of the parameters τ and α on the behavior of MembersOnly.

3.5.1 Evaluation Setup

For comparison, we use two propagation approaches: *CopyMyGroups*, where nodes transmit a list of every group they are members of to every contact they meet, and *CopyEverything*, where nodes transmit all group membership information they know to every contact they meet. For both of these protocols, the consolidation component is to simply take the newest version of any membership information as truth. While groups remain static throughout these simulations, we specifically evaluate the convergence time, giving insight into how dynamic groups would perform.

Average group completion percentage captures the speed and pervasiveness of group membership list propagation by tracking the completion percentage of a group over all nodes and all groups. Note that all nodes have access to an oracle with all correct group membership lists strictly for the purpose of metric computation. This metric will increase as soon as any node becomes aware of any subset of members for any group. The higher the metric's value is, the faster the system is at propagating group information. Both the normal propagation speed and the deletion attack effectiveness are measured using this metric. It is appropriate for the deletion attack, since the goal of the attackers is to delete as many members from every local membership list as possible, hindering propagation.

For the addition attack, the *average percentage of corrupt groups* captures how corrupt local membership lists are. A conservative approach is taken to say that a node's view of a group membership list is corrupt if that node (falsely) believes at least one attacker is actually a member of the group. It is the attackers' goal to corrupt as many groups as possible, driving the metric up. Hence, the lower the metric's value is, the better the system is at protecting against the addition attack.

All metrics are evaluated over time. Each of the evaluated systems eventually converges to either 0 or 1 for all metrics, and therefore it is most interesting to see how the curves progress over time, and how they look relative to one another. The exact time values are not as important as the characteristics of the curves, since these values change with properties of the network such as movement speed, transmission range, number of nodes, etc. Essentially, even though attackers may lose out in the end, there can be periods of vulnerability where attackers can make gains.

All simulations use the ONE [29] simulator and the random waypoint model, with nodes moving between 3 and 7 meters per second. There are a total of 250 nodes divided into 5 non-overlapping groups of either 50 nodes (if there are no attackers), 47 nodes (if there are 15 attackers), or 45 nodes (if there are 25 attackers). The transmission range of each node is 250m and the world size is 3.5 km x 3.5 km. All data points are the average of 10 runs with 95% confidence intervals. There are data points every 50 simulation seconds; however, many markers have been omitted for clarity. For all simulations, $\gamma = 0.75$.

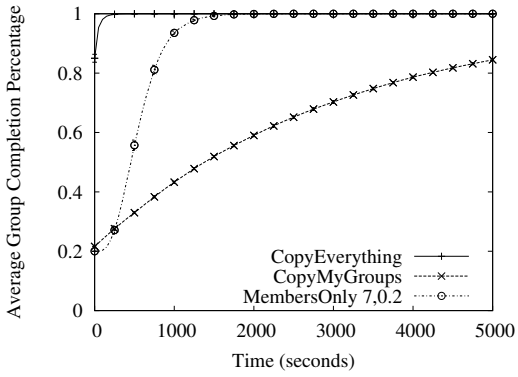

Figure 3.2: Group Completion

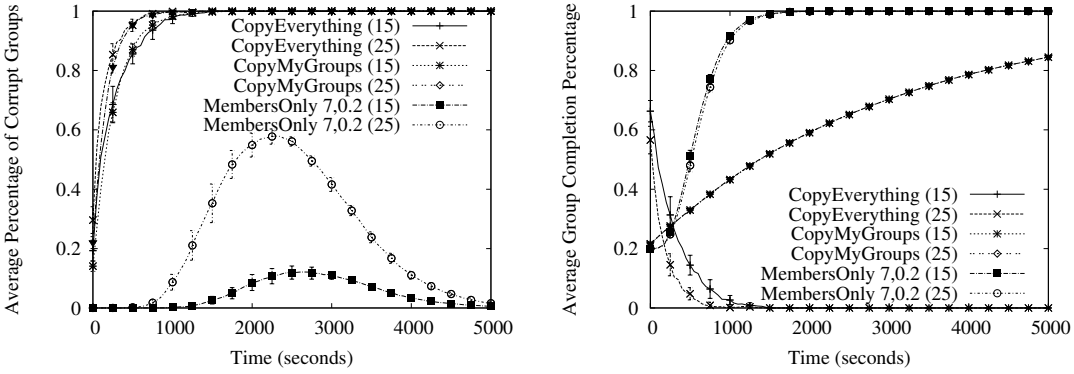

Figure 3.3: (a) Addition Attack, (b) Deletion Attack

3.5.2 Comparative Evaluation

To understand the impact of the propagation algorithm, MembersOnly is compared to both *CopyMyGroups* and *CopyEverything* by looking at membership list propagation speed as well as the effectiveness of the addition and deletion attacks. The number of malicious nodes, if any, in the simulations are denoted by parentheses next to the system name. The two numbers next to MembersOnly represent the parameters α and τ. As previously described, $\alpha = 7$ for these simulations, which constrains the choice of τ from around 0.13 to around 0.24, which handles up to 25 attackers. Therefore, we chose $\tau = 0.2$.

Propagation algorithms aim to spread membership lists throughout the network. As expected, *CopyEverything* shows the optimal speed since it epidemically disseminates all information (see Figure 3.2). Virtually all nodes are correctly aware of all membership lists in around 250 seconds. In contrast, *CopyMyGroups*, is relatively slow since it only transmits a list of groups a node is a part of during each contact, not a membership list of those groups. Therefore, to reach 100%, every node would have to come in contact with every other node. By the end of the simulation, at 5,000 seconds, this approach reaches only

around 85% completion. MembersOnly, which transmits group membership lists for all groups a node is a part of, starts off slightly slower than the other systems since, for security reasons, it waits for sufficient evidence before accepting information. However, after a sufficient amount of evidence is collected, nodes propagate the information very quickly.

Once attackers are introduced into the system, it is interesting to consider the average percentage of corrupt groups. For the addition attack (see Figure 3.3(a)), the higher the percentage, the more penetration the attackers gain, and hence the less resistant the system is to the attack. *CopyEverything* is slightly worse than *CopyMyGroups*. However, both are terrible at resisting the addition attack since the attack nodes persistently claim to be part of every group they know of. When a node meets enough attack nodes, it is convinced that at least some of the attack nodes are part of the group. This node then propagates that false information, convincing other nodes to do the same. This degenerating process is quite fast for both systems. In contrast, MembersOnly is more careful and considers the absence of information (i.e., a membership list without an attacker on it) as evidence against that information. Hence, the attackers can gain a temporary advantage with some nodes. However, in the long run, the attackers will not be able to overcome the honest nodes. The period of time where the metric is non-zero is a vulnerability period where some nodes wrongfully believe attackers are part of a group. As expected, the duration and prominence of this period increases as the number of attack nodes increases. However, even with 25 attackers, MembersOnly keeps the vulnerability period limited, and eventually the percentage goes to zero.

Finally, in the deletion attack, the attackers try to disrupt the propagation of group membership lists. With *CopyEverything*, membership lists quickly propagate and some gains are made (see Figure 3.3(b)). However, attackers continuously promoting membership lists with only themselves eventually cause a larger and larger number of nodes to believe that the membership list is actually blank. This results in the percentage going to zero, indicating the attack was successful. Interestingly, in *CopyMyGroups* the attack is not only unsuccessful, but useless since *CopyMyGroups* is immune from this attack because only a list of groups is propagated, never a membership list of those groups. Hence, there is no way for an attack node to convince another node of any membership list, let alone a blank one. The result of the attack on MembersOnly is simply a shift in time of the curve. The attackers are able to convince nodes to delay accepting membership lists as true for some time. However, eventually nodes running MembersOnly receive enough evidence from honest nodes to override the evidence given by attack nodes. Therefore, attackers simply delay the inevitable.

In summary, although *CopyEverything* is extremely fast, it is also extremely susceptible to both addition and deletion attacks. While *CopyMyGroups* is immune from deletion attacks, it is very susceptible

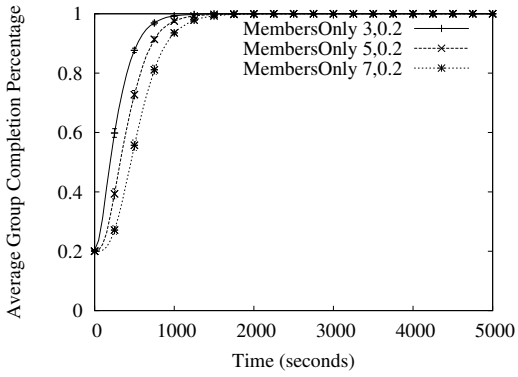

Figure 3.4: Group Completion

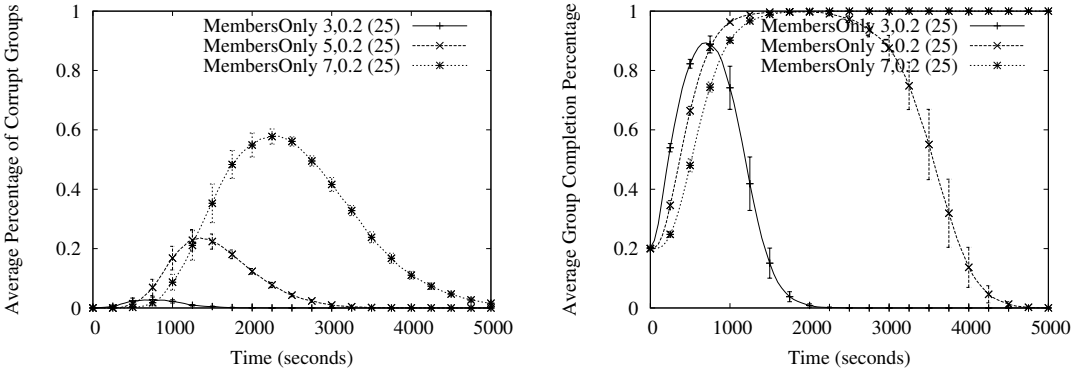

Figure 3.5: (a) Addition Attack, (b) Deletion Attack

to addition attacks and also too slow for practical use. In comparison, MembersOnly is both resistant to addition and deletion attacks, and can propagate group membership at a quick speed.

3.5.3 Parameter Evaluation

The parameters of MembersOnly determine both the propagation speed and resiliency to attack. Particularly, there is an interesting interplay between α and τ, which was explored in the analysis from Section 3.4. Recall, that as α decreases, the propagation speed should increase, since nodes can more quickly reach the threshold γ. However, as the analysis shows, this also decreases the valid range of τ. If τ is too low, addition attacks will succeed, and if τ is too high, deletion attacks will succeed.

When there are no attackers in the network, smaller values of α result in faster group information propagation due to the a smaller amount of positive evidence required to reach γ (see Figure 3.4). With addition attacks, MembersOnly successfully defends against the attack for all values of α (see Figure 3.5(a)), because $\tau = 0.2$, which is always greater than the minimum τ value, according to the model. Recall that lower values of α actually drop the range numerically, while shrinking it, and hence, lower values of α for

the same τ result in *better* protection against an addition attack. However, in the event of a deletion attack, the value of τ is greater than the maximum τ value for $\alpha = 3$ and $\alpha = 5$, according to the analysis. Essentially, MembersOnly *cannot* defend against the deletion attack for these two values of α, while it can for $\alpha = 7$ (see Figure 3.5(b)).

3.6 Group-based Routing

Obtaining quick and accurate group information about a network opens the door for a wider range and greater efficiency of routing protocols. One immediate result is the ability to efficiently perform anycast routing, which attempts to deliver a message to at least one member of a particular group [20]. Anycast routing is useful as a stand-alone routing technique for many scenarios. For instance, in emergency response networks, it may be more beneficial for an injured person to contact any emergency responder rather than a particular one. It is also useful as a means to improve unicast routing by first anycasting a message to a member of the destination's group, and then unicasting it from there.

To understand the impact of the accuracy of group membership on routing protocols, we evaluate a basic single-copy anycast routing protocol that utilizes group information to make routing decisions. We show that routing protocol performance is very dependent on the underlying group membership management. MembersOnly can improve routing performance by close to 8% under certain attack scenarios, in relation to current popular systems. For simplicity, a single-copy protocol was implemented, where replicas of messages are not created [53], and hence resource management is less important. This basic protocol acts as a building block for more advanced anycast routing techniques [36].

3.6.1 Anycast Routing and Attacks

One prominent building block for routing in DTNs is *direct delivery*, where a node simply holds a message until it comes in contact with the destination of that message. This building block allows for a store-and-carry approach to DTN routing and can even act as a very basic stand alone protocol. Similarly, a popular building block for anycast routing is to perform direct delivery to *destination groups* instead of destination nodes. This very simple protocol stores and carries all messages that are destined for a group rather than a node, until it meets a target group member. Since the message was successfully delivered (at least in the eyes of the deliverer) to the group, delivery is consider successful.

This group-based anycast routing protocol is reliant on the underlying group system for quick and accurate group information and so it is interesting to see how malicious nodes spreading bad group infor-

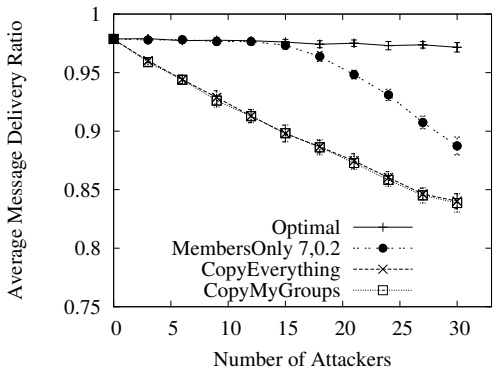

Figure 3.6: Routing Performance

mation affect the performance of the protocol. The malicious nodes attempt to perform multiple *black hole* attacks, where the goal of each is an aggressive addition attack to get on as many membership lists as possible. By getting on multiple membership lists, attackers can intercept and drop messages destined for those groups.

3.6.2 Performance Comparisons

The goal of this evaluation is to see how different group membership management approaches affect the performance of anycast routing under attack scenarios. We implemented the basic anycast routing protocol described above in the ONE simulator. For group membership management, we evaluate the performance of the anycast routing protocol using each of the following approaches: *MembersOnly*, *CopyEverything*, and *CopyMyGroups*. Additionally, we implemented an oracle module to provide a baseline that gives the routing protocol perfect group information at all times.

All simulations were done using the same parameters as before, except with a total of 150 nodes and groups of size $50 - A/3$, where A is the number of attackers. Messages are sourced from random non-malicious nodes and are destined for a particular group, ensuring that the group is different from the group of the message source. Every node sources a single message at a random time during every 200 second interval. Each message is 50kB in size, and buffers are large enough to hold all messages. Message delivery ratio, the metric used, is the total number of messages successfully delivered over the total number of messages sourced in the network. Every data point is the average of 10 runs.

By utilizing MembersOnly, the group-based anycast routing protocol achieves an approximately 8% improvement over current group approaches during some attack scenarios (see Figure 3.6). Furthermore, since the vulnerability window is relatively small during a low to moderate attack level, malicious nodes had trouble getting on more than a few local membership lists. Hence, MembersOnly performs close to

optimal much longer than other protocols in this environment. Conversely, in both *CopyEverything* and *CopyMyGroups*, many membership lists are quickly compromised and hence routing performance is significantly hurt.

3.7 Conclusions and Future Directions

We have presented *MembersOnly*, a local and robust group propagation and consolidation protocol that both quickly and accurately distributes group membership lists, even in the presence of multiple malicious nodes. We have shown via analysis and simulation that MembersOnly can withstand multiple types of attacks while still delivering membership lists quickly. Finally, we have shown that even the most basic group-based routing protocol can gain a performance advantage when using MembersOnly over existing protocols.

There are many interesting avenues for future work. First, one could investigate automated group creation. By analyzing trends in mobility and communication patterns, many types of groups, including social and geographic, can be automatically detected. Second, to improve the attack resistance of MembersOnly even further, one could utilize past information to help detect and limit malicious behavior in a network.

Chapter 4

EBR: Efficient Unicast Routing in DTNs

One of the primary hindrances to wide-spread DTN use is the lack of an efficient unicast protocol. Current work in DTN routing protocols leverage epidemic-style algorithms that trade off injecting many copies of messages into the network for increased probability of message delivery. However, such techniques can cause a large amount of contention in the network, increase overall delays, and drain each mobile node's limited battery supply. In this chapter, we present a new DTN routing algorithm, called Encounter-Based Routing (EBR), which maximizes delivery ratios while minimizing overhead and delay. Furthermore, we present a means of securing EBR against black hole denial-of-service attacks. EBR achieves up to a 40% improvement in message delivery over the current state-of-the-art, as well as achieving up to a 145% increase in goodput. Also, we further show how EBR outperforms other protocols by introduce three new composite metrics that better characterize DTN routing performance.

4.1 Unicast Motivation and Challenges

Delay and disruption tolerant networks transport application data by creating a "store and forward" network where no infrastructure exists. Although end-to-end connectivity may not be available between two nodes, DTN routing protocols instead take advantage of temporal paths created in the network as nodes encounter their neighbors and exchange messages they have been asked to forward. Since there are no guarantees that a route will ever be available, many current DTN routing protocols apply epidemic-style techniques [55], leveraging the fact that an increased number of copies of a particular message in the network should improve the probability that the message will reach its intended destination. However, such techniques come at a high price in terms of network resources, resulting in the rapid depletion of buffer space and energy on resource-limited devices, the rapid depletion of available bandwidth, and the potential to greatly increase end-to-end delay.

A number of routing protocols have been proposed to enable data delivery in such challenging environments [4, 10, 17, 18, 19, 32, 45, 51, 52, 57, 63]. However, many of these protocols trade overhead and

computational complexity for increased successful delivery. This overhead expresses itself as more traffic in the network creating more contention in clusters of high connectivity and increased energy consumption for nodes exchanging messages. Furthermore, many DTN protocols make routing and forwarding decisions based on advertised contact information, allowing for denial-of-service attacks over the already intermittently connected network. All of these effects can decrease overall network performance.

One method to mitigate this overhead is to identify key properties in the network that allow for more intelligent forwarding and message replication decisions. For example, in many human-centric environments targeted by DTNs, such as disaster scenarios and certain vehicular networks, different classes of nodes naturally tend to have more node encounters than others. The main contribution of our research in this chapter capitalizes on this network property to design a DTN routing protocol that uses local observations about a node's environment. Our protocol, Encounter-Based Routing (EBR), uses an encounter-based metric for optimization of message passing that maximizes message delivery ratio while minimizing overhead both in terms of extra traffic injected into the network and control overhead, as well as minimizing latency as a second order metric. Furthermore, we present a security component to our protocol that protects against denial-of-service attacks aimed at eliminating copies of messages in the system. To fully evaluate EBR, we propose the use of three composite metrics, which clearly illustrate the interplay between fundamental metrics like message delivery ratio, goodput, and end-to-end delay. We then use these metrics to evaluate EBR and compare it to the major protocols developed for DTNs, showing improved performance and overhead. EBR achieves up to a 40% improvement in message delivery over the current state-of-the-art, as well as achieving up to a 145% increase in goodput.

4.2 DTN Routing Protocol Taxonomy

DTN routing protocols can be classified as either *forwarding-based* or *replication-based*. *Forwarding-based* protocols keep one copy of a message in the network and attempt to forward that copy toward the destination at each encounter. In contrast, *replication-based* protocols insert multiple copies, or replicas, of a message into the network to increase the probability of message delivery. Essentially, replication-based protocols leverage a trade-off between resource usage (*e.g.*, node memory and bandwidth) and probability of message delivery. Although all replication-based protocols take advantage of this trade-off, these protocols can be further separated into two classes based on the number of replicas created: *quota-based* and *flooding-based*.

Flooding-based protocols send a replica of each message to as many nodes as possible, whereas quota-

Classification	Previous Work
Forwarding	Jain *et al.* [26], DSR [28], AODV [44]
Flooding-based Replication	Epidemic, Prophet [32], MaxProp [10], RAPID [4]
Quota-based Replication	Spray and Wait [51], Spray and Focus [52]

Table 4.1: Taxonomy of DTN routing protocols

based protocols intentionally limit the number of replicas. Assume that m_t indicates the maximum number of unique messages (excluding replicas) that have been created prior to some time t. Then an upper bound on the total number of messages (including replicas) in the network at time t is $m_t \cdot L$, where L is the maximum number of replicas for any given message. L can be a probabilistic or discrete variable. Given these definitions, a *quota-based* routing protocol can be defined as follows:

> A replication-based routing protocol is **quota-based** if and only if L is independent of the number of nodes in the network (assuming the characteristics of the network, such as storage, bandwidth, and mobility, allow for every node to have a replica of every message).

Conversely, any replication-based protocol where L is dependent on the number of nodes in the network is defined to be *flooding-based*.

These definitions allow us to classify routing protocols into three groups (see Table 4.1). Traditional Internet routing protocols (*e.g.*, IP [50]) and ad-hoc routing protocols (*e.g.*, AODV [44], DSR [28]) are forwarding-based, since nodes along a route forward messages toward the destination without storing or creating extra replicas of the messages. Forwarding-based approaches for DTNs have been proposed [22, 53], but are limited in their effectiveness due the instability or even non-existence of routes from any particular node to the destination. One forwarding-based approach, proposed by Jain *et al.* [26], utilizes future knowledge about node mobility and specific node encounters to improve the protocol (*e.g.*, knowledge that a node will encounter a bus at noon that will have access to the Internet). However, the availability of such future knowledge constitutes a special class of DTN networks and such approaches will not work in general.

Epidemic routing is an obvious example of a flooding-based protocol, since the number of replicas in the system is directly dependent on the number of nodes in the system. One of the major flooding-based protocols for DTNs is MaxProp [10]. MaxProp is flooding-based, since, if resources and mobility allow, it is possible for every node in the network to have a replica of the same message. Other examples of flooding-based DTN protocols include Prophet [32], RAPID [4] and PREP [45]. Prophet attempts to use informa-

tion about the likelihood of nodes encountering particular destinations to optimize the exchange of messages. RAPID orders messages through the use of utility functions, with the goal of intentionally maximizing specific metrics (*e.g.*, delay). PREP, a variant of Epidemic Routing, assigns priority to messages based on costs to destination as well as expiration time, and uses this priority to determine which messages should be deleted or transmitted when buffer or bandwidth is constrained respectively. In an attempt to mitigate the inherent resource burden from flooding-based protocols, many of these protocols specify complex optimizations, making implementation harder and error-prone. These optimizations are tuned and tweaked for performance in different environments.

Recent work by Erramilli *et. al* recognizes similar problems with current DTN routing protocols and proposes techniques to utilize properties of nodes, such as contact rate, when making forwarding decisions [18, 17]. They are concerned with choosing the best node(s) to forward messages to based on utility values. This technique, however, can result in flooding-like behavior if many encountered nodes have high utility values. On the other hand, if many encountered nodes have low utility value, messages may never leave the source nodes.

The main problem with flooding-based protocols is their high demand on network resources, such as storage and bandwidth. This led to work in developing quota-based protocols. Spray and Wait [51] is a quota-based protocol where an upper bound on the number of replicas allowed in the network is fixed during message creation. Spray and Wait breaks routing into two phases: a *spray* phase, where message replicas are disseminated, and a *wait* phase, where nodes with single-copy messages wait until a direct encounter with the respective destinations. A follow-up protocol called Spray and Focus [52] uses a similar spray phase, followed by a focus phase, where single copies can be forwarded to help maximize a utility function. While both Spray and Wait and Spray and Focus succeed in limiting some of the overhead of flooding-based protocols, their delivery ratios suffer.

While quota-based protocols are much better stewards of network resources than their flooding-based counterparts, one possible criticism is their inability to successfully deliver a comparable amount of messages. We show this to be false by developing a quota-based protocol using an encounter-based routing metric that has extremely low routing overhead, while maintaining delivery ratios better than or comparable to current flooding-based protocols.

4.3 Encounter-based Routing (EBR)

The primary goal of a DTN routing protocol is to obtain high message delivery ratios and good latency performance, while maintaining low overhead. However, current flooding-based protocols (*e.g.*, MaxProp [10], RAPID [4]) achieve high delivery ratios at the expense of excessive network resource usage, and current quota-based protocols (*e.g.*, Spray And Wait [51], Spray and Focus [52]) that reduce this overhead are not able to achieve comparable delivery rates.

In response, we present Encounter-based Routing (EBR), a quota-based DTN routing protocol that achieves high delivery ratios comparable to flooding-based protocols, while maintaining low network overhead. This improvement in delivery ratio is accomplished by taking advantage of the following observed mobility property of certain networks: *the future rate of node encounters can be roughly predicted by past data*. This property is useful because nodes that experience a large number of encounters are more likely to successfully pass the message along to the final destination than those nodes who only infrequently encounter others. Many networks experience this phenomenon; examples include disaster recovery networks, where ambulances and police tend to be more mobile and bridge more cluster gaps than civilians, and vehicular-based networks, where certain vehicles take popular routes.

Since EBR is a quota-based routing protocol, it limits the number of replicas of any message in the system, minimizing network resource usage. Additionally, EBR bases routing decisions on nodes' rates of encounters, showing preference to message exchanges with nodes that have high encounter rates. These routing decisions result in higher probability of message delivery, avoiding routes that may never result in delivery and so reducing the total number of message exchanges.

In EBR, information about a node's rate of encounter is a purely local metric and can be tracked using a small number of variables. Therefore, EBR is able to maintain very low state overhead, as compared to other protocols that can require up to $O(n)$ routing messages exchanged during *every* contact connection, and $O(n^2)$ routing state locally stored (*e.g.*, MaxProp [10], Prophet [32]). A further strength of EBR is that its message replication rules are simple to understand and implement, as opposed to complex rules found in many protocols, minimizing the chance of bugs and reducing computational complexity (*e.g.*, the resources in terms of CPU cycles required to operate the protocol).

4.3.1 Algorithm

Every node running EBR is responsible for maintaining their past rate of encounter average, which is used to predict future encounter rates. When two nodes meet, the relative ratio of their respective rates of en-

counter determines the appropriate fraction of message replicas the nodes should exchange. The primary purpose of tracking the rate of encounter is to intelligently decide how many replicas of a message a node should transfer during a contact opportunity.

To track a node's rate of encounter, every node maintains two pieces of local information: an encounter value (EV), and a current window counter (CWC). EV represents the node's past rate of encounters as an exponentially weighted moving average, while CWC is used to obtain information about the number of encounters in the current time interval. EV is periodically updated to account for the most recent CWC in which rate of encounter information was obtained. Updates to EV are computed as follows:

$$EV \leftarrow \alpha \cdot CWC + (1 - \alpha) \cdot EV.$$

This exponentially weighted moving average places an emphasis proportional to α on the most recent complete CWC. Updating CWC is straightforward: for every encounter, the CWC is incremented. When the current window update interval has expired, the encounter value is updated and the CWC is reset to zero. In our experiments, we found an α of 0.85 and update interval of around 30 seconds allow for reasonable results in a variety of networks. These parameter choices are further elaborated upon in Section 4.5.

Since EV represents a prediction of the future rate of encounters for each node per time interval, the node with the highest EV represents a higher probability of successful message delivery. Therefore, when two nodes meet, they compare their EVs. The number of replicas of a message transferred during a contact opportunity is proportional to the ratio of the EVs of the nodes. For two nodes A and B, for every message M_i, node A sends

$$m_i \cdot \frac{EV_B}{EV_A + EV_B}$$

replicas of M_i, where m_i is the total number of M_i replicas stored at node A. For example, assume node A has 4 replicas of a message M_1 and 8 replicas of a message M_2. Furthermore, assume node A, with $EV_A = 5$, comes in contact with node B, with $EV_B = 15$. Node A sends $\frac{15}{5+15} = \frac{3}{4}$ of the replicas of each message. Therefore, node A transmits 3 replicas of message M_1 and 6 replicas of message M_2.

Algorithm 1 presents the basic form of EBR, where W_i represents the current window update interval parameter.

Algorithm 1 *EBRRouting*

```
if time ≥ nextUpdate then
  EV ← α · CWC + (1 − α) · EV
  CWC ← 0
  nextUpdate ← time + W_i
end if
if Contact C available then
  for All messages M_i in local buffer do
    m_i ← M_i.numOfReplicas
    m_send ← ⌊m_i · EV_c / (EV_c + EV)⌋
    Send m_send replicas of M_i to node C
  end for
end if
```

4.3.2 Generalizing EBR

In this section, we prove that EBR adheres to the definition of a quota-based protocol (as described in Section 4.2) and show the relevant bounds, both for the simple version, where L, the maximum number of replicas of a message, is discrete, and for a more general version, allowing the use of probabilistic L values.

For discrete L values, it is easy to show that EBR is quota-based. Along with its data, every message contains a value indicating the maximum number of replicas into which this current message is allowed to be split. As an example, assume an application at node A creates a message with the maximum allowable replicas set to 10. Assume node A encounters node B and, based on the EBR protocol described in Section 4.3.1, wishes to transmit 8 replicas. Then, A creates a copy of the message for node B and assigns B's maximum allowable replicas to 8. Furthermore, A resets its maximum allowable replicas to 2. Continuing this procedure in a recursive fashion maintains the bound set by the initial message.

However, L values are not limited to a discrete maximum number of replicas. The discrete structure can easily be relaxed into a probabilistic structure, while maintaining meaningful (yet probabilistic) bounds. Probabilistic L values can allow for less sensitivity to exact network conditions. When using discrete L values, changes to the initial number of message replicas allows for a fundamental tradeoff between message delivery ratio, goodput, and average latency (see Section 4.5). Using probabilistic L values and changing the variance and mean can allow applications to compromise and not require exact decisions about the number of allowable replicas.

While any distribution may be used in this probabilistic model, the Gaussian distribution allows for immediate, eloquent properties that help establish the bound on the number of messages in the network. In this case, the application specifies the mean and variance of the distribution, instead of a discrete number. Assume a node A wishes to split the message M into two replicas, M_A and M_B. Node A must follow the following EBR message splitting rule:

If $M \sim N(\mu, \sigma^2)$, then it can only be split into $M_A \sim N(\mu_A, \sigma_A^2)$ and $M_B \sim N(\mu_B, \sigma_B^2)$ such that $\mu = \mu_A + \mu_B$ and $\sigma^2 = \sigma_A^2 + \sigma_B^2$.

For example, a message with mean 10 and variance 5 may be split into two messages, one with mean 8 and variance 4, and one with mean 2 and variance 1. It may not, however, be split into a message of mean 8 and variance 4, and one with mean 7 and variance 1. As a further note, EBR maintains the ratio of mean to variance for all message splits.

This message splitting rule preserves the Gaussian distribution for the two newly created replicas. This is due to a result from statistics known as Cramer's Theorem:

- If $X + Y \sim N(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$,
  then $X \sim N(\mu_x, \sigma_x^2)$ and $Y \sim N(\mu_y, \sigma_y^2)$.

We now demonstrate that this general version of EBR is a quota-based replication protocol, and establish an upper bound, by proving the following theorem:

Theorem 4.3.1 *Let S be a schedule of future message creations. Let t be an arbitrary future time. Assume*

$M_1, M_2, ..., M_i \in S$ are all the messages created before time t. Assume each message M_i has a Gaussian random variable (for notational ease, we refer to this directly as the message M_i), with mean μ_i and variance σ_i^2, that represents the maximum number of replicas the current message is allowed to be split into.

The upper bound on the maximum number of message replicas in the system is:

$$U \sim N\left(\sum_{j=1}^{i} \mu_j, \sum_{j=1}^{i} \sigma_j^2\right).$$

Proof: Let U be the sum of all message replicas in the system. Assuming messages never split, there will be i messages in the system, each with mean μ_i and variance σ_i^2. We utilize the following rule of linearity for Gaussian distributions (the converse of Cramer's Theorem):

- If $X \sim N(\mu_x, \sigma_x^2)$ and $Y \sim N(\mu_y, \sigma_y^2)$, then $X + Y \sim N(\mu_x + \mu_y, \sigma_x^2 + \sigma_y^2)$.

Therefore,

$$U = \sum_{j=1}^{i} M_j \sim N\left(\sum_{j=1}^{i} \mu_j, \sum_{j=1}^{i} \sigma_j^2\right).$$

Now assume a message, $M_j \sim N(\mu_j, \sigma_j^2)$ is split into $M_{j1} \sim N(\mu_{j1}, \sigma_{j1}^2)$ and $M_{j2} \sim N(\mu_{j2}, \sigma_{j2}^2)$ such that $\mu_j = \mu_{j1} + \mu_{j2}$ and $\sigma_j^2 = \sigma_{j1}^2 + \sigma_{j2}^2$ (the message splitting rule of EBR). Then by the same linearity

rules, $M_j = M_{j1} + M_{j2}$, leaving U unchanged. **QED**

One minor issue to address is that the statistical rules and theorems each assume true Gaussian distributions. However, it does not make sense in our system for a message M to hold a negative value. The probability of this occurring can be made sufficiently small by forcing the application to choose sufficiently low variances for corresponding means (which can never be below zero).

4.4 Securing EBR

The decision regarding how many replicas of a message a node should transmit to a contact depends completely upon the ratio of both parties' encounter values. Therefore, a malicious node can convince a node following protocol to transmit virtually any percentage of replicas to it. One of the most worrisome results is the possibility of a denial-of-service (DoS) attack where malicious nodes act as "black holes". Malicious nodes performing this attack advertise an ultra-high encounter value, causing all contacts to send almost all replicas to them. The malicious nodes then simply delete these messages, attempting to stop, or at least slow, message delivery.

Work by Burgess *et. al* shows that two popular types of denial-of-service attacks, dropping all messages (which we refer to as black hole denial-of-service) and flooding the network with fake messages, result in similar network degradation [9]. This degradation does not cripple the network because malicious nodes suffer from the same level of intermittent connectivity as non-malicious nodes. We have chosen to consider the case of black hole DoS attacks. This is because EBR is a low-overhead quota-based protocol, and hence extra flooding is not as big a concern as black holes. In quota-based protocols, non-malicious nodes do not flood messages, real or fake, and should simply drop messages with a high number of copies, since they are malicious.

To determine how vulnerable EBR is to black hole DoS attacks, we performed a series of simulations where a certain percentage of the nodes are malicious. Malicious nodes always advertise an exceptionally high encounter value, and immediately delete any message replicas obtained. Each data point is the average of 10 runs, and small 95% confidence intervals are shown. A vehicular mobility model is used, which is explained, along with simulation parameters, further in Section 4.5. The results of this experiment, shown in Figure 4.1, indicate that network performance can be hindered with a relatively small number of malicious nodes. However, matching the work done by Burgess *et. al*, additional malicious nodes are not able to cripple the network. These results indicate that it is necessary to provide an optional solution that prevents DoS attacks. Users not minding the decrease in performance may choose not to implement this solu-

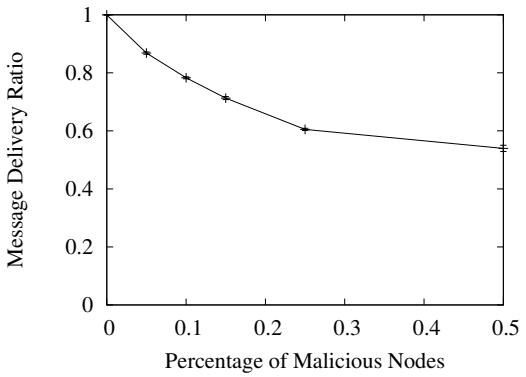

Figure 4.1: MDR in Attack Scenarios

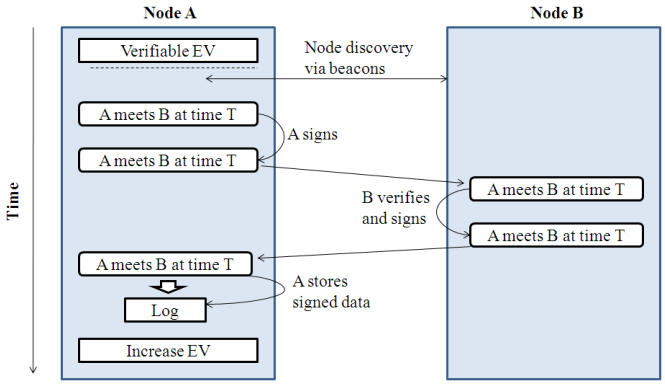

Figure 4.2: Timestamp Protocol

tion. However, providing a solution is necessary for those users more concerned about maximizing network performance. The penalty for choosing the solution is that there must exist a means of digitally signing data as well as binding keys to identities, such as PKI.

The insight of the solution comes from the observation that an encounter value can *never* be altered unless an external event (e.g., coming in contact with another node) occurs. Therefore, proving that the encounter value was altered only during an external event assures other nodes that the node in question is not individually faking the value. Now, of course, nodes can still collude to artificially inflate their encounter values; this case will be considered shortly. Note that the goal is to prevent the artificial increase, not decrease, of encounter values.

The protocol works as follows. Assume node A comes in contact with node C, and node C wishes to send data to node A. The goal is for node A to offer acceptable evidence to node C that the encounter value is not forged. To give acceptable evidence for this, node A must keep a list of transactions in which all previously encountered nodes digitally sign a time stamped message stating that "node A met me at time T". A graphical illustration of this is given in Figure 4.2. Node A can then offer all of these mes-

sages to node C, and allow node C to recompute node A's encounter value from scratch. If the recomputed value is equal to the value provided by node A, then node C can confidently transmit replicas to node A.

It is possible, even probable, that inherently trustworthy nodes are present in the network. For instance, in disaster recovery networks, police and emergency responders can be considered highly trustworthy entities. These nodes can be utilized to sign, or *checkpoint*, actual encounter values. This checkpointing process allows a node to delete all previous transactions and simply start with the new, signed encounter value. Checkpointing nodes verify the encounter value in the same fashion as mentioned above and then provide a signed encounter value back to the node. Checkpointing nodes must be trusted by all nodes in the network since previous transaction data is deleted after a signed encounter value is obtained (e.g., a node is checkpointed by a checkpointing node).

It is possible for colluding nodes to artificially inflate each other's encounter values by signing multiple "fake" meeting messages. This is a difficult problem, and we have not discovered a clear-cut solution. However, using statistical techniques, nodes diligent in looking for abnormal contact rates can mitigate the damage. If a node legitimately meets another node or group of nodes very frequently, it can lessen its chances of raising a false red flag by simply not storing some of the meetings, and not updating its encounter value for those meetings. A more thorough investigation of this is future work.

4.5 Evaluation

The primary goal of our evaluation is to show that EBR achieves a high message delivery ratio and good latency, while maintaining extremely low overhead. To demonstrate this, we first present the metrics used in our evaluation, followed by a brief description of the mobility models. Finally, we present a comprehensive evaluation of EBR in comparison to five other popular DTN routing protocols. To perform our evaluation, we use the Opportunistic Network Environment simulator (ONE) [29], which is a simulation environment designed specifically for disruption tolerant networks.

4.5.1 Metrics

Although traditional evaluation metrics provide a good understanding of the performance of a network, the evaluation of many current DTN routing protocols is hindered by the limited, and sometimes misleading, metrics used. To give a clearer, more complete picture of the evaluation, we consider three traditional performance metrics as well as introduce three composite metrics.

Traditional performance metrics include average message delivery ratio (MDR) and end-to-end message latency, while resource usage, or *resource friendliness* can be captured by goodput. Goodput is defined as the number of messages delivered divided by the total number of messages transferred (including those transfers that did not result in a delivery). In a resource constrained network, effective use of available storage can be captured by the number of messages dropped due to buffer overflows. We evaluated this metric in all of our scenarios; however, since it closely correlates to goodput, those results were omitted due to space constraints.

While these three traditional metrics provide a comprehensive view of the communication in DTNs, many protocols trade off effectiveness in one metric for effectiveness in another. Composite metrics are able to penalize protocols for performing poorly in individual primary metrics, giving a more complete picture of protocol performance. We consider three composite metrics to illustrate the relative relationship between the primary metrics. The *MDR x Average Delay* metric takes MDR and penalizes it for having a poor end-to-end delay, allowing for a more complete picture. Similarly, the *MDR x Goodput* metric looks at MDR and penalizes it for having poor goodput, giving a view of the network stewardship along with traditional MDR. Finally, the *MDR x Average Delay x Goodput* metric looks at MDR and penalizes it both for poor average delay and poor goodput. It is important to note that the absolute value of composite metrics is more or less meaningless by itself, since the metrics are artificial in nature. Therefore, when comparing protocols using composite metrics, one should consider the protocols' relative performance to one another. Further note that to maintain the standard of "higher is better", average delay is always inverted when used in composite metrics.

4.5.2 Mobility Models

Since DTNs can operate in many different environments, we use three different mobility models in our evaluation, specifically chosen to encompass a wide variety of DTN environments: a map-driven model simulating a vehicular network, an event-driven model simulating a disaster scenario [35], and a traditional random waypoint (RWP) model.

The vehicular-based map-driven model, which is part of the ONE simulator, limits node movement to actual streets found on an imported map, an approximate 5 km x 3 km section of downtown Helsinki, Finland. Approximately 15% of the nodes were configured to follow pre-defined routes (like tram lines) with speed between 7 and 10 m/s, the default for trams in the ONE simulator. The rest of the nodes were divided into four groups of nodes and four groups of "points-of-interest" (POI). Each node group was assigned different probabilities of picking the next node from a particular group of POIs to simulate the phe-

nomenon that people often visit certain areas of a city more frequently than others based on their profession, age and other factors. The speed of these nodes varied between 2.7 and 13.9 m/s, the default for car simulation in ONE.

The role-based, event-driven disaster mobility model [35], presented in Chapter 2, captures distinct movement patterns of roles as they react to external events. For this model, we simulate four equally spaced disaster events and a hospital. 50% of the nodes are civilians that flee from the events, 25% are ambulances that oscillate to and from events and a centrally located hospital, and 25% are police personnel who at first gravitate towards an event, but then react by "patrolling" the area in a random walk fashion. Police and ambulances always travel between 17 and 20 m/s, unless stopped. Civilians always travel between 1 and 4 m/s, unless stopped.

Finally, we simulate the routing protocols with a traditional random waypoint model. For these simulations, nodes are relatively slow moving, since the disaster scenario and vehicular models are relatively fast moving. Nodes move between 0.5 and 1.5 m/s, and pause at destinations for some time between 0 and 120 seconds.

For the disaster and random waypoint mobility models, the simulation area is 3 km by 3 km. For all simulations, the transmission range of each node is 250 m.

4.5.3 Performance Results

To demonstrate the effectiveness of EBR, we perform two groups of simulations on each of the three mobility models. To illustrate how each of the protocols reacts to changes in node density, we vary the number of nodes in the network starting at 26, followed by 51 to 251 in increments of 50, while keeping the area constant. The extra node represents a hospital in the middle of the simulation area for the purpose of the disaster scenario mobility model. To illustrate how each protocol reacts to varying network loads, we vary the per-node offered load by adjusting the number of messages sent per minute per source from 1 (lower load), to 2 (medium load), to 4 (higher low). Following this comparative evaluation, we evaluate how EBR reacts to changes in two local parameters: the popularity counter weighting constant (α) and the number of initial replicas per message.

In all simulations, we keep the area constant, the packet size constant at 25 KB, and the buffer space constant at 1 MB. Each simulation lasts for one simulated hour. Unless otherwise noted, each data point is the average of at least 10 runs, with 95% confidence intervals displayed. Due to the large amount of time required to simulate MaxProp in ONE, it was only evaluated fully for 26, 51, and 101 nodes, and is the average of four runs for 151 nodes, and is not evaluated for higher numbers of nodes. MaxProp is

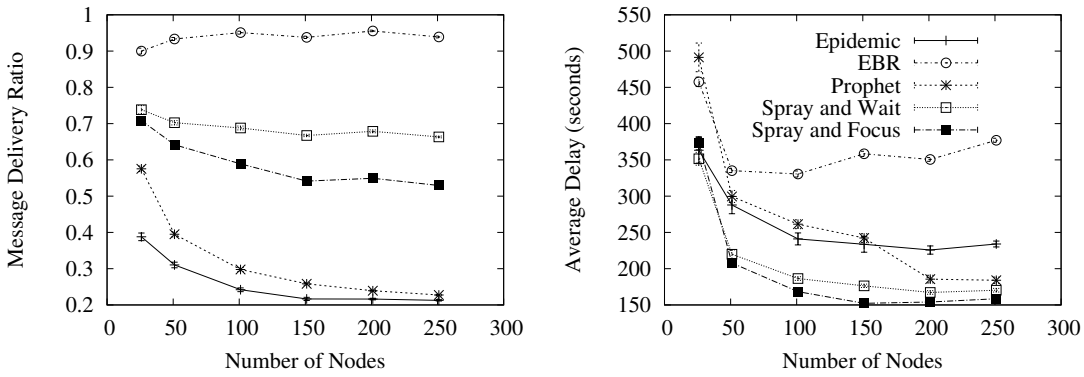

Figure 4.3: Vehicular: Varying number of nodes (a) MDR, (b) Average Delay

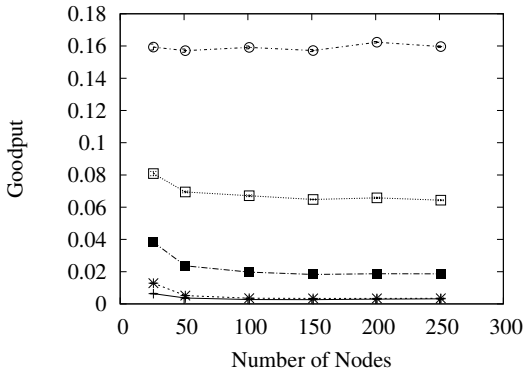

Figure 4.4: Vehicular: Varying number of nodes - Goodput

omitted from the evaluation using the vehicular mobility model due to the large amount of time required to simulate it.

Comparative Results

We evaluate EBR against five other popular protocols: (1) basic epidemic [55], (2) Prophet [32], (3) Spray and Wait [51], (4) Spray and Focus [52], and (5) MaxProp [10]. To enable a comparison between EBR and Spray and Focus, we implemented Spray and Focus to use an EBR-style encounter value (EV) to optimize delivery ratios in the focus phase. When nodes running Spray and Focus are in the focus phase, they hand-off single-copy messages to nodes with a higher EV.

First, we present the results from the vehicular mobility model. Note that MaxProp is not included in this set of simulations due to the large amount of time necessary to simulate it on the ONE simulator. EBR performs extremely well in terms of MDR, compared to the other quota-based protocols, Spray and Wait and Spray and Focus (see Figure 4.3(a)). Two factors account for this. First, the mobility model fits perfectly into the assumptions of EBR, namely that past information on rate-of-encounters is a good esti-

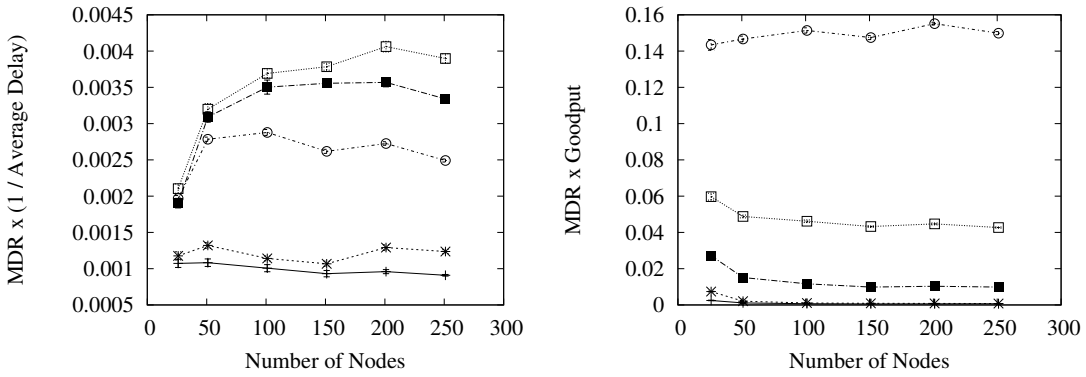

Figure 4.5: Vehicular: Varying number of nodes (a) MDR x Average Delay (b) MDR x Goodput

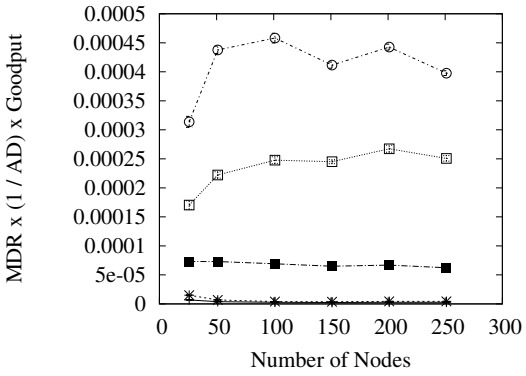

Figure 4.6: Vehicular: Varying number of nodes - MDR x Average Delay(AD) x Goodput

mator for future rate-of-encounters. Second, the network utilization seems to be correlated to MDR in this scenario, most likely due to constrained buffer space. EBR is, by far, the most resource friendly, as shown by the goodput metric (see Figure 4.4). While EBR seems to have unfavorable delay, this is, in part, due to a high MDR (see Figure 4.3(b)). Since delay is computed only over messages that have been delivered, it is deceptive to view delay alone since many protocols quickly deliver messages that take a small number of hops, and do not deliver most high-hop messages. The composite metrics, showing a more complete picture, further illustrate the power of EBR.

Second, we present the results from the disaster mobility model. As expected, in terms of MDR, MaxProp performs the best (see Figure 4.7(a)), due to its aggressive use of network resources. Closely following is EBR, which is never greater than 9 percentage points away from MaxProp. This is significant since EBR is much less demanding on network resources, yet can achieve a comparable MDR. Spray and Wait, which performs closest to EBR in terms of goodput (yet still significantly worse), performs noticeably worse in MDR. The reason EBR performs much better than Spray and Wait is due to the role-based characteristics of the disaster scenario mobility model. Both ambulances and police are highly active,

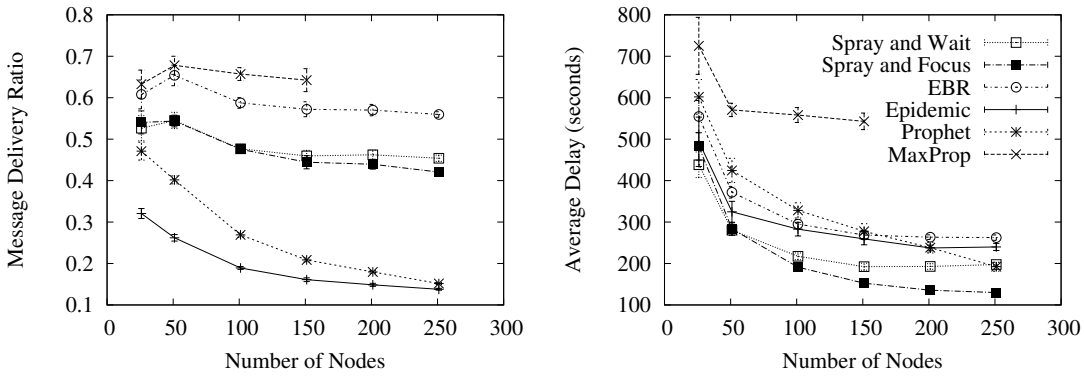

Figure 4.7: Disaster: Varying number of nodes (a) MDR, (b) Average Delay

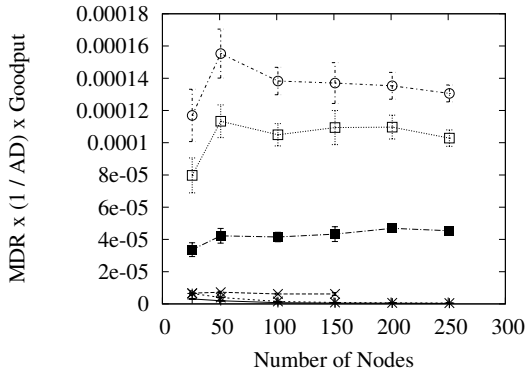

Figure 4.8: Disaster: Varying number of nodes - MDR x Average Delay(AD) x Goodput

more-so than civilians, and so EBR's assumption about predicting the rate of encounters using past data holds true. Furthermore, the goodput is significantly higher using EBR because if a large number of copies reach a high-encounter node, that node will not forward many of these copies to low-encounter nodes. This helps keep the network resource usages much lower than Spray and Wait. Note that both Prophet and Epidemic collapse as the number of nodes increases. In terms of latency, MaxProp performs worst, whereas Spray and Focus performs expectedly well (see Figure 4.7(b)).

Finally, the random waypoint model is considered. In terms of MDR (see Figure 4.9(a)), the gap between EBR and Spray and Wait is closer than with the disaster scenario (notice the change in scale). However, as the number of nodes increases, the gap becomes larger. The sudden increase at 50 to 100 nodes is due to the density finally becoming adequate for good delivery. Past this point, there is a minor decrease in performance for EBR, Spray and Wait and Spray and Focus and a more dramatic decrease for Prophet and Epidemic. We believe the poor performance of MaxProp is due to the relatively small buffer size. In terms of latency, Spray and Focus again performs the best (see Figure 4.9(b)); however, EBR consistently performs better than MaxProp. As expected, goodput strongly favors EBR.

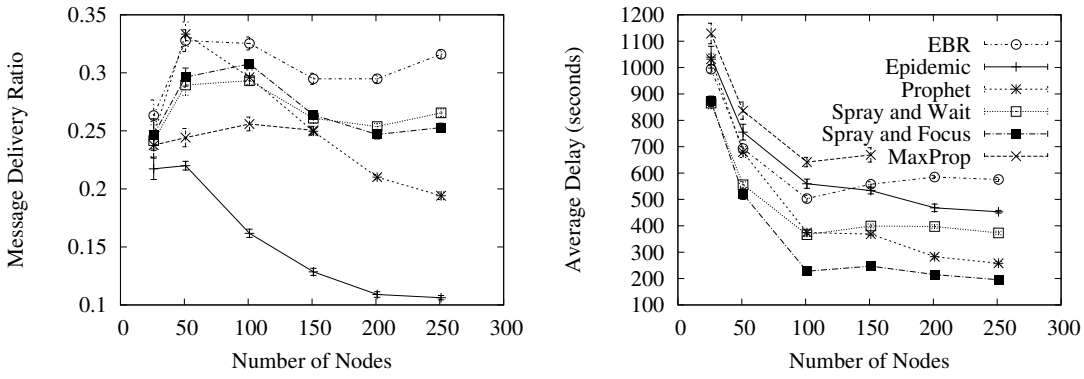

Figure 4.9: RWP: Varying number of nodes (a) MDR, (b) Average Delay

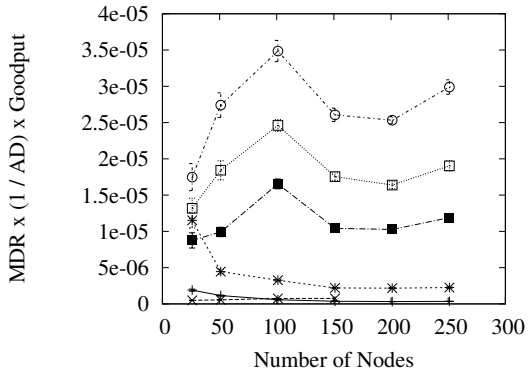

Figure 4.10: RWP: Varying number of nodes - MDR x Average Delay(AD) x Goodput

In the second group of simulations, the offered load is varied from 1 to 2 to 4 messages per source per minute. Due to space constraints, we only present results for the disaster mobility model and random waypoint model. Additionally, we only include the results for MDR, delay and the three-way composite metric. For the disaster scenario, MaxProp and EBR perform expectedly well, with all protocols suffering as the offered load increases (see Figure 4.11(a)). The average latency, however, shows MaxProp performing much worse than other metrics (see Figure 4.11(b)). Furthermore, as the offer load is increased from 1 to 4 messages per source per minute, EBR performs better than both Prophet and Epidemic. This is due to EBR's sharper drop in MDR as offer load increases. Spray and Focus and Spray and Wait perform the best, as expected. When combining all primary metrics, EBR performs at a high level, and the gap between EBR and Spray and Wait does not quickly close (see Figure 4.12).

When the offered load is varied using the RWP mobility model, the MaxProp data is averaged over three runs, with all other data averaged over ten runs. Due to the more uniform nature of per node rate of encounters, EBR does not perform as well as it does in the disaster scenario mobility model. However, in terms of MDR, it is still in the top tier, and performs higher than all others with lower offered loads

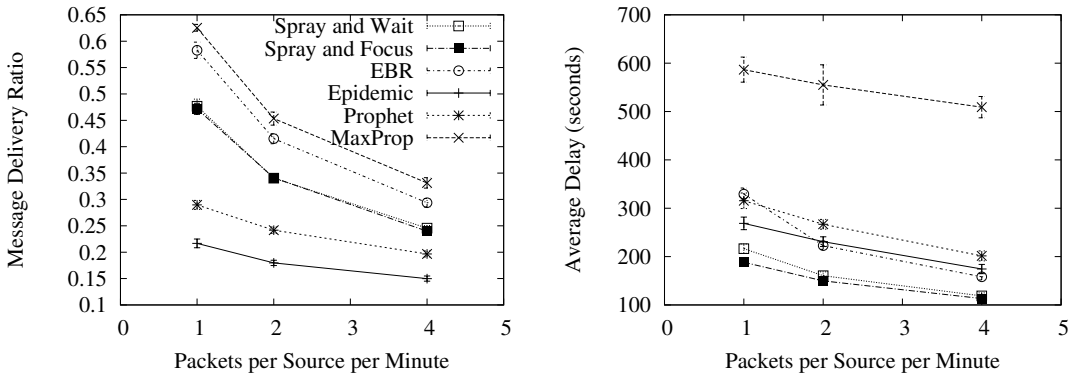

Figure 4.11: Disaster: Varying load (a) MDR, (b) Average Delay

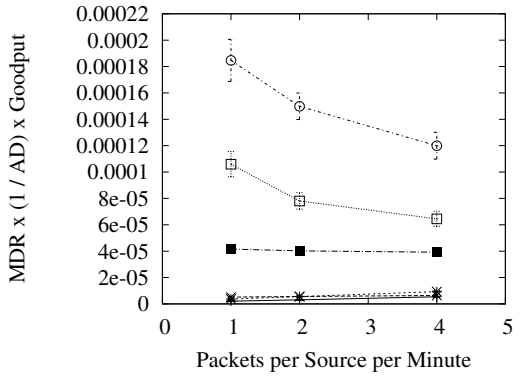

Figure 4.12: Disaster: Varying load - MDR x Average Delay(AD) x Goodput

(see Figure 4.13(a)). In terms of latency, as the offered load increases, the gaps between protocols tends to close (see Figure 4.13(b)). Finally, when combining all primary metrics, we notice that EBR performs at the highest level, primarily due to low overhead, and reasonable MDR and latency (see Figure 4.14).

EBR Parameter Results

To determine how EBR reacts to changes in internal parameters, we evaluate EBR against itself using different parameter settings. Due to space constraints, we only present results for the disaster scenario mobility model and only vary the number of nodes in the system. To evaluate the impact of the weight of the current rate of encounter in the EV counter, we vary α from 0.5 to 0.85. Additionally, to capture the tradeoff between resource usage and delay, we vary the starting number of message copies between 5, 11, and 20. Therefore, a total of 6 lines are shown per graph. Again, due to space constraints, we only present the graphs for the primary metrics, not the composite metrics.

In terms of MDR, α does not make a substantial difference. However, the number of initial copies does. As the number of nodes grows larger, EBR using only 5 copies starts to perform best, with EBR

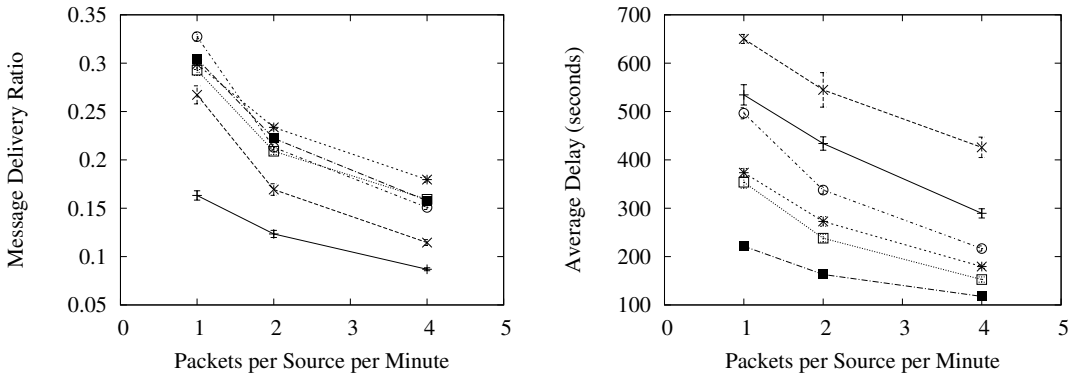

Figure 4.13: RWP: Varying load (a) MDR, (b) Average Delay

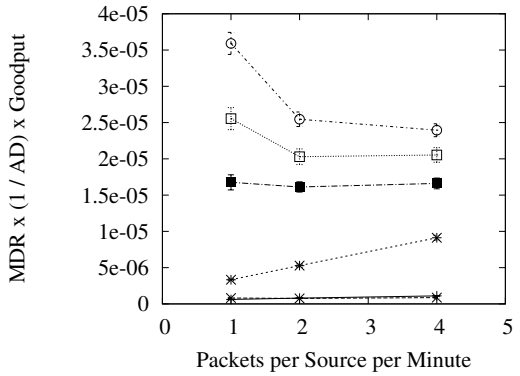

Figure 4.14: RWP: Varying load - MDR x Average Delay(AD) x Goodput

using 11 copies within a few percentage points (see Figure 4.15(a)). However, in terms of average delay, EBR using 5 copies performs significantly worse than with both 11 and 20 copies (see Figure 4.15(b)). Again, changing the value of α has little effect. The goodput is significantly greater when the number of copies is small, as expected (see Figure 4.16). In total, when not considering latency, a small number of copies, such as 5, allows for good performance of EBR. However, when latency is considered, a bit of a trade off must be made. Therefore, we have chosen to compromise and recommend a value of 11 initial copies as default to EBR.

4.6 Conclusions and Future Directions

The ability to efficiently and effectively route data through intermittently connected networks is of critical importance to DTNs. Many current routing protocols utilize flooding-based techniques to obtain relatively high message delivery ratios. This, however, comes at the expense of overwhelming network resources, mainly bandwidth and storage. Resource outages then lead to reduced performance in clustered

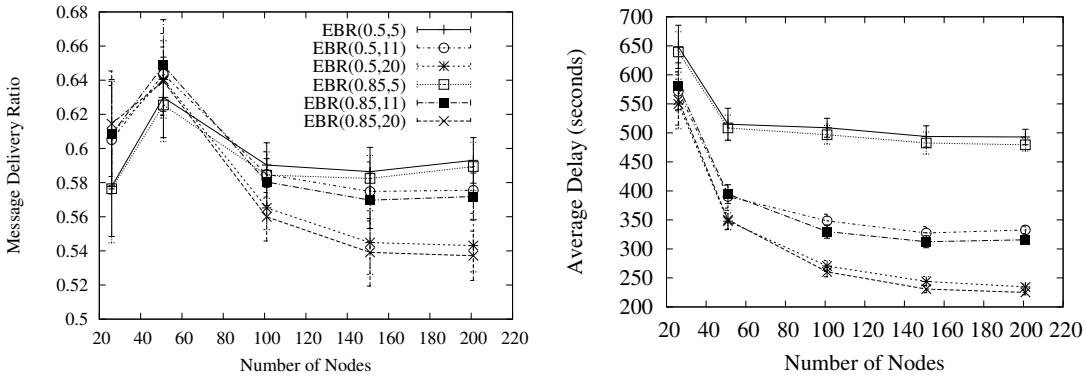

Figure 4.15: Disaster: Varying number of nodes (a) MDR, (b) Average Delay

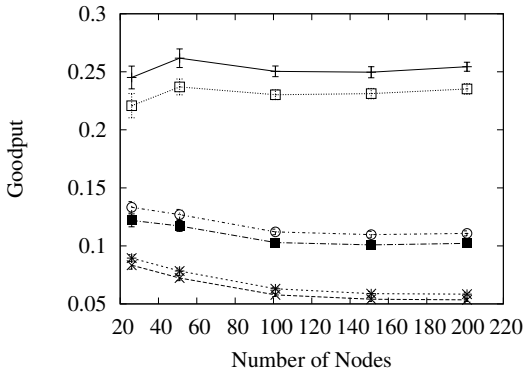

Figure 4.16: Disaster: Varying number of nodes - Goodput

areas, due to congestion, as well as energy strain on the devices. Filling all available buffer space with message replicas can hinder an application's ability to store local data. Additionally, overloading the network channel hinders one-hop protocols that do not rely on routing. Unfortunately, protocols that allow for low network resource utilization generally are not able to obtain comparable delivery ratios. We have shown that basing routing decisions on the encounter rate of a node can increase the delivery ratio. Our Encounter-Based Routing protocol (EBR) provides comparable or better message delivery ratios than current flooding-based protocols, while maintaining extremely low resource utilization.

There are many interesting future directions for encounter-based routing. First, it would be useful to evaluate EBR using probabilistic splitting rules, as described in Section 4.3.2. More specifically, to analyze the MDR, average latency, and goodput tradeoffs when the variance of the number of replicas is increased for all nodes, as well as when the variance is non-uniform for all nodes. Following this, one could explore, both mathematically and experimentally, distributions other than Gaussian. A second future direction is exploring the effects of using a second order derivative in terms of number of encounters. Currently, EBR only considers the current rate of encounters and averages this rate using an exponentially weighted av-

erage to account for both older and newer data. If EBR used a second order derivative, it would consider the *change* in rate of encounters over time and this trend could be used to distribute an appropriate number of message replicas.

Chapter 5

Achieving Anycast in DTNs by Enhancing Existing Unicast Protocols

Many DTN environments, such as emergency response networks and pocket-switched networks, are based on human mobility and communication patterns, which naturally lead to groups. In these scenarios, group-based communication is central, and hence a natural and useful routing paradigm is anycast, where a node attempts to communicate with at least one member of a particular group. Unfortunately, most existing anycast solutions assume connectivity, and the few specifically for DTNs are single-copy in nature and have only been evaluated in highly limited mobility models. In this chapter, we propose a protocol-independent method of enhancing a large number of existing DTN unicast protocols, giving them the ability to perform anycast communication. This method requires no change to the unicast protocols themselves and instead changes their world view by adding a thin layer beneath the routing layer. Through a thorough set of simulations, we also evaluate how different parameters and network conditions affect the performance of these newly transformed anycast protocols.

5.1 Anycast Motivation and Challenges

As mobile, wireless devices increase in popularity, intermittent connectivity becomes the norm. Robust communication to and from these devices requires protocols that are disruption tolerant by nature, with little reliance on static infrastructure. The natural communication patterns of these *disruption tolerant networks* (DTNs) differ from traditional Internet-based communication in that their structure is derived from node interaction and mobility. Furthermore, these networks can be highly heterogeneous and include smart phones, sensors, laptops, and even vehicles. Despite this heterogeneity, the historical Internet-style design principle of point-to-point communication (e.g., unicast) has dominated the DTN realm, severely hindering what could be a rich and diverse medium for new applications.

DTNs are critical to supporting environments that are human-centric by nature, such as emergency response, social networking and community networks [13]. The key interesting feature is that the communication supported in these networks is based on group-based human interaction [13, 49]. Such groups

can be geographic in nature, such as all the people on the same bus; social in nature, such as friends using mobile devices; or role-based, such as firefighters or police officers [35]. While user-based communication is naturally supported by network-level unicast communication, this group-based communication is more naturally supported by *anycast*, where communication with at least one member of a particular group is considered a success.

Two examples help illustrate the benefits of anycast communication in DTNs. First, in emergency response networks composed of groups such as police officers, ambulances, and civilians, group communication clearly trumps individual communication. A civilian is more likely to request the help of *any* police officer rather than a particular one. Similarly, police officers are more likely to request any ambulance, as opposed to a specific one. Second, in community DTN networks, which may be a composition of pedestrian social networks, vehicular networks, and local bus networks, group communication can also be very useful. As buses become equipped with Internet-able gateways, individual cars on the road would have incentive to contact any bus, instead of a specific one, for its gateway capability. Furthermore, pedestrians may be more interested in using the network to call for any cab, as opposed to a specific one. Interestingly, anycast can also be useful for enhancing unicast in DTNs. Essentially, smarter unicast routing protocols can be designed to contact nodes geographically affiliated with a target destination node as a first step towards contacting the target node itself [25].

The goal of anycast routing is to reach at least one node (the specific one does not matter) in a particular group. In connected environments, basic anycast routing techniques are relatively straightforward, since messages can be unicast to a particular node in the group that has the lowest cost (i.e., quickest response) [8, 41]. This technique, however, does not work in disconnected environments, since it is extremely difficult to predict which of the nodes of the group would even get the message, let alone be able to respond the fastest. In such a disconnected and unpredictable environment, anycast protocols must instead be smarter and attempt to truly reach *any* node in the group. While existing routing techniques for DTNs seem to lend themselves well to supporting anycast routing, current approaches to anycast in DTNs are very limited in scope, focusing on single-copy routing and/or targeted for highly constrained mobility patterns [20, 14, 25].

Current DTN routing protocols [34, 32, 10, 4, 52] are built on top of two key mechanisms: direct delivery, to support one-hop communication, and utility-based forwarding, to help guide messages to their destinations. Although both of these mechanisms can be used to support anycast routing, current unicast protocols are designed to route to a specific node as a destination and not to a group. The goal of the research presented in this chapter is to investigate the use of these mechanisms in an anycast setting

and how they need to be adapted to support groups as destinations. Additionally, we take this one step further and present a protocol-independent anycast layer that exposes group information in a meaningful way to unicast routing protocols, enabling these protocols to run unmodified to support anycast routing. If existing unicast protocols can be adapted to work in an anycast scenario, users could take advantage of a wide array of existing protocols that have been fine-tuned and thoroughly evaluated in many environments. These newly enhanced protocols could also be evaluated under a wide range of parameters and network conditions.

The main contributions of the research presented in this chapter are three-fold. First, we explore the mechanisms used in current DTN routing protocols and show how these mechanisms can be adapted to allow for anycast communication. Second, using these modified mechanisms, we present a protocol-independent anycast layer that allows current unicast protocols to run unmodified in an anycast mode. Third, we explore how the resulting anycast protocols perform in various environments, and specifically what features of an environment should guide a user in which anycast protocol to choose. We show that the choice is actually different in the anycast case, as opposed to the unicast case, since there are more factors to take into account. In particular, we show that important factors when choosing an anycast protocol include group size, resource constraints and mobility levels, and the presence of an acknowledgment scheme. Interestingly, the presence of a back channel (free and quick intra-group communication) does not have a major impact on the selected metrics.

5.2 Approaches to Group-based Communication

In group-based DTN scenarios, anycast can be used as the core routing paradigm. While there exists anycast solutions for connected environments [8, 41], these solutions all rely on stable end-to-end connectivity. In other words, they work under the assumption that the network allows for select group members to be reliably contacted and cannot be easily adapted to the disconnected and heavily partitioned environments found in DTNs. The challenge in disconnected and unpredictable environments is that anycast solutions cannot simply pick the "best" group member according to some metric and then use unicast techniques to reach it. Instead, they must take a group-based view where groups are the destination, not individual nodes.

The key to achieving anycast in DTNs lies in exposing knowledge of the groups in the network to the routing protocol, and having the routing protocol directly act on that knowledge. Similar to unicast protocols, anycast protocols can use single- or multi-copy techniques. In single-copy approaches, messages

are either held until the destination is met (e.g., direct delivery) or forwarded through intermediate nodes via a utility metric. While single-copy techniques work in some environments, they are unreliable in unpredictable DTN environments, since even the best guesses at which node to forward to are often wrong. Unfortunately, there has been little work on anycast in DTNs, all of which focuses on single-copy routing. One attempt at anycast routing in DTNs explores the problem by evaluating different routing metrics for selecting forwarding nodes [20]. However, this approach only analyzes single-copy routing. Furthermore, nodes are all stationary, with the exception of a few mobile nodes that act as message carriers, presenting a very constrained environment for evaluation. A second anycast technique, also using single-copy routing, attempts to utilize genetic algorithms to explore route decisions [14]. This work, however, assumes all mobility, including future mobility, is deterministic and known ahead of time, which is not a good assumption in most DTNs.

One DTN unicast protocol that incorporates elements of grouping is BubbleRAP [25]. This is a social-based forwarding method that first attempts to send messages to the destination's local community, which then can more effectively send the message to the actual destination. Nodes carry around a global ranking, determining how "central" they are to the network, and a local ranking, determining how central they are to their local community, which are used as hints to reach the destination node. However, BubbleRAP is designed for improving unicast, not anycast, and is inherently a single-copy technique and so is limited by the same problems as other single-copy approaches for anycast in DTNs.

It is worth noting that the related concept of multicast in DTNs, where the goal is to deliver a copy of the message to *every* member of the destination group, has briefly been considered. One simulation study, where no new protocols were proposed, considered existing multi-copy unicast routing protocols in a multicast context [3]. The simulation results of existing protocols were interesting. However, these results do not apply to anycast scenarios since the end goals are very different. Most specifically, a primary result was to include a considerable amount of redundancy to reach all group members. This would not hold true in anycast scenarios, where only *one* member need be reached.

Similar to unicast protocols, anycast in DTNs can benefit from managed replication. Unfortunately, it is not possible to directly use the multitude of currently available multi-copy DTN unicast protocols for anycast, since they do not have a group-based view of the world and operate on individual nodes. However, given our goal of deploying unmodified unicast protocols on top of an anycast layer, it is useful to discuss a few of the most relevant protocols. Current multi-copy unicast protocols can be characterized into two main groups, flooding-based protocols and quota-based protocols [34]. Flooding-based protocols, such as epidemic [55], Prophet [32], MaxProp [10], and RAPID [4], do not attempt to put a hard limit on

the number of times a message can replicate, and instead focus on smart buffer management and transmission ordering techniques to handle the potentially large number of replicas in the network. These protocols are more appropriate for environments not heavily constrained by limited resources. Quota-based protocols (e.g., Spray and Wait [51], Spray and Focus [52], and EBR [34]), on the other hand, set a hard limit on the number of times a message is allowed to replicate. Limited replication is guaranteed by attaching a quota to every message that indicates the number of replicas the message can split into in the future. The quota is split during each replication and a new quota is carried around with each replica, ensuring the total number of replicas of a particular message never exceeds the original quota for the message. These protocols are more suitable for resource-constrained environments.

While there unfortunately is a lack of effective DTN anycast protocols, it is important to note that there is a wide range of unicast protocols, each suited for different types of environments. Given the diversity of both DTN environments and DTN unicast routing protocols, it would be advantageous to be able to utilize these protocols for anycast routing. We next discuss which of the routing mechanisms commonly found in unicast protocols can be used for anycast and how they need to be modified to support groups as destinations.

5.3 Enabling Anycast in DTNs

Anycast, like other DTN routing protocols, needs mechanisms to guide replication, forwarding, and buffer management decisions. In this section, we show how common mechanisms found in unicast routing protocols can be adapted for anycast use. In particular, we show that a thin, protocol-independent anycast layer sitting directly below the routing layer can allow a wide-range of unicast protocols to run unmodified in anycast mode. In addition, we present a new DTN routing architecture that accounts for both unicast and anycast routing.

5.3.1 Group Management

One of the main requirements for anycast communication is access to information about group membership. Essentially, there must be a means of informing the routing protocol which groups the current contact belongs to. Such a *group management component* may store a nodeID-to-group table in memory, and update that table as it receives new group information. A simple approach to group management lets each node carry its own group information throughout the network [25]. While sufficient for the discussions in this chapter, such approaches are inherently susceptible to malicious attacks on the group member-

ship lists. A complex and robust form of group management is MembersOnly [37], which we presented in Chapter 3.

5.3.2 Routing Mechanisms

Most current DTN routing protocols perform two steps during a given contact opportunity: direct delivery, which supports one-hop delivery of messages, and utility-based forwarding, which guides messages and replicas towards their destination. These two steps can be enhanced to support a group-based view, instead of a node-based view, enabling anycast routing. Current protocols that follow this two-step process include Direct Delivery (with a non-existing utility step), Prophet, MaxProp, RAPID, and Spray and Focus (a follow-up protocol to Spray and Wait).

Direct delivery (referred to as *DD*) supports one-hop delivery of messages, where if the node has a message destined for the contact, that message is immediately transmitted to the contact. DD works by checking every message's destination ID against the contact node ID, as provided by the network layer. If the destination ID matches the contact node's ID, that message is immediately forwarded to the contact. To support anycast, DD must instead check every message's destination ID (which, in the case of anycast is a group ID) against groups that the contact is a member of. To support this, anycast routing protocols must obtain both a node ID from the network layer and a corresponding group ID from the group management component. If the destination ID matches any of the group IDs, that message is immediately forwarded to the contact.

After all messages destined for the contact are delivered, the protocol switches to the utility step (referred to as *Utility*). Based on the contact, a utility function is computed or looked up, and used to decide which, if any, of the stored messages to replicate, which order to send messages, and which order to drop messages. For unicast protocols, these utilities are *node-based*; every node the protocol knows about has a utility attached to it. An example utility is the probability of meeting a particular node. Utility values are updated either periodically or when contacts occur. Similar to DD, Utility can be adapted to support anycast by using group-based utility values where all routing policies work on groups instead of nodes. In other words, each node stores utilities for all *groups*, not individual nodes, that they are aware of. This, in essence, transforms groups into virtual nodes from the node's perspective. When a contact occurs, the node updates the utility for the contact's group(s) instead of the contact's actual node ID. This enables the routing protocols to capture mobility characteristics (such as meeting frequency) of groups instead of individual nodes.

5.3.3 Protocol Independent Anycast Layer

In the previous subsection, we have shown that the identification and modification of two commonly used routing techniques in DTNs, namely DD and Utility, can allow most current unicast protocols to operate in an anycast mode. Essentially, any unicast protocol that follows the two-step process can easily be adapted to support anycast. When the DD and Utility steps for a unicast protocol are transformed into their anycast counterparts (DD-A and Utility-A), the protocol's view of the world turns from node-based to group-based. While it is useful to know how to perform these transformations, it would also be beneficial to support anycast without modifying the unicast protocols at all. Since both DD-A and Utility-A work on group IDs instead of node IDs, it is possible to simply present a *group view* of the network to a unicast routing protocol, and have that protocol work as if it were an anycast protocol. The *group view layer* (GV layer) takes both the node ID from the network layer and the corresponding group ID from the group management component, and passes the group ID to the protocol *in the node ID field*. This presents a view to the routing protocol where every group is actually a virtual node, and every time a member of that group is encountered, it is like that single virtual node was encountered. This view combined with DD and Utility is equivalent to DD-A and Utility-A, enabling the unicast protocols to run in anycast mode unchanged.

5.3.4 Building a New DTN Routing Architecture

To properly integrate our anycast layer, we now describe a network architecture for DTNs that incorporates both unicast and anycast routing. Routing in DTNs has traditionally been unicast in nature, and therefore the network layer simply passed raw contact information to the unicast protocol, which then appropriately routes application data. However, anycast in DTNs require a slightly more complex architecture, since group IDs must be relayed to the routing protocol in addition to simple contact information.

Figure 5.1 illustrates a DTN network architecture, focusing on routing, that incorporates both anycast and unicast capabilities. Note that there are two primary tracks that lead up and down the stack: the anycast track and the unicast track. The unicast track is identical to before, the network and application layer can directly interact with the unicast routing protocol. The anycast track includes the group management component directly above the network layer. This allows both the node IDs of the current contacts, as well as their respective groups, to be passed to the anycast routing protocol. More specifically, the network layer passes node ID information about each of the nodes currently in communication range to the group management component. This group management component then looks up the corresponding group ID for each node ID, and attaches that information. This pairing is then sent either directly to

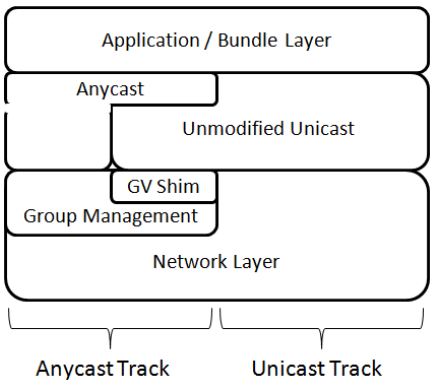

Figure 5.1: Architecture

the anycast routing protocol (the left side of the anycast track), or to the GV layer (the right side of the anycast track). The GV Layer, presenting a group-centric view of the world, indicates to the routing protocol which groups it is in contact with by substituting group IDs for the node ID fields. This gives the illusion that an entire group is actually a single (virtual) node, and allows many unicast protocols to run unmodified.

5.4 Evaluation

The goal of our evaluation is to explore the behavior of the anycast adapted versions of a few representative unicast DTN routing protocols in different environments. These evaluations can then help determine which protocol should be use in which DTN environment. In general, unicast DTN protocols are affected by characteristics such as resource constraints, mobility levels and acknowledgments. However, anycast scenarios are also affected by other characteristics such as group size and group back channel availability (i.e., whether group nodes can communicate via an out-of-band back channel). We evaluate how both the traditional unicast characteristics as well as new anycast characteristics affect the transformed anycast protocols.

Our results show that the most important network characteristics for anycast protocols are group size, resource constraints, mobility levels, and the presence of an acknowledgment scheme. Interestingly, the presence of a back channel does not have a major impact on the selected metrics. Where acknowledgments can flush out already delivered messages, a back channel simply allows instantaneous intra-group communication, and can be used as a means to increase the spread of acknowledgment messages. Essentially, once one member of the group obtains a message, it can then use the back channel to instruct all of its other group members to flood out an acknowledgment for the message just received.

5.4.1 Anycast Protocols

In our evaluation, we include representatives of different classes of single- and multi-copy DTN routing, including flooding- and quota-based protocols, each of which was implemented in the ONE simulator [29]. Using the protocol-independent anycast layer, the flooding-based Prophet [32] becomes Prophet-A and the quota-based Spray and Focus [52] becomes Spray and Focus-A, where the focus function is time since last meeting the destination (i.e., the group). Note that, for Spray and Focus, the initial quota was set to 11, which allows a good trade-off of resource usage and performance. In addition, we include a direct delivery anycast protocol (DD-A), as well as a pure epidemic protocol (Epidemic-A). Finally, we included an optimized epidemic approach where bandwidth is unlimited and message sizes are negligible (EpidemicOracle-A). This gives an upper bound on what anycast protocols can hope to achieve. Note that Epidemic-A by itself does not give optimal performance since there is no guiding factor in which messages to transmit first. In many cases, Epidemic-A may choose the "wrong message", which in turn does not allow the "right message" to be sent before the contact is broken. In addition to the anycast layer, we implemented two additional features that can be turned on and off: flooded acknowledgments and group back channel connectivity. These features allow us to explore in-depth the network characteristics that affect anycast performance.

Note that there is no additional overhead, outside of what is incurred by the group management component, in these enhanced protocols. In fact, many of the protocols store *less* data, since the number of utility values is decreased in the enhanced version.

5.4.2 Metrics and Simulation Environment

The first, and primary, metric used for the evaluation is *message delivery ratio* (MDR), which is the ratio of total messages sent divided by total messages received. Due to the intermittently connected environment, not all messages sent will be delivered, and hence this metric gives important information about a routing protocol's effectiveness. The second metric is *average delay of delivered messages*, which is the average end-to-end delay over all delivered messages. This metric indicates how quickly routing protocols can deliver messages, which is important since many messages lose relevance after being delayed for a long period of time. This metric is more meaningful if the MDR's of all protocols being compared are similar, since only delivered messages are included. The third and final metric is *average overhead ratio*, which is the total number of transmissions divided by the total number of received messages. Intuitively, this ratio indicates the average number of transmissions required for each message delivered, giving an indication of resource use. The lower the overhead ratio, the less strain there is on network resources (battery life, net-

work bandwidth, etc.). This is an important metric to indicate the "resource-friendliness" of the protocols.

All simulations were run in the ONE [29] simulator using the built-in community mobility model. This model simulates the movement of pedestrians, cars, and buses in the city of Helsinki, Finland. In all simulations, there are a total of 126 nodes: 80 pedestrians traveling between 0.5 and 1.5 meters per second, 40 cars traveling either between 2.7 and 13.9 meters per second (around 10 to 50 km per hour) in the normal case or between 2.7 and 3 meters per second in the slow case, and 6 buses traveling between 7 and 10 meters per second. This allows for an appropriate density to evaluate DTN protocols. All nodes can transmit at a distance of 100m and speed of 2Mbps, except two of the buses, equipped with high-speed interfaces, which can transmit at a distance of 1000m and a speed of 10Mbps. The nodes choose a location to move to with weighted probabilities (with real Helsinki hotspots being given higher probability), wait for a random amount of time between 0 and 120 seconds, and then repeat. All simulations are run for 4000 seconds, the world size is 4.5km x 3.4km, and node buffer sizes are 5MB. All data points are an average of 10 runs with a surrounding 95% confidence interval.

Nodes 0 to 39 are pedestrians, 40 to 79 are cars, 80 to 119 are pedestrians, and 120 to 125 are buses. In these scenarios, groups are simply chosen sequentially, and are disjoint. For instance, if the group size is 8, then nodes 0 to 7 are in group 0, 8 to 15 are in group 1, etc. This helps keep similar types of nodes in the same group (for example, pedestrians that have a high probability of visiting a particular hotspot), while at the same time allowing for a straightforward way of experimenting with different group sizes.

5.4.3 Performance Evaluation

The goal of this evaluation is to determine how different factors affect the transformed anycast protocols. The simulations are divided into two groups, a resource-constrained (RC) environment and a non-resource-constrained (Non-RC) environment. Group sizes are evaluated at 1 (which is equivalent to unicast), 2, 4, 8, and 16. Evaluation across this range indicates how anycast protocols perform when moving away from unicast and towards larger group sizes. For each set of simulations, the availability of acknowledgments and the availability of a back channel to help spread acknowledgments faster are both considered. There are three configurations: (1) no ACKs and no back channel, (2) ACKs and no back channel, and (3) ACKs and a back channel.

Resource-Constrained

In this set of simulations, the transformed anycast protocols attempt to deliver messages in a resource-constrained environment. Message sizes vary between 500k and 1M, and a message is generated by a ran-

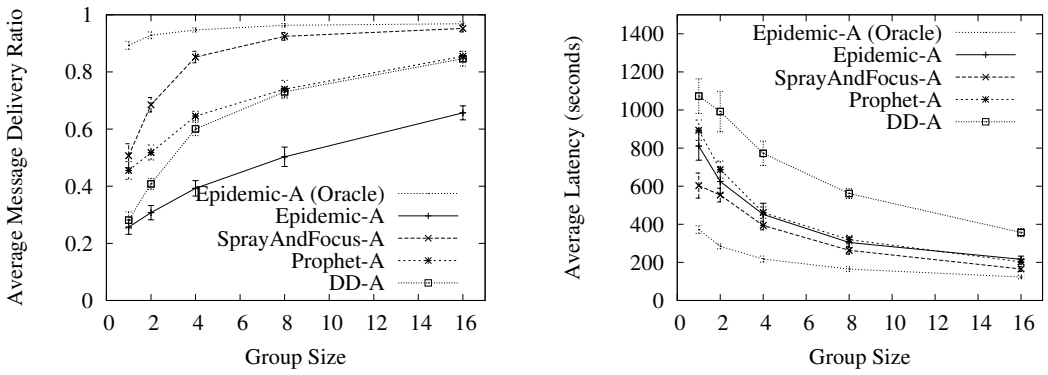

Figure 5.2: RC, no acks, no back channel (a) MDR, (b) Delay

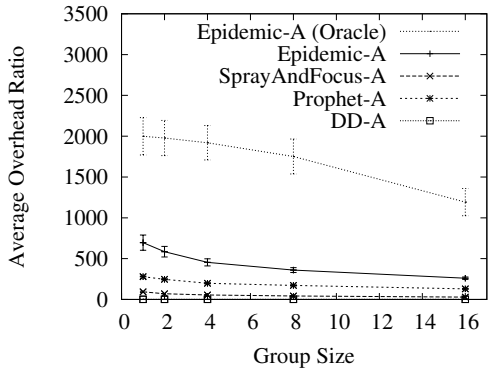

Figure 5.3: RC, no acks, no back channel - Overhead

dom node every 25 to 35 seconds, destined to a random group. In this scenario, cars are traveling at normal speeds (2.7 to 13.9 meters per second, as previously mentioned).

With no acknowledgments and no access to a back channel (see Figure 5.2), anycast behavior is actually quite different than unicast behavior. In a unicast environment (when the group size is 1), quota-based protocols (e.g., Spray and Focus-A) perform only slightly better than smart flooding-based protocols (e.g., Prophet). However, the gap between quota-based protocols and flooding-based protocols becomes much larger as the group size increases (as shown in Figure 5.2 (a)). With larger groups, it gets increasingly easy to meet a group member, and hence limiting replication is not hurting the network. In this resource-constrained environment, flooding-based protocols (particularly Epidemic-A) quickly overwhelm resources (as shown in Figure 5.3) to the extent that only a small fraction of buffered messages are transmitted during every brief contact opportunity. This indicates that group size is very important for protocol performance. Another surprising result is that as group size increases, DD-A actually performs as well as Prophet-A, although Prophet-A still unsurprisingly delivers messages quicker (as shown in Figure 5.2 (b)). This is because nodes running DD-A are able to eventually meet most groups when

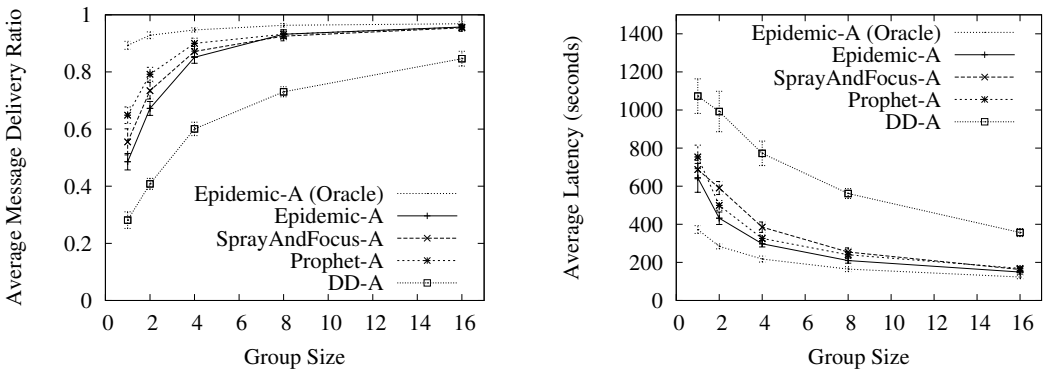

Figure 5.4: RC, acks, no back channel (a) MDR, (b) Delay

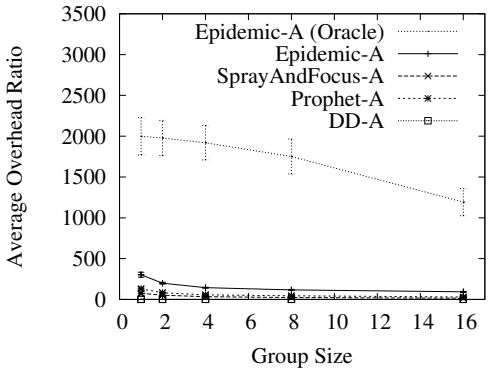

Figure 5.5: RC, acks, no back channel - Overhead

group sizes are large, without causing any strain on resources (no message drops, not as much worrying about contact duration, very little congestion, etc.). Overall, quota-based protocols are recommended in resource-constrained environments, particularly if group sizes are large and there is reasonable mobility.

Next, consider the presence of acknowledgments, but the lack of back channel access, as shown in Figure 5.4. It is immediately clear that acknowledgments drastically improve the performance of flooding-based protocols since these protocols, including Epidemic-A and Prophet-A, consume the most resources, to the point that they are near optimal, along with Spray and Focus-A, when the group size is above 8 (as shown in Figure 5.4(a)). Quota-based protocols are slightly improved, and DD-A is not improved at all since there is already a limit of at most 1 copy of every message in the network. Average delay also decreases slightly for all protocols. In terms of overhead, a significant improvement is seen, since wasteful transmissions (e.g., messages that have already been delivered) are minimized. As expected, the overhead ratio is much lower, especially for flooding-based protocols (as shown in Figure 5.5)

Finally, consider the presence of an acknowledgment scheme and back channel access. In other words, if a group member receives a message destined for their group, that member immediately informs all other

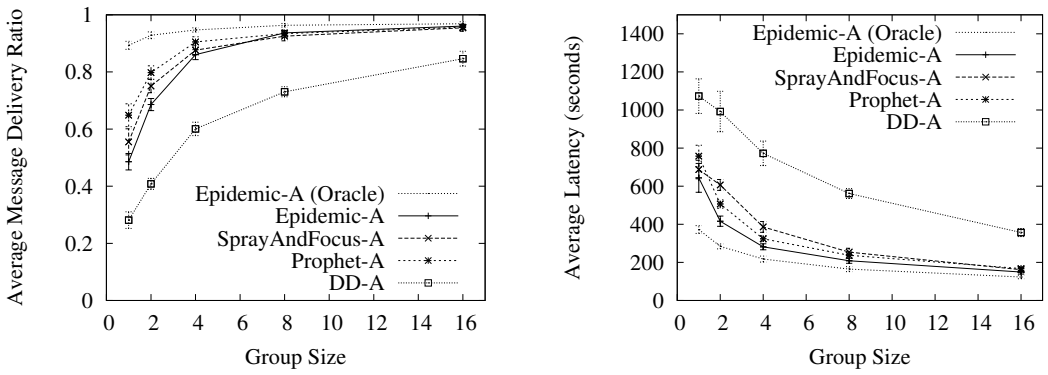

Figure 5.6: RC, ack, back channel (a) MDR, (b) Delay

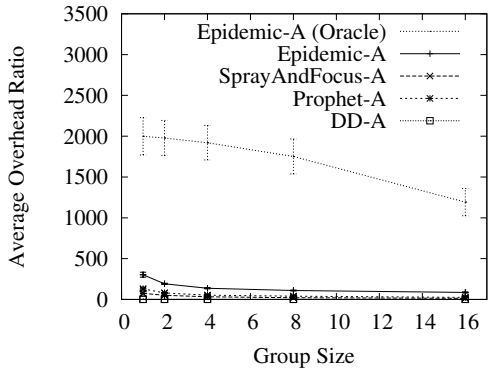

Figure 5.7: RC, ack, back channel - Overhead

group members to transmit ACKs for that particular message. Interestingly, access to a back channel does not bring a significant improvement to any metric, as shown in Figure 5.6. This is because back channel access is expected to have the most effect when group sizes are large (since there are more nodes to initially spread ACKs), but when group sizes are large and ACKs are used, all protocols perform at a high level and hence the back channel ACKs do not help much.

Non-Resource-Constrained

In this set of simulations, the transformed anycast protocols attempt to deliver messages in a non-resource-constrained environment. Message sizes vary between 50k and 100k, and a message is generated by a random node every 50 to 70 seconds, destined to a random group. Furthermore, cars travel much slower (hence, contact time is less of a resource constraint) at speeds of between 2.7 and 3 meters per second, which also has the effect of allowing less node mixing. In addition to the first set, this helps us see how protocols react to different ends of the resource spectrum.

As before, we first consider the scenario where there are no acknowledgments and no back channel

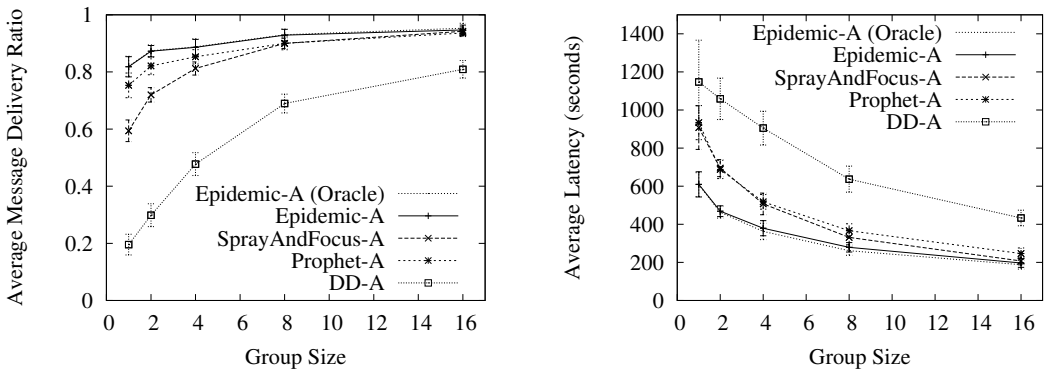

Figure 5.8: Non-RC, no acks, no back channel (a) MDR, (b) Delay

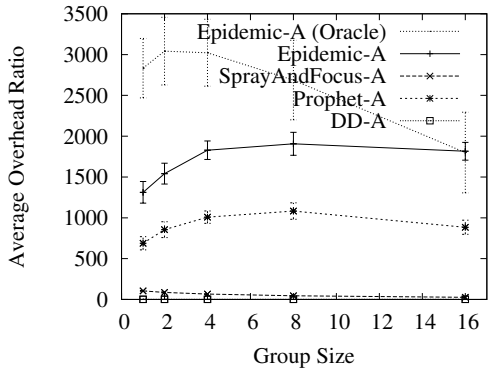

Figure 5.9: Non-RC, no acks, no back channel - Overhead

access (see Figure 5.8). What is immediately clear is that flooding-based protocols perform better than quota-based protocols in both MDR and average delay, particularly when group sizes are small, which mirrors unicast observations. This is because their flooding-based approach is appropriate in less resource-constrained environments, since adding extra messages to the network is more helpful than harmful when the messages are smaller and message production is less frequent. Furthermore, the reduced mobility allows less node mixing, which hinders quota-based protocols since many of the replicas may stay clustered together. One very interesting observation that differs from unicast behavior, is that after group sizes become reasonably large, the MDR difference between flooding and quota based protocols is almost negligible. This is because it is relatively easy to meet groups, even in lower mobility environments, when they are large and hence both types of protocols can perform well. In terms of delay, Epidemic-A performs best, as expected, with the flooding and quota-based protocols performing similarly. However, all are similar with large group sizes (16 or greater). Overhead results are as expected, with Epidemic-A and Prophet-A being the least resource-friendly and Spray and Focus-A being extremely resource-friendly. Therefore, if resources such as battery life or buffer size are severely limited, quota-based protocols are

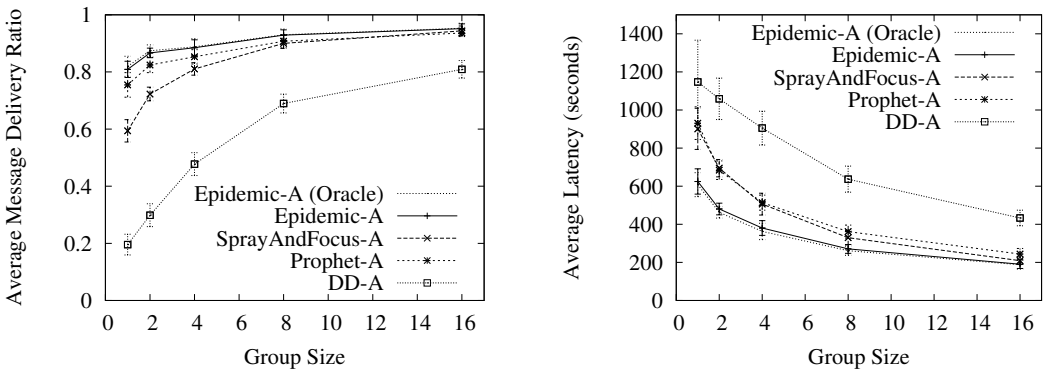

Figure 5.10: Non-RC, acks, no back channel (a) MDR, (b) Delay

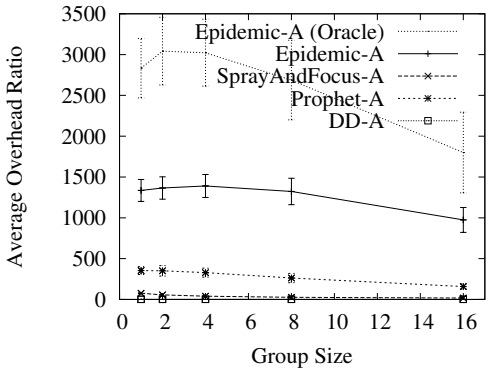

Figure 5.11: Non-RC, acks, no back channel - Overhead

the best choice.

It is important to note that overhead cannot be compared to the first set of simulations, and must be compared only in a relative fashion between protocols in the second set of simulations. Due to the message sizes being much smaller in this second set, many more transmissions per contact occur, meaning the overhead ratios for the protocols are much greater in this set than the first. However, this does not mean that the resource utilization is greater, since the message sizes are much smaller.

The second case shows the results when an acknowledgment scheme is added to the protocols, but there are no back channels. Interestingly, this does not significantly affect either the MDR or the delay metrics, as shown in Figure 5.10. This is due to the low resource utilization to start with, and hence, freeing up the resources does not significantly impact the performance. It does, however, have a very large effect in terms of overhead. Prophet-A and Epidemic-A have their overhead significantly lowered, since many extra and useless transmissions are eliminated. For the same reasons as in the first set of simulations, adding a back channel for each group does not significantly change any of the metrics, as shown in Figure 5.12.

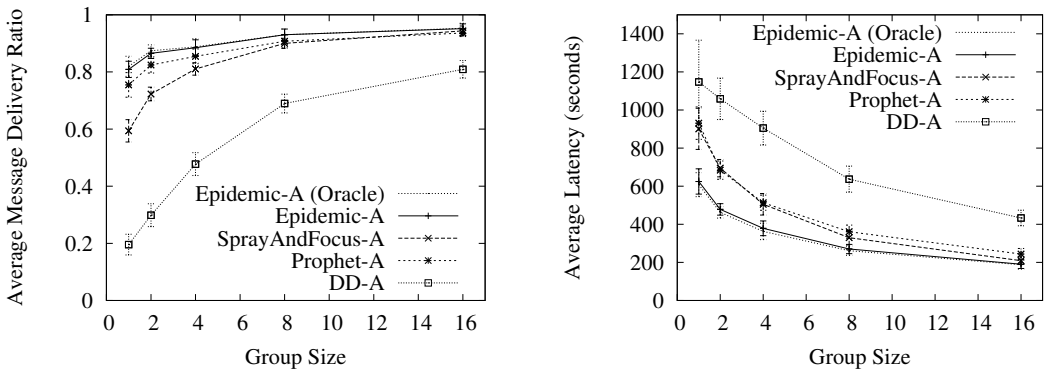

Figure 5.12: Non-RC, ack, back channel (a) MDR, (b) Delay

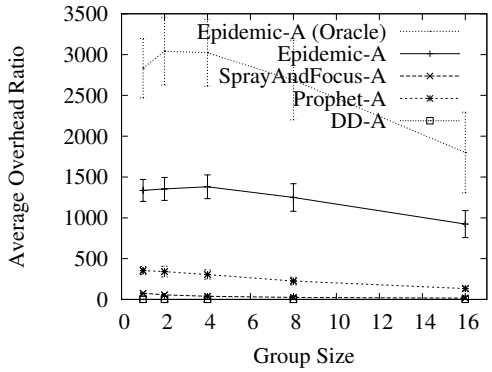

Figure 5.13: Non-RC, ack, back channel - Overhead

5.5 Conclusions and Future Directions

Groups are fundamental entities in many human-centric DTN environments, making effective and efficient anycast communication important. In this chapter, we have proposed an overarching approach that allows many current DTN unicast protocols to be enhanced to support anycast communication. It is shown that this approach can be protocol-independent, and hence is implemented in a thin shim beneath the routing layer, essentially changing the world view of the protocol. Using this approach, we were able to enhance many popular unicast routing protocols, including flooding-based protocols such as Prophet and epidemic, and quota-based protocols such as Spray and Focus. A thorough simulation-based evaluation of these newly enhanced protocols indicates that many factors, such as group size, the presence of an acknowledgment scheme, and the resource-constraints of the environment have a significant impact on anycast performance.

There are many avenues to be explored in future work. First, using the knowledge learned from the evaluation presented, one could specifically create anycast protocols from scratch and evaluate them against

the enhanced unicast protocols. Second, it would be interesting to explore the idea of what we refer to as *preferred anycast*, where the goal is to reach any member of a particular group, but it would be preferred to reach a specific subset of that group. This brings the idea of hierarchy into play.

Chapter 6

Exploring Dynamic Manycast in DTNs

In this chapter, we further support the concept of group-based communication in DTNs by exploring the paradigm of *manycast routing*, where the goal is to meet k members of a group of size m. This very general paradigm inherently includes other group-based routing concepts such as anycast and multicast. Efficiently handling manycast requests at the routing layer allows for both feasible and flexible manycast applications. Our manycast exploration takes a three pronged approach. First, the relative difficulty of manycast requests is quantified via analysis, which greatly deepens our theoretical knowledge of how challenging the general paradigm is in a DTN environment. Second, to understand how different replication-based classes of DTN routing protocols respond to and handle manycast requests, extensive simulations are performed in multiple types of network environments. These results show that any DTN manycast protocol must *dynamically react on a per-message basis* by dynamically changing their routing approach to achieve maximum results. Third, using the conclusions drawn from the analysis and simulation results, we present a DTN manycast meta-protocol that selects the appropriate routing technique based on the current request and network conditions.

6.1 The Importance of Manycast

As the previous chapters illustrate, group-based communication is critically important to human-centric DTNs, since groups are a fundamental element in the structure of human mobility and communication. Unfortunately, unicast communication has so dominated the Internet that group-based communication paradigms, like anycast and multicast, are difficult to implement effectively. However, DTNs have given us a clean slate with a very different underlying structure. Given the intermittent connectivity expected in DTNs, communication is typically supported by some type of message replication, which naturally enables many non-unicast communication paradigms, as shown in the previous chapter. Essentially, with multiple replicas of a message, multiple destinations can be reached, enabling group-based communication, like anycast, multicast and broadcast. However, it is very important not to be biased by the demands of

Internet-based applications when considering group-based communication in human-centric DTNs. Traditional multicast, where all members of a group are guaranteed to receive the same messages, is not only difficult in DTNs, but may not even be the correct approach.

Instead of forcing one communication paradigm on all DTN applications, it is interesting to consider the whole space of communication as captured by the concept of manycast, where the end goal is to reach some k out of the m members of a group. In essence, manycast can be thought of as an umbrella paradigm that spans the space of single-endpoint and group-based communication simply by specifying the size of the group and/or the size of the recipient pool. For example, manycast can be configured to achieve anycast by setting the recipient pool to be one unspecified member of the group (i.e., $k = 1$). Similarly, multicast can be captured by setting k to m. However, these popular extremes only represent two ends of the broad spectrum, and, while useful, are not alone sufficient. Consider a sensor network that needs to collect a statistically significant sample of readings from a group of sensors. Anycasting would clearly be insufficient, and multicasting to the entire group would be extremely inefficient. In this case, the application should have the flexibility to specify the target number of nodes to reach, with the network dynamically responding to meet that specific request.

While the goal of flexible communication opens DTNs to a wide range of new applications, providing efficient manycast in a DTN environment has many challenges. Along with the standard challenges of all DTN communication, such as intermittent connectivity, heavy partitioning, high variance in resource constraints, and the lack of instantaneous end-to-end paths, manycast has the added difficulty of handling two new routing parameters, the target group size and the target number of group members to reach, which can vary with each application request. Additionally, any group-based communication must be integrated with a group management protocol [37], where knowledge of which nodes are in which groups is propagated throughout the network. We have previously explored group management with MembersOnly, described in Chapter 3, which is compatible with our manycast protocol.

To achieve the full potential of manycast in DTNs, it would be beneficial to expose some of these difficulties to an application in a meaningful way that can guide the application in its decision as how best to send its data. For example, if the network is relatively well connected, an application may try to reach more members of a group. To provide this information, it is necessary to understand how "difficult" it is to achieve manycast, in all of its dimensions, in a DTN. Furthermore, understanding the difficulty of a manycast request is a necessary first step towards making routing and replication decisions. Unfortunately, almost all of the existing work on DTN routing has exclusively considered unicast. Of the little work on group-based routing, it has been shown that anycast requires little replication for success [36],

while multicast requires a lot of replication for success [3]. These preliminary results indicate that there is a wide variance in difficulty of manycast requests, depending on the application-specified target number of group members to reach.

The contribution of our research in this chapter can be seen through our three-pronged approach towards the understanding and development of manycast in DTNs. First, we perform an extensive analysis to increase the theoretical understanding of the *fundamental difficulty* of achieving manycast in a DTN environment. By visualizing this space through extensive analysis, we are able to draw conclusions about the difficulty of anycast, multicast, and all points in-between. One of the most interesting observations is that *loose multicast*, where reaching almost all nodes in a group is considered a success, is a substantially easier paradigm than strict multicast, and should probably be considered a core component of group-based routing in DTNs. Second, we perform a simulation-based study that incorporates the naturally challenging DTN environment to understand how these factors, in addition to replication rate, affect the success of manycast communication. We show that quota-based protocols, where replication is strictly limited, perform best when k is relatively small and flooding-based protocols, where replication is not strictly limited, perform best when k is relatively large. Furthermore, we show that the ideal transition point from quota- to flooding-based protocols is highly dependent on the environment, in particular the mobility. From this study, we show that for a manycast protocol to be effective in a DTN environment, it must *dynamically change its routing and replication approach on a per-message basis*. Finally, based on our analysis and simulation results, we develop a DTN manycast meta-protocol that selects the appropriate routing and replication technique based on the current request and network conditions.

6.2 Manycast in DTNs

Manycast is a very general paradigm that spans the space of single- and group-based communication, giving applications a high degree of flexibility in their choice of destinations. A manycast request can be defined using two parameters: m and k. The parameter m is the size of the destination group in the request. This parameter is likely to come from the network itself, or from a distributed group management component running on the network [37], as opposed to the actual application. The parameter k is the target number of nodes to reach in the destination group to satisfy the manycast request. Therefore, an (m, k) manycast request will be successful if a copy of the message is delivered to *at least k of the m nodes in the destination group*. The power of manycast is that it is general enough to include both anycast (i.e., $k = 1$) and multicast (i.e., $k = m$), as well as more traditional routing paradigms such as unicast (i.e., $k = m = 1$)

and broadcast (i.e., $k = m = n$, where n is the number of nodes in the network).

The flexibility of manycast routing opens the door to a rich DTN application space. The inherent support of anycast and multicast alone brings some degree of flexibility to applications. For example, in a DTN used in a disaster zone, individuals in need of help can contact *any* emergency responder instead of a specific one. Furthermore, it is useful to transmit information such as building blueprints to *every* emergency response team leader.

However, the true power of manycast lies in the space between anycast and multicast. Building on the emergency response example, first responders who initially survey a scene may conclude that it is necessary to bring in a certain number of ambulances and/or firefighting vehicles. Manycast would allow these responders to request k of the m available vehicles, while anycast would be insufficient (only requesting one) and multicast would be inefficient (requesting all of them). Revisiting the sensor network example from the previous section, a manycast protocol that could deliver a COLLECT message to at least k of the m sensor nodes would allow for a statistically significant sample to be reached, without the inefficiency of reaching all sensors. Another interesting avenue for DTN applications is security. Contacting centralized trust authorities is very difficult due to the inherent lack of instantaneous end-to-end paths. Manycast would allow an application to contact a subset of distributed CAs, a primitive that is necessary for *threshold cryptography*, allowing a more robust form of trust in the network [60]. Even mobile social networking applications can benefit from the flexibility offered by manycast protocols. For instance, many smart-phones and handheld gaming devices have built-in WiFi and Bluetooth, which can be used for multiplayer gaming when friends are in close proximity. Manycast would enable gaming applications to find k of one's local group of m friends to join a game.

To the best of our knowledge, manycast has not been investigated in the context of a DTN environment. In mobile ad-hoc networks, manycast has been shown to be quite useful [12]. However, in ad-hoc networks, end-to-end paths are assumed to exist much of the time and manycast can be supported through the building of partial multicast trees in the network. Such solutions for ad-hoc networks are not directly applicable to the DTN environment, since there is no expectation of such a high degree of connectivity or stability. Essentially, multicast or manycast trees will likely have a very short lifetime. Therefore, DTN manycast protocols must attempt to leverage replication, which does not require knowledge of the exact route, or even the exact set of nodes, that will receive the message.

Since manycast is a general form of group-based routing, existing work in anycast and multicast for DTNs is also relevant. As we have shown in the previous chapter, anycast is a highly useful and practical routing paradigm for DTNs [36]. While anycast has been considered in wired network scenarios [8, 41],

it has only briefly been explored in DTN environments, where the exploration has been limited to single-copy routing and/or highly constrained mobility [20, 14]. Multicast, the other extreme of the manycast paradigm, has only very briefly been considered in DTN environments. In particular, a simulation-based study that explored how existing protocols handled multicast requests showed that a considerable amount of redundancy was necessary to reach all group members [3], a somewhat unsurprising result that is further confirmed by our work.

In this chapter, we thoroughly explore the general concept of manycast in a DTN environment using both analysis and simulation. Our results span from anycast to multicast and include all points in-between. In particular, our work gives insight into how routing and replication techniques must change from "easy" anycast requests to "hard" multicast requests, and how they can determine where the transition points are.

6.3 Analysis of Manycast Difficulty

The first step in understanding manycast in a DTN environment is understanding how the *difficulty* of a request changes as the parameters k and m change. Since these parameters are highly dynamic, the variance in difficulty of manycast requests is very high. Determining the difficulty of a request is a necessary prerequisite to understanding how to best route and replicate the message. For instance, anycast requests are considered relatively easy, requiring little replication to satisfy [36]. At the other extreme, multicast requests are considered relatively hard, requiring a lot of replication to satisfy [3]. This section thoroughly analyzes the difficulty of all requests, precisely quantifying how that difficulty varies with respect to k and m.

6.3.1 Mathematical Analysis

The fundamental difficulty of a request can be defined in a probabilistic fashion. Given n nodes in the network, a group size m and a recipient pool size of k, $P(m, k)$ is the probability of satisfying a manycast request (m, k). To capture mobility and node contacts, assume a node meets other nodes uniformly at random, and can expect to meet c nodes per time unit. Furthermore, assume messages expire after t time units from creation. To enable our mathematical analysis, we assume routing is done via *direct delivery*, where only the source node ever replicates a message, and only does so to deliver the message to a member of the target group who does not forward it further. This simplistic, but parametrized, system model is assumed for mathematical analysis only. In Section 6.4, we use a more realistic environment for

in depth evaluation.

Problem: Given the previously described system model, compute $P(m, k)$, which represents the *probability that a copy of a generated message successfully reaches at least k of the m destination group members.*

The problem of manycast success is analogous to the following bin-ball problem. Assume there are n balls labeled 1 through n representing the nodes. Since the sender does not count, $n - 1$ is more precise, however this is irrelevant to the computation. Further, assume that balls are picked one at a time, with equal probability, and the label of the picked ball is recorded. Balls are replaced after each pick. An experimenter has a total of $c \cdot t$ picks, since this is the total number of non-unique nodes the source node can expect to meet before the message expires. There is one target group with m members. Assume, without loss of generality, that the destination group members are the first m balls and have the labels 1 to m. Therefore, after $c \cdot t$ balls have been picked, we want to determine the probability that the experimenter sees *at least k unique* balls with labels less than or equal to m.

As a quick example, assume $n = 3$, $ct = 2$, $k = 2$, and $m = 2$. All possible sets of picks include (1,1), (1,2), (1,3), (2,1), (2,2), (2,3), (3,1), (3,2), (3,3). Since $k = 2$, of these sets, only $(1, 2)$ and $(2, 1)$ meet the requirements for success. Therefore, the difficulty of this request, under the described system parameters, is $\frac{2}{9}$. As another example, consider $n = 4$, $ct = 3$, $k = 2$, and $m = 3$. If we work through this case, we obtain a difficulty of $\frac{42}{64}$, demonstrating how fast the space explodes: the number of possible sets is n^m.

Solving the whole problem of determining the probability that after ct picks, one sees at least k labels less than or equal to m is quite complex. To make the problem more tractable, we divide it into two steps: (1) given ct, the chance of getting exactly u unique picks (referred to as $f(ct, u)$), multiplied by (2) given u unique picks, the chance of seeing at least k values less than or equal to m (referred to as $g(u, k, m)$). These two steps are iterated over all reasonable values of u, which range from k, since anything below k unique picks cannot result in success, to the minimum of n and ct, since the number of unique picks cannot exceed either the total number of balls or the total number of picks. Therefore, P is defined in terms of f and g as follows:

$$P(m, k) = \sum_{u=k}^{min(ct,n)} (f(ct, u) \cdot g(u, k, m)).$$

Recall that f captures the chance of getting exactly u unique values given ct picks. There are two ways this can occur: (1) the first $ct - 1$ picks contain the u unique values needed, and so the last pick must be a duplicate, or (2) the first $ct-1$ picks contain $u-1$ unique values, and so the last pick must be unique. Note that the chance of the last pick being a duplicate if there are already u unique values is $\frac{u}{n}$. Similarly, the

chance of the last pick being unique if there are already $u-1$ unique values is $1-\frac{u-1}{n}$. We can therefore write f using the recurrence relation

$$f(ct,u) = f(ct-1,u)\cdot\frac{u}{n} + f(ct-1,u-1)\cdot(1-\frac{u-1}{n}).$$

The initial conditions of the recurrence are as follows. If there are any picks, there must be at least one unique pick, hence $f(ct,0) = 0$. If there is one pick, there must be exactly one unique value, hence $f(1,1) = 1$, and $f(1,u) = 0$ if $u \neq 1$.

Next, recall that g captures the chance of seeing at least k values less than or equal to m, given u unique picks. Seeing *at least* k values means seeing exactly k values or seeing exactly $k+1$ values or seeing exactly $k+2$ values, etc, up to seeing exactly m unique values less than or equal to m. We therefore introduce another variable, l, that ranges from k to m, and focus on computing the probability of seeing *exactly* l values less than or equal to m. This turns out to be a relatively simple counting problem. We first count the number of ways to see the l values less than or equal to m, and then count the number of ways to have the rest of the values greater than m. This is then divided by the total number of possible label combinations. Putting this all together, we define g as follows:

$$g(u,k,m) = \sum_{l=k}^{m} \frac{\binom{m}{l}\binom{n-m}{u-l}}{\binom{n}{u}}.$$

This completes the definition of $P(m,k)$, representing the difficulty of a manycast request to reach at least k nodes out of a group of size m. To help visualize the function, we implemented it in MATLAB, as described in the next subsection.

6.3.2 MATLAB Computation

To explore how manycast difficulty changes with varying request parameters, we implemented P in MATLAB and visualized the results over a wide range of system and user parameters. Memoization was used to both speed up and cut down on the memory consumption of the recursively defined f function.

The goal of this analysis is to understand how difficult it is for the target number of group members, k, to receive a message for a group size m. It is also important to understand how this difficulty changes as one or both of these parameters change. To capture how P varies with varying values of m and k, the results are presented as 3D graphs. The two control variables are m, which ranges from 1 (e.g., unicast) to the total number of nodes in the network (e.g., broadcast), and k, which ranges from 1 (e.g., anycast) to m (e.g., multicast). The z axis represents the probability of success, or $P(m,k)$. All graphs incorpo-

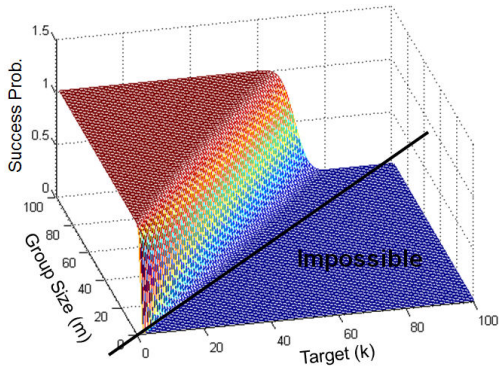

Figure 6.1: $P(m, k)$ for 100 Node Network

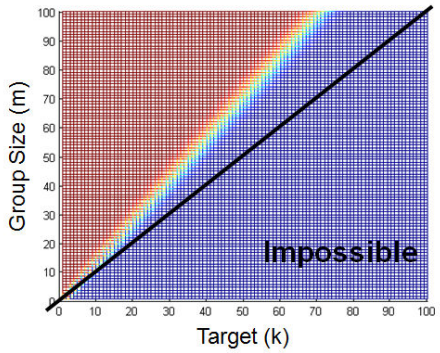

Figure 6.2: Top-Down View (100 Nodes; 2 Hour Expiration)

rate color to improve the quality of the presentation; however, we will explain how to read the gray-scale versions.

As a first step, we look at a moderately sized network, $n = 100$, with reasonable encounter rates, $c = 1$ *per minute*, and reasonable message expiration times, $t = 2$ *hours*. Starting at $k = 1$ and increasing to around $k = \frac{2}{3}m$, there are a relatively large set of values where the success probability is close to 1. Essentially, these requests should be relatively easy to satisfy (see a 3D representation of the success probability in Figure 6.1). There is then a somewhat narrow transition point where the success rate falls to values close to 0. This transition point is interesting, since this is where routing techniques may have to change to provide the more challenging levels of manycast. When k is close to m, there is another relatively large portion with values close to 0, indicating that these requests are relatively difficult to satisfy. Finally, there is a large zone labeled "Impossible", where $k > m$. These requests are impossible to satisfy, since one cannot deliver a message to $k > m$ group members if there are only m members in the group.

One of the most interesting features of these 3D graphs is the transition from easy to difficult. To better compare where these transitions fall, we also present 2D top-down graphs using color to indicate the

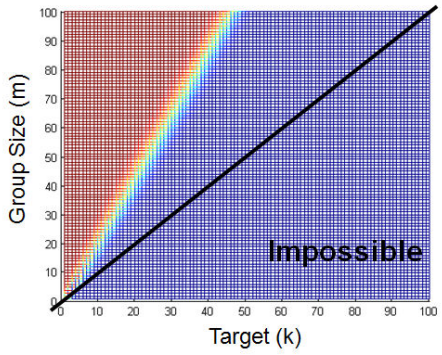

Figure 6.3: Top-Down View (100 Nodes; 1 Hour Expiration)

third dimension. These can be thought of as heat maps, where red indicates values closer to 1 and blue indicates values closer to 0. For example, Figure 6.2 is top-down view of the graph in Figure 6.1. For the gray-scale versions of these graphs, the upper left portion are the red values close to 1, the dividing lines are the transitional areas, and the lower right portion are the blue values close to 0.

Two interesting regions are the "slices" where $k = 1$ (along the y-axis in Figure 6.2), representing anycast requests, and $k = m$ (along the diagonal in Figure 6.2), representing multicast requests. As expected, anycast is close to 1 (red) and so can be satisfied very easily as long as m is not too small, while multicast is almost always 0 (blue) and so very difficult to satisfy unless m is very small.

Two interesting observations can be made about the transition from high delivery probability (left-red) to low delivery probability (right-blue). First, the transition happens relatively quickly, as indicated by the thin white band. This means that if the difficulty of an application's request is close to the transition point, it can increase its success drastically if it is willing to decrease k slightly. Second, the transition line is seemingly linear in nature. This means that if m decreases (e.g., nodes leave the group), then to keep a similar level of success, k must decrease proportionally. Hence, if the slope of the transition line is known, applications can adjust their requests accordingly when a group size changes without actually knowing the exact group size.

Message lifetime obviously has an impact on the success of a manycast request. We evaluate this effect by using using a message expiration time of 1 hour, essentially reducing $c \cdot t$ by one half. Since nodes now have a shorter amount of time to deliver messages, this increases the difficulty of all requests. All other system parameters remain the same. The result is a shift in the transition line, as shown in Figure 6.3. In essence, changing c or t results in a change in the slope of the transition line. Therefore, some requests that had a high probability of success changed to having a very low probability of success, indicating that message expiration time is a critical factor in determining success. Interestingly, the width and linearity

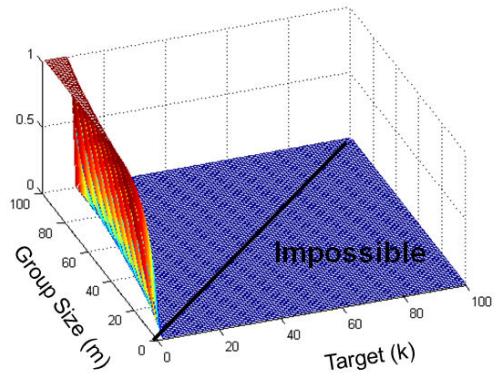

Figure 6.4: Very Quick Expiration (10 Minutes)

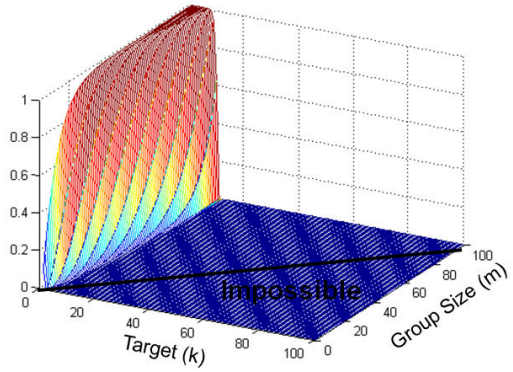

Figure 6.5: Very Quick Expiration (10 Minutes)

of the slope remained relatively unchanged. Since some applications would like to be able to predict the success probability of a given request within a given lifetime, we can also use these evaluations to find the minimum value of t, for a given k and m, that would result in a high success probability.

We have looked at manycast in a network with reasonable system parameters. However, it is also interesting to consider how manycast performs with extreme system parameters. Therefore, P has been reevaluated for very small and very large values of ct. Consider first a very small value of t, namely *10 minutes* (see Figure 6.4). As expected, the transitional region has shifted very close to the anycast "slice", indicating that, unless k is quite small, requests in general have a low chance of success. Perhaps more interesting, though, is that the linearity of the transitional region breaks. Instead, it seems more exponential in nature. This implies that even with a large value of m, k must be small to have a reasonable chance of success. To better illustrate what is occurring at lower values of m, Figure 6.5 is a rotated version of Figure 6.4. From this view, it can be seen that even anycast requests (e.g., $k = 1$) have a very low chance of success when m is small.

On the other extreme, consider a very large value of t, namely *5 hours*. The top-down view of this

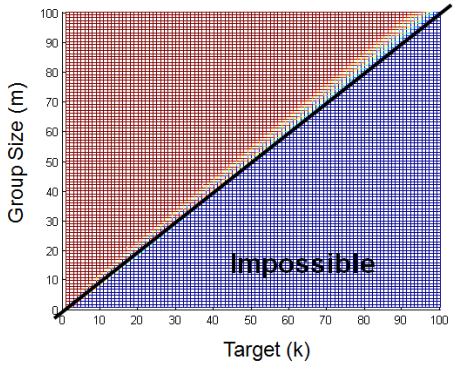

Figure 6.6: Very Long Expiration (5 Hours)

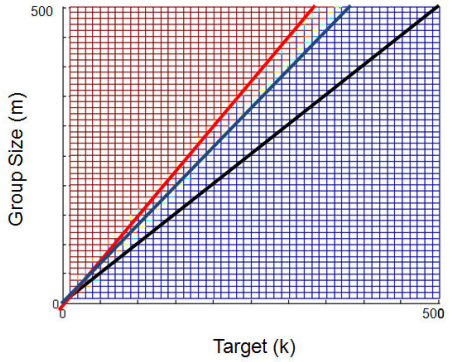

Figure 6.7: 500 Node Network

graph, shown in Figure 6.6, clearly indicates almost all requests can be satisfied with a high degree of certainty. However, it is interesting to note that multicast requests (e.g., $k = m$) still have a low probability of success, particularly when m is large. This further confirms that the multicast paradigm is simply too hard to satisfy in DTN environments. In fact, as the graph indicates, it is much easier to satisfy "almost all members of a group" than "all members of the group". We refer to the *almost all* paradigm as *loose multicast*, and will further show via simulation that loose multicast is substantially easier than strict multicast.

Finally, to understand the impact of network size, we evaluate a large network of 500 nodes. For this network, the contact rate is set to 2 nodes per minute and messages expire after 5 hours. The resulting graph, shown in Figure 6.7, further confirms a linear, thin transition line. As a visual guide, we have included solid lines indicating the left and right edges of the transition.

In conclusion, analyzing P for varying values of m and k, as well as with different system parameters, can lead to many interesting and useful observations. Some of the more prominent ones include: (1) the clear division of very high and very low probability regions, indicating the need for routing protocols to

dynamically shift their approach based on the application request and (2) a dramatic increase in success if the application is willing to relax requests that fall close to the transition line. To gain a better understanding of how real protocols in more realistic environments handle manycast requests, the following section continues the discussion of the difficulty of manycast in a simulation environment.

6.4 Simulation Study of Manycast

The difficulty analysis presented in the previous section gives insight into the relative difficulty of an (m, k) manycast request, which in turn provides guidance on routing-level decisions such as replication. This section incorporates realism into the equation by studying how effective different classes of DTN routing protocols are and how different types of mobility factor in. For this evaluation, a popular DTN simulator called the Opportunistic Network Environment (ONE) simulator is used [29].

6.4.1 Evaluation Criteria

To gain a broad understanding of manycast performance, two major components are explored: routing mechanisms and mobility.

Although designed for unicast communication, many of the current unicast protocols use mechanisms that can support manycast. One of our goals is to understand which routing mechanisms can best be used for manycast and so can be integrated into our target manycast protocol. Based on the level of replication used, these mechanisms can be divided into four classes: direct delivery, quota-based, flooding, and epidemic. *Direct delivery* is the most basic form of DTN routing, where a node simply carries around messages it sources until the destinations are directly met. No forwarding ever occurs, and hence this can be considered the most resource-friendly protocol. *Quota-based* protocols allow limited forwarding and replication to improve delivery, but reduce resource usage (e.g., Spray and Wait [51], Spray and Focus [52], and Encounter-based Routing (EBR) [34]). Essentially, every sourced message contains a quota, which is a hard limit on the number of replicas of the message allowed in the system. This is enforced by decreasing the quota of a message upon replication. Next, *flooding-based* protocols take advantage of abundantly available in-network storage and are allowed to freely replicate to any or all contacts, without limit (e.g., Prophet [32], MaxProp [10], and RAPID [4]). These protocols work well in highly disconnected environments; however, they can quickly overwhelm resources in resource-constrained environments. Finally, while technically a flooding-based protocol, Epidemic routing [55] attempts to replicate all messages to all nodes in the network. This is a popular protocol due to the fact that it is optimal, in terms of delivery ra-

tio and latency, if there are no resource constraints in the network. This protocol can be improved upon by smart buffer management techniques [45].

To properly evaluate how these protocols handle manycast requests, we choose to implement (or use existing implementations in the simulator) one protocol per class as a representative of that routing class: Direct Delivery, Spray and Focus, Prophet, and Epidemic. We also implemented a "group-based" version of these protocols, where destinations are groups, not individual nodes. We adapted the utility functions utilized by the protocols to capture group utility instead of node utility. This is done by having members of the same group "look" like the same node from the perspective of utility functions in the routing protocols. In other words, groups look and act like virtual nodes. We updated the utility functions used in the routing protocols for a particular group whenever a node meets a group member. The protocol labels in the results are appended with "-G" to further emphasize this.

The second evaluation criteria is mobility. In our analysis from the previous section, we assumed a very simple connection model, where a node had an equally likely chance of meeting any other node at any time. Simulation allows us to understand manycast in a wider range of mobility patterns. There are two main types of mobility that are critical to the understanding of DTN routing: unstructured mobility and structured mobility. By unstructured mobility, we mean that there is very little actual structure that can be extracted from the movement patterns of nodes (i.e., random waypoint and random walk [11]). Many DTN unicast protocols are analyzed by their performance in these types of unstructured mobility. For instance, the binary quota distribution technique used by Spray and Wait has been shown to be optimal in random mobility [51]. While less realistic, this type of mobility is generally easier to analyze. On the other hand, structured mobility can be thought of as mobility patterns that generally arise from nodes that follow different types of movement patterns, possibly related to their environment. For instance, in a disaster response scenario, emergency responders may be moving towards an event, civilians may be fleeing from it, and ambulances may be oscillating to and from it [35]. Another example is a community network, which could be composed of pedestrians, cars, and trams [16]. Structure from these networks (i.e., popularity) can be extracted and exploited for routing purposes [34, 18]. To explore manycast in both types of environments, our simulations use both random waypoint as well as the built-in community model of the ONE simulator.

6.4.2 Simulation Setup

The goal of our simulations is to understand how manycast requests perform under various classes of routing protocols and various types of DTN environments. Simulations are divided into two main classes re-

lated to the mobility pattern: unstructured and structured. The unstructured environment is random waypoint, with each node moving at a speed between 1 and 10 meters per second and waiting at the waypoint for a random period of time between 0 and 2 minutes. The structured environment is the built-in community mobility model, which places pedestrians, cars, and trams on a real map of Helsinki, Finland. Pedestrians walk at a speed of 0.5 to 1.5 meters per second, cars travel at a speed of 2.7 to 13.9 meters per second, and trams travel at a speed of 7 to 10 meters per second. These nodes follow intuitive routes to and from local hot-spots. The total map size for the random waypoint mobility model is 3.5km x 3.5km, while the structured mobility model is 4.5km x 3.4km. Within each of these two classes, we explore how the routers react in small groups (where $m = 16$) and larger groups (where $m = 32$). Each graph contains results from each of the aforementioned routing protocols, with the x-axis being the target number of nodes to reach (k), ranging from 1 (anycast) to m (multicast).

The total number of nodes in each simulation is 126. In the structured mobility model, there are 80 pedestrians, 40 cars, and 6 trams. Each node has a communication range of 100m, transmits at 256kbps, and has a buffer size of 5MB, except trams which have a communication range of 1000m, transmit at 10Mbps, and have a buffer size of 50MB. Messages are generated randomly by every node every 50 to 70 seconds, with a size randomly chosen between 500kB and 1MB. This setup allows for a somewhat resource-constrained environment. Each simulation is run for 4000 seconds and each data point is the average of 10 runs and includes a 95% confidence interval.

Simulations are evaluated using both group-based message delivery ratio (MDR) as well as group-based latency. MDR is defined as the number of successfully completed manycast requests (e.g., the message reached at least k of the m nodes) divided by the total number of manycast requests. The *Average MDR* is the average of each node's MDR. Latency, or delay, is defined as the time from message source until the time that the k^{th} node of the group received the manycast message. *Average delay* is the average of all message delays in the network. Note that a message can only have a delay if it was successfully delivered, and hence this metric should be viewed only in relation to the average MDR. If two protocols have widely differing average MDRs, then the average delay is less meaningful because intuitively, a high MDR usually means delivery to hard-to-reach groups. For this reason, we consider average MDR to be the *primary metric of evaluation* and the average delay to be the *secondary metric of evaluation*.

6.4.3 Structured Mobility

We first present results from our *structured mobility model*, specifically the community mobility model built into the ONE simulator. Within this class, we first consider a group size of 16. The first major ob-

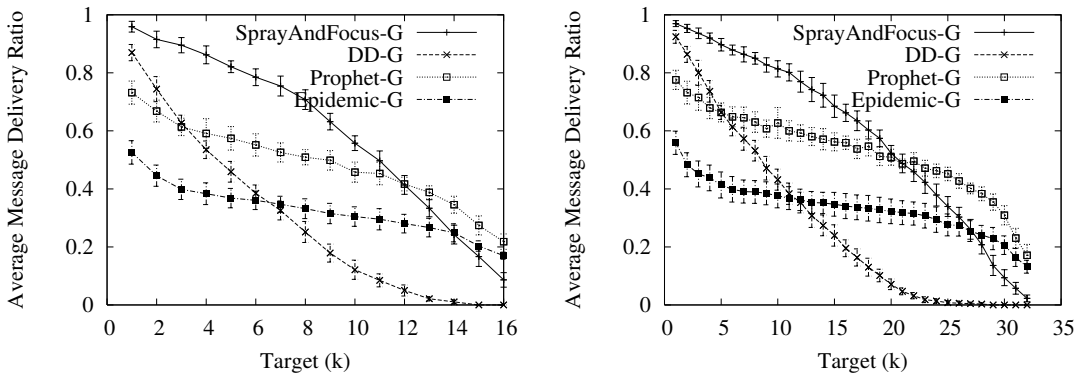

Figure 6.8: MDR - Structured Mobility: (a) 16 Node Groups, (b) 32 Node Groups

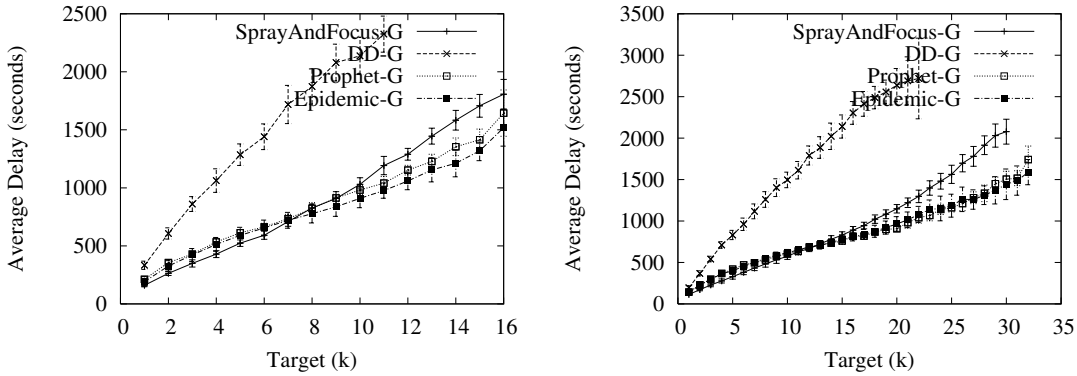

Figure 6.9: Delay - Structured Mobility: (a) 16 Node Groups, (b) 32 Node Groups

servation, as seen in Figure 6.8(a), is that no single protocol dominates all values of k in terms of message delivery ratio. This result immediately confirms that an efficient manycast protocol must dynamically shift techniques depending on the individual request. When $k < 8$, Spray and Focus clearly obtains the best performance; however, when $k > 11$, Prophet is superior. Note that the downward slope of Spray and Focus is greater than both Prophet and Epidemic. This exposes an interesting feature of quota-based protocols, in that they can be considered more risky than flooding-based ones. Essentially, quota-based protocols can perform very well when the target number of nodes to meet is relatively small. Limiting the number of replications keeps resources from being depleted, which can lead to message drops and missed contact opportunities, yet may still be sufficient for reaching the target number of nodes. On the other hand, such quota-based protocols perform very poorly when the target number of nodes is relatively large, since limiting the number of replications does not get the message out quickly enough to a large fraction of the network.

The results found in Figure 6.8(a) can be broken down further by considering four different regions, which we refer to as regions A, B, C, and D. Viewing results such as these in terms of discrete regions

hints at how a dynamic manycast protocol can be developed, which is explored in Section 6.5. We define region A as the region where Direct Delivery and quota-based protocols are the top performers. It can be seen that region A includes $k = 1$ (and hence anycast requests) and $k = 2$. Region B is defined as the region where quota-based protocols alone are superior. This region includes values of k from 3 to 9, in this figure. Region C is defined as the region where quota-based and flooding-based protocols are best. Hence this can be considered the region where k ranges from 10 to 13. And finally, region D is defined as the region where flooding-based protocols are dominant over all others. This includes values of k from 14 to 16 (and hence includes multicast requests).

It is important to comment on the behavior of pure epidemic routing. While epidemic routing is considered optimal when there are no resource constraints, it has been shown many times before that its performance is severely hindered when bandwidth, buffer size, and contact duration are limited [34, 45, 32, 36]. Our results further confirm this behavior for manycast.

When the group size is increased to 32, as shown in Figure 6.8(b), the characteristics of the graph stay the same. Primarily, the point at which flooding-based protocols overtake quota-based protocols stays in proportion to the group size. This is actually quite a significant observation since it provides further evidence that to keep the same success ratio, k must be increased proportionally to the increase in m. Recall that this behavior was seen as a linear transition line in the MATLAB evaluation. To be clear, in Figure 6.8(a) (when $m = 16$), the Spray and Focus MDR crosses the Prophet MDR at around $k = 11$; in ratio form, this is $\frac{11}{16} = 0.6875$. In Figure 6.8(b) (when $m = 32$), the two cross at around $k = 22$; in ratio form, $\frac{22}{32} = 0.6875$. Hence, the crossing point for quota-based and flooding-based protocols seems to occur in constant proportion to the group size.

Another interesting observation, when $m = 32$, is the relatively sharp drop-off as k approaches m. This further confirms the difficulty of multicast in DTNs, and gives support for the theory that applications willing to relax multicast requests will experience significantly higher success ratios. The relative ranges covered by regions A, B, C, and D, in relation to the group size, can be considered the same as with $m = 16$, due to the similar crossing points. Hence region A contains $1 \leq k \leq 4$, region B contains $5 \leq k \leq 18$, region C contains $18 \leq k \leq 26$, and region D contains $27 \leq k \leq 32$.

In terms of average delay, it is clear that Direct Delivery is substantially worse than the other protocols for all cases except anycast, where $k = 1$, as shown in Figures 6.9(a) and 6.9(b). This is because messages are carried only by the source nodes, and hence the source node itself would have to meet all k of the target nodes. It is interesting to note that while the resource-friendly property of Direct Delivery can help its average MDR in resource-constrained environments, it will not help its average delay. Therefore,

if delay is a critical factor for the application, a Direct Delivery routing protocol would be a poor choice. Another interesting observation is that, as noted with MDR, the average delay characteristics are similar between small and large groups. The other three protocols are relatively similar until k gets large. When $k \approx \frac{2}{3}m$, Spray and Focus starts to diverge. This reinforces the idea that flooding-based protocols perform best when k approaches m.

6.4.4 Unstructured Mobility

The second set of results evaluates manycast under *unstructured mobility* using the random waypoint mobility model. As before, we first present results where group sizes are relatively small, namely 16 nodes. In contrast to the previous results, Spray and Focus consistently performs at the highest level, as shown in Figure 6.10(a). This is due to unstructured, random mobility allowing message replicas to spread better throughout the network [51]. In structured mobility environments, protocols that limit replication have to deal with the possibility that most of the replicas will stay in a relatively local area. However, in unstructured, random mobility environments, nodes tend to have a higher degree of mixing. For this same reason, Direct Delivery also performs at a high rate for a longer period of time. Overall, this leads to the interesting observation that limiting replication is most beneficial to networks whose nodes mix well with one another. It is also worth noting the relatively sharp drop-off for Spray and Focus and Prophet from $k = 15$ to $k = 16$. This illustrates the difficulty of multicast in DTN environments.

In terms of dividing the figure into regions, there is no point where flooding-based protocols are convincingly better than quota-based protocols. Therefore, we can divide the graph into 3 regions, eliminating region D. Region A includes $k = 1$ and $k = 2$, where Direct Delivery and Spray and Focus both perform at a high level. Region B includes $3 \leq k \leq 14$, where Spray and Focus has a clear dominance over all other protocols. And finally, region C includes $k = 15$ and $k = 16$, where Spray and Focus as well as Prophet perform well.

With a larger group size, namely $m = 32$, the most interesting feature is the sharp drop-off in MDR as k approaches m, as seen in Figure 6.10(b) as well as the other MDR figures previously presented. This common thread indicates that *loose* multicast, where applications are satisfied if almost all of the group is reached, will have a much greater chance of success than strict multicast. It is therefore advantageous for DTN applications to accept and make use of loose multicast if they want to significantly improve their message delivery ratios. Note that, in our simulations, Epidemic is never superior to Prophet, since Prophet is better at managing resources efficiently.

To capture the trends, this figure can be divided to four regions, since there is a clear point when flooding-

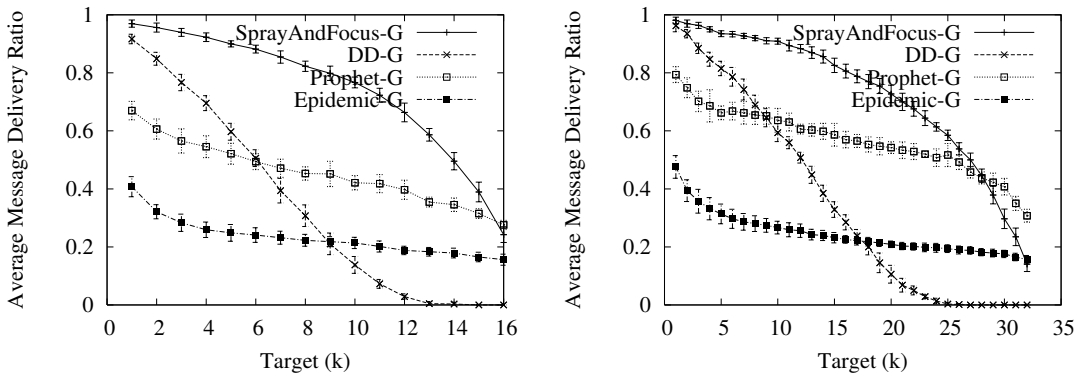

Figure 6.10: MDR - Unstructured Mobility: (a) 16 Node Groups, (b) 32 Node Groups

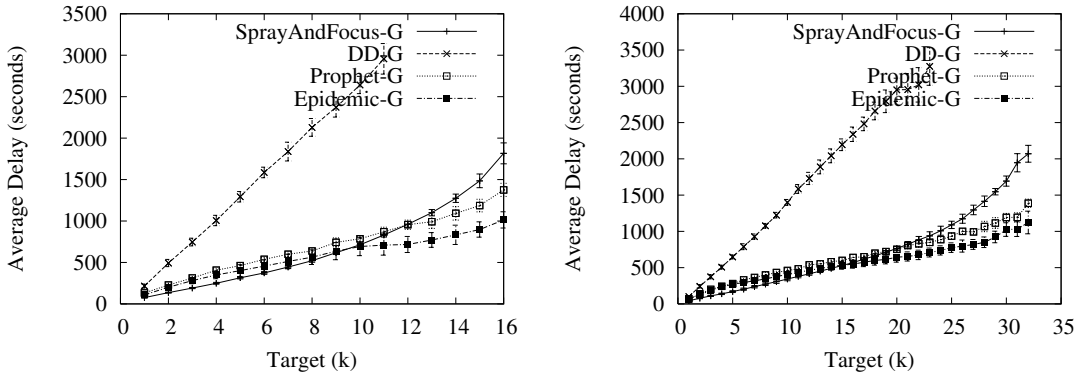

Figure 6.11: Delay - Unstructured Mobility: (a) 16 Node Groups, (b) 32 Node Groups

based protocols perform best. Region A can be viewed as the region where $1 \leq k \leq 3$. Region B contains the region where $3 \leq k \leq 24$. Region C can be defined as $25 \leq k \leq 29$. Finally, region D includes $30 \leq k \leq 32$.

The average delay trends of the protocols in the unstructured environment are similar to that of structured environments. As seen in Figures 6.11(a) and 6.11(b), Direct Delivery incurs the largest average delay by far for all cases except anycast. All of the protocols have a somewhat linear trend until k approaches m. Prophet, Epidemic, and Spray and Focus all quickly increase as k approaches m, with Spray and Focus being the most pronounced. This further emphasizes the difficulty of multicast requests, and strongly suggests that applications consider loose multicast.

6.5 A Manycast Meta-Protocol

Given the trends exposed in our evaluation, it is clear that no one protocol mechanism performs best all of the time. Therefore, a successful manycast solution for DTNs should be dynamic and change replica-

tion techniques *per request* as necessary to achieve the best performance. In this section, we present a discussion of general guidelines that can be used for handling requests, and build a manycast *meta-protocol* framework based on the observations from the previous sections. Essentially, the goal of this meta-protocol is to select a protocol from the appropriate replication class that maximizes the average message delivery ratio. This can be thought of as "staying on top of the curve".

There are three main factors to consider when deciding whether to use no replication, little replication, or a lot of replication. Additionally, these factors change with every distinct request, and hence must be re-evaluated per request. The first factor is the target number of nodes, k, of the request. If k is small, less replication is necessary to achieve success. If k is large, more replication is necessary. The second factor is the network and group characteristics. If the mobility of the network is structured or nodes do not mix evenly, then quota-based protocols may have a harder time properly distributing replicas. In this case, more replication may be necessary. On the other hand, if the mobility of the network is unstructured, where nodes mix relatively evenly, quota-based protocols are sufficient in many cases. Furthermore, the group size of the request's destination group will influence the decision. While not directly explored in this chapter, resources such as battery life also fall into the "network characteristics" property. If battery life is a major constraint, then less replication is desirable. The third factor is the application's tolerance to delay. This factor is dependent on the request and, hence, will change per request. If low delay is important, then Direct Delivery should never be favored.

Using these observations, a general framework for routing manycast requests can be constructed. Recall from Section 6.4 that the network and group characteristics, the second factor in our previous discussion, can be used to break the range of k into four regions. If k falls in region A, Direct Delivery or quota-based protocols can be used. If k falls in region B, quota-based protocols alone are superior. If k falls in region C, quota-based or flooding-based protocols can be used. And if k falls in region D, flooding-based protocols are preferred. Therefore, the meta-protocol will take the following steps:

1. Divide the k range into four regions based on the network and group characteristics: A, B, C, and D (note that some regions may be empty, such as region D as shown in Figure 6.10(a))
2. If the request is time-sensitive, eliminate region A, and extend region B to cover it
3. Consider the target number of nodes, k, and determine which region the request falls in
4. Select a routing protocol from the appropriate class based on the region

In a more algorithmic form, a general skeleton for the dynamic manycast protocol can be seen in Algorithm 2.

Algorithm 2 Dynamic Manycast Meta-Protocol

$m \leftarrow size(request.destGroup)$
$regions \leftarrow getRegions(networkState, m)$
if $request.timeSensity$ **then**
  $regions.B = regions.B \cup regions.A$
  $regions.A = EMPTY$
end if
$reg = whichRegion(regions, request.k)$
if $reg == A$ **then**
  $protocol = selectFromClass(DD \cup QUOTA)$
else if $reg == B$ **then**
  $protocol = selectFromClass(QUOTA)$
else if $reg == C$ **then**
  $protocol = selectFromClass(QUOTA \cup FLOODING)$
else if $reg == D$ **then**
  $protocol = selectFromClass(FLOODING)$
end if
return *protocol*

This algorithm can help a node decide which low-level protocol to use for routing a specific request. The algorithm should be run only at the source node. Any intermediate nodes simply route the message based on the protocol originally selected by the source node. Therefore, the overall process would be as follows. First, an application generates a manycast request. The meta-routing protocol at the source selects a low-level routing protocol to use for the request. Finally, the network routes the request, using the low-level protocol originally decided on by the source meta-routing protocol.

6.6 Conclusions and Future Directions

In this chapter, we have explored the concept of *manycast* routing, where an application desires to reach at least k of m members of a group, where m is the group size. This very general paradigm inherently incorporates more specific group-based paradigms such as anycast and multicast. Through thorough analysis and simulation, we have quantified the difficulty of manycast requests in relation to one another, and illustrated the need for a dynamic manycast protocol that changes techniques on a per request basis. Utilizing these discoveries, we demonstrated a practical approach to manycast routing by using a meta-protocol to appropriately select a low-level routing protocol based on network factors and the specific request.

There are many interesting future directions in regards to group-based communication and manycast. First, it would be interesting to understand how different DTN protocols interact with each other while running simultaneously. Our results from this chapter show that a dynamic manycast protocol is necessary to change the replication rate on a per packet basis. Taking this a step further, one could thoroughly explore how the replication decisions from one request affect the delivery rate and other metrics of sub-

sequent requests; in other words, explore the interplay between requests that are routed using different routing techniques. Furthermore, it would be beneficial to extend our results to include resources such as battery life, which will force a new trade-off regarding replication. Finally, an implementation of our meta-protocol, and its subsequent exploration on a live testbed such as DieselNet [62] would be a necessary step towards practical deployment.

Chapter 7

Conclusions and Future Directions

Human-centric DTNs are critical to supporting communication in environments where infrastructure cannot be, or should not be, relied on. As mobile devices become more ubiquitous, we expect the potential for multi-hop communication through these devices to be high. However, the lack of efficient unicast and group-based communication hinders their wide-spread use. In this thesis, we have identified and contributed to five components that will help facilitate the adoption of human-centric DTNs. In particular, we have shown that by taking advantage of the structure inherently found in these networks, traditional forms of communication can be efficiently performed in DTNs, and new group-based communication can be practically realized. First, we introduced two tools that help us understand and lay the foundation for advanced routing protocols. These include a high-level mobility model for disaster recovery networks and a local and robust group management system, capable of distributing group information accurately, even in untrustworthy environments. Second, we progressed the state-of-the-art in DTN routing by proposing three routing protocols and techniques to allow for efficient and effective unicast, anycast, and manycast.

In this thesis, we have presented five components that together progress the state-of-the-art in DTN communication. In addition to the individual future directions presented at the end of each chapter, there are many general future directions that affect all of the components. We now identify and briefly discuss three prominent ones: (1) evaluation of these protocols, and others, working together in the same environment, (2) merging together DTN protocols to more connected environments, and (3) implementation of the components.

First, future DTNs will likely have multiple routing paradigms, such as unicast and manycast, as well as multiple routing protocols within the same paradigm, operating simultaneously. Hence, individual protocols should work cooperatively to achieve maximum performance. Since common resources, such as storage and bandwidth, are limited in human-centric DTNs, these protocols should be aware of current resource levels and adjust their replication rates accordingly. Techniques from Thompson et al. [54] present a first approach on monitoring local resource availability, and merging these ideas into the protocols presented in this thesis would allow for a more cooperative environment.

Second, with the prominent increase in mobile, wireless devices, it is almost a certainty that future environments will include highly connected and highly disconnected nodes in the same network. Therefore, techniques to bridge DTN protocols with MANET routing protocols [44, 28, 2], particularly those that make use of in-network storage such as CNF [42], are a necessity. An initial investigation into this topic was presented by Ott et al. [39], where DTN was used to extend AODV functionality. While this is a good first step, future protocols must take into account the numerous challenges associated with hybrid networks, such as varying link quality, vastly different levels of connectivity, and devices having multiple network attachment points.

Third, the components presented in this thesis have been designed and evaluated using analysis and simulation. It would be very useful to implement some of these components, particularly MembersOnly and the routing protocols. Implementation would allow us to more thoroughly evaluate the characteristics of the protocols, and could be done using software routing techniques such as CLICK [30] and XORP [59]. With the recent increase in wireless and DTN testbeds, such as ORBIT [46] and DieselNet [62], the protocol implementations would allow for much more realistic characterizations than simulation can give. The ultimate goal would be for the protocols presented in this thesis to have production-level code associated with them, and can hence be deployed.

References

[1] ns2 network simulator. http://www.isi.edu/nsnam/ns/.

[2] Optimized link state routing protocol (OLSR), rfc3626, 2003.

[3] M. Abdulla and R. Simon. A simulation analysis of multicasting in delay tolerant networks. In *Proceedings of Winter Simulation Conference*, 2006.

[4] A. Balasubramanian, B. Levine, and A. Venkataramani. DTN routing as a resource allocation problem. In *Proceedings of ACM SIGCOMM*, 2007.

[5] C. Bettstetter. Smooth is better than sharp: A random mobility model for simulation of wireless networks. In *Proceedings of ACM MSWiM*, 2001.

[6] C. Bettstetter, H. Hartenstein, and X. Perez-Costa. Stochastic properties of the random waypoint mobility model. In *Proceedings of ACM MSWiM*, 2002.

[7] C. Bettstetter and C. Wagner. The spatial node distribution of the random waypoint mobility model. In *Proceedings of German Workshop on Mobile Ad hoc Networks (WMAN)*, 2002.

[8] S. Bhattachargee, M. Ammar, E. Zegura, N. Shah, and Z. Fei. Application layer anycasting. In *Proceedings of IEEE INFOCOM*, 1997.

[9] J. Burgess, G. Bissias, M. D. Corner, and B. N. Levine. Surviving attacks on disruption-tolerant networks without authentication. In *Proceedings of MobiHoc 07*, 2007.

[10] J. Burgess, B. Gallagher, D. Jensen, and B. Levine. MaxProp: Routing for vehicle-based disruption-tolerant networks. In *Proceedings of IEEE INFOCOM*, 2006.

[11] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. In *Proceedings of Wireless Communications & Mobile Computing (WCMC)*, 2002.

[12] C. Carter, S. Yi, P. Ratanchandani, and R. Kravets. Manycast: exploring the space between anycast and multicast in ad hoc networks. In *Proceedings of ACM MobiCom*, 2003.

[13] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Impact of human mobility on opportunistic forwarding algorithms. *IEEE Transactions on Mobile Computing*, 6(6):606–620, 2007.

[14] E. R. da Silva and P. Guardieiro. Anycast routing in delay tolerant networks using genetic algorithms for route decision. In *Proceedings of IDCS*, 2008.

[15] V. Davies. Evaluating mobility models within an ad hoc network. Master's thesis, Colorado School of Mines, 2000.

[16] F. Ekman, A. Kernen, J. Karvo, and J. Ott. Working day movement model. In *Proceedings of SIGMOBILE Workshop on Mobility Models for Networking Research*, 2008.

[17] V. Erramilli and M. Crovella. Forwarding in opportunistic networks with resource constraints. In *Proceedings of the Fourth ACM Workshop on Challenged Networks (CHANTS)*, 2008.

[18] V. Erramilli, M. Crovella, A. Chaintreau, and C. Diot. Delegation forwarding. In *Proceedings of ACM MobiHoc*, 2008.

[19] S. Farrell, V. Cahill, D. Geraghty, I. Humphreys, and P. McDonald. When tcp breaks: Delay- and disruption- tolerant networking. *IEEE Internet Computing*, 10(4):72–78, 2006.

[20] Y. Gong, Y. Xiong, Q. Zhang, Z. Zhang, W. Wang, and Z. Xu. Anycast routing in delay tolerant networks. In *Proceedings of IEEE Global Telecommunications Conference*, 2006.

[21] R. A. Guerin. Channel occupancy time distribution in a cellular radio system. *IEEE Transactions on Vehicular Technology*, 1987.

[22] D. Henriksson, T. Abdelzaher, and R. Ganti. A caching-based approach to routing in delay-tolerant networks. In *Proceedings of ICCCN*, 2007.

[23] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. In *Proceedings of ACM SigComm workshop WDTN*, 2005.

[24] P. Hui and J. Crowcroft. How small labels create big improvements. In *Proceedings of IEEE ICMAN*, 2007.

[25] P. Hui, J. Crowcroft, and E. Yoneki. Bubble rap: social-based forwarding in delay tolerant networks. In *Proceedings of ACM MobiHoc*, 2008.

[26] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. In *Proceedings of ACM SIGCOMM*, 2004.

[27] A. Jardosh, E. Belding-Royer, K. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks. In *Proceedings of ACM MobiCom*, 2003.

[28] D. B. Johnson and D. A. Maltz. *Mobile Computing*, chapter Dynamic source routing in ad hoc wireless networks, pages 153–181. Kluwer Academic Publishers, February 1996.

[29] A. Keränen, J. Ott, and T. Kärkkäinen. The ONE Simulator for DTN Protocol Evaluation. In *Proceedings of SIMUTools*, 2009.

[30] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek. The click modular router. *ACM Transactions on Computer Systems*, 18(3):263–297, August 2000.

[31] B. Liang and Z. J. Haas. Predictive distance-based mobility management for PCS networks. In *Proceedings of INFOCOM*, 1999.

[32] A. Lindgren, A. Doria, and O. Scheln. Probabilistic routing in intermittently connected networks. In *Proceedings of MobiHoc*, 2003.

[33] A. Menezes, P. van Oorschot, and S. Vanstone. Handbook of applied cryptography, 1996.

[34] S. Nelson, M. Bahkt, and R. Kravets. Encounter-based routing in dtns. In *Proceedings of IEEE InfoCom*, 2009.

[35] S. Nelson, A. Harris, and R. Kravets. Event-driven, role-based mobility in disaster recovery networks. In *Proceedings of ACM CHANTS*, 2007.

[36] S. Nelson and R. Kravets. Achieving anycast in dtns by enhancing existing unicast protocols. In *Proceedings of the ACM Workshop on Challenged Networks (CHANTS 2010)*, 2010.

[37] S. Nelson and R. Kravets. For members only: Local and robust group management in dtns. In *Proceedings of the ACM Workshop on Challenged Networks (CHANTS 2010)*, 2010.

[38] M. E. J. Newman. Analysis of weighted networks. *Physical Review E*, 70:056131, 2004.

[39] J. Ott, D. Kutscher, and C. Dwertmann. Integrating dtn and manet routing. In *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS, pages 221–228, New York, NY, USA, 2006. ACM.

[40] G. Palla, I. Dernyi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, pages 814–818, 2005.

[41] V. D. Park and J. P. Macker. Anycast routing for mobile networking. In *Proceedings of IEEE Military Communications Conference*, 1999.

[42] S. Paul, R. Yates, D. Raychaudhuri, and J. Kurose. The cache-and-forward network architecture for efficient mobile content delivery services in the future internet. In *Proceedings of Innovations in NGN: Future Network and Services*, 2008.

[43] M. R. Pearlman, Z. J. Haas, P. Sholander, and S. S. Tabrizi. On the impact of alternate path routing for load balancing in mobile ad hoc networks. In *Proceedings of ACM MobiHoc*, 2000.

[44] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *Proceedings of The Second IEEE Workshop on Mobile Computing Systems and Applications*, February 1999.

[45] R. Ramanathan, R. Hansen, P. Basu, R. Rosales-Hain, and R. Krishnan. Prioritized epidemic routing for opportunistic networks. In *Proceedings of MobiOpp*, 2007.

[46] D. Raychaudhuri, I. Seskar, M. Ott, S. Ganu, K. Ramachandran, H. Kremo, R. Siracusa, H. Liu, and M. Singh. Overview of the orbit radio grid testbed for evaluation of next-generation wireless network protocols. In *Proceedings of WCNC 05*, 2005.

[47] G. Resta and P. Santi. An analysis of the node spatial distribution of the random waypoint mobility model for ad hoc networks. In *Proceedings of ACM Workshop on Principles of Mobile Computing (POMC)*, 2002.

[48] M. Rossi, L. Badia, N. Bui, and M. Zorzi. On group mobility patterns and their exploitation to logically aggregate terminals in wireless networks. In *Proceedings of IEEE VTC*, 2005.

[49] J. Scott, P. Hui, J. Crowcroft, and C. Diot. Haggle: A networking architecture designed around mobile users. In *Proceedings of WONS*, 2006.

[50] IEEE Computer Society. Internet protocol, rfc 791, September 1981.

[51] T. Spyropoulos, K. Psounis, and C. Raghavendra. Spray and wait: An efficient routing scheme for intermittently connected mobile networks. In *Proceedings of ACM WDTN '05*, 2005.

[52] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility. In *Proceedings of IEEE PerCom*, 2007.

[53] T. Spyropoulos, K. Psounis, and C.S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In *Proceedings of (IEEE SECON)*, 2004.

[54] N. Thompson, S. Nelson, M. Bakht, T. Abdelzaher, and R. Kravets. Retiring replicants: congestion control for intermittently-connected networks. In *Proceedings of the 29th conference on Information communications*, INFOCOM, pages 1118–1126, Piscataway, NJ, USA, 2010. IEEE Press.

[55] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical Report CS-2000-06, Department of Computer Science, Duke University, April 2000.

[56] D. von Seggern. *CRC Standard Curves and Surfaces with Mathematics, 2nd ed.* CRC Press, 2007.

[57] B. D. Walker, J. K. Glenn, and T. C. Clancy. Analysis of simple counting protocols for delay-tolerant networks. In *Proceedings of ACM MobiCom CHANTS Workshop*, pages 19–26, New York, NY, USA, 2007. ACM.

[58] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393:440–442, June 1998.

[59] XOPR. Experimental open router platform. http://www.xorp.org.

[60] S. Yi and R. Kravets. Moca : Mobile certificate authority for wireless ad hoc. In *Proceedings of 2nd Annual PKI Research Workshop Program (PKI)*, pages 65–79, 2003.

[61] X. Yin, J. Han, and P. Yu. Truth Discovery with Multiple Conflicting Information Providers on the Web. In *IEEE Transactions on Knowledge and Data Engineering*, pages 796–808, November 2008.

[62] X. Zhang, J. Kurose, B. N. Levine, D. Towsley, and H. Zhang. Study of a Bus-Based Disruption Tolerant Network: Mobility Modeling and Impact on Routing. In *Proceedings of ACM MobiCom*, 2007.

[63] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: overview and challenges. *Communications Surveys & Tutorials, IEEE*, 8(1):24–37, 2006.