A FOG COMPUTING ARCHITECTURE FOR DISASTER RESPONSE NETWORKS

A Dissertation

by

HARSHAVARDHAN CHENJI JAYANTH

Submitted to the Office of Graduate and Professional Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

DOCTOR OF PHILOSOPHY

Chair of Committee,	Radu Stoleru
Committee Members,	Rabi Mahapatra
	Dezhen Song
	Alex Sprintson
Head of Department,	Nancy Amato

May 2014

Major Subject: Computer Engineering

ABSTRACT

In the aftermath of a disaster, the impacted communication infrastructure is unable to provide first responders with a reliable medium of communication. Delay tolerant networks that leverage mobility in the area have been proposed as a scalable solution that can be deployed quickly. Such *disaster response networks* (DRNs) typically have limited capacity due to frequent disconnections in the network, and under-perform when saturated with data. On the other hand, there is a large amount of data being produced and consumed due to the recent popularity of smartphones and the cloud computing paradigm.

Fog Computing brings the cloud computing paradigm to the complex environments that DRNs operate in. The proposed architecture addresses the key challenges of ensuring high situational awareness and energy efficiency when such DRNs are saturated with large amounts of data. Situational awareness is increased by providing data reliably, and at a high temporal and spatial resolution. A waypoint placement algorithm places hardware in the disaster struck area such that the aggregate goodput is maximized. The Raven routing framework allows for risk-averse data delivery by allowing the user to control the variance of the packet delivery delay. The Pareto frontier between performance and energy consumption is discovered, and the DRN is made to operate at these Pareto optimal points. The FuzLoc distributed protocol enables mobile self-localization in indoor environments. The architecture has been evaluated in realistic scenarios involving deployments of multiple vehicles and devices.

DEDICATION

To my grandparents, parents, family, friends and pets

ACKNOWLEDGEMENTS

This dissertation at the Department of Computer Science and Engineering at Texas A&M University was made possible due to the support of several agencies. I gratefully acknowledge the National Science Foundation, the Office of Naval Research, Nuview-Mehta for the fellowship, the Texas Commission on State Emergency Communications, the U.S. Department of Transportation and the Office of Graduate Studies at TAMU.

My first encounter with Dr. Radu Stoleru's exciting and inspiring style of teaching was in a Fall 2007 CSCE 689 class, and it made me consider research as a career. I enjoyed the exposure to recent research provided by his classes as well as the weekly meetings with the research group. His patient mentoring during the hundreds of hours we spent together solving problems allowed me to develop a unique set of skills necessary for successful research. Outside the lab, his friendly and helpful personality provided much needed relief from the high pressure research environment - those sailing and flying trips were super fun. Needless to say, my Ph.D. study under him was very enjoyable and fruitful. This dissertation would not have been possible without Dr. Stoleru's guidance. Thank you Sir. It has been a privilege to know you.

I would like to thank the members of my advisory committee, Dr. Rabi Mahapatra, Dr. Dezhen Song, and Dr. Alex Sprintson for their support. The helpful feedback allowed me to see this dissertation from a different perspective and identify avenues for betterment.

During my initial years at TAMU, I was an RA at the Academy for Advanced Telecommunications and Learning Technologies. I will always cherish those years spent with Dr. Walt Magnussen, Chris Norton, Laurie Ditto, Dr. Bob Arnold and

Dr. Ana Goulart. Thank you everyone for your support and help.

Past and present members of the LENSS Lab - Myounggyu, Amin, Mike, Wei, Mahima, Jay and others - thank you for your help during those tedious outdoor experiments and spending time answering my many questions.

I would like to thank my parents, Jayanth and Sheela, for always being there for me. My interest in computers was sparked by that 166MHz/16MB PC you bought for me (remember?). This dissertation would not have been possible without the solid foundation you provided during my early schooling years.

Houston, we don't have a problem. My extended family, in Houston and elsewhere, made me feel at home from day 1. Visiting you during holidays made me feel recharged and never homesick. Prasanna, Manju, Mads, Nina, Usha, Ananth, Mohan, Krishna, Prabhu, Meera, Dilli, Purnima, Arvind, Anu, Kiran, Hema (in no particular order!) - thank you all. It looks like I am the 9^{th} doctorate holder in the family - hopefully there will be many more in the future (looking at you Sindhuja, Kesh, Pranav, Pramukh, Anjali/Arjun, Kiran's kids).

I am fortunate to have known a large group of people back home in Bangalore, whose friendship I will always treasure: AK, Ashwin, ASO, Chikkadi, Gooby, Jammy, Kashi, Manju,Marrie, Ra, Roonie, Seena, Seti, Sha, Subbu, Sunny, Tho, Toori, Umax, Yak and many others. Dad's friends at BUSC and elsewhere - thank you for the fun times.

Dear Riddhi, you have been a constant source of love, help, support and food. Thank you for keeping me happy and sane. Blackie and Brownie: thanks for being a constant source of softness, warmth and snuggles.

Last but not the least, I would like to thank all my previous teachers as well as the staff at the Department of Computer Science and Engineering.

NOMENCLATURE

COTS	Consumer Off-The-Shelf
DoI	Degree of Irregularity
DTN	Delay Tolerant Network
DRN	Disaster Response Network
EOC	Emergency Operations Center
FEMA	Federal Emergency Management Agency
GPS	Global Positioning System
MA	Mobile Agent (in PDMM)
PoC	Proof of Concept
PDD	Packet Delivery Delay
PDV	Variance of PDD
PDR	Packet Delivery Ratio
PDMM	Post Disaster Mobility Model
QoS	Quality of Service
RSS	Received Signal Strength
SA	Situational Awareness
SV	Supply Vehicle (in PDMM)
USAR	Urban Search And Rescue

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE Page

LIST OF TABLES

TABLE Page

1. INTRODUCTION

Natural disasters cause loss of power and communication infrastructure in the affected area, and make the recovery process challenging. Search and rescue is one of the Emergency Support Functions as described by the Federal Emergency Management Agency in the U.S.A; Urban Search and Rescue (USAR) is a sub-function that deals with collapsed structures in urban areas. Since USAR first responders have always communicated using traditional means such as paper and paint on walls, recent research [1] has looked at providing them with new sensing modalities like low power wireless sensors and smart phones. DRNs use concepts from delay tolerant networking research to build a networking infrastructure that integrates these modalities and allows responders to share data. These DRNs are expected to handle data created by sensors that can be measured in kilobytes, to multiple gigabyte videos generated by smartphones.

However, the capacity of DRNs are finite and limited for many reasons. Node mobility in the area, which is leveraged by the DRN to mule data, is inherently unpredictable and random. This unpredictability makes it very difficult to find optimal end-to-end paths in the network. Because of the large deployment area and limited mobility, node contacts are sparse and the inter-contact time which is the primary component of the end-to-end packet delivery delay [2], can be measured in tens of minutes or even hours. Limited on-node storage buffers and contact bandwidth make optimal routing of data more challenging.

The crippled power infrastructure contributes to the complexity. Network devices

need to be powered by portable power sources that are in limited supply. While these devices can be duty cycled to save energy, the performance of the DRN will be severely impacted. Missing a node contact opportunity due to sleep scheduling is prohibitive since contacts in a DRN are very sparse. Some devices perform mission critical functions - as a result, they may not always be able to save power by sleeping.

Providing a high degree of situational awareness in an energy efficient manner becomes challenging in such environments. In this dissertation we present Fog Computing, a mobile ad hoc wireless network architecture for DRNs where traditional cloud computing is instantiated in a disconnected, mobile network called the "fog". Since large amounts of data will no longer pose a problem, cloud computing paradigms like file sharing and social networking that enhance teamwork are made possible. The key challenges addressed by Fog Computing are the improvement of situational awareness (SA) and ensuring energy efficiency, especially when there is a deluge of data in the network. SA is enhanced through a variety of means, mainly by improving the QoS metrics of the underlying DRN. System-wide energy efficiency at all layers of the network stack ensures that the Fog Computing architecture can be deployed using only battery powered COTS devices to last for about a week.

Increasing aggregate throughput in the underlying DRN will contribute towards increasing SA. The capacity of the DRN can be artificially increased by placing additional hardware in the area of deployment, since the number of contacts increases and the inter-node contact time decreases. Nodes can then use the improved capacity to deliver large amounts of data more reliably. However, the complexity of computing optimal locations at which to place hardware increases exponentially as the size of the deployment area increases. In this dissertation we investigate how such optimal locations can be computed such that the aggregate throughput and hence, the SA, can be improved.

Recent research has looked at algorithms which improve traditional QoS metrics like the average packet delivery delay (PDD) and packet delivery ratio (PDR). However, the variance of the packet delivery delay (PDV), commonly called jitter, remains un-addressed. In traditional networks, the PDD/PDV is typically measured in milliseconds - but in DRNs it can range from tens of minutes (trace based experiments in [3]) to even hours. This means that some packets may have a delivery delay much higher than the average delay. Since sensed data from the field is used to make decisions at the Emergency Operations Center (EOC), a large PDV becomes problematic since some critical data may arrive very late. Reducing the PDV can help improve the SA, by providing data more reliably. In this dissertation we look at how the PDV metric in a DRN can be controlled in a distributed fashion.

Energy efficiency in such a system is critically important. Area-wide deployment of portable power sources may not be practical, requiring manpower that could otherwise be used to rescue victims. Aggressive energy efficiency without regard to the performance of the network is equally dangerous, since low network performance could hamper situational awareness. In this dissertation we investigate the inherent trade-off present between energy efficiency and performance. More concretely, when a packet in the network is sent over multiple disconnected paths (i.e. replication in a delay tolerant network), nodes which do not lie on these paths can save energy by sleeping. Excessive replication can cause nodes to stay awake all the time, but very low replication hampers performance. It is this fundamental trade-off that is investigated.

Data with precise location is very important both during and after a fast paced process like disaster recovery. Victims can be easily located and tracked in the medical triage area, while responders in the field can quickly locate vital equipment. After the disaster, location tagged data can be very helpful in providing feedback to

responders and potentially improving the disaster recovery process. However, GPS which is a familiar technique, is prohibitive both energy and cost wise. Additionally, GPS may not be available in indoor areas such as the basement of a collapsed building. In this dissertation we investigate how self-localization can be performed in a fast and energy efficient manner.

1.1 Motivation

The motivating scenario for Fog Computing is a disaster whose affected area spans tens or hundreds of square miles. In this section we first describe this scenario in detail, followed by the mathematical mobility model used to capture movement in the area. The Post Disaster Mobility Model [4] is especially suited to model such disasters, but does not model US&R responder mobility. Some enhancements to PDMM are proposed, in order to correct this shortcoming.

Based on this motivating scenario, a set of requirements that are to be fulfilled by Fog Computing are enumerated in the next section. Each of these requirements improve the degree of SA during the disaster recovery process; they have been compiled based on multiple sources, including our interaction with Texas Task Force 1 members, various articles in newspapers and trade journals which describe the problems faced by responders during rescue, and our own proposed use of new technologies like fog computing, file sharing and social networking.

1.1.1 Motivating Scenario

Fog Computing addresses the needs of the disaster recovery process in the USAR application domain, as opposed to the medical triage area of a large disaster, as illustrated in Figure 1.1. When a disaster hits an urban metropolitan area that spans tens of square miles (2011 Joplin tornado) or hundreds of square miles (2011 Japan earthquake), power and communication infrastructure are rendered unusable. A

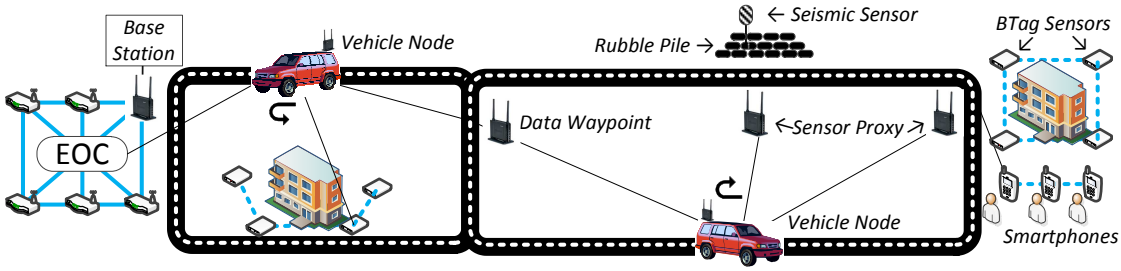


Figure 1.1: Schematic of a Fog Computing deployment showing all components (to be discussed later). Data generated by BTag and Seismic Sensors is ferried to the Base Station using Vehicle Nodes. A Data Waypoint improves the data transfer process by creating a contact opportunity between two Vehicle Nodes.

situation report about the 2011 Joplin Tornado [5], three days after the disaster, offers a glimpse into the situation: electric services are still being restored, a few cell sites have been restored, cell phones are being distributed and satellite telephone has been set up. In this kind of environment, presence of broadband internet access cannot be assumed, and blanketing a large urban area with battery powered communication hardware is near impossible. Providing data to USAR responders at high spatial and temporal resolution, with only tens of routers becomes a challenge. We assume that in such an environment, there are multiple collapsed buildings (buildings in Figure 1.1) or rubble piles in the affected area ("Rubble Pile" in Figure 1.1), and the Emergency Operations Center ("EOC" in Figure 1.1) is situated tens of miles away from the affected area. Limited internet connectivity is available only at the EOC. There is some mobility in the area ("Vehicle Node" in Figure 1.1) as medical supplies and rescued victims are transported from the field to the EOC.

1.1.2 Post Disaster Mobility Model

The PDM Model uses two main components to functionally model the interaction between survivors and rescue workers: "mobile agents" (MAs) and "Centers".

Centers are fixed, static areas on a city map and represent important areas such as the EOC or an evacuation center such as a stadium. PDM is a map based mobility model as opposed to random waypoint: each MA moves from one point to another based on a predefined set of roads. There are different categories of MAs like patrol cars, ambulances, volunteers and supply vehicles, each with its own movement model as described below. It is assumed that a networking device is present at each Center as well as on each MA. The PDM model is described formally as follows. There are multiple Centers C, of which two are special and compulsory: the EOC and the Triage. The categories of MAs are: Volunteers, Supply Vehicles, Ambulances and Patrol Cars. Each category has its own min and max speeds, and an agent belonging to that category chooses a speed uniformly between min and max at random, for each leg of travel. In the Volunteer Movement Model, each volunteer is placed at a randomly assigned home center $c_H \in C$ initially. Next, every volunteer individually chooses a random point within the entire map with 90% probability or chooses c_H with 10% probability, and travels to it along the shortest path. The process is repeated upon reaching the point. In the Supply Vehicle Movement Model, each SV is placed at a randomly assigned home center $c_H \in C$ initially. Then each SV individually chooses a center $c_d \in C$ at random and travels to it along the shortest path. The process is repeated upon reaching c_d. In the Patrol Car Movement Model, each car has a predefined list of centers $\{c_1, c_2, \ldots, c_n\} \in C \times C \times \cdots \times C$, and is placed at c_1 initially. Next, it travels to c_2 along the shortest path, and the process is repeated by choosing the next center in the list. In the Ambulance Movement Model, each ambulance is always assigned to the Triage initially. Next, each ambulance chooses a Center (including the Triage) at random to travel to, following which the ambulance always returns to the Triage. The process is repeated, resulting in a series of alternating Centers and Triages.

1.1.2.1 Enhancement

We add the "Urban Search and Rescue Worker (USAR)" category of MAs to the PDM mobility model. USARs operate in a area of fixed radius around a Center and move using random waypoint within that area. In the USAR Movement Model, each USAR member is placed at a predefined home center $c_H \in C$ initially. Then, every USAR agent individually randomly chooses a point uniformly within radius r of its c_H, and travels to it along the shortest path possible in the map. After reaching the point, the process is repeated. USARs need not visit c_H compulsorily.

1.2 Fog Computing Requirements

The FEMA equipment cache list [6] gives an idea of the size, cost and bulk of equipment currently used by US&R teams. Based on this list and interaction with Texas Task Force 1 US&R team members, we outline the responder requirements below.

1. File Sharing: Large rescue efforts typically involve many teams, sometimes from different countries. Data sharing among teams leads to a better understanding of the situation, and hence a high degree of SA. However, popular file sharing services available on today's desktops, like Dropbox and Amazon S3, cannot work in the absence of a connection to the central server. In Fog Computing, these file sharing methodologies have to function in the absence of high quality links or end-to-end paths, while the devices are physically spread over a large geographical area. A corresponding smartphone app should serve as an interface to this fog, allowing responders to share files, as well as allow third party developers to develop new apps using a published API.

2. Disconnected Social Networking: Social media like Twitter are increasingly being used by the public during the aftermath of disasters for communication and information dissemination [7][8]. Information sharing by responders during disaster recovery could possibly enhance the recovery effort. However, such traditional services cannot function in the absence of persistent internet connectivity. An equivalent service for Fog Computing will provide first responders with an opportunity to share information without requiring constant connectivity, while automatically synchronizing with the internet whenever internet access is available.

3. Network QoS Control: Data produced and consumed during disaster recovery can have vastly different QoS requirements. Some data could be mission critical where low packet delivery delay is acceptable. Large video files shared by responders need high throughput paths in the DRN. Other data could require low jitter. However, optimizing the network for one metric for one data stream could negatively affect one or more metrics for other data streams. Thus, the inter-dependency and dynamics of QoS control need to be investigated, following which, control of these metric should be offered to the user. The SA in this case is increased since data arrives more timely and reliably.

4. Accurate Localization in Indoor Environments: The affected area can be remotely monitored at the EOC by deploying sensors in areas of interest. For example, the air quality around the scene of a chemical leak in the basement of a building can be periodically monitored using low power wireless sensors. When such data is tagged with a precise location, it enhances the degree of SA since the quality of data is improved, meaning that responders can quickly and accurately pinpoint problems. Device localization is a primitive that is

assumed by many other services like routing and distributed data retrieval. However, localization technology like GPS may not be available indoors or in GPS-denied environments, or it could be cost prohibitive to deploy on all sensors. Even if it is available, periodically polling the GPS on a mobile device shortens the battery life. Existing WiFi based localization techniques can not always be used because the necessary infrastructure could be destroyed or unusable. Thus, there is a need for an inexpensive way of computing a device's location indoors, especially when the device is mobile.

5. System-wide Energy Efficiency: Each and every disaster recovery scenario is unique and dynamic, with respect to the number of deployed devices, the specific disaster recovery function performed by first responders, the length of time taken to recover from the disaster as well as the resources available. As a result, the expected system lifetime of each scenario can be different, or the amount of usable portable power in the form of batteries can vary, or the network performance requirements can be different. It is important to tune the deployed system such that the network performance is optimal given constraints on the available energy resources. In other words, the user needs to be able to control the system performance/system lifetime tradeoff to achieve the best possible performance given an anticipated time frame for disaster recovery.

6. Smart Victim Detection Under Rubble: Highly sensitive seismic sensors that pick up vibration from a rubble pile were used during the 9/11 emergency to locate trapped victims [9]. First responders can listen for human voices or activity through attached headphones, and can locate them by asking victims to tap on nearby pipes. However, the low frequency sound created by shaking buildings and nearby human activity interferes with this detection process [9].

Automatic noise filtering 1) eliminates the "All Quiet" condition required for seismic sensor use (which halts rescue efforts in the immediate surroundings); and 2) enables the re-deployment of on-site personnel to other areas of the disaster.

7. Digitized Building Information: Whenever USAR teams search buildings, the search status of the building is indicated using markings (called X-codes, FEMA or INSARAG format) painted in day-glo orange on walls (tagging), for the benefit of other teams. This includes data like the last search date/time, presence of hazards etc. Digitizing such tags using low power motes will provide the EOC with high situational awareness due to the variety of information that can be sensed on motes. By digitizing building tags and enabling automated data collection, resources can be allocated by the command center more efficiently.

8. Team Separation Detection: During USAR operations in a collapsed building, team members may become separated from each other due to falling beams, or they may lose vital tools like cement saws accidentally. We present an algorithm that lets each team member know of any separation in the team independent of the team size, even when the separation or "cut" occurs many hops away. Team separation detection delay is measured in seconds. An app installed on a smartphone alerts a first responder immediately after a tool or team member is detected as missing, enabling recovery from the situation within seconds.

2. RESEARCH CHALLENGES AND MAIN CONTRIBUTIONS

Fog Computing has been introduced and motivated in Section 1. In this section the key research challenges by this dissertation are enumerated. First, the dissertation statement is defined, followed by discussion about the two key research challenges espoused in this statement. A set of factors that contribute to the complexity of solving these challenges is listed. The rest of this dissertation is divided into multiple sections, all of which aim to answer these key challenges; a dissertation structure is proposed. Each of these sections answer some specific research questions, which are collected here. Finally, the research contributions of this dissertation are presented.

2.1 Dissertation Statement

The dissertation statement that concisely represents the design goals and the proposed dissertation is as follows:

> "Design Of A Fog Computing Architecture For Disaster Response Networks That Provides A High Degree Of Situational Awareness In An Energy Efficient Manner"

2.2 Key Research Challenges

The two major research challenges espoused in the above statement are:

1. Improved situational awareness through improved Quality of Service: The key to effective disaster response is a high degree of situational awareness. Fog Computing can improve situational awareness by providing large amounts of data with high spatial and temporal resolution. Improving QoS metrics like throughput, packet delivery delay, variance of the packet delivery delay and packet delivery ratio results in data that is delivered quickly and reliably.

2. Energy Efficiency: Since there is no usable power infrastructure available, the entire system has to be powered with portable power sources like include batteries, diesel generators and solar panels. While diesel generators need a constant supply of fuel, solar panels are expensive. Even if every Fog Computing device had a dedicated energy source, there is a need for energy efficiency. Because of limited personnel and human resources during an emergency, precious man-hours can not be spent maintaining the energy sources (replacing spent batteries, refilling diesel generators). Moreover, devices in these systems are critical and cannot stop functioning without compromising the whole system. With energy efficient devices, the system lifetime is extended as much as possible.

The following factors contribute to the complexity of meeting the above two major challenges:

1. Sparsity: Stable end-to-end paths in the network are very sparse because of the low node density, which causes frequent disconnections in the network. Routing data under these circumstances becomes a challenge.

2. Unpredictability: Both the node mobility in the area of deployment, and the data workload produced by Fog Computing devices are unpredictable. As a result, tasks such as routing data and sleep scheduling become difficult.

3. Limited resources: Fog Computing devices have limited radio bandwidth as well as on-node storage capacity. As a result, only a small amount of data can be transferred at each node contact, increasing the packet delivery delay QoS metric. The storage capacity is limited compared to the large data workloads - hence, retrieving data from the network can result in low throughput as well as increased delay.

4. Scalability: A complex architecture like Fog Computing should scale ideally. However, because of limited resources and the state maintained by several subsystems, scaling to thousands of devices becomes a problem.

5. Security: A sensitive application like disaster response necessitates secure transport of data. However, the computing and transportation overhead incurred by security strains the already limited resources like bandwidth and storage capacity.

2.3 Dissertation Structure

The rest of this dissertation will be divided into the following six sub-components (barring the next section which is survey of relevant literature), each of which contribute towards the goal of realizing the dissertation statement, given the assumptions, key research challenges and motivation. A very high level overview of each section follows.

1. Fog Computing Design Principles and Architecture: Based on the design requirements, a set of design principles is proposed, followed by a high level system architecture that incorporates these principles.

2. New Applications and Proof-of-concept Implementation: A few motivating applications that satisfy some of the Fog Computing requirements are described. The previously proposed high level architecture is instantiated on multiple heterogeneous devices. The software and hardware used in each device class is shown, followed by a list of new motivating applications that can be run on a particular device class.

3. Infrastructure Optimization in Fog Computing: The key challenge of improving SA is addressed by increasing the aggregate goodput in the network. Experi-

mentation and evaluation of this approach is presented.

4. QoS Aware Routing in Fog Computing: The key challenges of improving SA as well as energy efficiency are addressed, by controlling the PDD as well as PDV simultaneously and measuring their effect on energy consumption. An analysis of the tradeoff is presented, followed by experimentation and evaluation.

5. Energy Efficiency in Fog Computing: Simultaneously satisfying the key challenges of improving SA as well as ensuring energy efficiency is analyzed, and a mathematical approach to control the tradeoff is presented, followed by experimentation and evaluation.

6. Mobile Self-Localization in Fog Computing: SA is improved by tagging data with location. This section described an algorithm that can performing this task in GPS-denied or indoor environments, followed by experimentation and evaluation.

2.4 Research Questions

Here we concisely and concretely enumerate the research questions that are addressed in each of the six sub-components listed above, so that the dissertation statement can be realized, considering the assumptions.

1. Fog Computing Design Principles and Architecture:

   (a) What are some design principles that will make Fog Computing more user friendly, and prevent practical roadblocks during usage?

   (b) How can heterogeneous devices and protocols, each with their own capabilities, be integrated into a single architecture?

(c) What are the services used by and required of each layer of this architecture?

2. New Applications and Proof-of-concept Implementation:

   (a) Is it possible to implement the above architecture using COTS devices?

   (b) If so, what are the capabilities of each device's hardware?

   (c) What services are required of each device?

   (d) Can the system requirements be implemented as user installable apps, conforming to the above design principles and architecture?

   (e) Can third party developers create apps that can be used easily?

3. Infrastructure Optimization in Fog Computing:

   (a) Can placing additional devices (waypoints) increase aggregate throughput?

   (b) If so, is there an efficient heuristic for waypoint placement?

   (c) Is a polynomial time algorithm possible?

   (d) Is there a DTN routing scheme that allows lower delay than the others given a scenario?

   (e) How does vehicle speed and payload size affect the amount of data transferred?

   (f) Can devices automatically tune system parameters so that wireless data transfer is optimized?

4. QoS Aware Routing in Fog Computing:

(a) How can the variance of the packet delivery delay, commonly called jitter, be controlled in a DRN?

(b) Is it possible to obtain closed form equations to estimate both the mean and variance of the travel time for the USAR movement model?

(c) What are the effects of controlling a QoS metric, on other QoS metrics?

5. Energy Efficiency in Fog Computing:

(a) Which devices/components consume the most power, and under what circumstances?

(b) Is there a Pareto Front between system performance and energy consumption?

(c) How has recent research in the area advanced our knowledge of this Pareto Front?

(d) Is it possible for the user to tune the Fog Computing system to operate at a chosen Pareto optimal point?

6. Mobile Self-Localization in Fog Computing:

(a) How can we design a lightweight algorithm for multi-hop, self localization of mobile nodes in complex environments?

(b) What is the effect of sparse anchor density and anchor location accuracy upon indoor, RSS-based multi-hop self localization schemes?

(c) How can we design a fully distributed protocol for multi-hop self localization while minimizing network overhead?

2.5 Main Contributions

In light of the above research questions, assumptions and the dissertation statement, the contributions of this dissertation are as follows (per sub-component):

1. Fog Computing Design Principles and Architecture:
   (a) Recognizing and defining Fog Computing
2. New Applications and Proof-of-concept Implementation:
   (a) Completely battery powered, purely COTS implementation
   (b) New applications like File Sharing, Social Networking, Building Tags, Vibration Sensing
3. Infrastructure Optimization in Fog Computing:
   (a) Framework for placing extra hardware to maximize throughput
   (b) Optimizing device to device data transfers
   (c) Polynomial algorithm under special assumptions
4. QoS Aware Routing in Fog Computing:
   (a) Raven, a risk-averse outing protocol that allows control over QoS metrics, especially the jitter
   (b) Enhancements to Post Disaster Mobility Model to include USAR first responders
   (c) K-Shortest paths algorithm adopted to graphs with stochastic weights
5. Energy Efficiency in Fog Computing:

(a) Novel framework that intentionally excludes nodes from the routing process

(b) Dual objective non-linear optimization problem solved using evolutionary algorithms

6. Mobile Self-Localization in Fog Computing:

(a) FuzLoc, a distributed self-localization protocol

(b) Greatly improved accuracy by considering two hop neighbors

(c) Uncertainty is characterized using fuzzy logic, in a system of non-linear equations

3. LITERATURE SURVEY

Our motivating scenario is a large scale disaster (e.g., entire cities/regions are affected) and not a local, block-wide emergency in a city or a town. Unfortunately, recent history gives a few motivating examples like the earthquake in Haiti [10] and the tsunami in Japan [11]. In these incidents, the communication infrastructure is disrupted (i.e., cellular networks are completely or partially damaged) for weeks if not months, there are serious shortages of power (i.e. power sources like nuclear reactors are damaged), surveying the disaster area for survivors under the rubble takes from days to weeks (with some inspiring examples of survivors emerging after tens of days) and the EOC is flooded with sensing and multimedia data from the field. This febrile, fast pace environment lasts from one to several weeks, until the infrastructure is usable again.

In this section, state of art research related to the above motivating scenario is reviewed. First, DRNs are introduced and both academic as well as industrial state of art is reviewed. The set of motivating applications that we present in Section 5 is motivated by comparing them with recent research. Then, research related to the infrastructure optimization technique that we propose in Section 6 is discussed. Section 7 which presents a QoS aware routing scheme draws from both the DTN routing and stochastic graph theory research communities. State of art in both these areas is reviewed. The energy efficiency scheme in Section 8 is motivated by explaining its contributions compared to state of art research. FuzLoc, the mobile self localization scheme presented in Section 9 is discussed with respect to prior research in the localization area.

3.1 Disaster Response Networks (DRNs)

The disaster recovery process has seen a lot of development over the last few years owing to the increasing use of technologies like wireless networking and mobile computing. In the United States of America, the Federal Emergency Management Agency (FEMA) is responsible for coordinating the federal response to disasters and emergencies. It's mission is to "support our citizens and first responders to ensure that as a nation we work together to build, sustain and improve our capability to prepare for, protect against, respond to, recover from and mitigate all hazards" [12], and was established in 1803. FEMA's National Incident Management System (NIMS) defines an organized approach to emergency management.

The importance of communication during emergency response is stressed in one of the NIMS training modules [13]. It details numerous instances where data sharing, communication and information management can save lives. For example, during the Air Florida tragedy in 1982, coordination problems were experienced between response agencies because they could not communicate with each other. The document specifies that communications equipment should be reliable, portable and scalable; redundancy and resiliency should be implemented to ensure that data flow is uninterrupted.

We now focus on work that has improved the disaster recovery process from a computer networking and communications perspective.

3.1.1 Industrial State of Art

A mandated and standardized equipment list for FEMA US&R teams is available online [6]. Each task force maintains its own cache, containing over 16,000 items. The technical equipment details Project 25 (P25) [14] compatible 2 way portable wireless radios. A 120V AC powered base station is also mentioned, along with

battery powered repeaters. Such radio systems have a large radio range capable of covering large areas and are securely encrypted. However, only voice and data channels are available on such systems at very low data rates of 9.6kbps [15]. Since the P25 systems defines a physical as well as a MAC layer, it is difficult to integrate protocols meant for low power wireless like 802.15.4 (henceforth referred to as 15.4). Although reliable real time long range secure communication systems like Project 25 exist, they have several caveats like cost, difficult integration with other systems, high power requirements, physical bulk.

[16] has commercial offerings which accomplish network centric warfare. Based on the limited details available, the system offers robust middleware based on 802.11 and/or WiMax based networking. To the best of our knowledge, these systems assume an AC powered connected network and do not offer integration of low power smart devices. Additionally, they address spatially localized disasters or the triage area of disasters which occur over a large area.

3.1.2 Academic State of Art

Recent work in academia [17] has shown that using traditional means of communication like paper/pencil and sharing data using voice based radios is not scalable, and often leads to inaccuracies. Using networks that require significant infrastructure, like cellular networks for example, is not possible in a large scale natural disaster. Moreover, scalability is an issue with such solutions. Medical triage activity during a disaster has received significant attention from the research community. An initial version of the Wireless Internet Information System for Medical Response in Disasters (WIISARD) [18] project proposed a local client-server infrastructure. A middleware layer supported a publish-subscribe model where cached remote objects (CROs) are shared. The key contribution is that data consistency is assured across

clients, even when disconnections are caused by client mobility and unreliable links.

As mentioned in [19], the above setup (which used AODV-based routing) performed poorly across deployments. The next iteration of WIISARD [19] focused on characterizing the network properties during deployments. It's chief contribution is showing that network properties can vary during the different stages of disaster recovery due to the complex co-operation and data sharing that occurs among mobile first responders. As a solution, it proposes the WIISARD Communication Protocol (WCP), which is a gossip based protocol for data dissemination. IEEE 802.11 based hardware provide the networking backbone.

Another academic effort that improves disaster recovery in the medical triage area is the Advanced Health and Disaster Aid Network (AID-N) [20]. It presents the design of an electronic triage system for use on victims. Low power IEEE 802.15.4 based hardware replaces colored paper tags that are used to identify the severity of a victim's situation. A tiered architecture is proposed: level 1 contains an ad hoc network of embedded 802.15.4 based devices, level 2 consists of 802.11 based PDAs and laptops that personnel use to access victim information, and level 3 is a central high capacity server that stores information. Level 1 devices form an ad hoc mesh network that uses the Flows routing protocol [20] which was proposed as part of the CodeBlue project [20]. It is a spanning tree based protocol that uses hop by hop acknowledgments. Level 2 devices connect to a 802.11g based wireless router, which in turn connects to the level 3 central server. To summarize, AID-N provides a tiered multi-PHY architecture that uses mesh networking.

[21] is another 802.11 based wireless mesh network (WMN) tailored to provide effective medical response in the event of a disaster. Mobile clients like PDAs and laptops roam around the geographical area while being connected to the Internet via multiple backhaul connections [22]. Digital tags on patients [23] are read by

medical personnel using PDAs. Changes to such digital records are tracked and can be easily rolled back in case of conflict due to multiple simultaneous editing. Project RESCUE [24] provides an overview of a WMN for effective emergency response. [25, 26] argues for a WMN to be used in disaster response. It cites several shortcomings in several real use cases which provide a baseline comparison to such systems. In [27], a hybrid WMN makes use of wireless WANs as a backhaul link to access traditional networks. Several portable networked devices make use of routers affixed to lamp posts in order to achieve network connectivity. The SAFIRE project [28] deals with situational awareness for firefighters. Among the many problems dealt with are reliable data dissemination over ad hoc networks. Responders use a WiFi enabled tablet which uses a central push-pull method of data movement. The intended purpose is for use in a local emergency, and not a region wide disaster.

These WMNs are susceptible to disconnections, unreliable links due to user mobility as well as network partitioning when spread over a large geographical area. Delay Tolerant Networks (DTNs, discussed in the next section) have been used in a few disaster response networks. Our own previous work [29] proposes a multi-tiered 802.11 and 802.15.4 based architecture that *focuses on urban search and rescue* instead of the medical triage area (it's contributions are discussed in Section 3.1.5). The authors of [30] propose an Emergency DTN (EDTN) and recognize that TCP is a protocol that is unsuitable for use in DTNs. The main contribution of [30] is an analysis of the Session Initiation Protocol (SIP) for use in a DTN, and a subscribe-notify architecture. The authors of [31] comprehensively survey academia and industry for information about how information flows in disaster, and review state of art network simulators for their usefulness in simulating a large scale natural disaster. The main contribution here is a survey of responder requirements and a conclusion that a distributed system is much more efficient than a centralized top-

down command-and-control approach. In [32], the authors propose an opportunistic network that can glue together existing crippled infrastructure. It uses the HiBoP protocol [33] which is a context based routing protocol. Simulations are performed using this protocol in a disaster scenario. The main contribution of this paper is recognizing that DTNs can be used to bridge existing networks, and the use of the HiBoP protocol in simulations.

3.1.3 Applications in DRNs

Seismic sensing has been used alongside WSNs to predict volcano activity [34]. In this system, Mica2 nodes (called infrasound nodes) are outfitted with a condenser microphone which samples data at 102.4Hz. Twenty five consecutive samples were packed into a single 32 byte packet and transmitted to an aggregator node. Data is then sent to a wired high capacity node using a long distance wireless link. Data sampled using the onboard ADC on the infrasound nodes was initially distorted during radio transmission due to hardware voltage issues. The 4Hz noise caused by the radio is corrected using a smoothing filter. The paper then proposes a distributed event detection system designed to reduce bandwidth usage during times when the volcano is quiescent. An exponential weighted moving average algorithm is used to detect local events. Other examples of using seismic sensors to detect events include footstep characterization in intrusion detection systems [35, 36]. In [37], seismic data is analyzed both in the time and frequency domain. A method to differentiate walking footsteps from running is also presented, based on the Fourier Transform technique. Techniques involving the sampling and processing of seismic data are the focus of research in home intrusion detection systems and military area monitoring. In [38], the authors describe their experiences in deploying a wireless sensor network to monitor the Torre Aquila building. Environmental nodes sensed temperature,

while deformation nodes sensed the building's deformation by measuring the time taken for an optical pulse to travel through fiber optical cables. Vibration nodes used an accelerometer to measure movement. The heterogeneous data generated by these sensors was transmitted to a Gumstix based sink node using a modified version of Trickle, a tree based routing protocol.

Building monitoring systems have been proposed in recent research. In [39], the authors present a framework designed to improve information sharing during disaster rescue. Responders place RFID tags on buildings and program them with 802.11 enabled devices, which form a mobile ad hoc network.

3.1.4 Infrastructure Optimization in DRNs

Researchers have investigated the idea of placing additional hardware in the deployment area, in order to improve various performance metrics. In [40] the authors present a scheme to deploy static relays, called throwboxes. These devices artificially increase the number of contact opportunities in a DTN. The throwbox placement problem aims to maximize the capacity between nodes in order to increase the overall throughput. It is mathematically modeled as a mixed integer programming problem, and a *greedy* solution is proposed. There are three possible scenarios depending on how much is known about the network. The first one assumes that quantities like the traffic demand, average contact duration and average capacity between any two nodes are known. The second scenario assumes that only the information about contact opportunities (and not the traffic demand) is known. Lastly, the oblivious scenario assumes that nothing is known about the network. Since the first scenario is NP-Hard, a greedy approximation is proposed. The second scenario is solved tractably and a random solution is proposed for the third scenario. After the throwbox locations are computed, single, multi and epidemic routing algorithms are proposed and

analyzed. [41] studies the hardware architecture for such throwboxes in an attempt to increase the lifetime (discussed in Section 3.3).

The authors of [42] analyze the problem of augmenting existing networks with three types of solutions: base station based, wireless mesh based and disconnected store-and-forward relay based approaches. Base stations are connected by high capacity wired networks, mesh nodes connect to each other over wireless links, while the relay nodes cannot talk to each other due to the sparsity (they simple store data and relay it to passing mobile nodes). The first contribution of the paper is the observation that the performance improvement gained by adding x base stations can only be achieved by using $2x$ mesh nodes or $5x$ relay nodes. However, base stations require the *most* infrastructure (because of the wired backhaul link), and relay nodes *require the least*. The second contribution notices that adding just a few nodes of any kind is much better than having none at all and relying purely on data transfers between mobile nodes. Then two strategies for placing these nodes are proposed: uniform and non-uniform. A simple heuristic is used for the latter wherein the area of deployment is divided into a grid, and the number of nodes placed in a square is proportional to the amount of time spent by mobile nodes in that square. The problem of choosing relay strategies for deployed throwboxes is investigated in [43]. A Markovian model of a DTN is proposed, where the contact opportunities between any pair of mobile nodes is modeled as a Poisson process.

3.1.5 Contributions of this Dissertation

The contributions of this dissertation with respect to the above four areas is now detailed.

3.1.5.1 Industrial State of Art

The Fog Computing architecture is designed to process large amounts of data - a procedure that would be difficult when using voice based low bit rate P25 radios. This dissertation proposes new sensing and data sharing modalities. Smartphones, for example can record HD video which can be shared with other responders, while low power sensors periodically record and report sensor data. It is unclear how such paradigms can be integrated into the existing equipment mandated in the FEMA list. Our architecture is able to incorporate a wide variety of device classes, each of which have their own PHY layers. The most important difference is that our architecture can be fine tuned using a variety of algorithms to improve performance metrics, while also being able to control the performance-energy trade off. To summarize, these systems are unable to provide new sensing and data sharing modalities that generate data at high temporal and spatial resolutions. Some of these requirements are detailed in Section 1.2 of Section 1. The design principles adopted in this dissertation, detailed in Section 4.1, successfully incorporate these requirements.

3.1.5.2 Academic State of Art

The primary difference between the work presented in this dissertation and research like WIISARD [19] or AID-N [20] is that the former is designed for the needs of urban search and rescue personnel when operations occur over a large geographical area. While solutions that use wireless mesh networking[21, 25, 26] in the medical triage area are well investigated, this work addresses US&R operations like searching for survivors building rubble and building monitoring using low power 802.15.4 devices. As stated in [19], the major contribution is not WCP itself, but the characterization of link quality and human mobility patterns during the medical triage phase. To summarize, this work discusses the use of delay tolerant networking in the

larger US&R area as opposed to wireless mesh networking based solutions for the medical triage area.

Research has proposed the use of delay tolerant networking during disaster recovery [30, 31, 32]. The major contributions of this dissertation are a hardware/software architecture that incorporates delay tolerant networking, as well as *theoretical contributions* like algorithms and routing protocols designed to fine tune this DTN based architecture. The work presented in [29] can be thought of as a very preliminary version of the fog computing architecture. It's major contribution was the lack of assumptions about existing infrastructure. Solutions that enhance situational awareness through topology management, sound detection subnetworks and a message routing scheme. This dissertation has significantly different contributions that include QoS aware routing and cross layer energy efficiency.

Architecture-wise, a multi layered device architecture that is able to incorporate *heterogeneous hardware* is proposed in this dissertation. It's uniqueness lies in the fact that *all functionality is pushed towards the application layer* - meaning that it can be easily extended to applications like heritage building monitoring and even volcano detection or taking an animal census in jungles. For example, the Vibration Sensing App in Section 5.1.4 that is ideal for DRNs could easily be replaced with a seismic sensing application designed for detecting building vibrations. The theoretical contributions like QoS aware routing and Pareto-optimal energy efficiency *will still be applicable* in these scenarios. Thus, this Fog Computing architecture is much more versatile than the state of art systems.

Applications: The set of motivating applications presented in Section 5 have important but minor contributions. For example, the Vibration Sensing App in Section 5.1.4 implements a kNN based classifier method. A 1024-bin FFT is chosen to extract the characteristics of the sensed signal. This lightweight approach allows

the algorithm to be implemented on the low resource mote itself - and only the events are forwarded to high resource nodes, to alert first responders. The Building Monitoring App in Section 5.1.6 utilizes 802.15.4 based motes which are capable of sensing temperature and/or air quality - a feature that is not found in RFID tags.

Infrastructure Optimization: The key difference between the strategies presented above and this dissertation is the mobility model. We use the Post Disaster Mobility Model, and more specifically the Patrol Car Movement Model in this mobility model. This is because the dissertation focuses on disaster recovery scenarios, whereas the above work is suitable for application to general networks which are sparse and mobile. The patrol car model allows us to make justified assumptions about the deployment scenario, leading to better performance improvement. Comparing with [40], our problem formulation proposes a MIP-based solution and not a greedy one. Traffic demand is *not* assumed to be known. The relaying strategy is single path and source routing based. We deal with a slightly different problem where instead of maximizing the data rate between mobile nodes, we focus on optimizing the throughput of data delivered to the EOC. This objective is based on an observation that no matter how high a data rate a mobile node has, limited vehicle movement bounds the possible throughput, and hence, the amount of unique data passed to a vehicle has to be maximized. Compared to [42], we do not investigate the possibility of augmenting the DRN with base stations or mesh nodes because of the infrastructure required - which is difficult to come by in a disaster. The relay placement strategies presented in [42] are heuristic based and based on a general mobility model, whereas the one in this dissertation is tailored to the PDM model and uses a mixed integer program to solve the problem. Moreover, in the absence of any upper limit on the number of throwboxes, we provide a quick polynomial time placement solution.

3.2 Quality of Service and Routing in DRNs

Chapter 7 of this dissertation proposes a QoS aware routing framework, called Raven, for DRNs. It's unique contribution is its ability to control the variance of the packet delivery delay in a delay tolerant network. With this mind, let us look at related work in the area.

3.2.1 Routing in DRNs

We draw upon a large body of research experiences in the field of delay tolerant networking (DTN). [44] advocates the use of DTN to provide situational awareness in a disaster. The proposed system provides elementary social networking by helping victims disseminate information while not overwhelming the system. Dieselnet and the DOME testbed [45] provide rich information about implementing routing protocols, providing services and a public DTN testbed using WiFi devices mounted on buses covering a large area. In [46], data is collected from sensors deployed in a wildlife tracking environment leveraging the frequent movement of zoologists and scientists in the area.

A DTN routing protocol can be classified based on its approach towards five major criteria:

1. Redundancy: forwarding/limited replication/unlimited replication
2. Contact Bandwidths: limited/unlimited
3. Storage Resources: limited/unlimited
4. Knowledge of Mobility: none/partial/learned/complete/oracle
5. Target Metric: mean delivery delay/ energy efficiency/other

A short overview of recent DTN routing protocols in the context of the five criteria follows. Epidemic [47] routing performs unlimited replication and relays a copy of the message to every node possible, assumes unlimited resources and does not attempt to leverage mobility. The authors of [48] have analyzed the trade off between replication and energy efficiency in an epidemic-like routing protocol. Spray and Wait [49] performs limited replication of a message according to a user specified parameter, while retaining other assumptions. A mathematical analysis of this protocol is available in [50]. Prophet [51] and MaxProp [52] leverage mobility in a DTN by selecting relays based on historical encounters such that the likelihood of delivery is maximized. RAPID [3] allows the user to target any metric, including the delivery delay, and works by estimating the marginal utility gained in replicating a packet to a host. Such protocols are often called utility based protocols since a relaying decision is made by comparing a node's utility to its contact's. [53] uses MCMC methods to perform utility based relay selection in a DTN. Scoop [54] is a DTN multicast routing protocol which leverages locally observed information about mobility to minimize the delivery delay. In [55], information about the social function of DTN nodes (humans) is used to perform limited replication routing. The R3 [56] protocol unifies mesh/MANET/DTN routing paradigms by using learned information about the distribution of link delays to perform replication based routing. DTN routing is most optimal when all future inter-node contacts are well known. In [57] a comprehensive linear programming formulation of DTN routing with limited contact bandwidths is presented, with the assumption that interrupted transfers can be resumed. However, it cannot be applied to opportunistic DTNs knowledge of all future contact durations is needed. In [58], the authors mention that jitter is an applicable QoS metric but propose two additional metrics.

3.2.2 Stochastic Graph Theory

Problems involving graphs with either multidimensional or probabilistic edge weights have been investigated. An overview of the shortest path problem with probabilistic edge weights is found in [59]. In [60] a stochastic shortest path algorithm is used to solve a route planning problem in the presence of uncertain traffic delays. [61] uses expectations of link delays to solve a network routing problem.

3.2.3 Contributions of this Dissertation

3.2.3.1 DTN Routing

Raven can either forward or replicate messages, assumes limited contact bandwidths, assumes unlimited storage, possesses partial knowledge of mobility and targets the multiple metrics simultaneously. Based on the taxonomy in [3], Raven can be placed in a new "P6" category for protocols which assume unlimited buffers and limited contact bandwidth. The primary motivation for assuming unlimited storage capacity is the falling price of storage cost (0.07 USD per GB in 2009; today it is 0.03 USD per GB). Recent COTS WiFi routers like the Netgear WNDR3700v2 have a USB port, which means that several terabytes of storage can be provisioned by connecting a USB drive. This is practically unlimited in comparison to the average compressed HD video or photograph taken with a smartphone camera which is about 1GB and 4MB respectively. The assumption of limited contact bandwidth is justified by the experiments in [62].

The above protocols do not allow control of the PDV, with the possible exception of utility based routing protocols where a target metric can be specified. These utility based protocols need *the user* to provide the protocol with a formula which can estimate the target metric, in this case the PDV. As the complexity of the mobility model increases, estimating this metric becomes less trivial and more challenging

mathematically. Raven on the other hand provides the user with a single paramter ρ which when increased, automatically lowers the PDV. Thus, no complex mathematical modeling on the user's part is needed. Additionally, the above protocols do not allow for *simultaneous* and *intentional* control of multiple target metrics; the effect these protocols have on the QoS metrics may be incidental. Very few protocols (like SprayWait) are able to choose between forwarding ($L = 1$) and controlled replication ($L > 1$) when required, which is important at high data loads since uncontrolled replication can be harmful. Raven controls replication using the K system parameter. It should be noted that Raven is designed keeping the PDM mobility model in mind and could theoretically work with mobility models; but this is outside the scope of this dissertation and is future work. Prior art which is most similar to Raven is [63] in the sense that it is a DTN routing protocol for disaster response networks and uses the PDM mobility model. However, [63] focuses on reducing the energy spent on communication whereas Raven is optimized for controlling multiple metrics (most notably the PDV) *including* the energy consumption.

3.2.3.2 Stochastic Graphs

To the best of our knowledge, the K-Shortest Paths problem has not been extended to graphs with stochastic weights, and such graphs haven't been used to model DTNs. Note that some works use the word "stochastic" to refer to the problem of delivering broadcast or multicast traffic to nodes with a probabilistic guarantee, but the similarity ends there.

3.3 Energy Efficiency in DRNs

Energy efficiency is an important concern to a DTN's user: high or non-uniform energy consumption can lead to network failure, low performance or violation of user constraints. Lowering the already low node density means that the node inter-contact

time increases, decreasing the capacity of the DTN. These concerns are doubly important in highly challenged environments such as the post-disaster recovery process and DRNs. Here we describe some recent research in the area of energy aware delay tolerant networking.

3.3.1 Software Based Approaches

These aim to minimize radio usage by reducing neighbor discovery overhead and/or reducing data transfers. The number of transmitted messages has been a metric in the performance evaluation of many DTN routing protocols [49] [63]. The scheme in [64] saves energy by adjusting the contact probing frequently optimally. For data transfers, epidemic routing is a simple protocol where each packet is replicated to every encountered node; this energy-inefficient approach has been the concern of recent research. The authors of [48] model energy efficient epidemic routing as an optimal control problem, while the authors of [65] propose that a packet be transmitted only when the number of neighbors reaches a threshold so as to reduce energy. Markov chains are used to model message dissemination performance of two routing protocols [66], and an optimal dynamic forwarding policy is derived. Yet another approach is to limit the maximum number of replicas a packet can have in the network [49]. Network coding [67] [68] [69] reduces the number of bits transmitted and hence the energy used by the radio. Forwarding packets to socially close nodes may increase performance, but simultaneously causes rapid energy depletion [70].

3.3.2 Hardware Based Approaches

Such approaches are far and few. In [71], the authors propose the use of a multi-tier platform that involves a long range, low bitrate radio that can wake up a short range, high bitrate radio. An optimization problem maximizes the number of bytes transferred while meeting a power consumption constraint. Using traditional low

power, short range radios like 802.15.4 has been shown to be un-optimal for sparsely populated DTNs [72]. Energy efficient MAC protocols are useful for traditionally dense networks like MANETs, but inefficient for sparse DTNs with high inter-contact times. A different approach involves controlling the mobility of nodes to alter network performance [73], but in this dissertation we assume that node mobility is externally controlled.

3.3.3 Contributions of this Dissertation

Intercontact routing [63] is perhaps the most similar to the work in this dissertation work in the sense that it uses the post disaster mobility model as well as graphs with stochastic weights; however the objective of this dissertation is to quantify the effect of excluding certain nodes from routing on the performance (i.e., a hardware based approach), and is not concerned with the number of transmitted messages.

While the above approaches aim to be awake for *every* contact, this dissertation has a completely different idea of *intentionally sleeping during node contacts* and examining the effect on routing performance. The work in [74] formulates the delay-energy tradeoff as a control problem, but there is no mention of Pareto optimality. It is also not known if the approach valid for non-probabilistic forwarding based protocols (i.e., where a packet is forwarded to a node based on probability p, a system parameter).

3.4 Mobile Self-localization Techniques

Existing work on localization can be classified as range-based/range-free or as anchor-based/anchor-free, although a few techniques do not fall cleanly into these categories. We review here a broad set of localization techniques, typically classified as range-based or range-free.

3.4.1 Range-based Localization Methods

These methods require an estimate of the distance or angle between two nodes to localize and may operate in both absolute and relative coordinate systems. GPS is a familiar range-based method that uses the time of arrival of signals from satellites to obtain a precise location in latitude-longitude format. Map stitching and graph embedding are used to localize nodes based on inter-node distances [75, 76] but are centralized and computationally intensive. Methods requiring specialized hardware include precise measurement of acoustic phase difference [77, 78], optical sensors/reflectors [79, 80] and time difference of arrival (TDoA) [81, 82]. Estimation based methods are also used to localize [83, 84]. [85, 86] uses surveying to predetermine RSSI values at any point in the area of deployment. Distance is inferred from RSSI through this data. Hybrid methods have also been used [83]. [87] uses precise infra-red ranging in combination with a grid based fuzzy logic approach. [88] proposes using RSS-Distance fuzzy rules to perform crisp localization, when there are anchors placed at the four corners of the deployment area. In [89], time-of-arrival and RSS data is fused together using Bayesian inference, following which fuzzy optimization is used to compute a crisp location. Typical drawbacks for these methods include higher computational loads, increased node size, higher energy consumption and increased cost. A lighter weight solution uses fuzzy logic to locate cellular phones in a hexagonal grid in a cellular network [90]. It assumes a fixed number of anchors but handles mobility very well. The computation and refining are not suitable for a resource-constrained computation platform like a MicaZ node. This was the inspiration for this work. A frequent requirement is the presence of at least three anchors so that necessary uniqueness and geometric constraints are satisfied.

3.4.2 Range-free Localization Methods

These methods are typically used in systems where connectivity is the metric of choice and actual geographic distance is less important. Hop counting is a technique frequently used in these scenarios, where the distance between two nodes is inferred from the number of hops a packet takes and is based on some assumed or measured average hop length [91]. A major drawback is that it fails in networks with irregular topologies such as those with a concave shape [92, 93]. Mobility incurs large overhead since all hop counts must be refreshed frequently. Centroid [94] performs GPS-free indoor localization by simply averaging the co-ordinates of the anchors each node hears. It typically requires high anchor density and fails in common geometric configurations, like when all anchors are on the same side of the node. APIT [95] is a similar method which divides the area of deployment into triangles formed by anchors and then estimates the location. It assumes a large anchor density and higher radio ranges for the anchor nodes. A hybrid method [96] that uses RSS and connectivity is worthy of note. Other state of the art methods [97, 98, 99, 100, 101, 102, 103] include those that use radio interferometry [104, 105, 106], use extra hardware [107] or are event based [108]. Analysis of the minimum achievable error has also been made [109]. [110] uses hypothesis testing to infer the location of a node, by using RSS PDF distributions gathered by anchors nodes through surveying. The fundamental difference between fuzzy logic and hypothesis testing is that while the former gathers learned intelligence and applies it to a given input, the latter tests all possibilities using tools like the Generalized Likelihood Ratio Test (GLRT).

3.4.3 Contributions of this Dissertation

Compared to the above three works, FuzLoc does not assume the presence of additional sensing capabilities [87] [89] and provides localization solutions when there

are less than three anchors [88] as well as computing the location as an area, a feature not found in previous work. In [110], building each PDF by surveying requires a large sample space. The computation involved in building the PDFs and performing the tests is not suitable for embedded devices unlike FuzLoc. The complexity associated with hypothesis testing increases with increasing number of anchors as well as the deployment area, since there are more tests to be conducted (with multiple tests, the accepted hypothesis from the previous test is used in the next test). With fuzzy logic, the number of rules increases with the anchor density but is independent of deployment area. Fuzzy rules require only storage while hypothesis testing requires computation and storage.

4. FOG COMPUTING DESIGN PRINCIPLES AND NETWORK ARCHITECTURE

In this section the design principles behind Fog Computing are discussed. These are general system design principles which we have adopted keeping in mind the requirements to be satisfied by Fog Computing as well as some practical issues affecting today's disaster response efforts. Next, system network architecture that implements these design principles is proposed. Each of the layers in this architecture has specific functions, which are described.

4.1 Design Principles

Based on first hand accounts of USAR deployments and responder requirements, we decide on a set of principles that govern our design of Fog Computing. A list of applicable system design principles can be found in [111].

1. Unmodified COTS Devices: Governmental organizations are increasingly adopting COTS devices because of the available support and software, at fairly economical prices as compared to a custom platform. In many instances, USAR responders have used their own personal iPhones during disasters to email photographs of rubble piles. In any case, one cannot assume a "jailbroken" device where one can have complete control, as is fairly common with hardware platforms used in academic research. Instead, the system has to be designed such that stock capabilities of popular COTS devices are sufficient, so that devices can be borrowed and setup easily. A custom routing protocol or user replaceable batteries are not possible on the iPhone, as an example.

2. Open Standards and Protocols: Standardized protocols, preferably of international scope, are emphasized. Certain WiFi channels are allowed in Japan but not in the USA; such issues should be planned for. At every layer of the system, open formats and widely supported protocols make integration of hardware with other international teams much easier.

3. App Oriented Design: Because complexity is pushed towards the application layer, updating the system becomes easy and does not need recompiling/reflashing the entire device, especially during disasters. When deployed on a large scale over a variety of heterogeneous devices, This design principle will significantly reduce roadblocks encountered in platform adoption. At the same time, simplicity in these complex apps is necessary: when a human-computer interface is present, having more than three buttons will cause the device to be left behind in a vehicle, instead of being used by first responders.

4. "Premature Optimization is the Root of All Evil": With Fog Computing we first build a proof-of-concept implementation that captures most of the required functionality, and then iteratively optimize the system based on deployment experiences. For example, we trade performance for simplicity in the source routing optimization, by using a simplistic vehicle movement model. The gained simplicity makes it easier to deploy Fog Computing as a whole with limited manpower, providing us with valuable experience which we can then use in the next iteration of the source routing protocol.

4.2 High Level Network Architecture

A network architecture for Fog Computing is shown in Figure 4.1. Being designed for COTS devices and open protocols, there is a necessity for incorporating heterogeneity at all levels. For example, there are multiple protocols being used at

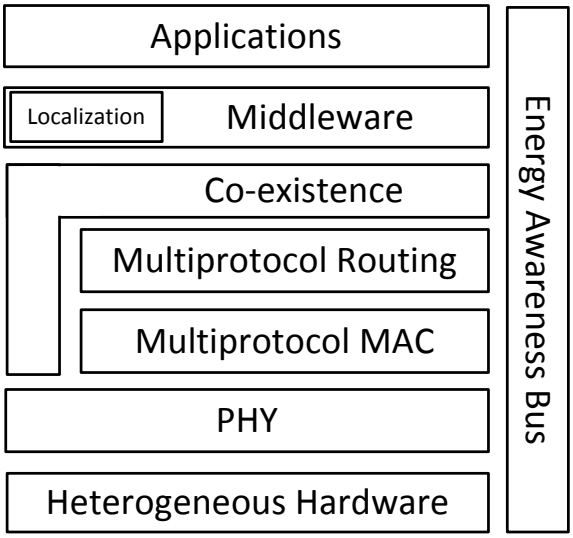


Figure 4.1: System architecture

the PHY, MAC and Routing layers. 802.15.4 is a popular PHY choice for low power sensing systems, whereas 802.11/WiFi is widely used in smartphones and COTS end user devices. In addition, a device may or may not have additional hardware like GPS and USB storage devices. All these components are important to the design of Fog Computing. A Coexistence layer manages and coordinates the various PHY/MAC/Routing protocols. *However, multiple protocol coexistence is not explored in this dissertation and is left as future work.* The middleware layer acts an abstraction layer to the Applications used in Fog Computing. It contains the QoS aware routing protocol that takes data from the applications and computes the node to which this data is to be transmitted (via the lower layers). A Localization layer provides precise location information that may be used as data quality enhancement by an Application, or by the Middleware layer to optimize the aggregate goodput of the system. For localization to work, it needs low level access to hardware. Energy Awareness is implemented as a cross-layer bus that interacts with all of the above layers. The bus can compute a work schedule for PHY given an expected network

usage estimate by the Middleware layer, which in turn creates its estimate from the installed Applications and their usage pattern.

The precise functions of each of these components is enumerated below:

1. Heterogeneous Hardware: Provide basic hardware functionality to the upper layers

2. PHY (Network Physical Layer): Provide access to the radio

3. MAC (Network Medium Access Control): Provide control primitives for the radio to access the physical medium, provide link layer robustness to the upper layers

4. Routing (Network Routing): Provide basic device addressing functionality, choose a link for data forwarding

5. Multiprotocol Coexistence: Optimize and tune each PHY protocol so that interference between multiple PHY protocols is minimized

6. Middleware: Provide Fog Computing API functionality to the applications, provide QoS aware routing functionality to the applications, provide waypoint placement functionality

7. Localization: Provide the geographic location of the host device

8. Applications: Provide an interface into Fog Computing for the user, provide file sharing and social networking services to the user

9. Energy Awareness Bus: Duty cycle each hardware component based on network performance requirements

5. NEW APPLICATIONS AND PROOF OF CONCEPT IMPLEMENTATION

Fog Computing provides new sensing and sharing modalities to first responders during the disaster recovery process. Some of the design requirements presented previously can be implemented as standalone applications, presented below, while some others need to be implemented system wide. The high level architecture from the previous section is then instantiated on three separate device classes that representing the whole spectrum of computing power, from ultra low power sensors to resourceful routers. Finally we show how each of these new applications can be implemented on one or more of the device classes according to resource requirements.

5.1 New Motivating Applications for Fog Computing

We now describe the conceptual working of seven new motivating applications designed to satisfy some of the Fog Computing requirements. After describing the hardware architecture in the next section, the implementation details of these new apps are discussed.

5.1.1 File Sharing App

The app oriented architecture of Fog Computing enables third parties to add piecewise functionality without worrying about lower layer functions like routing or storage. As an example, a file storage and social networking service for first responders can be very useful and improve situational awareness. A traditional file storage service in the cloud has the following functionality and properties: i) The

Parts of this section reprinted with permission from "Distressnet: A disaster response system providing constant availability cloud-like services" by H. Chenji, W. Zhang, R. Stoleru, and C. Arnett. Ad Hoc Networks, vol. 11, no. 8, pp. 2440-2460, 2013. Copyright 2013 by Elsevier.

ability for a client to upload a file to the cloud, without specifying a destination node; ii) The ability for a client to retrieve a file, without specifying the location of the file; and iii) Data robustness due to intelligent replication performed by the cloud service back end. The file sharing application allows authenticated users to share data with other users, or groups of users. As an example, one can imagine a team of responders sharing the layout of an explored building along with current hazards, with other nearby teams. An important feature that emphasizes the need for such a service is that the destination for data sent is unknown - it is simply stored in the Fog and accessed by anyone who connects to the Fog Computing infrastructure.

The File Sharing App can be seen as a client/front end, designed to be used by a first responder, that connects to a File Sharing Service server/backend and authenticates itself. The clients can then use the Fog API to ADD/DELETE/MODIFY files which they own (as determined after authentication), as well as specify availability metrics depending on the importance and criticality of the data file. They can also share selected files with other clients or users. Examples of files that users can upload include video taken using a smartphone's camera. Users can also specify whether they want to backup these files to an external storage provider like Amazon S3 or Flickr. If a client wishes to use their own external account, an encrypted query, using the provider's API (S3's API or Flickr's API), is sent to a special device that has both the File Sharing Service as well as internet access. The HTTP header and body is intercepted by the File Sharing Service and sent to the internet Gateway using the Fog's underlying network.

5.1.1.1 Fog API

The Fog API insulates the user, first responders in this case, from the inner workings of the underlying network. It aims to provide a service similar to those

offered by cloud storage services like Dropbox and Amazon S3. Three primitives called ADD, DELETE and MODIFY provide an interface into the Fog. The ADD primitive uploads a file into the Fog. This file could consist of a tweet as produced by the Social Networking App (to be presented next), or a photo from the user's phone as produced by the File Sharing App. The MODIFY/DELETE primitives allow users to modify or delete files that they own.

When an ADD request is sent from the File Sharing App to a nearby device running the File Sharing Service, the file is first transferred locally to the device. Then, the file is replicated on multiple Fog devices according to the criticality of the data. The actual Fog devices that are chosen as endpoints depend on the output of several algorithms to be presented later in this dissertation. A MODIFY operation causes the File Sharing Service to send the difference between the current version of the stored file and the new incoming file to the Fog devices which contain the original. These devices will locally modify their copy of the file and push it to another user's File Sharing App upon connection. The DELETE operation simply sends a low overhead message to the Fog device which says that the local copy of the file on the device should be deleted. The synchronization between the devices running the File Sharing App and File Sharing Service occurs as follows: when an App discovers a Service nearby, it can supply a list of files stored locally and ask for changes to those files. The Service then replies with a list of changes, which the App can apply to its local copy.

5.1.1.2 Security

API level security is available if the user chooses external service providers - thus providing encryption on an end to end basis for the user. Users need not disclose their existing external credentials in order to use the Fog. We consider Amazon

S3 as an example of an external RESTful cloud storage service provider. Files are uploaded to S3's servers using a published API which offers both a REST and a SOAP interface using XML. When a user signs up, a secret key is assigned. This secret key is then used alongside HMAC-SHA1 in order to authenticate all HTTP (optionally, HTTPS) requests. Whenever a user wishes to place a file in S3, a challenge string is first constructed based on a predefined ruleset, which then serves as the "message" in HMAC-SHA1. The output, which is a base64 encoded string, forms part of the HTTP request header. The entire HTTP header and body (if applicable) is then sent to Amazon by the client.

5.1.2 Social Networking App

The Social Networking App is a user client that provides services like inter-team messaging, and allows victims to publicly tweet their status. Users can communicate with other users, or mass message other teams or team members. They can also choose to post messages to an external Twitter account. It is important to note that such messages will be available in the Fog, as well as on their Twitter profiles - thus, an external service like Twitter is mirrored in the Fog. All data from Twitter is pulled regularly by an internet gateway and sent to the Fog (Figure 5.1) so that the data stays synchronized. Primitives specific to social networking, such as "following" a user or replying to a messages, are handled by a Social Networking Service that runs on the same device as the File Sharing Service, such that the Fog API is used by the Service to synchronize data.

5.1.3 Self-Localization App

The Self-Localization App is designed for use by low power mobile nodes in GPS denied environments. For example, a team of responders investigating a collapsed building can deploy a Sensor on each member as well as on tools of importance, like

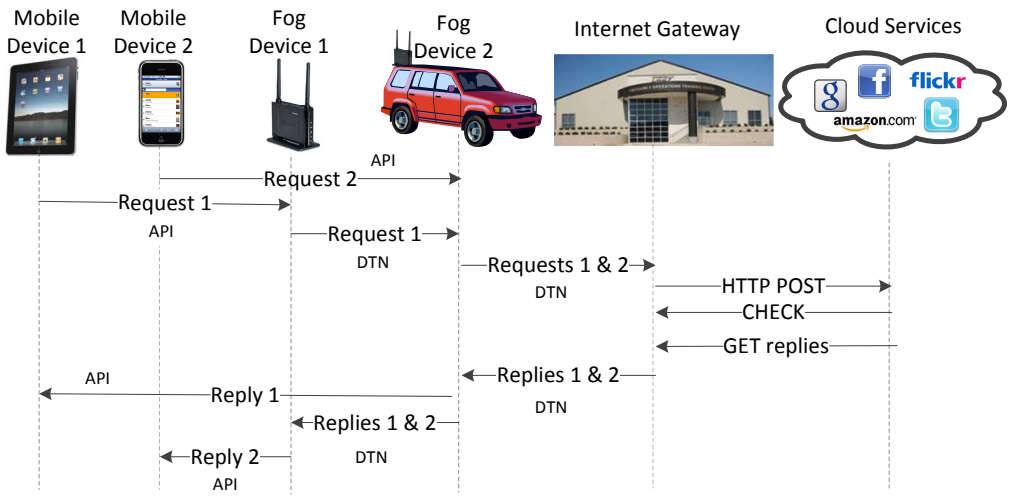


Figure 5.1: Sequence diagram for accessing cloud and social networking services (e.g., Twitter).

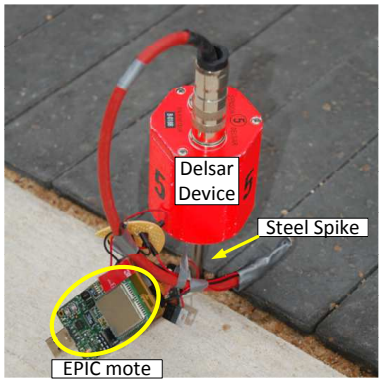


Figure 5.2: The Delsar Life Detector attached to a Sensor class device.

a cement saw. In the event that a tool is misplaced but still within radio range, it can be easily located using this app. The mathematical framework behind this app is discussed in Section 9.

5.1.4 Vibration Sensing App

The FEMA US&R equipment cache list [6] mentions Delsar Life Detection sensors: a steel spike (Figure 5.2) that is driven into rubble which responders can then

Algorithm 1 k-NN Classifier

1: **for** each $s_i \in s_1 \dots s_{gn}$ **do**
2: Compute $d_i \leftarrow \sqrt{(F_1 - s_i^{f_1})^2 + (F_2 - s_i^{f_2})^2}$
3: **end for**
4: $r_1 \dots r_k \leftarrow$ The k smallest d_i
5: $groups \leftarrow$ Union of groups that each of $r_1 \dots r_k$ belong to
6: $G \leftarrow$ most common group in $groups$

monitor for voices or knocks from victims. Upon manually probing the rubble at different places, the victim can be localized and rescue operations can commence. However, there are sources of noise like footsteps and vibration from nearby vehicles which are also picked up. The goal of the Vibration Sensing App is to automatically detect and classify the source of vibration. To profile these sources the steel spike of the sensor was driven into a small wedge in a pavement outside our building on campus. Three sources of noise/data were profiled: a stone dropped from a height, footsteps of pedestrians and a knock made by a hammer on a pavement. The fixed-point in-place 1024-bin FFT is shown in Figure 5.3.

It is important to note that the amplitude of the signal alone cannot be used to classify a source. Hence, two features were extracted from the FFT: (i) average value of the frequencies weighted by their respective amplitudes (f_1) and (ii) the mean amplitude of the frequencies (f_2). We then used these features in a simple KNN (k-nearest neighbor) classifier. Suppose that we have g different types of data $G_1 \dots G_g$ and n samples for each group, for a total of gn samples $s_1 \dots s_{gn}$. Let each sample be a vector consisting of two features $[f_1, f_2]$. The KNN classifier first needs to be trained using these samples. Training consists of storing each sample and its corresponding group in memory. Now, given a new sample $S = [F_1 F_2]$ that needs to be classified, Algorithm 1 can be used to calculate the group G that S belongs to,

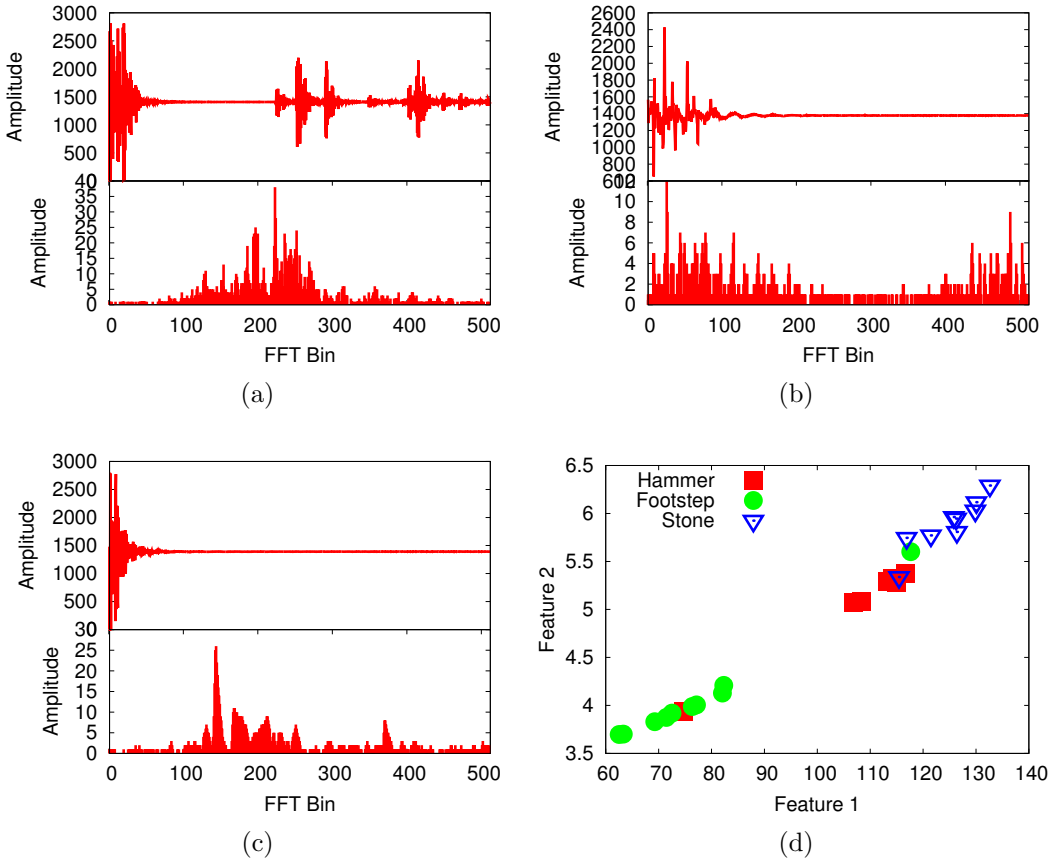


Figure 5.3: Spectrum and signal of (a) stone drop, (b) footstep and (c) hammer strike. (d) shows the classifier results based on 2 features

thus identifying the source of the vibration.

5.1.5 Team Separation Detection App

USAR operations in an unexplored large areas with low visibility and potential hazards (e.g., collapsed tunnel, chemical spills) are dangerous. Any incidents involving team member separation or loss of vital tools (a "cut" in the network) can slow down the victim rescue process because of unnecessary delays. To meet the need for a separation detection method, we develop an app that enables each team member to monitor connectivity to a team leader *multiple hops away*, and warns a team member

of physical separation from the team leader.

The Team Separation Detection App is inspired by the distributed cut detection algorithm presented in [112]. A node in an electrical network containing a current source will see a change in its potential when there is a partition in the network, enabling it to detect changes in network topology. Similarly, every node n in a computer network maintains a positive scalar value called the "state" $st(n)$, updating it using the formula $st(n) = (\sum_{N(n)} st(i))/(|N(n)| + 1)$, where $N(n)$ is the set of one hop neighbors of node n. A special source node S injects a value I by updating its own state using the formula $st(S) = \frac{I+\sum_{N(S)} st(i)}{|N(S)|+1}$. The state of each node converges, given a network topology. A node in the same partition as S after a cut will see its state converge to a higher value, otherwise it drops to zero. The convergence time is fast and the maximum delay in experiencing such a change is bounded, as shown in [112]. Thus, this algorithm helps first responders detect accidental separation in the team by using only one hop communication.

5.1.6 Building Monitoring App

The primary motivation for the Building Monitoring App was the current state of art, where USAR personnel use paint on walls (Figure 5.4) to store information about the current search status of a structure. This includes information like the number of survivors inside, the location of chemical hazards if any and the most recent date/time that the structure was searched [113]. This information is most likely to remain constant and not change very often. A Building Monitoring Service runs on low power sensors and stores the data programmed into it by using a client front-end like the Building Monitoring App. One or more of these sensors can be deployed in or around a building. Vehicles in the vicinity can automatically gather data from these sensors and mule it using other vehicles to the EOC.

Figure 5.4: Markings indicate that the buildings have been searched

5.1.7 Sink Election App

Imagine that the Building Monitoring App presented previously has been installed on several sensors which are deployed around a disaster struck building. Not all of the senors in this sensor subnetwork may have access to passing vehicles due to the topology. It is therefore necessary to choose a single sensor from this subnetwork to act as a data aggregator which can offload data to passing vehicles. A set of sensors running the Sink Election App can be modeled as a set $M = \{m_1 \dots m_n\}$ of deployed sensor nodes. Let $batt(m_i)$ be the residual battery charge of node m_i, $beac(m_i)$ the number of beacons received from vehicles passing by, and $uniq(beac(m_i))$ be the number of unique vehicles encountered by m_i. Now, Let $sc(m_i) > 0$ be a scoring function that assigns the eligibility score to any node m_i based on the above three parameters. The sink selection problem can be formulated as the choosing of a

Algorithm 2 Sink Election

```
1: procedure SinkElec(For node i)
2:     sc(sink) ← 0
3:     if ScoreTimerFired then
4:         Calculate sc(m_i)
5:         if sc(m_i) > sc(sink) then
6:             Broadcast sc(m_i) to DAGRoot
7:         end if
8:     end if
9: end procedure
10: procedure SinkElecRoot(For the DAGRoot)
11:     if ElecTimerFired then
12:         sink ← id(Max(receivedscores))
13:         Broadcast sink to all children
14:     end if
15: end procedure
```

particular m_i such that we:

$$\text{maximize} \qquad sc(m_i) \wedge uniq(beac(m_i)) \tag{5.1}$$

$$\text{subject to} \qquad beac(m_i) > 0 \tag{5.2}$$

$$batt(m_i) > 0 \tag{5.3}$$

The solution is simple: a sensor which has access to the most number of unique vehicles is given higher priority during sink election. We make use of RPL's routing tree structure to perform sink election and resolve conflicts among candidates. This ensures that there is no additional messaging overhead (e.g., for neighbor discovery), while at the same time providing unicast communication in the subnetwork.

Algorithm 2 shows the distributed sink election using RPL's tree structure. The scoring function sc used was $sc(m_i) = 3 \times uniq(beac(m_i)) + 2 \times beac(m_i) + batt(m_i)$. This simple, computationally inexpensive function gives more weight to the number

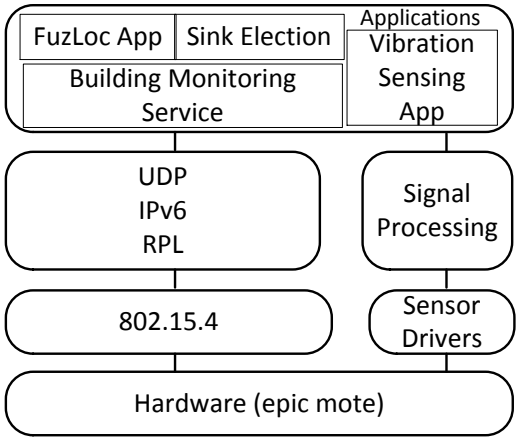


Figure 5.5: The hardware/software architecture of Fog Sensors.

of unique vehicles seen by the node, while at the same time preferring nodes which frequently see vehicles. Nodes, after computing their score regularly (Step 4) and comparing it with the current sink's score (Step 5), send the score to the root of the DAG structure built by RPL. Because frequent sink changes are considered resource intensive, the root conducts election only at specified intervals. During this election process, potential candidates are compared and the node with the highest score is chosen (Step 12). In case of a tie, the node with the lowest id is chosen. The identity of the new sink is then broadcast to each child node, till the leaf nodes are notified as well (Step 13).

5.2 Hardware/Software Architecture

Here we discuss how the above high level architecture can be implemented on various categories of devices. There exist three distinct classes of devices in the Fog: Sensors, Smartphones and Routers. We now describe the hardware used to implement these three classes, following by a listing (Table 5.1) of the actual devices used in Fog Computing, some of which may represent multiple device classes depending on their functionality.

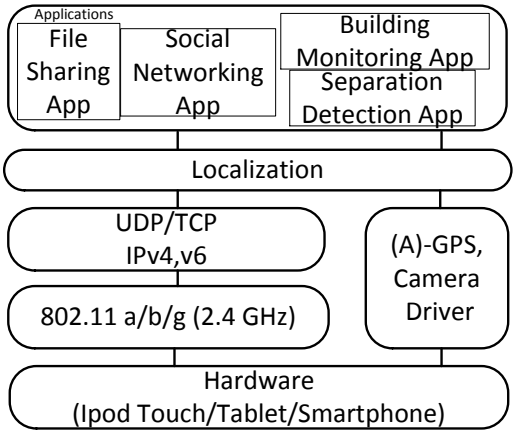


Figure 5.6: The hardware/software architecture of Fog Smartphones.

5.2.1 Fog Sensors

Sensors are typically low power, battery powered wireless sensor network platforms such as an EPIC [114] mote. The data sensed may be either mission critical or informational. Being heavily duty cycled, they are designed to last for several weeks with a single charge. 802.11 support is rarely found on these devices, with 802.15.4 or no networking being more common. Applications deployed on such devices include sensing and services (Building Monitoring Service, Sink Election App, Vibration Sensing App, and Self-Localization App in Figure 5.5). The BLIP stack provides UDP connectivity over the IPv6 provided by 6lowpan. In Fog Computing, we use RPL [115], an IPv6 routing Protocol for low power and lossy networks, as the default routing protocol. Fog Sensors can upload data into the Fog using a Fog Router as a proxy (to be described shortly).

5.2.2 Fog Smartphones

Smartphones refer to popular network centric consumer electronics like tablets, smartphones and laptops which have networking capabilities, but have limited re-

sources. Fog Smartphones are primarily used by first responders to access data stored in the Fog. They provide a rich interface to the data collected in the field, while also providing some functionality themselves. Not as resource constrained as Fog Sensors, most devices have 802.11 capability and are designed to last a few days on a single charge. Smartphones also have various sensors such as GPS, cameras, microphones and accelerometers. They are usually incapable of routing or advanced networking capabilities and have limited, but not scarce, resources. Apps installed on these devices are more of data consumers than data generators. The hardware platform used was the iPod Touch as well as the iPad. We were limited to the application layer since the SDK does not allow non-trivial modifications to the operating system for security reasons. The network stack on these devices consists of TCP/UDP over IPv4/v6 and 802.11. The fact that 802.11 IBSS mode was readily supported out of the box made us choose iOS over Android.

Fog Computing apps and services that run on Fog Smartphones including the Building Monitoring App (BTag App in Figure 5.6) that is used to program Fog Sensors running the Building Monitoring Service once they are deployed with relevant information. The File Sharing and Social Networking Apps (as shown in Figure 5.6) are also implemented on Smartphones, as is the Team Separation Detection App.

5.2.3 Fog Routers

Routers are portable, battery powered devices which provide basic wireless networking functionality and are deployed in the field. An example is a common 802.11 router found in most homes today. They can be assumed to have expansion ports to provide additional functionality like persistent storage or cellular connectivity. These can either be static or deployed inside a vehicle. The hardware platform used in the Mikrotik RB433UAH routerboard which has 3 MiniPCI slots and 2 USB 2.0 ports,

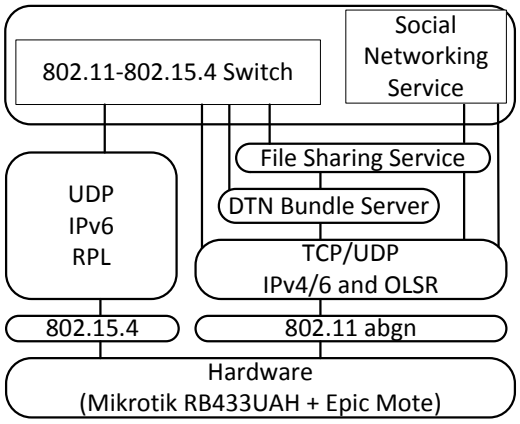


Figure 5.7: The hardware/software architecture of Fog Routers.

allowing for two 802.11abgn wireless cards configured for 2.4 and 5 GHz respectively. It also has 512MB of NAND flash and 128MB of RAM. OpenWRT [116] is an open source GNU/Linux based operating system compatible with this router, which was chosen because of the openness and the wide range of software and support available. The USB port can be used to provide functionality like a new physical layer such as 802.15.4 (to communicate with Fog Sensors), enhanced storage like a USB flash drive or both. The networking stack used in is 802.11abgn below IPv6/v4 and UDP (Figures 5.7). Since most COTS WiFi compliant devices support only the 2.4GHz band, we decided to use the 5GHz interface exclusively for routing. DHCP is provided on the 2.4GHz interface for clients to connect. All routers have statically assigned IPs - router n has an IP of 192.168.50.n for its 5GHz interface and 192.168.n.1 for the 2.4GHz interface. Each router can handle 255 end user devices - they are assigned IPs in the 192.168.n.0/24 range.

Fog Routers employ delay and disruption networking. For implementing DTN functionality, we used the IBR-DTN [117] implementation which is readily available as a package for OpenWRT. A "Bundle" (RFC 5050) is the primary data unit in

DTN. Each DTN node is identified by a URI like $dtn://dn.zigbeegateway1$. A client application with an ID of $mote1$ can connect via an API to the local "Bundle Server". Then, any traffic intended for this app will simply need to be addressed to $dtn://dn.zigbeegateway1/mote1$. Examples of functions provided by the bundle server API includes registering the application name (e.g., "mote1"), setting a destination, requesting encryption or authentication or custody transfer and setting the lifetime. All communication in the DTN layer can be encrypted and authenticated, as defined in RFC 6527. Each DTN node first generates, pre-deployment, a 2048-bit RSA public/private key pair. The public keys of all nodes are aggregated and shared among all DTN nodes. This data is used for bundle encryption in a public-key cryptography fashion. It is to be noted that bundle encryption is always between source-destination and authentication is always on a single hop basis. Bundle authentication uses the HMAC-SHA1 message authentication cipher, which encrypts a message based on a key. In this case, the key is a pre-shared plaintext key of arbitrary length that is different from the public/private keys.

DTN is implemented as an overlay network of nodes where multiple local clients can connect to a local DTN server (Bundle server in Figure 5.7) in the application layer. A special DTN app on the router which can talk to 802.15.4 based Fog Sensors as well as the DTN server provides DTN proxying functionality ("802.11-802.15.4 Switch" in Figure 5.7). The File Sharing and Social Networking Services have the capability to interface with Fog Smartphones by acting as a Fog API provider, while simultaneously talking to the Bundle server in order to replicate data on other Fog Routers.

5.3 Fog Computing Device Listing

A comprehensive listing of all the Fog Computing devices is available in Table 5.1.

Component	**Class**	**PDMM Function**	**Installed Apps/Services**
BTag Sensor	Sensor	None	Building Monitoring Service, Sink Election App, Self-Localization App
Seismic Sensor	Sensor	None	Vibration Sensing App, Self-Localization App, Sink Election App
Smartphone	Smartphone	Mobile Agent (USAR)	File Sharing App, Social Networking App, Building Monitoring App, Team Separation Detection App
Data Waypoint	Router	Center	None
Base Station	Router	Center	File Sharing Service, Social Networking Service
Vehicle Node	Router + Sensor	Mobile Agent (Ambulance, Patrol Car, Supply Vehicle, Volunteer)	File Sharing Service, Social Networking Service
Sensor Proxy	Router + Sensor	Center	File Sharing Service, Social Networking Service

Table 5.1: Summary of the devices in the Fog Computing Architecture, their corresponding device classes, their mobility pattern according to the PDMM and the installed Apps and Services.

6. INFRASTRUCTURE OPTIMIZATION IN FOG COMPUTING

The Fog Computing architecture [118] consists of battery powered devices that have been deployed by responders over a large geographical area. Mobility in the area is limited and unpredictable, while the contact bandwidth is finite. The inter-contact time can be measured in tens of minutes or even hours. The opportunistic nature of routing means that the optimal use of each contact opportunity is crucial in ensuring high performance. In this section we investigate how the aggregate throughput in the network, an important QoS metric, can be increased by increasing the number of contact opportunities by statically placing additional hardware (Fog Router class devices) in the area. These devices, called Data Waypoints (Table 5.1), act as caches that can be used by mobile nodes in the area to store data. By carefully placing devices at optimal locations, like those that are frequented by vehicles, one can increase the number of contact opportunities.

6.1 Introduction

First we introduce the data flow model in Fog Computing so that the data workload can be quantified. The unique nature of Fog data flows, where the destination is simply the Fog and not a concrete address consisting of a hostname, is expressed in the form of firm and potential flows.

6.1.1 Data Production and Consumption Model

The main idea behind servicing apps in Fog Computing, as shown in Figure 6.1, is to mirror a traditional external service (like Flickr, Amazon S3 or Twitter) in

Parts of this section reprinted with permission from "Distressnet: A disaster response system providing constant availability cloud-like services" by H. Chenji, W. Zhang, R. Stoleru, and C. Arnett. Ad Hoc Networks, vol. 11, no. 8, pp. 2440-2460, 2013. Copyright 2013 by Elsevier.

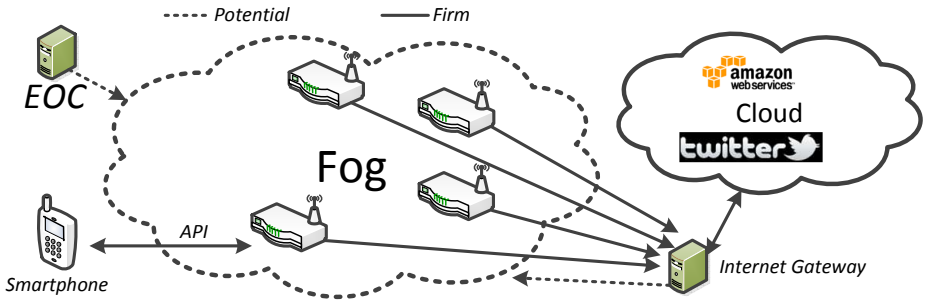


Figure 6.1: Firm and potential flows in Fog Computing.

a local network, containing a mixture of mobile and static vehicles and devices. Mobile teams visit multiple points of interest (Centers in PDMM parlance) in the affected area. Devices may be deployed at various locations/points of interest for unsupervised monitoring of the environment - for example, a Seismic Sensor along with a Sensor Proxy (Table 5.1). These points of interest eventually act as sources and *destinations* of data in the Fog. As shown in Figure 6.1, special points of interest are the EOC, and an internet gateway (a Base Station in Table 5.1) which may or may not be co-located with the EOC. The internet gateway provides access to traditional cloud services, but using a high cost link. Internet access for large amounts of data may be high cost and resource demanding. Therefore, and this is a key idea in Fog Computing, the remote cloud's services are provided locally in the Fog (i.e., the instantiation of a cloud in the intermittently connected Fog). Because accessing services directly from the Cloud incurs a high cost, if at all possible, it is much more efficient to instead access services from the Fog.

6.1.2 Firm and Potential Flows

As shown in Figure 6.1, there are several data flows (source-destination pairs). Some flows are *firm flows*: these have a pre-decided destination and source, while other flows are *potential flows*. These potential flows represent data to be stored

somewhere in the Fog, without a specific destination. For each potential flow, there is an associated *availability metric* which ranges from 0-100% and denotes the importance of the data. An availability of 100% means that data will be available on all Fog devices, whereas an availability of 25% means that data will be available on at most a quarter of Fog devices. More critical data will have higher availability. Examples of data and its availability metric include data generated by BTag Sensors (Table 5.1), which is not critical and has an availability of 33% (i.e., the BTag Sensor Data will be probabilistically stored on about 30% of existing destinations in the Fog), and data generated by Seismic Sensors (Table 5.1) which is critical and requires 100% availability (i.e., the Seismic Sensor Data will be stored on all existing destinations in the Fog). These firm and potential flows can be specified by the user.

6.2 Problem Formulation

Suppose that the current scenario is represented by an instance of the Post Disaster Mobility Model (PDMM). Based on the movement patterns of Mobile Agents in the area as well as data flows in the Fog, a set of candidate locations for placing Data Waypoints can be computed. From this candidate set, an optimal set that maximizes the aggregate throughput can be computed. For ease of modeling, we assume a reduced version of the Post Disaster Mobility (PDM) Model [4] that consists of only Patrol Cars. The newly placed Data Waypoints will be treated as Centers in PDMM parlance (Table 5.1).

6.2.1 Preliminaries

Consider n patrol cars $V = \{V_1 \dots V_n\}$, with the path of each vehicle being a loop and hence representable by a closed polygon. For a vehicle $v \in V$, let this polygon be called $Path(v)$, the speed of the vehicle be $Speed(v)$, the time taken by a vehicle

to go from point A to point B both on $Path(v)$ and along it, be

$$Time(A, B, Path(v)) = Dist(A, B, Path(v))/Speed(v)$$

Let the set of data sources be S and the set of destinations, D. Each source or destination is compulsorily present in the set of Centers of the PDMM instance. Any deployment additionally has a set of firm flows F, with each flow $F_i \in F$ having a data source $F_i^{src} \in S$, a destination $F_i^{dst} \in D$ and the size of the data F_i^{data} that can be sent from the source to the destination. Note that a node may act as a source as well as a destination in different flows. A similar set of potential flows P is also defined, but each potential flow $P_i \in P$ has only a source $P_i^{src} \in S$, an availability $0 < P_i^{avail} < 1$, the maximum data size P_i^{data} but no destination. We construct the set of modified potential flows Q, from P such that each flow $P_i \in P$ is assigned each destination $d \in D$. Let $Z \subseteq (F \cup Q)$ be the final set of selected flows such that:

- Every firm flow is included, i.e, $F \subseteq Z$
- The availability of each potential flow is satisfied. Mathematically, for every $P_i \in P$, there are at least $|D| \times P_i^{avail}$ flows in Z, for which the source is P_i^{src}, chosen from Q.

A "waypoint" is a router placed at the intersection of the paths of two vehicles $v, w \in V$ such that data can be dropped by v and picked up by w or vice versa. Let X be the set of all possible waypoint locations (which is where ever the paths of any two vehicles intersect). A solution set for each flow $Z_i \in Z$ means a sequence of alternating vehicles and data waypoints that are capable of carrying data from the

source to the destination, i.e, a set

$$\{Z_i^{src}, v_i^1, x_i^1, v_i^2, x_i^2 \ldots Z_i^{dst}\}, v_i \in V \text{ and } x_i \in X$$

The time taken for data to flow using the solution set for a Z_i will then be

$$T(Z_i) = T(Z_i^{src}, x_i^1, Path(v_i^1)) + \cdots + T(x_i^n, Z_i^{dst}, Path(v_i^n))$$

The maximum size of data that can be transferred on a flow depends upon the *Data Transferred per Contact (DTC)* quantity of each contact in its solution set: Z_i^{data} can now be defined as

$$Z_i^{data} = min(contact(v_i^1), contact(v_i^2) \ldots contact(v_i^n))$$

where $contact(v)$ is the maximal DTC that is possible between a node and the vehicle v traveling at a speed $Speed(v)$. The DTC is determined experimentally, as explained in the next section.

The Waypoint Placement problem is now defined as follows - given an upper bound X_{max} on the number of waypoints (e.g., limited available hardware), we choose $X^* \subseteq X$ such that:

- For each flow $Z_i \in Z$, the solution set contains vehicles which are found in V and waypoint locations which can be found in X^*
- The aggregate throughput $\sum_{z \in Z} \frac{z^{data}}{T(z)}$ is maximized
- Optionally, the cardinality of X^* is less than X_{max}

To efficiently solve the above problem, we first create a representative graph G. To construct this graph, first create a vertex for each unique source and unique

destination. Next, create a vertex for every possible waypoint location $x \in X$. Draw an edge between any two vertices m_i and m_j whenever a single vehicle v passes through both the vertices. The key intuition here is that the weight of this directed edge is nothing but the time taken by the vehicle v to physically transport the data from m_i to m_j. In order to account for the periodic movement of vehicles, let the time taken by v to arrive at m_i, called $arr(v)$ be a function of $Path(v)/Speed(v)$. The weight of the edge in question will then be $arr(v) + Time(m_i, m_j, Path(v))$. $T(Z_i)$ can then be modified to take this into account by adding $arr(v)$ to each term appropriately.

Let a binary selection vector $\mathbf{c} = [c_0, c_1 \ldots, c_{|X|}]$ denote whether a possible location $x_i \in X$ is chosen to be a data waypoint (1) or not (0). Let the subgraph G^* denote the graph formed by G by removing vertices indexed by a 0 in $\mathbf{c}$. The problem is now to find a binary vector $\mathbf{c}$ such that, operating on G^* we:

$$\text{maximize} \qquad \sum_{i=0}^{|Z|} \frac{Z_i^{data}}{T(Z_i)} \tag{6.1}$$

$$\text{subject to} \qquad T(Z_i) \neq \infty \tag{6.2}$$

$$\sum_{i=0}^{|X|} c_i \leq X_{max} \tag{6.3}$$

Constraint 6.2 ensures that there is always a path in G^* between the source and destination for each flow. This is because the delay for a nonexistent edge will be set to inf, or equivalently, G^* can be made fully connected with the newly created edges having a very large weight. Constraint 6.3 ensures that the number of waypoints deployed is less than or equal to the maximum possible. This problem can be recognized as a binary integer programming problem. Popular heuristics include the branch-and-bound algorithm, which is available in MATLAB as `bintprog` or

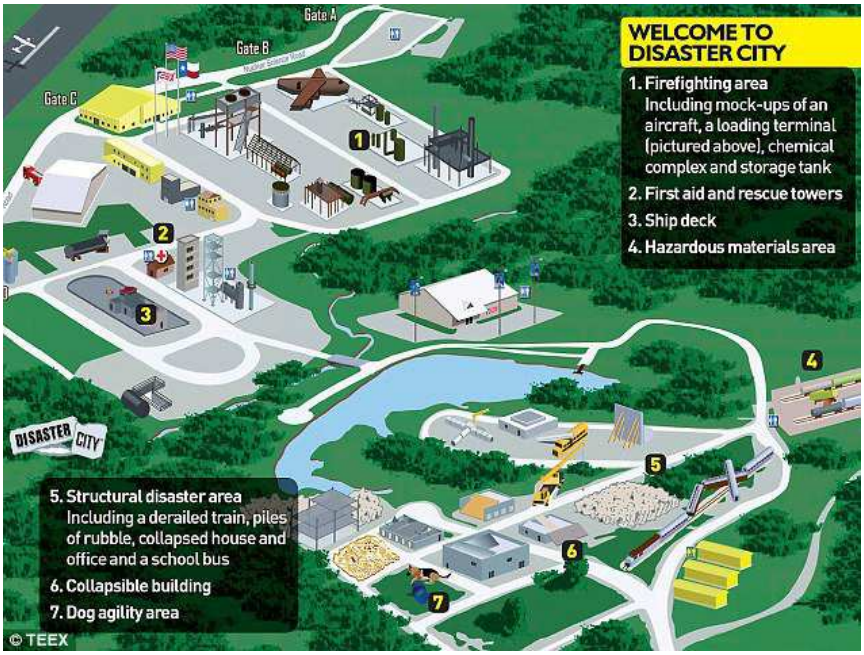


Figure 6.2: Schematic showing the layout of Disaster City

various algorithms available in ILOG/CPLEX.

6.3 Performance Evaluation

In this section we present the performance evaluation of some of the Fog Computing Apps as well as the waypoint placement algorithm. The accuracy of the Vibration Sensing app has been evaluated in Disaster City (Figure 6.2), which is a comprehensive 52-acre training facility for emergency responders with extremely realistic wrecks, including several rubble piles of wood and concrete.

First, we describe how the Fog can be deployed in Disaster City, following a hypothetical disaster. The deployment effort needed from first responders, as well as the data needed from the deployment area is discussed. We progressively evaluate the system and the service internals by first finding the optimal payload size (DTC) for 802.11 and 802.15.4 transfers for a given speed. Then, after verifying the correctness

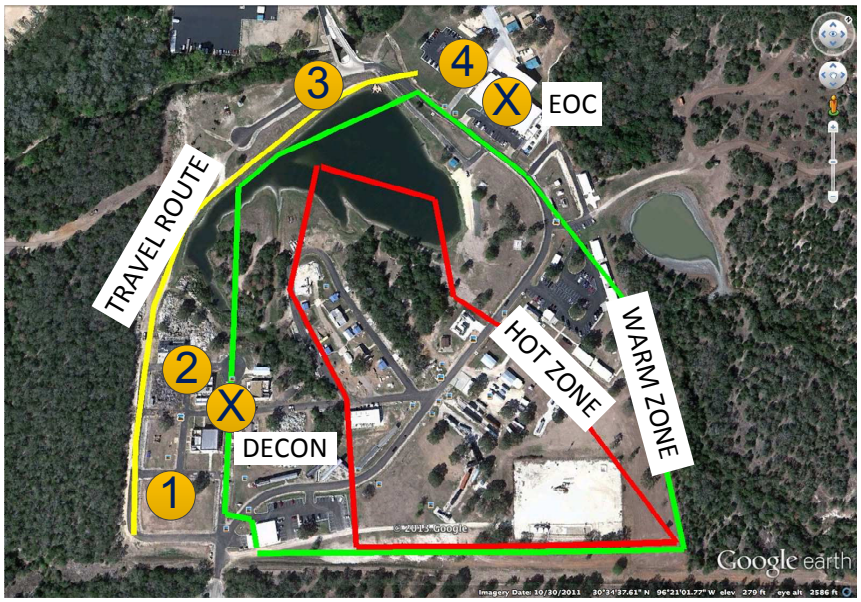


Figure 6.3: Map of deployment at Disaster City during Summer 2012

of the Sink Election algorithm using EPIC motes, we perform a simple experiment (Expt 1) which does not attempt to optimize the DTC. A simulation helps us choose the best routing protocol for this experiment. In a second deployment (Expt 2) we then show the improvement in aggregate goodput even in the presence of DTN security overhead, by the use of optimal payload size, source routing and optimal Data Waypoint placement. These experiments involved three vehicles and various data sources, and was conducted on our campus. The accuracy of the Vibration Sensing App w.r.t classifying the source of noise is shown.

6.3.1 Deployment Scenario

The Fog was deployed in Disaster City during Summer 2012 as part of an exercise involving first responders. The objective of this deployment was to test the practicality of deploying during a disaster, and not to test the performance over many days. At 10 A.M. on May 17, the Incident Commander convened a meeting

at the Command Post (EOC in Figure 6.3). Participants consisted of various groups specializing in technology such as UAVs, ground robots, marine robots and delay tolerant networking. The situation summary was as follows:

> A major train derailment involving an estimated 12 cars of a freight train has occurred in the SE sector of Disaster City, TX. The derailment is adjacent to a producing oil well, underground oil distribution line, above and below ground power lines, commercial and private residences. Unknown at this time: damage to utilities and infrastructure but power is out in the area. Unknown if any car(s) are leaking but residents in the area report burning eyes and difficulty breathing per EMS/Fire dispatch. Unknown casualties. Engineer stated that train had radiological cargo; however the manifest has not yet been obtained. A full RECON by HAZMAT and aerial surveillance must be conducted in order to establish appropriate DECON. Winds are out of the north west. Initial HOT Zone has been established, see map. Any RECON by personnel shall be coordinated with HAZMAT. Responder medical and Command post located next to EOTC.

The area of the disaster was 0.081 square miles. No human movement was allowed in the HOT zone due to danger of contamination. Any robot entering the HOT or WARM zones had to be decontaminated at DECON before being handled by a human. The objective of our team was to provide networking in the area, so that data from the DECON area (DECON in Figure 6.3) could reach the command post. The only travel route to the DECON area was through a dirt road (TRAVEL ROUTE in Figure 6.3). Additionally, data from the marine robots (location "3" in Figure 6.3) also needed a path to make it to the Command Post. Location 2 had

a collapsed strip mall that the US&R team would have investigated. After a group huddle, we decided to deploy four routers in the area (locations 1,2,3,4 in Figure 6.3). Location 1 was a Data Waypoint, location 2 was a Sensor Proxy due to the presence of BTag Sensors nearby, and location 3 was a Data Waypoint. A Base Station (4) was placed at the EOC. The entire deployment effort took about two hours including travel time. Deploying a node was easy, since they were all mounted on tripods and were battery powered. BTag Sensors were attached to the strip mall rubble using duct tape and were programmed using Smartphones/BTag App carried by our team. Additionally, a few pictures were sent using Smartphones/File Sharing App from the node at location 1 to the Base Station 4 at the EOC. Service optimization using source routing was not performed due to the limited time available and the small size of the deployment area. Epidemic routing was used to route packets. The entire experiment, including setup and debugging took about six hours.

6.3.2 Optimizing the DTC: 802.11 Based

The amount of data transferred between a static and mobile node is dependent on the size of the payload and the speed of the vehicle, keeping other factors constant. This important observation that the data transferred per contact (DTC) depends on the payload size enables us to design better data aggregation algorithms. Two separate experiments show the effect of speed and payload size upon both 802.11 and 802.15.4 transfers.

6.3.2.1 Effect of Vehicle Speed

The DTC for Wifi contacts is shown in Figure 6.4(a,b). We observe that as speed increases, the DTC decreases, for both secure and insecure transfers. This is to be expected since increased mobility results in lesser contact time and also degradation of link quality. Considering a base speed of 15mph, an increase of

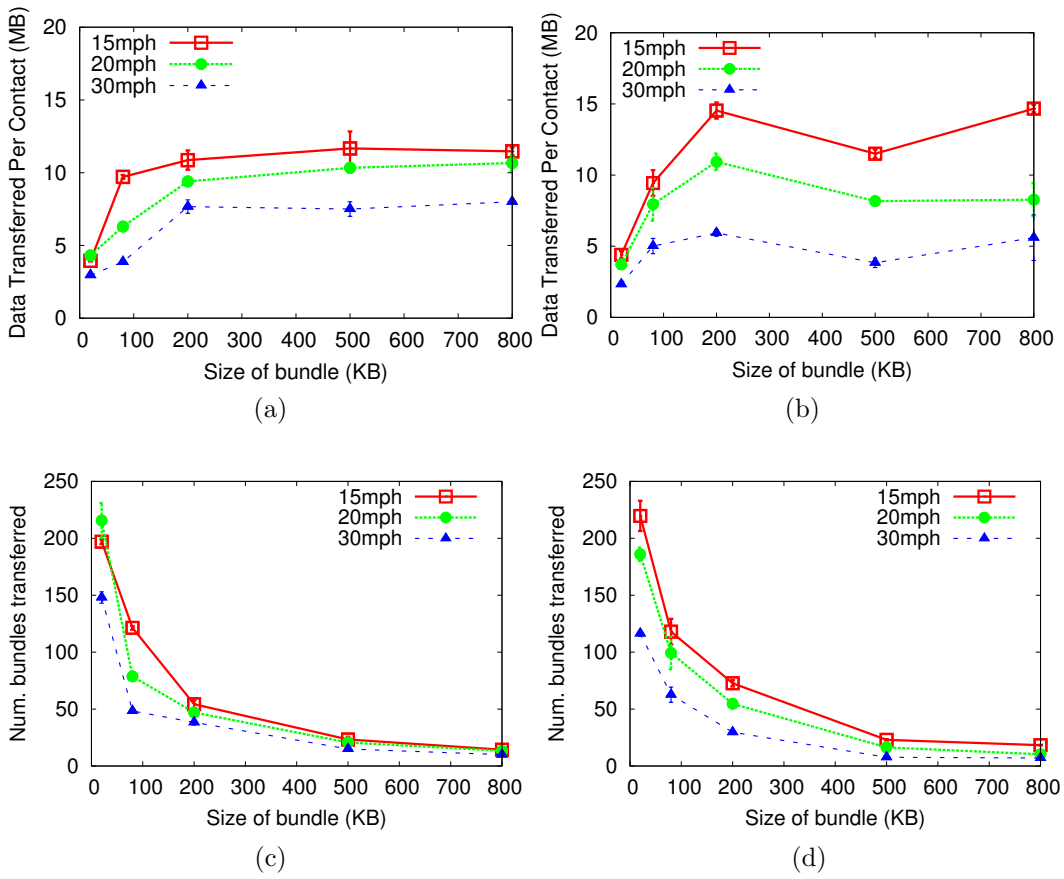
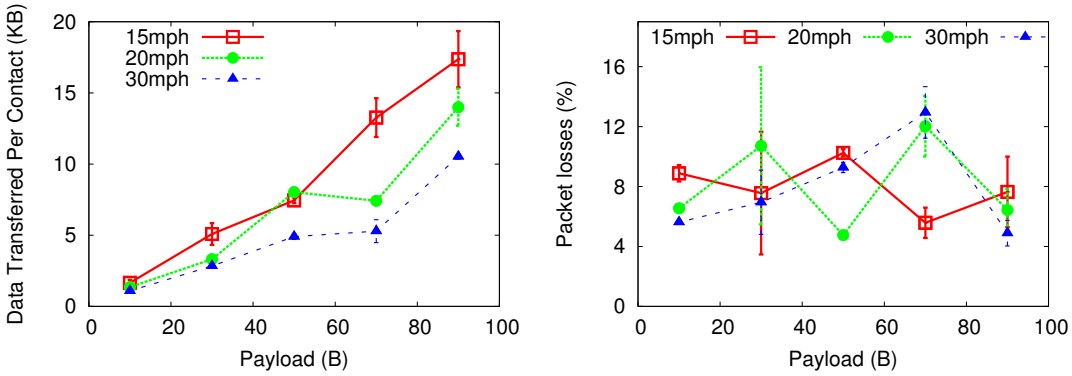


Figure 6.4: Per-contact performance for WiFi: the DTC without security (a) and with security (b); the number of bundles transferred without security (c) and with security (d).

1.3x/2x to 20/30mph should intuitively result in a throughput decrease of 0.75x/0.5x respectively. Without security and when averaged over all bundle sizes, these ratios were in practice found to be 0.86x (+14.67%) and 0.63x (+26%). With security enabled, they were 0.72x (-4%) and 0.42x (-16%) - this is expected since security incurs overhead. The number of bundles transferred can be seen in Figure 6.4(c,d). The ratios for insecure transfer were 0.91x/0.63x, whereas for secure transfers, it was 0.81x/0.49x. We conclude that the number of transferred bundles decreases sub-linearly with increase in speed.



(a) (b)

Figure 6.5: Zigbee contact performance - (a) data transferred per contact and (b) packet loss per contact.

6.3.2.2 Effect of Payload Size

The payload size affects the number of unique bundles created in each node's queue. A large number of bundles stored on a router demands more resources for local processing. As a result, DTC is low at small bundle sizes. We also observe a flattening of the goodput curve at larger payload sizes (Figure 6.4(a,b)), above around 200KB. This implies that bundles with at least 200KB are optimal for transferring between routers. Concretely, with a baseline of 20KB, the expected ratios for 80, 200, 500, 800 are 4x/10x/25x/40x. Without security, these ratios when averaged over all speeds are: 1.77x/2.49x/2.63x/2.68x. With secure transfers, we achieve 2.15x/3x/2.25x/2.73x. These results lead us to believe, based on empirical data, that the DTC increases with the square of the expected ratio up to a "critical" bundle size, after which it remains constant. A similar effect holds true for the number of bundles transferred - the number stays almost constant once a threshold bundle size is reached.

6.3.3 Optimizing the DTC: 802.15.4 Based

The effect of payload size and mobility on the DTC two nodes over 802.15.4 can be seen in Figure 6.5. A stationary node was placed 5 feet above the ground, while a mobile node was placed in the passenger seat of a vehicle. The vehicle made multiple, regular runs at speeds of 15, 20 and 30mph, transferring at each run multiple packets. These packets had a payload size between 10 and 90B. The software used was IPv6/UDP using Blip2 on TinyOS. The maximum MTU for 6lowpan/IPv6 is 100B - however, fragmentation was found to be unstable and hence unusable. Therefore, we limited our payload size to roughly 100B in our experiments. Each transferred packet used application layer acknowledgments. Each data point represents two separate runs using identical parameters.

6.3.3.1 Effect of Vehicle Speed

The effect of vehicle speed upon DTC can be seen in Figure 6.5a. As expected, DTC decreases with increasing speed. DTC shows degradation consistent with speed: the ratios for 20/30mph considering 15mph as the baseline are 0.76x/0.55x, which are +1.3%/+10%. Packet losses (Figure 6.5b) are independent of speed: the respective ratios are 0.99x and 1.01x when averaged across all payload sizes. We conclude that speed has a marked, linear effect on DTC, but packet loss percentages are independent of speed and constant.

6.3.3.2 Effect of Payload Size

The effect of payload size upon DTC is surprising: given the same contact time/speed, it increases with increasing payload size. This means that there is a constant overhead involved in the processing, sending and ACKing of packets - increasing the payload size does little to affect this overhead, but results in a large DTC.

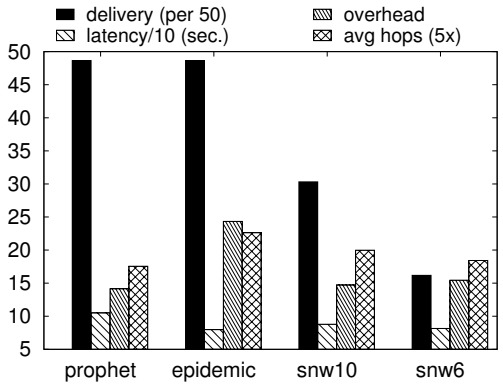


Figure 6.6: Evaluation of latency, delivery rate, overhead and average hop count for four different DTN routing protocols in simulation.

Given a baseline of 10B, the ratios for 30/50/70/90B (3x/5x/7x/9x) are 2.74x, 4.97x, 6.33x, 10.22x. The respective change against their expected ratios (assuming a linear relationship) are -8.65%, -0.61%, -9.51%, 13.54%. Thus, choosing the biggest possible payload size results in maximal DTC (within the limits of fragmentation). For packet loss (%), the ratios were 1.2x/1.15x/1.45x/0.9x - showing that empirically, the largest payload size will suffer relatively fewer losses.

6.3.4 Simulation Based Evaluation

The DTN simulator chosen for the task of choosing a routing protocol was the Opportunistic Network Environment simulator (TheONE). The paths of vehicles were digitized using the Google Earth GIS software. The entire setup in simulation consists of 26 nodes - 5 data sources each at Sources 1 and 2, three routes with 5 vehicles on each route and a EOC node. The vehicles move at a speed uniformly chosen between $\mathcal{U}$(19, 21) mph by default. The transmission range is 13m, in order to allow for a multi hop network. Each data source sends a packet of size $\mathcal{U}$(95, 105)KB every $\mathcal{U}$(20, 30) seconds to the EOC. The total simulation time for each scenario is 1h unless specified otherwise.

The combined performance of four DTN routing protocols can be seen in Figure 6.6. Epidemic routing aims to deliver messages by delivering a copy of the data to every neighbor that it encounters. It has one of the highest delivery rates and the lowest latency. A caveat is the overhead which is the number of extra messages created while routing per delivered packet, and the average hop count. In PRoPHET routing, each node maintains the probability of each of its neighbors being able to deliver a packet to a given destination. The delivery rate is same as that of Epidemic, but the latency is 26 seconds higher. SNW refers to the Spray and Wait protocol which aims to bound the number of copies of a packet in the network by a given parameter. For 2 values of 10 and 6, the SNW protocol has a very low overhead and hop count, but has a low delivery rate and high latency. We chose Epidemic routing for its high delivery rate and low latency.

6.3.5 Throughput Optimization and Waypoint Placement

6.3.5.1 Setup

In order to evaluate the waypoint placement algorithm, we designed a deployment (Figure 6.7) involving three cars and three flows with three data producing/consuming nodes. Flow1 in the following text refers to data sent from Source1 to the Base, Flow2 is from Source2 to the Base, and Flow 3 is from Source1 to Source 2. Flows 1 and 2 were firm flows, whereas Flow3 was a potential flow from S2 to the Fog with a 33% availability. For the sake of flow diversity, Flow3 was made firm by choosing S2 as a destination. Two possible waypoint locations were WP1 and WP2. There are four configurations possible with two locations: none (config0), both (config3), WP1 only (config1) and WP2 only (config2). The goodput for each flow was experimentally measured for all possible waypoint configurations.

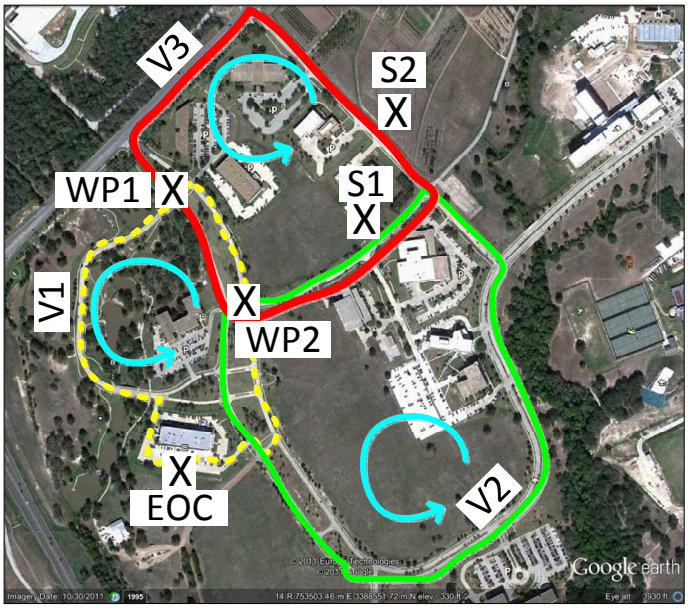


Figure 6.7: Map of deployment: S1,S2 are sources, V1,V2,V3 are vehicles, WP1,WP2 are Data Waypoints, X are locations.

6.3.5.2 Expt. 1: Epidemic Routing Without DTC Optimization

For this experiment all nodes performed epidemic routing. Results for the goodput and delay are presented in Figure 6.8(a,b). Payload size was chosen to be 100KB, each flow generated data at every 20 seconds, vehicles moved at around 20mph. It has to be noted that the sources themselves act as data waypoints due to the nature of Epidemic routing - hence, the goodput improvement between the configurations is not that high as compared to the following experiment. config3 proved to be optimal by providing the highest aggregate goodput across all the flows. If we consider only flows 1 and 2, since Flow3 does not need any additional waypoints (since the same vehicle passes through sources 1 and 2), configs 1 and 3 are almost equal.

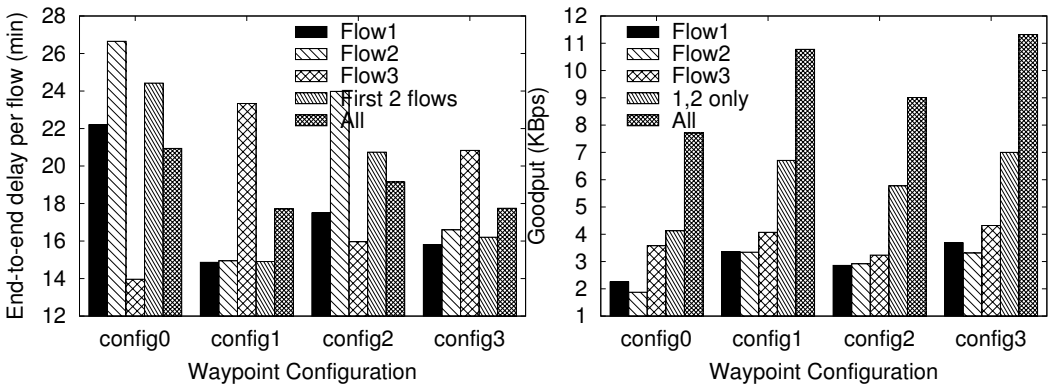


Figure 6.8: Expt. 1: DWP performance in terms of (a) latency and (b) goodput.

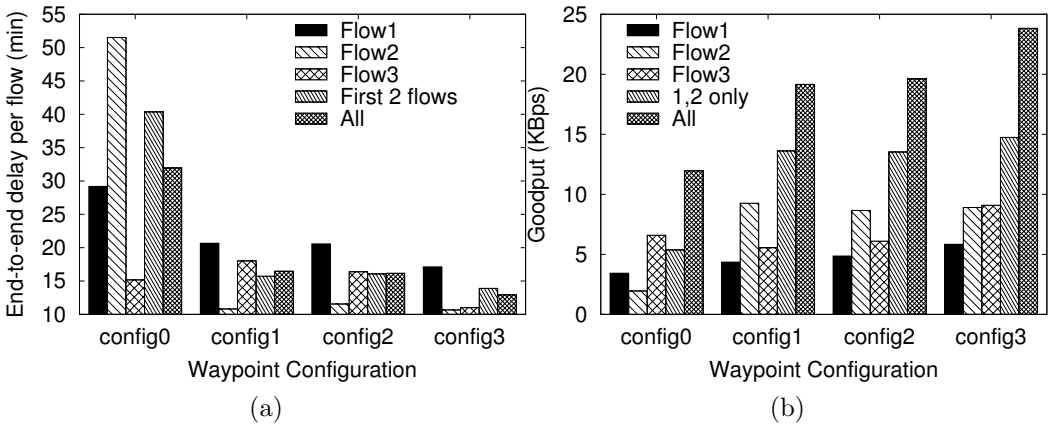


Figure 6.9: Expt. 2: DWP performance in terms of (a) latency and (b) goodput.

6.3.5.3 Expt. 2: Source Routing With DTC Optimization

For this experiment, values from the WiFi and 802.15.4 contact experiments were used to determine the payload sizes of flows. In addition to source routing replacing Epidemic routing of the previous experiment, security at the DTN layer was enabled. Flow2 was converted to a 802.15.4 flow with the vehicle picking up data from Source2 using 802.15.4 instead of Wifi. Flow1 was still Wifi based - this meant all deliveries to

Source2 were made over Wifi. The payload sizes for Wifi and 802.15.4 were chosen to be 300KB and 90B respectively. However, once a vehicle picked up 90B packets, they were marshaled into 300KB DTN bundles. Data was generated every 30 seconds.

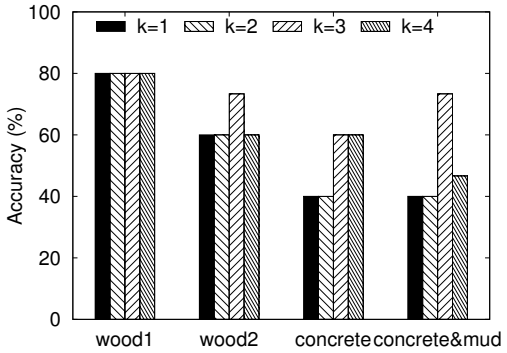
As a result (Figure 6.9) we see a marked increase in the aggregate goodput as compared to the previous experiment. Because of source routing, unnecessary copies of bundles were not created, leading to efficient and non-redundant per-contact data transfer. However, the maximum delay in config0 is high because there is no data replication (and only opportunistic contact between vehicles), but when waypoints are present, the delay is comparable in spite of the increased payload size and overhead due to security. We conclude that using source routing and choosing the payload size optimally results in a 2x increase in goodput.

6.3.6 Apps: Vibration Monitoring

We evaluated the Vibration Sensing App in Disaster City on three different rubble piles: one consisting of wooden rubble (Figure 6.10a), one of concrete, and another with a combination of concrete and mud. In the latter one, the soft mud dampens the vibrations caused inside the pile and hence makes detection with a seismic sensor difficult. Samples for different types of events were gathered at each of these piles: a stone drop, a footstep and a hammer strike. Half of the samples were used to train the KNN classifier, and the other half to evaluate performance. All samples were taken at slightly different strike intensities and distances from the sensor.

Results are shown in Figure 6.10b. "wood1" represents samples taken at the wooden pile with the default sensitivity threshold of 25 and "wood2", at a threshold of 50. A higher threshold implies lower sensitivity. This higher threshold was not possible on the two other piles since the sensor could not register soft knocks and events. We conclude that a $k = 3$ provides for optimal performance from the KNN

(a)



(b)

Figure 6.10: (a) Wooden rubble pile in Disaster City and (b) KNN classifier accuracy as a function of k

classifier with an average detection of accuracy of 73.33% independent of type of rubble, strike intensity and the distance from the seismic sensor.

6.3.7 Apps: Separation Detection

The effect of separation upon the state of a team member is shown in Figure 6.11. An experiment was conducted inside an urban building where iPod touches running the separation detection app were given to each member. "One" is the team leader and hence injects a constant state into the network. Initially, all the team members were present in a single room until time 30. Then, One and Two separated from Three by going into another room. As a result, the state of Three drops to zero since it is no longer connected to One, and the states of Two as well as One increase and converge (time $40 - 60$).

Then, One and Two move around in the large room with lot of metallic wall sized objects, causing disconnection. This disconnection is temporary and does not signal a separation. Later, Two returns to the same room as Three at time 95. As a result, the state of Three increases for time $100 - 110$ due to the residual state brought by Two, but both of them quickly decrease to zero at 110 since they are no longer

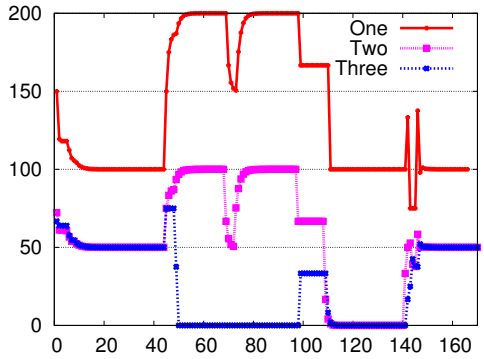


Figure 6.11: Graph of state versus time for a team of three responders.

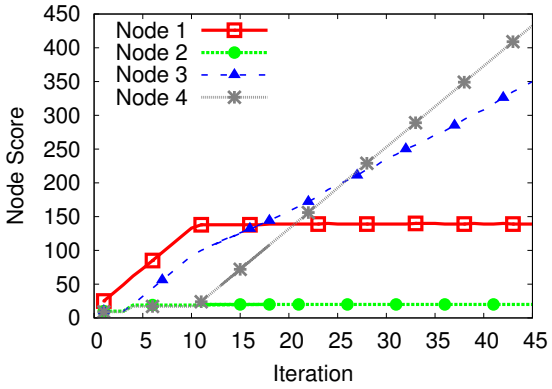


Figure 6.12: Sink election evaluation.

connected to One. Finally, One reunites with Two and Three at 140 causing all of their states to converge once again to their initial values. The average detection delay, looking at each of the three separation events and the corresponding state at that time, is $\frac{(45-30)+(98-95)+(143-140)}{3} = 7s$. The detection delay for separation as opposed to rejoining is a little longer because of the guard interval before a node declares a neighbor as disconnected.

6.3.8 Apps: Sink Election

The results of running a sink election algorithm using RPL/Blip2/TinyOS on EPIC motes is shown in Figure 6.12. Four BTag Sensors were deployed outdoors, while a fifth node in a vehicle acted as a beacon with an interval of 2 seconds. The battery voltage was read every 2 seconds, the election was conducted every 10 seconds, and data was generated every 60 seconds. Because the experiment was conducted outdoors, we had to set the transmission power to -15dBm (9.9mA draw) to simulate a multi hop network. The vehicle drove in loops at around 10mph first around nodes 1 and 2, then around nodes 3 and 4 starting at iteration 10. We see that the node with the highest score elected as the sink (Node 1 first, and then Node 4) - this verifies the correct execution of our algorithm. Interestingly, Node 3 was able to receive a few stray beacons due to radio irregularities. However, this did not disturb the election or cause a switch in the elected sink in the long run (up to iteration 45).

7. QOS AWARE ROUTING FOR FOG COMPUTING

As defined previously in this dissertation, DRNs are disruption tolerant networks designed to deliver mission critical data during disaster recovery, while operating with limited energy resources. While QoS is desired, it is difficult to offer guarantees because of the unpredictable nature of mobility in such DRNs. The variance of the packet delivery delay (PDV, more commonly called jitter), an important QoS metric which in DRNs is measured in tens of minutes instead of milliseconds, has not been sufficiently addressed in recent research. Smartphones used by first responders generate large data workloads, causing the PDV to further degrade. Reducing packet replication at these workloads will lower energy consumption, but reduces the packet delivery ratio (PDR). The complex interplay between these QoS metrics remains unclear, making their control difficult. In this section we present Raven [119], a routing protocol for DRNs that offers control over QoS metrics, especially the PDV. Stochastic graph theory which deals with probabilistic edge weights having a mean and variance is used to model PDMM. A stochastic version of the K-Shortest Paths algorithm routes data over multiple paths simultaneously. The dynamics between performance and energy consumption is analyzed mathematically, and its control is demonstrated.

7.1 Introduction

Natural disasters cause loss of power and communication infrastructure in the affected area, and make the recovery process challenging. Search and rescue is one of the Emergency Support Functions as described by the Federal Emergency Management Agency in the U.S.A; Urban Search and Rescue (USAR) is a sub-function that deals with collapsed structures in urban areas. Since USAR first responders have always communicated using traditional means such as paper and paint on walls, recent research [1] has looked at providing them with new sensing modalities like low power wireless sensors and smart phones. DRNs use concepts from delay tolerant networking (DTN) research to build a networking infrastructure that integrates these modalities and allows responders to share data. These DTNs are expected to handle data created by sensors that can be measured in kilobytes, to multiple gigabyte videos generated by smartphones. Such heterogeneous data can adversely affect the performance of DTNs that are not designed for carrying such workloads.

It is difficult for DTNs to provide hard QoS guarantees primarily because mobility in the area, which is leveraged by the DTN to mule data, is inherently unpredictable and random. This unpredictability makes it very difficult to accurately estimate the node inter-contact time, which is the primary component of the end-to-end packet delivery delay [2], and remains an open problem. Limited buffers and bandwidth contribute to the complexity of estimating the PDR and energy consumption for a given workload. Recent DTN research has looked at algorithms which improve, but not guarantee, traditional QoS metrics like the average packet delivery delay (PDD) and PDR. However, the PDV metric un-addressed. Why is the PDV important? In traditional networks, the PDD/PDV is typically measured in milliseconds - but in DTNs it can range from tens of minutes (trace based experiments in [3]) to even

hours. This means that some packets may have a delivery delay much higher than the average delay. Since sensed data from the field is used to make decisions at the Emergency Operations Center (EOC), a large PDV becomes problematic since some critical data may arrive very late.

Designing a DRN now poses several challenges: *Is it possible to control the PDV in a DTN? If so, what are the implications on the PDD, PDR & energy consumption?* Recent research [48][49][47] has shown that packet replication, used to combat uncertain mobility through redundancy, reduces PDD but results in higher PDR & energy consumption. Forwarding based techniques have lesser overhead but need more a priori information (which is difficult to obtain in an opportunistic DTN) to outperform replication based protocols. Analysis of the PDD/PDR/energy in DTNs have not included the PDV to the best of our knowledge. In research related to finance and traffic engineering, decision making in the presence of uncertainty is called risk-aversion. A risk-averse user will prefer a strategy whose reward has lower variance (i.e., more predictable) but a higher mean (i.e., lower reward). It is this concept of risk-aversion that we wish to make available to a first responder who is using an opportunistic DTN during disaster recovery (reward here is the PDD and the PDV represents uncertainty). Unfortunately, there is neither a DTN framework that jointly analyzes PDD, PDV, PDR & energy, nor an algorithm that is able to control the quantities simultaneously.

The Raven routing framework provides its users with the ability to control QoS: PDD & PDV via risk-aversion, PDR & energy consumption via replication. Raven (Risk AVersE routing in dtNs) models mobility in the disaster area using the Post Disaster Mobility (PDM) model [4]. A "stochastic multigraph", where multiple edges with probabilistic weights is possible between vertices, represents a mathematical abstraction of the PDM scenario. Important geographical locations in a disaster

called Centers are mapped to vertices, and DTN data mules (called Mobile Agents), to edges. The risk associated with each path between source and destination is calculated using a mean-risk model. Source routing is performed by selecting the least risk paths between the two vertices, using a K-Safest Paths algorithm. In order to route data between USAR responders around a Center, a forwarding decision is made wherein a packet is passed to the responder who represents the least risk in reaching the destination.

7.2 Motivation

The mobility model used in this section is the Post Disaster Mobility Model (PDMM) [4], which has been defined in Section 2. First, a simple example scenario that uses PDMM is constructed, and the corresponding stochastic multigraph is calculated. The concept of least risk path is then introduced as a generalization of the shortest path when both the mean and variance of the delay are considered.

7.2.1 Example PDMM Scenario

An example scenario is depicted in Figure 7.1. Following a disaster, the EOC has been setup, a collapsed building (RUBBLE) has been identified for search and rescue operations and a medical TRIAGE area has been setup resulting in three Centers. For simplicity only three categories of MAs are shown: ambulances, supply vehicles and USARs. USARs move around the Triage and Rubble in an area of fixed radius shown by the dotted line.

7.2.2 Risk as an Alternate Path Optimality Metric

Using the above example scenario, we now illustrate how the shortest path according to the PDD metric changes when its variance (PDV) is taken into account. This example motivates the need for an alternate path optimality metric that in-

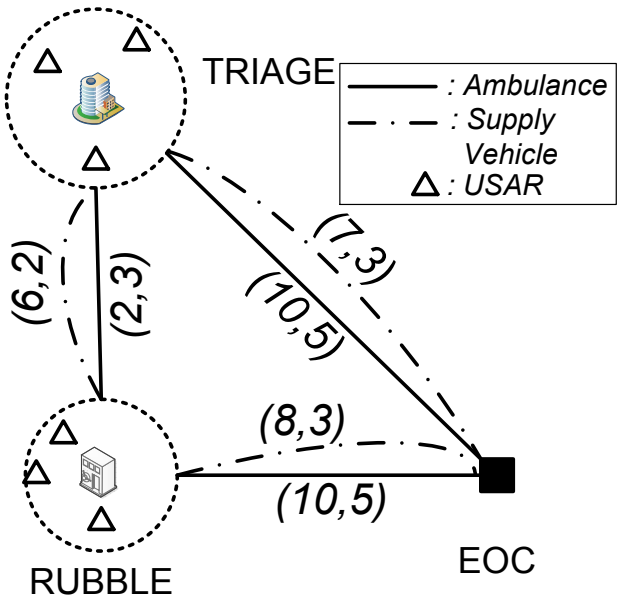


Figure 7.1: A simple scenario with 3 Centers and 3 Mobile Agents: ambulances, supply vehicles and USARs. Numbers next to a path indicate the (mean,variance) of the travel delay in minutes along that path, for the category of Mobile Agent represented by the line type (solid vs dashed).

corporates the second moment, the PDV metric in this case, in addition to the first moment or the PDD. Suppose that data is routed from one Center to another, following a "route by area" paradigm. The travel delay is a major contributor to the delivery delay. Minor contributors like queuing delay are neglected. The travel time between Centers is different for each type of Mobile Agent (MA) - and has a mean and a variance because the speed has a min and max, and this distribution is unique because of the different movement models. The scenario in Figure 7.1 is represented using a stochastic multigraph in Figure 7.2. A stochastic multigraph in this context is a graph whose vertices represent Centers and an edge represents a MA category that visits the two incident Centers. Assuming that the travel time distributions are Gaussian, the delay distribution of a path is the sum of the distributions of its constituent edges. The PDD and PDV for multiple paths from Rubble to EOC in

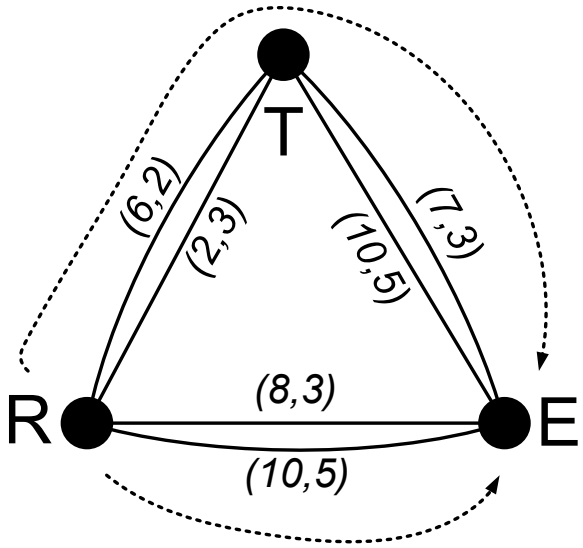


Figure 7.2: Stochastic multigraph for the above example scenario in Figure 7.1. Vertices R, T and E correspond to the Rubble, Triage and EOC centers respectively. The edges represent Mobile Agents and have the same weights.

the above scenario (Figure 7.2) is shown in Table 7.1. When the data has a deadline of 8 minutes, path 1 is optimal since it has the highest probability that the delay will be less than 8 minutes. However, when the deadline changes to 6 minutes, path 2 is optimal. What if there is no specified deadline? How can one determine the best (w.r.t delay) path in such a scenario? One solution is to use a combination of the mean and variance, called risk, as a path optimality metric instead of just the mean. The best path will then be the "least risk path".

7.2.3 Effect of Large Data Workload

Here we make a case for the necessity of QoS control at large data workloads. High quality data like 1080p video when recorded during disaster recovery can be valuable but can also run into several gigabytes. First responders who use DTNs during disaster recovery can be expected to use smart phones which have such recording and storage capabilities. However, in state of art research that contain experiments

	Path	Leg1	Leg2	PDD	PDV	P(X<6)%	P(X<8)%
1	R→E	SV	-	8	3	25.2493	50 ⋆
2	R→T→E	Amb	SV	9	6	30.8538 ⋆	43.3816
3	R→E	Amb	-	10	5	21.1855	34.4578
4	R→T→E	Amb	Amb	12	6	15.8655	25.2493
5	R→T→E	SV	SV	13	5	8.0757	15.8655
6	R→T→E	SV	Amb	16	7	7.6564	12.6549

Table 7.1: Enumerating the mean and variance of the delivery delay along multiple paths for routing data from R to E. All values are in minutes. SV stands for supply vehicle, Amb stands for ambulance. Path 2 is optimal for a deadline of 6 minutes, while path 1 is optimal for a deadline of 8 minutes. What is optimal for a deadline of 6 minutes is not necessarily optimal for a deadline of 8 minutes.

performed on real DTN testbeds, the load imposed on these systems can be measured in a few kilobytes (Table 3 in [56] and 1.5KB per packet). We examine the performance of several DTN routing protocols as the load on the network is increased to a few GBs. The results are shown in Figure 7.3. The mobility model used was the Post Disaster Mobility Model (PDMM) [4] and the setup simulated a disaster in a large city with a street map, consisting of an EOC, a Triage and seven collapsed buildings. The protocols chosen were Prophet [51], MaxProp [52], SprayWait [49] (L=3, binary mode) and RAPID [3]. The load on the network (X axes in Figure 7.3) was spread across three flows, and all packets were generated at $t = 0$. As the total load increased from 30MB to 3.6GB, both the PDD (Figure 7.3a) and PDV (Figure 7.3a) increased by about 150% on average. The reason that the PDD (which is calculated for delivered packets only) remains constant after 1.8GB is that the DTN becomes saturated and delivers the same number of packets irrespective of the load; it is confirmed by the fact that the PDR (not shown) decreases from 40% at 1.8GB to 20% at 3.6GB. To conclude, the quality of service rapidly deteriorates as the DTN capacity saturates - and there is a need for a routing framework which allows a user

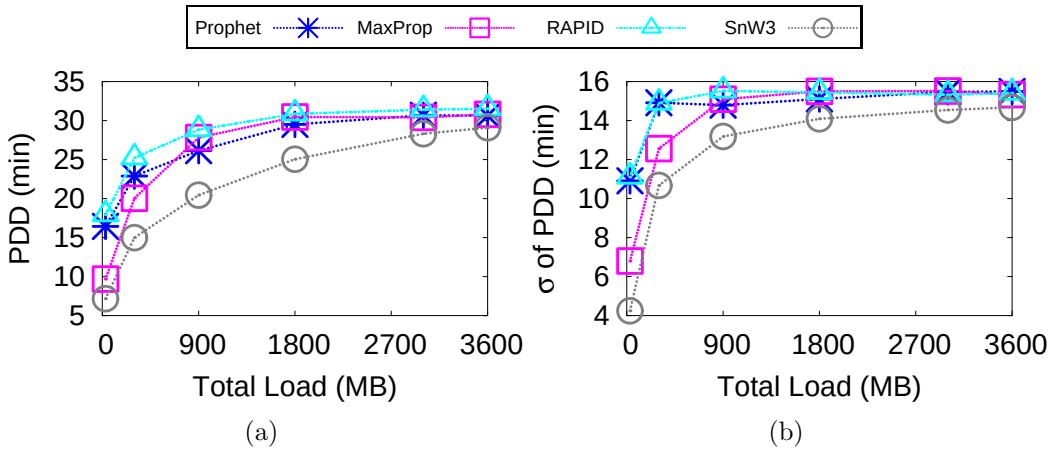


Figure 7.3: Performance of several DTN routing protocols as data workload increases: (a) PDD and (b) standard deviation of PDD. Radio bandwidth was 4MBps (approx. TCP throughput on 802.11 54Mbps) and the buffer size was 1TB.

to prioritize one QoS metric, including the PDV, over another.

7.3 The Raven Routing Framework

In this section we first present the problem formulation followed by a detailed explanation of how Raven works, culminating in a distributed protocol. This section poses two research questions: 1) Is it possible to incorporate risk-aversion in a DTN by controlling the PDV? If so, how can it be done using a routing protocol? Is replication necessary? and 2) What are the effects of enabling risk-aversion and replication on quantities like PDR and energy consumption? This section addresses question 1, whereas the second question is answered in the following Section. In the context of the PDM mobility model, the routing problem can be seen as the union of two subproblems: routing between Centers and routing within Centers. Following a description of these subproblems, the process of building the stochastic multigraph by estimating the travel time distributions is discussed. A formal definition of risk is then derived, followed by the K Safest Paths algorithm.

7.3.1 Preliminaries

A set of centers C and their locations L_c is known. The set of Mobile Agent categories is M, and each category m has n_m agents. For each category m, the min and max speeds of the agent are known. The radius R around each Center in which USAR agents operate is specified. Collectively, these quantities are called the PDM scenario $\mathcal{P}$. The risk-aversion factor ρ and the number of paths K is provided by the user. The scenario is not completely known because of the randomized movement. Previous research has shown that replication, where data is sent on multiple paths, improves performance in scenarios where the movement is random, contacts are sparse, and the node inter-contact time is large [49][47]. Thus, we design Raven to include replication.

7.3.2 Problem Formulation

The problem formulation involves the PDM model, stochastic multigraphs resulting from the PDM model, and risk-aversion. In a scenario represented by the PDM model, there are two types of data flows possible: between Centers and within Centers. Statically deployed chemical sensors deployed around a Center, for example, need to report data to the EOC periodically. Such a data flow is an example of the Center to Center model. Data from sensors is first collected on the static node at the Center and is then sent to the static node at the EOC using Mobile Agents other than USARs. The EOC on the other hand, may choose to push information to the USAR agents working at a Center. This is an example of a hybrid flow, because data from the EOC is first sent to the static node at the Center, where it will be disseminated among USAR agents using the "within Centers" flow.

7.3.2.1 Problem Formulation (General)

Given a DTN scenario represented by the PDM mobility model with Centers C and Mobile Agent categories M, calculate the $K \geq 1$ least risk paths between source $S \in C \cup M_{USAR}$ and destination $D \in C \cup M_{USAR}$. The risk associated with a path is calculated based on a user defined risk-aversion coefficient $\rho \geq 0$, where $\rho = 0$ means the PDD should be minimized over the PDV and $\rho \to \infty$ means the PDV should be minimized over the PDD.

This formulation optimizes the risk but not the PDR & energy consumption because varying K (replication) only has a coarse grained effect on PDR/energy, whereas varying ρ (risk-aversion) has fine grained effects on PDD & PDV. Now the general problem formulation can be split into 2 subproblems because of hybrid flows:

7.3.2.2 Problem Formulation (Routing Between Centers)

For a source $c_s \in C$ and destination $c_d \in C$, calculate K least risk paths where each path P is a set of alternating Centers and Mobile Agents.

7.3.2.3 Problem Formulation (Routing Within Centers)

Given a Center c, a set of USAR agents MA_U around c, perform a binary forwarding decision when a USAR agent $u_1 \in MA_U$ carrying a packet (with source in $c \cup MA_U$ and destination in $c \cup MA_U$) encounters another agent u_2, such that the risk in delivering the packet is minimized.

Routing between Centers has been explained in the previous section. Here we provide a short overview of routing within Centers. An example scenario is shown in Figures 7.4a and 7.4b. USAR agents U1 and U2 move using Random Waypoint around a Center. In Figure 7.4a, agent U1 is carrying a message for the Center, and is in contact with U2. We assume that agents know their current waypoint - in this

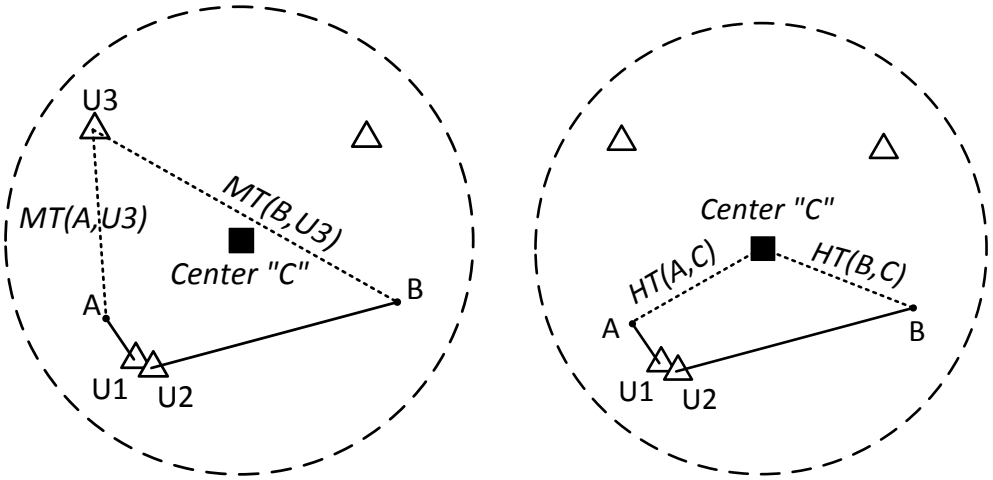


Figure 7.4: Routing within Centers: when two USAR agents U1 and U2 meet, with current waypoints A and B respectively, U1 decides whether or not to forward a packet to U2. (a) If the message is destined for the Center, the decision is made based on the risk present in the travel time to the Center from A and B. (b) If destined for another USAR agent U3, the decision is made based on the risk in the expected travel time between A or B to U3.

case it is the point A for U1 and B for U2. U1 computes the mean and variance of the travel time from A to C, and from B to C. The result is two sets of means and variances. The decision to forward the packet to U2 is made if the A-C path has a lesser risk than the B-C path. Similarly, in Figure 7.4b, U1 has a message destined for another mobile agent U3. The decision to forward is made based on the travel time from A-U3 and B-U3. These two scenarios show the need for *hitting times* and *meeting times* respectively (explained below).

7.3.3 The Stochastic Multigraph and its Construction

A stochastic multigraph $\mathcal{S}$ maps the Centers C of the PDM model to vertices and Mobile Agent categories M to edges, while the edge weights are not scalars but random variables. In a stochastic multigraph $\mathcal{S} = G(V, E)$ each edge $e \in E$ has an

associated mean μ_e and variance σ_e^2 corresponding to the travel time distribution. The weight of an edge W_e is defined as a pair of scalars $W_e = (\mu_e, \sigma_e^2)$. We only consider stochastic time-independent graphs where W_e do not change over time. A *stochastic path* p in $\mathcal{S}$ is a graph path whose edges and vertices are present in $\mathcal{S}$. The weight of this path W_p is the sum of the distributions of the edges in the path. If the distributions are Gaussian:

$$W_p = \sum_{e \in p} W_e = (\mu_p, \sigma_p^2) = (\sum_{e \in P} \mu_e, \sum_{e \in p} \sigma_e^2) \tag{7.1}$$

Prior to constructing $\mathcal{S}$, the travel time distributions for each Mobile Agent category between each pair of Centers need to be calculated. In recent mobility model research, the time taken for two nodes to meet each other when starting from different positions is called the meeting time (if both nodes are mobile) or the hitting time (if one node is mobile) [50]. In the PDM mobility model, we define the meeting time between any two Mobile Agents u_1 and u_2 as the time taken for them to come in contact with each other when starting from different positions $L_{u_1}(0)$ and $L_{u_2}(0)$ and moving according to their respective movement models. If their radio range is R,

$$MT(u_1, u_2) = \min_t \{t : ||L_{u_1}(t) - L_{u_2}(t)|| < R\} \tag{7.2}$$

The hitting time (HT) as defined in [50] is a special case of the meeting time, defined when one of the nodes is static. In the context of PDM, HT is defined between a Mobile Agent and a Center:

$$HT(ma_1, c) = \min_t \{t : ||L_c - L_{ma_1}(t)|| < R\} \tag{7.3}$$

7.3.3.1 Routing Between Centers

In the stochastic multigraph as shown in Figure 7.2, each edge represents the travel time between the two Centers, through a particular category of Mobile Agent. That is to say, the mean and variance of each edge is the time taken for an Ambulance (for example) to travel from the Rubble to the Triage. This is nothing but the hitting time (Equation 7.3), where ma_1 is an ambulance and c is the Triage. Since the ambulance starts at the Rubble, $L_{ma_1}(0) = L_{rubble}$ and $L_c = L_{triage}$. The PDM model is map based and hence closed form solutions for HT are highly dependent on the underlying map as well as the locations of the Centers. Therefore, values for HT, and hence the edge weights, need to be derived from simulation. The TheONE DTN simulator, when given a map and the list of Centers, can place Mobile Agents at any intersection and calculate the travel time to another Center. The Dijkstra algorithm is used to choose the shortest path between two points in the map. This way, the HT values can be stored in a lookup table and accessed later.

7.3.3.2 Routing Within Centers

USAR agents move using the Random Waypoint (RWP) mobility model in a fixed radius around a Center. RWP has been analyzed mathematically in recent research - including closed form formulae for HT and MT. However, closed forms for the *distribution* of HT/MT are not available. We present some results from our work on the HT/MT distributions for RWP that is in progress. Suppose that USAR agents whose radio range is R move using the RWP model in a square area with area A and side d with average velocity $\bar{v}$ (the Center c is at the center of this square as in Figure 7.4a). The average leg length in RWP has been shown to be $L = 0.5214d$ and the corresponding time taken is $\overline{t} = \frac{L}{\overline{v}}$. If the Center c is at location (x, y), then

Algorithm 3 Construction of $\mathcal{S}$

Input: PDM scenario $\mathcal{P}$, ρ, K
Output: Stochastic graph $\mathcal{S}$
1: Create $|C|$ vertices in $\mathcal{S}$
2: Draw $|M|$ edges between each pair of vertices in $\mathcal{S}$
3: **for** edge $e(a, b, m)$ where $a, b \in C$ and $m \in M$ **do**
4: **if** Movement model of m does not involve a or b **then**
5: Delete e and **continue**
6: **end if**
7: $\mu(e), \sigma^2(e) \leftarrow \text{HT}(m,\ L_b)$ s.t $L_m(0) = L_a$ ▷ By simulation
8: $e \leftarrow \mu(e), \sigma^2(e)$
9: **end for**
10: **return** $\mathcal{S}$

the hitting time for a USAR agent u_1 is:

$$E[HT(c, u_1)] = p_{1h} \cdot t_{1h} + \frac{\overline{t}}{\pi^2 R f(x, y)} \cdot (1 - p_{1h}) \tag{7.4}$$

$$f(x, y) = \frac{36}{d^6} xy(d - x)(d - y) \text{ and } p_{1h} = \frac{2RL}{A}$$

$$Var[HT(c, u_1)] = \frac{1 - p_{1h}}{p_{1h}^2} \cdot \overline{t}^2 \tag{7.5}$$

where p_{1h} is the probability that u_1 hits c in the first leg itself and t_{1h} is the time taken to do so. The analysis for meeting time (Figure 7.4b) is similar, but in this case the location of u_3 is not available to u_1 or u_2 (as opposed to the location of the Center in the previous paragraph). To overcome this shortcoming, we average $MT(u_1, u_3)$ over all possible starting locations of $L_{u_3}(0) = L^0_{u_3}$:

$$MT(u_1, \overline{u_3}) = \frac{1}{A^2} \int_{\mathcal{A}} MT(u_1, u_3)\, \mathrm{d}L^0_{u_3} \tag{7.6}$$

Because of the complexity of the above formula, values are obtained from simulation.

The algorithm for constructing $\mathcal{S}$ is shown in Algorithm 3. First, a vertex is

created for each PDM Center (Step 1). Since each edge represents a Mobile Agent category, $|M|$ edges are drawn between each pair of vertices (Step 2). However, some edges (Step 3) are redundant - for example, a Patrol Car may not visit all Centers. Therefore it is necessary to cull some edges based on whether the movement model allows for agent to visit those two Centers (Steps 4-6). If the edge is allowed, the mean and variance to be assigned is first calculated (Step 7). This is nothing but the hitting time HT for an agent m_1 of category m, when it starts at Center a and hits Center b (Equation 7.3). The mean and variance of the HT is assigned to that edge (Step 8).

7.3.4 Quantifying Risk

A stochastic path is considered "risky" if there is a high probability that a sample from $\mathcal{N}(\mu_p, \sigma_p^2)$) will deviate far from the expected value (μ_p) [120]. In order to quantify this "risk", we adopt the mean-risk probability model [121]. The risk R_e of an e in $\mathcal{S}$ is defined as $R_e = \mu_e + \rho * \sigma_e$ where the *risk-aversion coefficient* $\rho \geq 0$ is a user specified quantity. It represents how important the variance of the path weight is to the user. A ρ of zero in stochastic routing chooses the path with the least mean. Similarly, for a path p its risk R_p is defined as $R_p = \mu_p + \rho * \sigma_p$. However, it is not equal to the sum of the risks of its edges:

$$R_p = \sum_{e \in p} \mu_e + \rho * \sqrt{\sum_{e \in p} \sigma_e^2} \quad \Rightarrow \quad R_p \neq \sum_{e \in p} R_e$$

In order to choose K least risk paths, we adopt the K-Shortest Paths (KShP) problem in deterministic graphs to stochastic graphs. A "safe" path of two paths p_1 and p_2 is the one with the lower risk $min(R_{p_1}, R_{p_2})$. The objective of KSfP is to choose the K safest paths of a stochastic graph $\mathcal{S}$, given a source node and a

Algorithm 4 K-Safest Paths Algorithm (KSfP)

Input: Stochastic multigraph $\mathcal{S}$, K, source s, dest. d
Output: $\mathbb{K}$, a set of K paths in $\mathcal{S}$ between s and d
1: **for** edge e in $edges(\mathcal{S})$ **do**
2: $\quad weight(e) \leftarrow \mu_e + \rho * \sigma_e^2$ ▷ Modified Risk Formula
3: **end for**
4: $\mathcal{S}' \leftarrow \mathcal{S}$ with edge weights as above
5: $\mathbb{K} \longleftarrow$ Apply K-Shortest paths on $\mathcal{S}'$ with src/dest s/d
6: **return** $\mathbb{K}$

destination node. K is a natural number. Existing algorithms for KShP (classical version) include a modified Bellman-Ford algorithm that stores the top K shortest paths at each pass instead of storing only the shortest (JGraphT library), and Yen's algorithm [122]. Certain shortcomings of the Bellman-Ford algorithm, such as the diversity and disjointness of computed paths, are outside the current scope and left as future work. A better algorithm can serve as a drop-in replacement. The stochastic shortest path problem (KSfP with $K = 1$) is a non-convex combinatorial problem [60]. A dynamic programming approach is incorrect since sub-paths of optimal paths are not optimal. The risk of a path is not a linear combination of the risks of the edges, but is in fact non-linear as seen above ($R_p \neq \Sigma R_e$). We therefore propose the use of variance instead of the standard deviation for simplicity. While dimensional homogeneity is not present due to the use of variance which is the square of the standard deviation, the implementation of KSfP becomes straightforward and simple. The algorithm is shown in Algorithm 4. The stochastic graph is first converted into a deterministic graph (Steps 1-3). The edge weight is computed using the modified risk formula $R_p = \mu_p + \rho * \sigma_p^2$. Each edge e in the stochastic graph $\mathcal{S}$ is assigned a deterministic edge weight (Step 2). The modified graph $\mathcal{S}'$ is now completely deterministic (Step 4). Any KShP algorithm can now be applied (Step 5). The result is a set of paths $\mathbb{K}$ that have the least risk.

Algorithm 5 The Raven Routing Protocol (includes RbC)

Input: PDM scenario $\mathcal{P}$, ρ, K, source s, destination d

1: Build $\mathcal{S}$ using Alg 3 input $\mathcal{P}, \rho, K$
2: **if** s and d are Centers **then** ▷ Routing Between Centers
3: $\mathbb{K} \leftarrow$ Use Alg 4 input $\mathcal{S}, K, s, d$
4: Source route along the paths in $\mathbb{K}$
5: **else if** s is a USAR, d is a Center **then**
6: $c_s \leftarrow$ the Center around which s works
7: Use Alg. 6 with input s, c_s, ρ till packet reaches c_s
8: Apply Raven at c_s with input $\mathcal{P}, \rho, K, c_s, d$
9: **else if** s is a Center, d is a USAR **then**
10: $c_d \leftarrow$ the Center around which d works
11: $\mathbb{K} \leftarrow$ Use Alg 4 with input $\mathcal{S}, K, s, c_d$
12: Source route along $\mathbb{K}$ still packet reaches c_d
13: Use Alg. 6 at c_d with input c_d, d, ρ till packet reaches d
14: **else if** s is a USAR, d is a USAR **then**
15: $c_s, c_d \leftarrow$ the Centers around which s, d work
16: **if** $c_s = c_d$ **then**
17: Use Alg. 6 with input s, d, ρ till packet reaches d
18: **else if then**
19: Use Alg. 6 with input s, c_s, ρ till packet reaches c_s
20: Apply this algorithm at c_s with input $\mathcal{P}, \rho, K, c_s, d$
21: **end if**
22: **end if**

7.3.5 The Raven Routing Protocol

The main algorithm is shown in Algorithm 5. It is performed when a packet is generated at a source node s which may be a USAR Mobile Agent or a Center. The first step constructs the stochastic multigraph $\mathcal{S}$ (Step 1) using Algorithm 3 which is explained below. The next steps implement the hybrid data flow model as explained in Section IIIA. The first possibility is that both source and destination are centers (Step 2). This is essentially Routing Between Centers (RbC). The K Safest Paths algorithm is used (Step 3) to find the set of K paths as explained in the previous section. The packet is then source routed along these paths. If the source is a USAR agent (Step 5), the packet is first routed to the agent's Center (Steps 6-7) using

the Routing Within Centers (RwC) algorithm which is explained below, and then the RbC algorithm is used as if the packet was created at the agent's Center. The workflow is similar if the destination is a USAR agent (Steps 9-13). If both source and destination are USAR agents working at the same Center (Steps 14-16), then the RwC algorithm is applied (Step 17). If not, then a combination of the previous strategies is applied (Steps 18-20).

7.3.5.1 RwC - Routing Within Centers

The RwC algorithm is shown in Algorithm 6. Unlike RbC, it is not based on a stochastic graph and does not replicate. If the source happens to be a Center (Step 1), then the packet is forwarded to the first USAR agent in contact (Step 2) and RwC is applied recursively (Step 3). When a USAR agent u_1 meets agent u_2 (Step 5), a forwarding decision needs to be performed. The first step in RwC is to exchange each other's current waypoints (Steps 6-7). The basic idea in RwC is to compute the risk present in u_1 or u_2 meeting the destination. As explained before, the risk is a linear combination of the mean and variance of the travel time distribution. Computation of the travel time depends on whether the destination is a static Center c (Steps 8-12) or a mobile agent u_3 (Steps 13-17). If the destination is static, the hitting time is computed to yield a mean and variance for each of u_1 and u_2 (Steps 9-10) and the corresponding risks are computed (Steps 11-12). Similarly, if the destination is mobile, the meeting time formula is used (Steps 14-15) and the risks computed (Steps 16-17). It is to be noted that since neither u_1 nor u_2 know the location of u_3, the average of MT is taken (as explained before Equation 7.6). Finally, if the other agent u_2 has a lower risk (Step 19), the packet is forwarded (Step 20).

Algorithm 6 Routing Within Centers

Input: Source s, Destination d, ρ

if Source s is a Center c **then**
    Forward packet to first USAR u in contact
    Break and apply this algorithm with input u, d, ρ
end if
Source s is a USAR u_1 in contact with USAR u_2
$A \leftarrow$ Current waypoint of u_1
$B \leftarrow$ Current waypoint of u_2
if Destination d is a Center c **then**
    $\mu_1, \sigma_1^2 \leftarrow \mathrm{HT}(u_1, L_c)$ where $L_{u_1}(0) = L_A$ ▷ Equation 7.4
    $\mu_2, \sigma_2^2 \leftarrow \mathrm{HT}(u_2, L_c)$ where $L_{u_2}(0) = L_B$ ▷ Equation 7.4
    $R_1 \leftarrow \mu_1 + \rho \times \sigma_1^2$ ▷ Modified Risk Formula
    $R_2 \leftarrow \mu_2 + \rho \times \sigma_2^2$ ▷ Modified Risk Formula
else if Destination d is a USAR agent u_3 **then**
    $\mu_1, \sigma_1^2 \leftarrow \mathrm{MT}(u_1, \overline{L_{u_3}})$ where $L_{u_1}(0) = L_A$ ▷ Equation 7.6
    $\mu_2, \sigma_2^2 \leftarrow \mathrm{MT}(u_2, \overline{L_{u_3}})$ where $L_{u_2}(0) = L_B$ ▷ Equation 7.6
    $R_1 \leftarrow \mu_1 + \rho \times \sigma_1^2$ ▷ Modified Risk Formula
    $R_2 \leftarrow \mu_2 + \rho \times \sigma_2^2$ ▷ Modified Risk Formula
end if
if $R_2 < R_1$ **then**
    Forward packet to u_2
end if

7.4 Analysis

In this section mathematical analysis on the coexistence of risk-aversion and replication as well as their effects on the QoS metrics is presented and the performance evaluation is discussed. The interdependence of ρ and K is shown in Table 7.2. Borrowing terminology from [3], an intentional effect changes a metric *by design*, whereas an incidental effect does so *indirectly*.

7.4.1 General Problem Formulation

Suppose that we have n normal distributions in the set $\{P\}$ (representing the paths from a given source to a given destination in the stochastic multigraph). Each of these distributions P_i has an associated mean and variance $(E[P_i], V[P_i]) =$

	PDD	PDV	PDR	Energy
ρ	Intentional	Intentional	Incidental	Incidental
K	Incidental	Incidental	Intentional	Intentional

Table 7.2: The type of effect each Raven parameter (in rows) has on the QoS metrics of interest (in columns).

(μ_i, σ_i^2). A risk-aversion coefficient $\rho \geq 0$ assigns a scalar quantity called risk $(= \mu_i + \rho * \sigma_i)$ to each P_i according to the mean-risk probability model. When one wants to be risk-averse, this set of n paths is ordered according to the risk of each path, resulting in a set $\{Q\}$ such that:

$$\{Q\} = Q_1, Q_2, Q_3, \ldots, Q_n \text{ where } Q_i \in P$$
$$i < j \Rightarrow (E[Q_i] + \rho * \sqrt{V[Q_i]}) < (E[Q_j] + \rho * \sqrt{V[Q_j]})$$

Without replication, data will be sent only on the path Q_1. However, with replication, the first $K \geq 1$ elements of $\{Q\}$ which are $\{Q_K\} = \{Q_1, Q_2, \ldots, Q_K\}$ are chosen. The delivery delay will now have a mean of $mean(\rho, K) = PDD = E[W]$ and a variance of $var(\rho, K) = PDV = V[W]$ where $W = min\{Q_1, Q_2, \ldots, Q_K\}$. These paths can be assumed to be i.i.d because of the assumptions of 1) infnite buffer and 2) the physical travel is the major contributor to the packet delivery delay. The c.d.f of the minimum of a set of independent distributions is defined as follows:

$$\begin{aligned} P(W \leq x) &= 1 - P(Q_1 > x)P(Q_2 > x)\ldots P(Q_K > x) \\ &= 1 - \prod_{i=1}^{K} P(Q_i > x) = 1 - \prod_{i=1}^{K}(1 - P(Q_i \leq x)) \\ &= 1 - \prod_{i=1}^{K}(1 - \Phi\left(\frac{x - E[Q_i]}{\sqrt{V[Q_i]}}\right)) = 1 - \prod_{i=1}^{K}(1 - \Phi_i) \end{aligned} \tag{7.7}$$

where $\Phi(x)$ is the c.d.f of the standard normal distribution and Φ_i is, with abuse of notation, defined as scaling $\Phi(x)$ to a distribution with non-standard mean $E[Q_i]$ and variance $V[Q_i]$. Once we have the c.d.f, the mean and variance are:

$$P(W \leq x) = F_W(x) \text{ and } f_W(x) = F'_W(x)$$

$$mean(\rho, K) = E[W] = \int_{-\infty}^{\infty} x f_W(x)\, dx \tag{7.8}$$

$$var(\rho, K) = V[W] = \int_{-\infty}^{\infty} (x - E[W])^2 f_W(x)\, dx \tag{7.9}$$

7.4.2 Results

Result R1: ρ has an incidental effect on PDR & energy while K has an intentional effect. For a given K, ρ and a packet/source/destination, a set of paths $\{Q_K\}$ is constructed. Let J be the union of all the Centers present on these K paths. The number of relayed messages, and hence the energy, is proportional to the cardinality $|J|$. This is because if a Center is present on multiple paths, the packet will be relayed to it only once since infinite buffers are assumed. Similarly the PDR is proportional to the number of paths K. For two different ρ, there is no guarantee that $|J|$ or K will change since ρ only changes the order of paths and not the number of paths. Thus, ρ only has an incidental effect on the PDR & energy. K has an intentional effect on the energy/PDR since the number of paths as well as $|J|$ (converges to the total number of Centers) are guaranteed to increase.

R2: $mean(\rho, K) \to -\infty$ as $K \to \infty$. This is because as K increases, the minimum of K normally distributed random variables will decrease. While the result is intuitive, a proof of this statement for the general case is difficult owing to the complexity of solving Equation 7.8 for a non-standard normal distribution. However, this result has been proved for K standard normal distributions [123] and is known

as the extreme first order statistic. Suprisingly, such a result for variance does not seem to hold even for two random variables [124]. This result has been confirmed by the authors using Monte Carlo simulations on stochastic multigraphs extracted during simulation. As a corollary, $mean(\rho, K_1) < mean(\rho, K_2)$ if $K_1 > K_2$.

R3: As $K \to \infty$, the effect of ρ will be less and less pronounced. In other words, it is difficult to be risk-averse at high K. This is because 1) the order of Φ_i in Equation 7.7 does not matter since it is a product, and 2) ρ only changes the order but not the number of selected random variables.

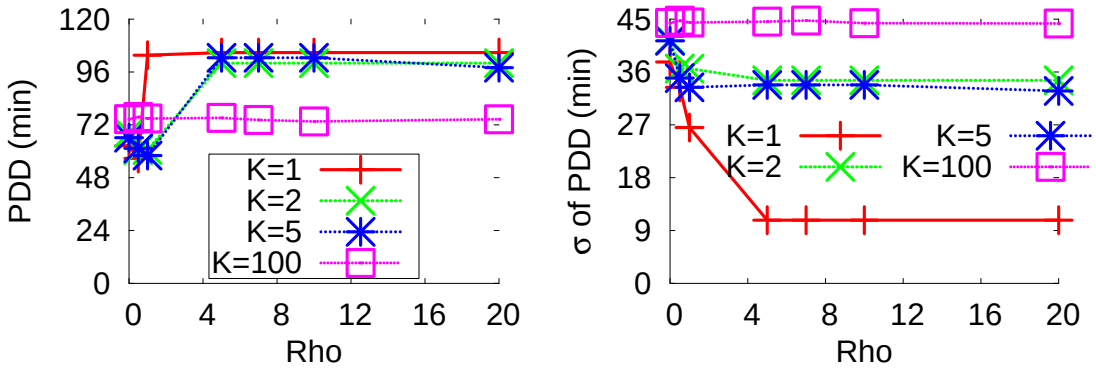
R4: $m_1 \leq mean(\rho, K = 1) \leq m_2$ as $\rho \to \infty$, where m_1 is the mean of the P_i with the smallest mean and m_2 is the mean of the P_i with the smallest variance. The proof is trivial since an increasing ρ chooses a lower variance by definition.

R5: If $\rho_1 > \rho_2$, $mean(\rho_1, K = 1) > mean(\rho_2, K = 1)$. In order to understand this slightly counter-intuitive result, imagine the $\mu - \sigma$ Pareto frontier of a graph where each distribution P_i corresponds to a point $(x, y) = (\mu_i, \sigma_i)$. As ρ goes from 0 to ∞, the distribution minimizing $\mu + \rho * \sigma$ will be the one with smallest mean, then the next one on the $\mu - \sigma$ Pareto frontier, and so on until the bottom-most distribution is selected. Using this graph, it is easy to see that $mean(\rho_1, K = 1) > mean(\rho_2, K = 1)$.

7.5 Performance Evaluation

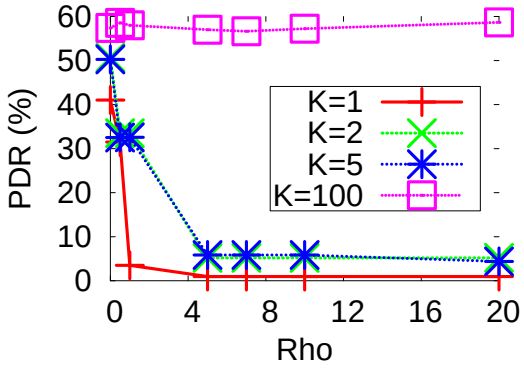
Evaluations were performed in simulation using TheONE simulator, with the PDM mobility model, using the Helsinki street map. Trace based evaluation is difficult since movement traces of first responders are not readily available. A large disaster in the city has was simulated. An EOC and TRIAGE are setup, with 7 collapsed buildings where urban search and rescue is to be performed, resulting in 9 total Centers. 3 ambulances and 3 supply vehicles move in the city using their respective mobility models. Both categories have a speed of (0,40) m/s. 10 volunteers

move using the Volunteer Mobility Model with their home centers as the EOC and a speed of (0,4) m/s. A patrol car moves using its mobility model among collapsed buildings 1, 2 and 4 with a speed of (0,40) m/s. A speed multiplier simulation parameter changes the min and max speeds of each Mobile Agent proportionately. Data is sent on three flows simultaneously and all packets are created at $t = 0$. Unless specified, the default load is $1GB$ per flow and the bandwidth of each node's radio is $4MBps$. The simulated time is 10000 seconds and each data point is averaged over 25 random runs. It took about 75 processor-hours of wall clock time to gather data for this section. The four metrics of interest are the three QoS metrics (PDD, PDV, PDR) and the energy consumed by the system. The number of relayed messages is an indication of the latter since the on board radio draws a large current (as compared to the storage device, for example). For reasons of dimensional homogeneity, we compare the standard deviation of PDD (called PDS for brevity) instead of the PDV. First, the effect of varying K and ρ upon the PDD and PDV is measured. With an increasing ρ, the PDV should decrease because risk-aversion favors a lower variance. Based on the results, we tune Raven by choosing a particular K and ρ for comparison with other state of the art protocols. The following sections demonstrate the effect of increasing load, increasing node speed and increasing radio bitrate upon the four metrics of interest. The protocols chosen for comparison are RAPID [3], Prophet [51], MaxProp [52] and SprayWait [50] (with $L = 3$). We chose $L = 3$ to demonstrate that simply fixing replication at a low number will not improve performance at high loads. RAPID is a utility based routing protocol tuned to minimize the PDD based on the marginal utility of replicating a packet, whereas Prophet and MaxProp attempt to characterize the mobility and replicate packets only to better hosts, i.e., those with a higher probability of meeting the destination.

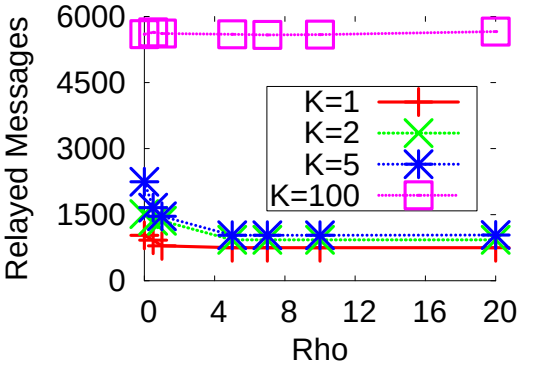


(a) Effect of ρ upon the PDD

(b) Effect of ρ upon the PDS



(c) Effect of ρ upon the PDR



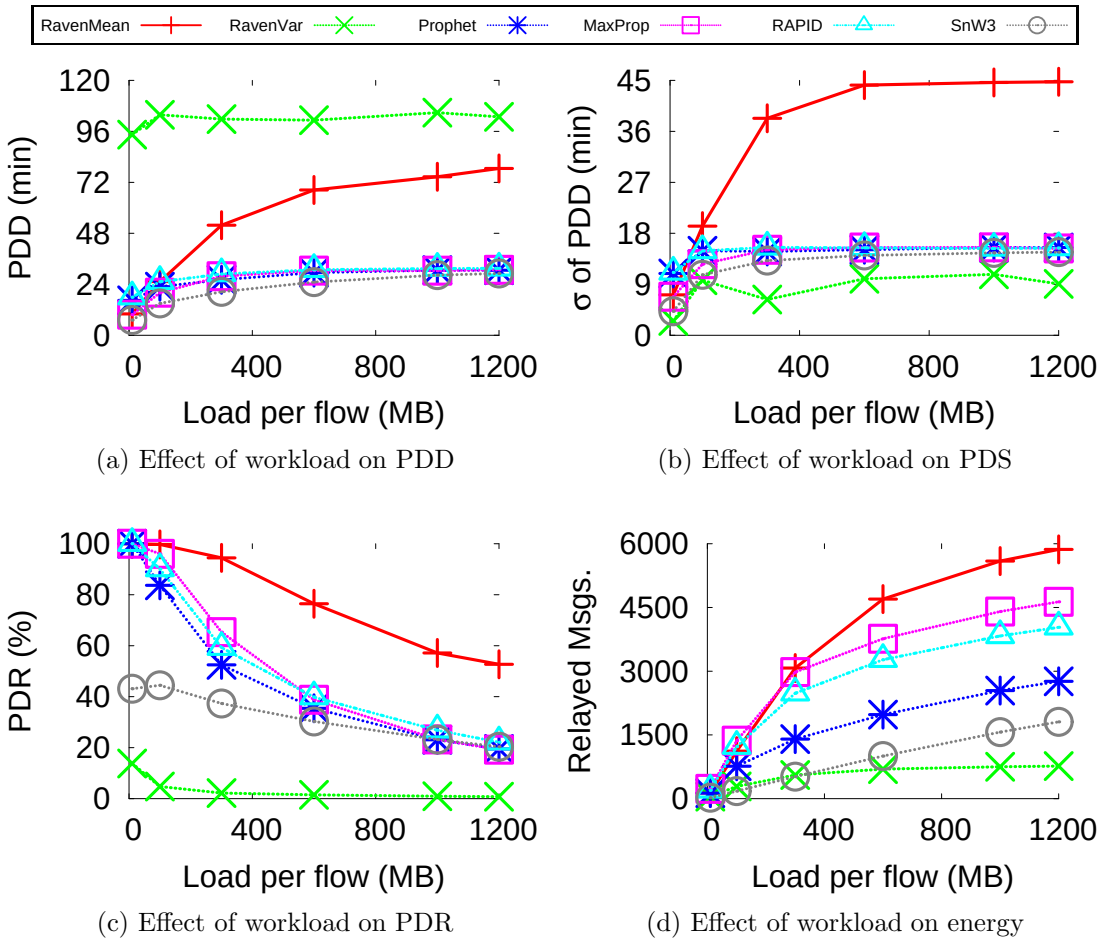
(d) Effect of ρ upon energy

Figure 7.5: Behavior of the QoS metrics as ρ and K change at a workload of 1GB per flow, 4MBps bitrate and 1x speed.

7.5.1 Tuning Raven

For a fair evaluation, two variants of Raven are constructed: RavenMean and RavenVar, which are designed to minimize the PDD and PDV respectively. In order to configure the two variants with appropriate K and ρ, experiments were conducted. The performance for varying K and ρ is shown in Figure 7.5. One immediate observation is that as ρ increases, the PDV decreases (Figure 7.5b) but at the cost of increased PDD (Figure 7.5a). This is an expected result since a higher ρ places more

emphasis on the variance of the paths and causes the algorithm to choose paths with lesser risk. Increasing ρ causes marked improvement at lower values while only incremental improvement is observed at higher ρ. The value at which ρ saturates depends upon the topology, which decides the Pareto frontier for the paths. Thus, choosing a ρ is highly dependent on the topology: the number of Centers as well as the speed of Mobile Agents. The PDR is very low (Figure 7.5c) since the simulation time was 10000 seconds (166 mins) and the increasing PDD causes packets to be delivered outside the simulation window. The PDD in (Figure 7.5a) is calculated *based on the delivered packets only*; since the PDR is high for high K, so is the PDD for high K. Result R3 is confirmed in the sense that at K, the effect of ρ decreases causing the metrics to stay constant. R1 is demonstrated since ρ has only an incidental and minor effect on the energy consumption (Figure 7.5d), causing the metric to stay fairly constant compared to the PDR. With an increase in K the amount of replication in the network increases. As expected, since a single packet travels on more and more paths simultaneously, the PDD decreases with increasing K (Figure 7.5a) and the PDR increases as well (Figure 7.5c). But this comes at a cost - both the PDV (Figure 7.5b) and the energy consumption increase with K (Figure 7.5d) for a given ρ. An infinite K is equivalent to Epidemic - a packet travels on all possible paths through the network. When a packet is flooded, the limited contact bandwidths are used inefficiently through redundant data transfers, causing some packets to be delivered quickly while other packets stay in the queue. As before, K follows the law of diminishing returns, showing only incremental changes at high K. Based on the above experiments, we choose $\rho = 0, K = 100$ for RavenMean and $\rho = 10, K = 1$ for RavenVar.



(a) Effect of workload on PDD

(b) Effect of workload on PDS

(c) Effect of workload on PDR

(d) Effect of workload on energy

Figure 7.6: Behavior of the QoS metrics as load per flow changes. Workload per flow was 1GB, speed multiplier was 1x and bitrate was 4MBps unless changed.

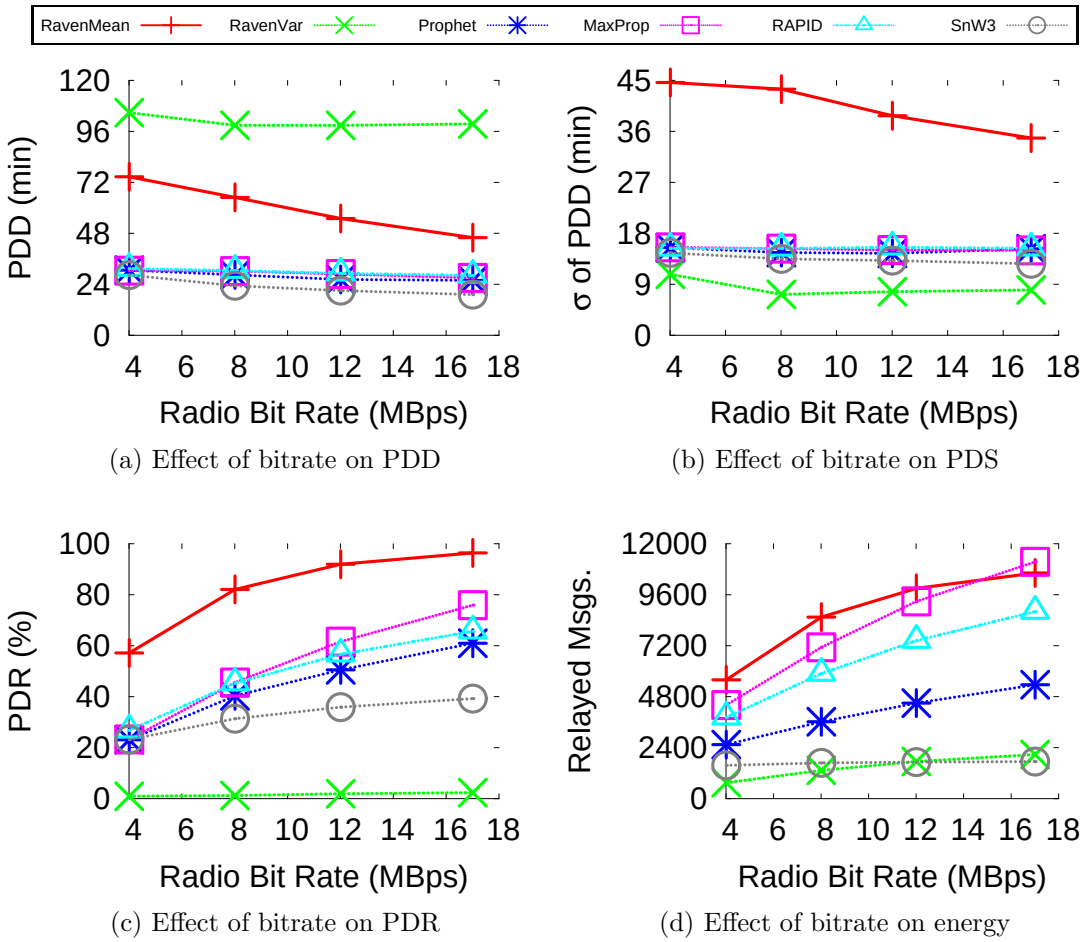
7.5.2 Effect of Load

Figure 7.6 shows the effect of steadily increasing load (the data generated per flow) on the four QoS metrics of interest. Before discussing the PDD, one needs to keep the PDR (Figure 7.6c) as reference since the PDD is calculated only for delivered packets. PDR decreases as load increases because the contact bandwidth is finite. RavenMean has the highest PDR since it has a very large K, while RavenVar has the lowest, since $K = 1$. All other protocols, except SprayWait, perform uncontrolled replication and

hence suffer at high loads. At the reference value of 1GB, RavenMean has 3x the PDR of other protocols. The contact bandwidth between nodes is finite and limited and so, packets have to wait longer in the transmission queue at each node as the size of the queue increases along with the load. Therefore, the PDD increases with load (Figure 7.6a). The PDD for RavenMean is the more than other protocols (2x more) - but only because it delivers more packets. The other protocols have comparable PDR and thus have comparable PDD. Normally, the PDV is proportional to the PDD - but because of risk-aversion, RavenVar can force a lower PDV in exchange for a higher PDD (Figure 7.6b). Because of low K, RavenVar is unaffected by the load. Since all of the presented DTN routing protocols treat packets as independent and do not tune their replication based on the load, the energy consumed increases for increasing load (Figure 7.6d). RavenMean consumes 1.5x more energy than Rapid, but delivers 3x as many packets. MaxProp replicates while waiting for an ACK to clear a message from its buffer, causing it to have the highest energy consumption among other protocols.

7.5.3 Effect of Radio Bitrate

The contact bandwidth is the major bottleneck in modern DTNs: while mass storage devices have become cheaper and faster leading to larger on-node buffers, the bandwidth of wireless devices has not kept pace. The effect of increasing radio bitrate on performance is shown in Figure 7.7. With increasing contact bandwidth, nodes can transfer more data during an opportunistic contact. The PDR increases almost linearly (Figure 7.7c). MaxProp takes maximum advantage of the increase, causing its PDR to appreciate by almost 4x. In comparison, RavenMean's PDR increases 1.5x because it has a fixed value of K that does not change with the workload. SnW3 is able to show only a 2x since its replication factor does not change, while Rapid and



(a) Effect of bitrate on PDD

(b) Effect of bitrate on PDS

(c) Effect of bitrate on PDR

(d) Effect of bitrate on energy

Figure 7.7: Behavior of the QoS metrics as bitrate changes. Workload per flow was 1GB, speed multiplier was 1x and bitrate was 4MBps unless changed.

Maxprop show a 3x increase. The PDD (Figure 7.7a) decreases as expected but not for RavenVar since its K does not change. Its PDV (Figure 7.7b) is fairly constant but the PDV of other protocols decreases. As the contact bandwidth increases towards infinity, the PDV should decrease towards zero. The number of relayed messages (Figure 7.7d) increases since the node buffers are cleared more frequently owing to increased bandwidth, allowing more data to be moved in the DTN.

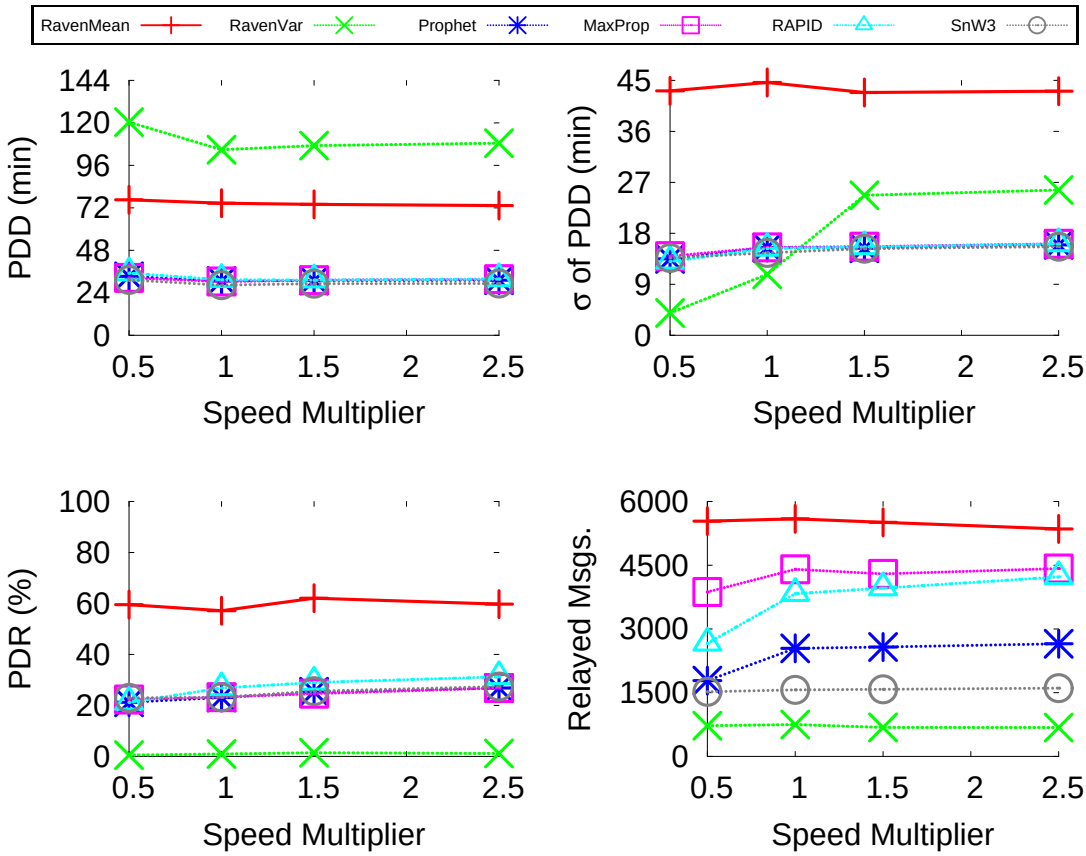


Figure 7.8: Behavior of the QoS metrics as node speeds change. Workload per flow was 1GB, and bitrate was 4MBps.

7.5.4 Effect of Node Speed

Changing the nodes' speed gives an opportunity for other protocols to improve their estimates of node mobility patterns. This is because as speed increases, the length of each contact opportunity decreases proportionately, keeping the total contact duration across all node pairs roughly constant (barring randomness since the movements are not cyclical). However *the number of contacts increases*. Protocols which maintain state can improve the quality of their metadata by exchanging it with more and more contacts. However, uncontrolled replication (i.e. replicating till an ACK is received for the packet) hinders the protocols' performance (Figure 7.8).

Both the minimum and maximum speed of each Mobile Agent was multiplied by the factor shown on the X axis. While multipliers of 2.5 are impractical (100mph road speed in a disaster struck area), it is shown for completeness. The PDR (Figure 7.8c) for RavenMean and RavenVar remain constant since a source routing approach is used and the travel time distributions in the stochastic multigraph change proportionately with changes in node speed. RAPID sees an increase from 21% to 31% since it is utility based and higher speeds mean more contact opportunities during which a replication decision is to be made. Similarly Prophet and MaxProp, which update their estimates of delivery likelihood at each contact, are able to replicate to better hosts and make more efficient use of contact bandwidth. Number of relayed messages (Figure 7.8d) include the control channel packets of very low size that these protocols use to exchange metadata; hence it increases with increasing contact opportunities. It stays somewhat constant for RavenMean/RavenVar/SprayWait since they do not exchange metadata and the replication factor is fixed. The PDD (Figure 7.8a) decreases by about 150s for the other protocols since they are able to make better routing decisions. Similarly, the PDV (Figure 7.8b) increases for protocols which display a decrease in PDD.

7.6 Conclusion

We have presented Raven, a DTN routing protocol that allows the user to control the PDV as well as other QoS metrics. It uses multigraphs with stochastic weights to model the PDM mobility model. K least risk paths are chosen by a node using the K Safest Paths algorithm, and is used to perform source routing. USAR hosts forwarding packets to each other based on the risk involved in delivering the message to their Center. Mathematical analysis of Raven shows several interesting properties that describe the coexistence of risk-aversion and replication. Raven is able to out-

perform other protocols based on metrics chosen by the user. When configured for minimizing PDD, it is able to deliver 3x more packets than other protocols.

8. ENERGY EFFICIENCY IN FOG COMPUTING

Disaster Response Networks (DRNs) are designed to assist first responders during the recovery period following a large scale disaster, where the lack of end to end paths as well as unpredictable mobility is common. The system lifetime of deployed DRNs is critical to successful recovery, as are performance metrics such as packet delivery delay. As a node sleeps more and more, the number of contact opportunities are reduced, impacting performance. In this section we investigate this Pareto frontier [125] between system lifetime and network performance. Given a desired deployment duration, the system is tuned to extract the best possible performance.

8.1 Introduction

Energy consumption patterns in wireless ad hoc networks have been studied extensively. Reducing usage of the radio interface saves energy. The radio is mainly used for two purposes in a DRN: relaying messages and discovering contacts. Because of sparse node density and large node inter-contact times, it is imperative that a node not miss any contacts. The number of relayed messages can be reduced using various techniques at the routing layer. Optimal node wakeup intervals can be determined either by an algorithm or by using external hardware that alerts the node when a contact is in range.

This section aims to not just reduce the energy consumption, but to quantify the effect on system performance while doing so. The existence of a Pareto front

is motivated, and later proved using simulation. The ability to operate a DRN at various Pareto-optimal points means that the users (first responders in this case) can actively control the performance/energy tradeoff. One of the requirements for discovering this front is the ability to mathematically model both the energy consumption and performance of the routing protocol in terms of the protocol-specific parameters. Raven, which is the routing framework used in the Fog, uses stochastic multi-graphs to model mobility in the area. Using this graph, a user-specified number of the most optimal paths between source and destination are computed, and a packet is routed on these paths simultaneously. The source routing approach used by Raven makes it easy to compute an expectation of system performance as well as energy consumption. A genetic algorithm is used to solve the dual objective problem and discover the Pareto-optimal points. The contributions of this section are as follows: 1) the first (to the best of our knowledge) framework that is able to discover the Pareto front between performance and energy consumption and models it as a dual objective optimization problem, and 2) an algorithm to estimate the performance and energy consumption of Raven.

8.2 Motivation

In this section we motivate this section by analyzing various methods to save energy at the system level in a DRN, positing the existence of a Pareto front, and finally investigating the possibility of operating at various Pareto optimal points on this front.

In a DRN, mobile vehicles are used to mule data to and from static nodes (deployed at collapsed buildings, for example). Being spread over a large geographical area with few resources, the mobility in a DRN is very sparse. As a result, the node inter-contact time is on the order of tens of minutes or even hours: it is typical for

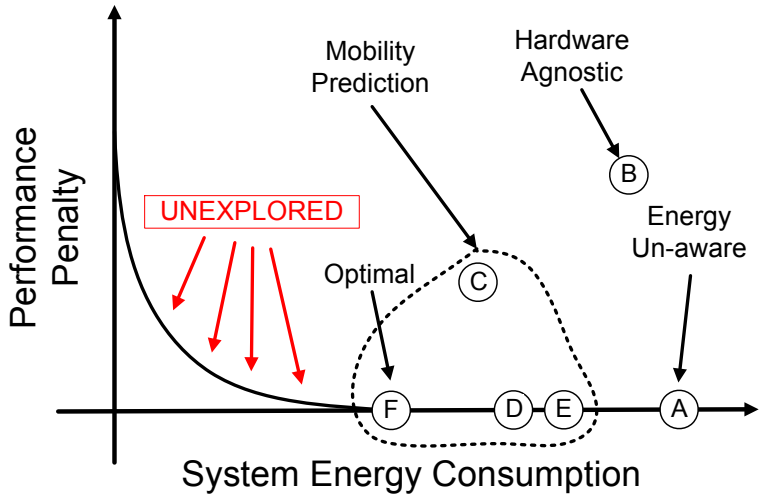


Figure 8.1: Schematic showing the Pareto front between system performance and energy consumption. Points A-F show where state of art methods lie.

a node to spend only about 20% of its lifetime in contact with at least one other node. This quantity is henceforth called the *contact time (CT)*. This means that a node can afford to sleep in the inter-contact time (about 80%) and still not affect the performance by missing any contacts.

One of the trivial methods of energy saving in a DRN is to not save any at all - each node stays awake all the time. Network performance is maximal, but so is the energy consumed ("A" in Figure 8.1). One simple optimization is to have the node sleep in the inter-contact time. Either mobility prediction or hardware assistance can be used to wake up the node. For example, [41] uses long range auxiliary radios to detect a vehicle and subsequently wake up the node. In such a scheme ("E" in Figure 8.1), there is a reduction in energy consumed but no loss in performance. These prediction schemes are not completely optimal; replacing it by an oracle ("D" in Figure 8.1) could possible save a little more energy.

It is important to note that saving energy purely in software (by minimizing the number of transmitted messages at the routing layer) or only on the radio interface

Device	Radio Power	Base Power
Sensor	0.06W	0.006W
Smartphone	0.7W	0.2W
Router	0.72W	3W

Table 8.1: Power profiles of various devices. Sensor: based on a 3V Epic mote with 802.15.4. Smartphone - based on 3.7V HTC Evo 4G with 802.11. Router - based on 12V Mikrotik RB433UAH router with 802.11.

(by enabling 802.11 PSM mode) will not save much energy. This is because the energy consumed by the radio interface in a typical WiFi router is much less than that consumed by the base board. Low power sensor networks, however, have the opposite characteristic. Table 8.1 compares a WSN mote, a hand held WiFi smartphone and a WiFi router. As we can see, the power consumed by a router's radio (0.72W) can be 5x smaller than the base board (3W). A scheme that aims to minimize the number of relayed messages, as compared to "A", will have low performance but will save a little energy ("B" in Figure 8.1).

Depending on the routing protocol that is used, not all contacts may involve transfer of data. For example, in Rapid [126] a node transfers packets only to a node with higher utility. An optimization that could be performed here is to have a low cost low capacity backhaul link, such as a satellite link, that nodes can use to query the buffer contents of any other node and thus calculate the utility or any similar metric. As a result, a node can infer whether a vehicle has any packets that could be transferred; if not, it can sleep through the contact. Such a scheme ("F" in Figure 8.1) represents the most energy that can be saved without any performance penalty. The percentage of contact time that involves useful transfer of data in relation to the node lifetime is called the *useful contact time (UCT)*.

Given the above background, we investigate in this section the existence of a

Pareto front in a DRN. The major intuition is that if some nodes can be *excluded from the routing process*, then those nodes can save energy by sleeping. However there is a performance penalty involved; it is the aim of this section to quantify this trade off and investigate the feasibility of operating at various points on the Pareto front. It has to be mentioned that knowledge of data flows in the network is essential. If every node in the network is either a source or destination, then no Pareto front exists. However, if some nodes simply route data and are neither a source or destination, then such a Pareto front exists.

8.3 Preliminaries and Problem Formulation

In this section the problem is formulated and a solution is derived. First, the concepts of firm and potential flows are introduced. These flows are nothing but source-destination pairs which quantify the data workload in the network. We then introduce our previous work Raven, which is a risk-averse routing framework for DRNs. Finally the problem formulation, which incorporates the sub-problem of converting potential flows to firm flows, is quantified. A genetic algorithm based solution is then proposed.

8.3.1 Problem Formulation

The major insight in this section is that if a Fog device (such as a Data Waypoint) is not on one of the paths chosen by Raven to route data, it can sleep and save energy. Thus, the number of unique nodes present in the set of paths for all flows is an indicator of the energy consumption. This number can be reduced in a variety of ways: in the process of converting potential flows to firm flows (choosing Centers such that flows overlap) or in Raven (by reducing K or by choosing a set of paths other than the shortest). However, these techniques affect the system's performance - and therefore, there is a Pareto frontier between performance and energy consumption.

Additionally, the user can control the operating point along this frontier by simply changing system parameters like those used by Raven.

The disaster area consists of c centers $C_1 \ldots C_c$ and is represented by the stochastic multigraph $\mathcal{S}$. There are f firm flows $F_1 \ldots F_f$ and their sources $\{F_i^S\}$ and destinations $\{F_i^D\}$. p potential flows $P_1 \ldots P_p$ have their availabilities $A_1 \ldots A_p$ and sources $\{P_i^S\}$. Thus the total number of flows is $h = f + \sum_{i=1}^{P} \lceil cA_i \rceil$. Each of these h flows has an associated parameter $k_1 \ldots k_h$ that is used by Raven (the number of paths to compute). The user has specified a global Raven parameter K: as a result, $1 \leq k_i \leq K$. When Raven computes paths for each of these h flows, let the number of unique Centers in the union of these paths be $\bar{c} \leq c$.

Now since $\mathcal{S}$ is stochastic, each path in this graph has an associated mean and variance (equal to the sum of means/variances of the constituent edges), and is hence a distribution. The path weight represents the packet delivery delay because the physical travel delay in a DTN is a major component of the packet delivery delay [2]. When a packet is sent on k paths simultaneously, the expected delay is the minimum of the delays of the k individual paths; it follows that the per-flow packet delivery delay is the minimum of k normally distributed random variables. For the flow numbered i with paths parameter k_i, the delay is:

$$D_i = min\{D_{i1}, D_{i2}, \ldots, D_{ik_i}\} \tag{8.1}$$

where each D is a distribution and not a scalar. A closed form expression for this minimum of several random variables is not trivial. For h flows, the overall packet delivery delay distribution is:

$$D = \frac{\sum_{i=1}^{h} D_i}{h} \tag{8.2}$$

where to re-emphasize, all quantities except h are normally distributed random variables with a mean and a variance. Using the mean-risk probability model, the risk of this distribution D is

$$risk(D) = E[D] + \rho * \sqrt{V[D]} \tag{8.3}$$

Given the above notation, our objective is to minimize the delay (which is called risk when variance is taken into account, i.e., $risk(D) := E[D]$ when $\rho = 0$) as well as $\bar{c}$. There are three parts to this problem: (1) the conversion of potential flows to firm flows, (2) applying Raven to each of the firm flows so that the delay distribution D can be estimated, and (3) tuning Raven's K parameter so that energy and $risk(D)$ are minimized. The above three problems are solved in a single optimization problem as follows. To convert a potential flow P_i into $\lceil cA_i \rceil$ firm flows, consider a binary vector V of length c. The ith element V_i corresponds to Center C_i: $V_i = 0$ if the Center is not chosen as a destination, and $V_i = 1$ otherwise. The sum of this bit vector should be $\lceil cA_i \rceil$. Repeating this procedure for p potential flows, the length of V becomes pc. Because Raven needs a K parameter for each flow, V is augmented with h more integers. Thus, the vector V of length $(pc + h)$ can now be used as input. It is a dual objective non-linear program:

$$\min_{V} \quad \textsc{Risk}(\mathrm{V}),\ \textsc{Uniq}(\mathrm{V}) \tag{8.4}$$

$$\text{s.t.} \quad \sum_{j=1}^{c} V_{(i-1)c+j} = \lceil cA_i \rceil,\ i = 1 \ldots p \tag{8.5}$$

$$1 \leq V_i \leq K,\ i = (pc+1) \ldots (pc+h) \tag{8.6}$$

Constraint 8.5 deals with potential flows. It stipulates that the total number of

firm flows created (by setting a Center's bit) for each potential flow equals $\lceil cA_i \rceil$. Constrain 8.6 makes sure that the K parameter needed by Raven cannot exceed the global value of K specified by the user. Equation 8.4 involves two procedures RISK and UNIQ which calculate $risk(D)$ and $\bar{c}$ respectively.

Procedure 7 details the calculation of the two objectives. Steps 1-7 involve the creation of firm flows from potential flows: for each potential flow (Step 1), if the bit corresponding to a Center is set (Step 3), a new firm flow is created (Step 4). Now that all h firm flows have been created, Raven is applied to each of these (Step 9) along with the Raven parameter K (Step 10). The resulting set of k_i paths are collected and duplicates are removed (Step 11). The number of unique vertices in these paths is nothing but the number of unique Centers (Step 13) since each vertex in the stochastic multigraph corresponds to a Center. The delay distribution, which is the path weight of each of these paths is calculated (Step 14), averaged (Step 15) and the risk is calculated (Step 16).

8.3.2 Solution

Equation 8.4 is a dual objective, non-linear optimization problem. Because of its complexity, a stochastic optimization approach is preferred, as opposed to a deterministic one. Evolutionary algorithms are specially suited to solve multi-objective problems - and genetic algorithms (GAs) are the most popular variety of evolutionary algorithms. We use the NSGA-II algorithm to solve Equation 8.4, owing to its speed and low complexity. It is also able to handle disconnected Pareto fronts.

The input to the algorithm, vector V, is referred to as a "chromosome" in GA parlance. It is a string of numbers, either binary or real valued (in this case, integer valued). Through multiple GA operations like crossover, selection and mutation, new candidate solutions are generated in a stochastic fashion. The current set of

Procedure 7 RISK(V) and UNIQ(V)

Input: vector V, firm flows $\{F\}$, K, ρ

for $i := 1 \ldots p$ **do**
  for $j := 1 \ldots c$ **do**
    if $V_{(i-1)c+j} == 1$ **then**
      $F \leftarrow F \cup$ (a new firm flow between P_i^S and C_j)
    end if
  end for
end for
$paths \leftarrow \Phi$
for $i := 1 \ldots h$ **do**
  $\{Q_k\} \leftarrow$ (Raven with s/d F_i^S & F_i^D, k-parameter V_{pc+i})
  $paths \leftarrow paths \cup \{Q_k\}$
  $D_i \leftarrow \min\{Q_1, Q_2, Q_3 \ldots Q_K\}$
end for
UNIQ(V) $= \bar{c} \leftarrow$ unique nodes in $paths$
RISK(V) $\leftarrow (\sum_{i=1}^{h} E[D_i] + \rho * \sqrt{V[D_i]})/h$
return RISK(V),UNIQ(V)

candidate solutions (the "population") is evaluated (the "fitness" is calculated) and filtered to retain only non-dominated solutions. NSGA-II gives preference to Pareto optimal points that are situated far away, so as to capture both extremes of the front. The crossover operator used is Simulated Binary Crossover (SBX), since the chromosome is real valued. Mutation occurs according to the Polynomial operator. Selection happens in a Binary Tournament fashion.

8.4 Performance Evaluation

In this section we present the performance evaluation of our energy saving scheme. First, the existence of the Pareto front is confirmed by implementing NSGA-II and solving Equation 8.4, for a given scenario with Centers, Mobile Agents, potential and firm flows. Using a DTN simulator, the mathematical modeling is validated by running Raven at the Pareto points and verifying that the delay decreases when energy increases and vice-versa. Two of these points (the extremes) are chosen, and

the Raven protocol is then evaluated at each of these Pareto optimal points, and compared with state of art DTN routing protocols such as MaxProp, RAPID and Prophet. Metrics used for comparison are packet delivery delay (PDD), the packet delivery ratio (PDR), the number of relayed messages (REL), and the total awake time (TAT). REL refers to the total number of packet transfers in the network *across all nodes*. TAT is the sum total of the awake time across *Centers only*, assuming that nodes stay awake only during those contacts where is transfer of data and that Mobile Agent nodes are powered by the vehicle's battery. The former assumption is justified by the existence of a low bitrate backhaul link which allows nodes to determine each other's buffer contents, as explained in Section 2.1. We have chosen REL in addition to TAT for fair comparison: only Raven is optimized for Center-only energy savings (TAT) while other protocols aim to reduce REL across all nodes. TAT is a better metric that UCT since the TAT represents all nodes of interest (namely Centers), whereas UCT is an average across all nodes.

8.4.1 Obtaining the Pareto Front

On the Helsinki street map, an EOC, a Triage and 25 Centers (collapsed buildings) were setup, their locations chosen randomly. A firm flow was setup from Building 1 to 9 as well as a potential with availability of 0.1 originating at the EOC, for a total of 4 ($= 1 + \lceil 27 \times 0.1 \rceil$) flows. Three ambulances, three supply vehicles and 10 volunteers moved according to their respective mobility models. The stochastic multigraph for this scenario was computed.

A Java implementation of NSGA-II was provided by the jMetal package: the parallel version where each chromosome is evaluated in a separate thread was chosen. After implementing Algorithm 7 within the jMetal framework, a Pareto front with 22 points was obtained and is shown in Figure 8.2. As expected, energy consumption

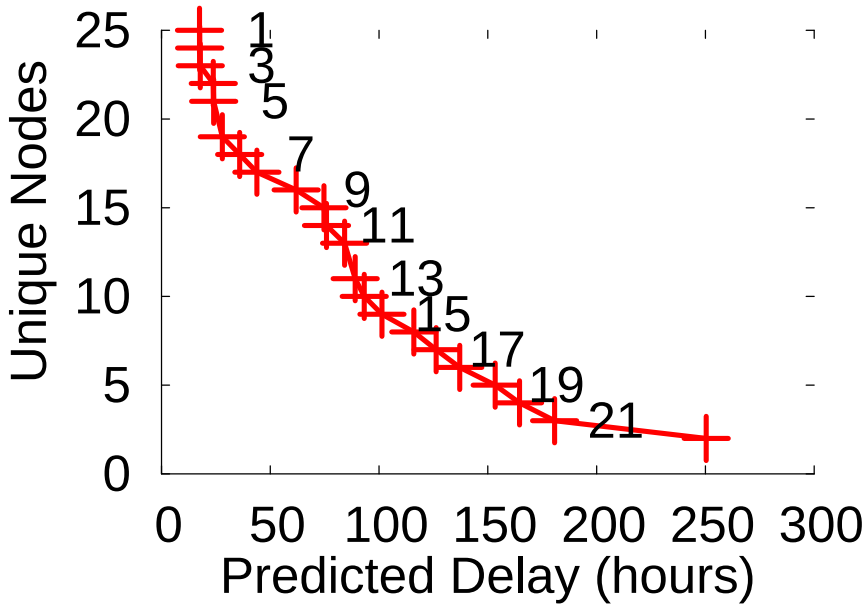
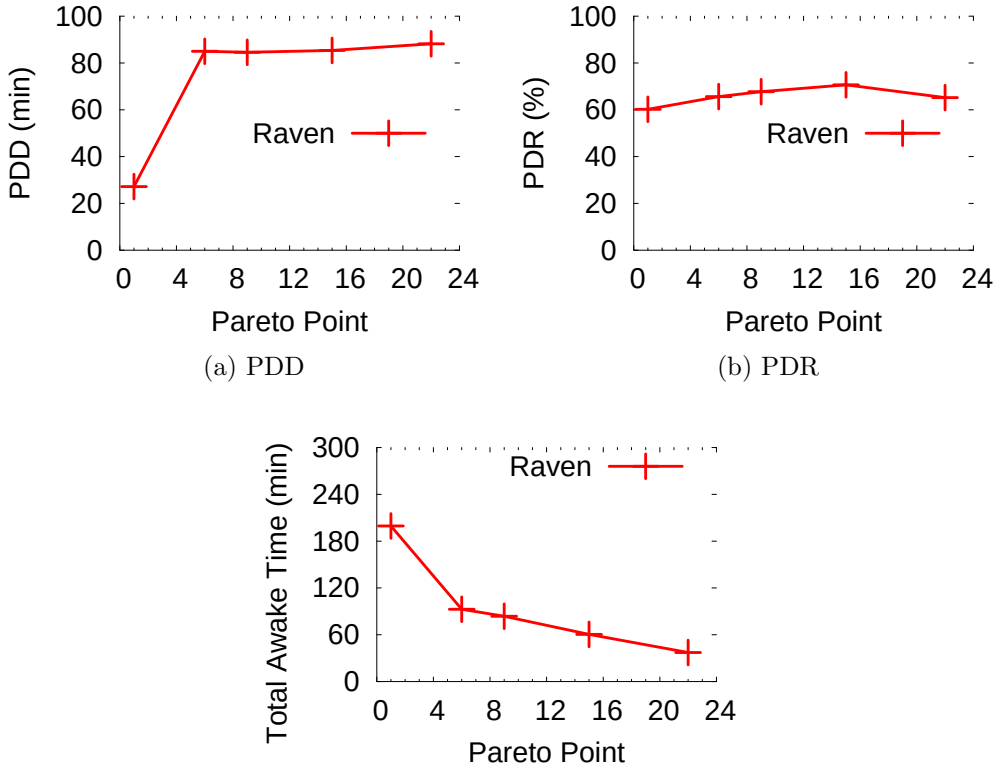


Figure 8.2: The Pareto front as obtained from simulation.

(Y axis) can be minimized only at the cost of increased delay (X axis). Pareto Point 1 utilizes all the 27 (= 25 shown on the graph + 2 source/destination) Centers with K values of [197 164 196 144] for each of the 4 flows, while Point 22 used only 4 Centers with K values of [2 4 3 4].

This setup initially had long run times, due to the fact that the constraint handling extensions of NSGA-II were not implemented. The probability of obtaining a randomly generated chromosome that satisfied the potential flow conversion constraints (Equation 8.5) was low, and decreased as the number of Centers increased. We developed an improved chromosome generation subroutine that produced high quality initial solutions. First, elements $1 \ldots pc$ of the input vector V were set to zero. Then, for each sub-vector of V corresponding to each of the p potential flows, $\lceil cA_i \rceil$ elements were randomly chosen and set to unity, satisfying Equation 8.5. As a result, each generation (iteration) in NSGA-II had valid chromosomes that were not



(a) PDD

(b) PDR

(c) TAT

Figure 8.3: The effect of operating at different Pareto optimal points upon (a) packet delivery delay, (b) packet delivery ratio and (c) the total awake time.

discarded, leading to evaluation of more and more chromosomes, and the discovery of many more Pareto-optimal points. The run time was reduced dramatically, resulting in quicker experimentation.

8.4.2 Verifying the Pareto Front

The performance of Raven as it is made to operate at various Pareto points is shown in Figure 8.3. Data for this and subsequent sections was obtained using TheONE, a Java based opportunistic network emulator at the packet level and *not* NSGA-II. The data workload per flow was 300MB, all created at $t = 0$ and the radio

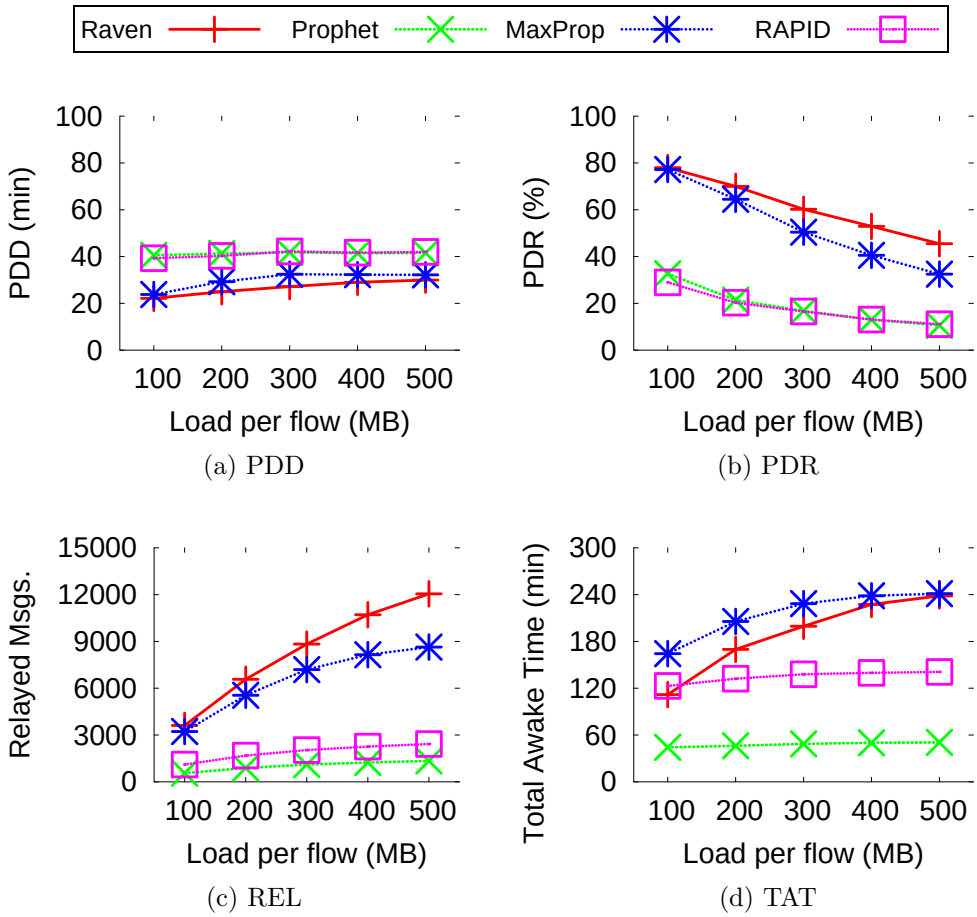
bitrate was 8MBps. The entire simulation lasted for 166.67 minutes and each data point is averaged over 200 random runs. Point 1, as defined in Figure 8.1 optimizes the PDD at the cost of high energy consumption. This is confirmed in simulation since the PDD for Point 1 (Figure 8.3a) is the lowest, whereas the TAT (Figure 8.3c) is the highest. The TAT represents the system-wide energy consumption of the nodes at Centers - thus a high TAT means high energy consumption. A similar observation holds for Point 22, confirming the correctness of the problem formulation presented in Equation 8.4. Surprisingly, the PDR (Figure 8.3b) is fairly constant across all Pareto points. This can be explained by the fact that each potential flow results in a different set of firm flows for each Pareto point. The TAT decreases by about 66% as we move towards the energy-optimal end of the Pareto front, since the average K value per flow decreases. To summarize, the NSGA-II based optimizer provides the user with a variety of Pareto-optimal points, each of which represents a unique balance between performance and energy consumption: for a ~5 minute increase in PDD, the average energy consumption across *all active nodes* is decreased by ~66%.

8.4.3 Delay Optimal Raven

We first operate Raven at the delay optimal Pareto point (Point 1 in Figure 8.2), and compare it with state of art protocols. Increasing the data workload per flow will saturate the network while the contact opportunities remain the same, resulting in higher PDD and lower PDR. Increasing the radio bitrate increases the amount of data that can be transferred per contact - the expected result is that PDD should be lower and PDR higher. The results are shown in Figure 8.4.

8.4.3.1 Effect of Workload

The PDD of Raven is higher than the other protocols (Figure 8.4a) but only because it delivers more packets than the others (Figure 7.6c), and the PDD is only

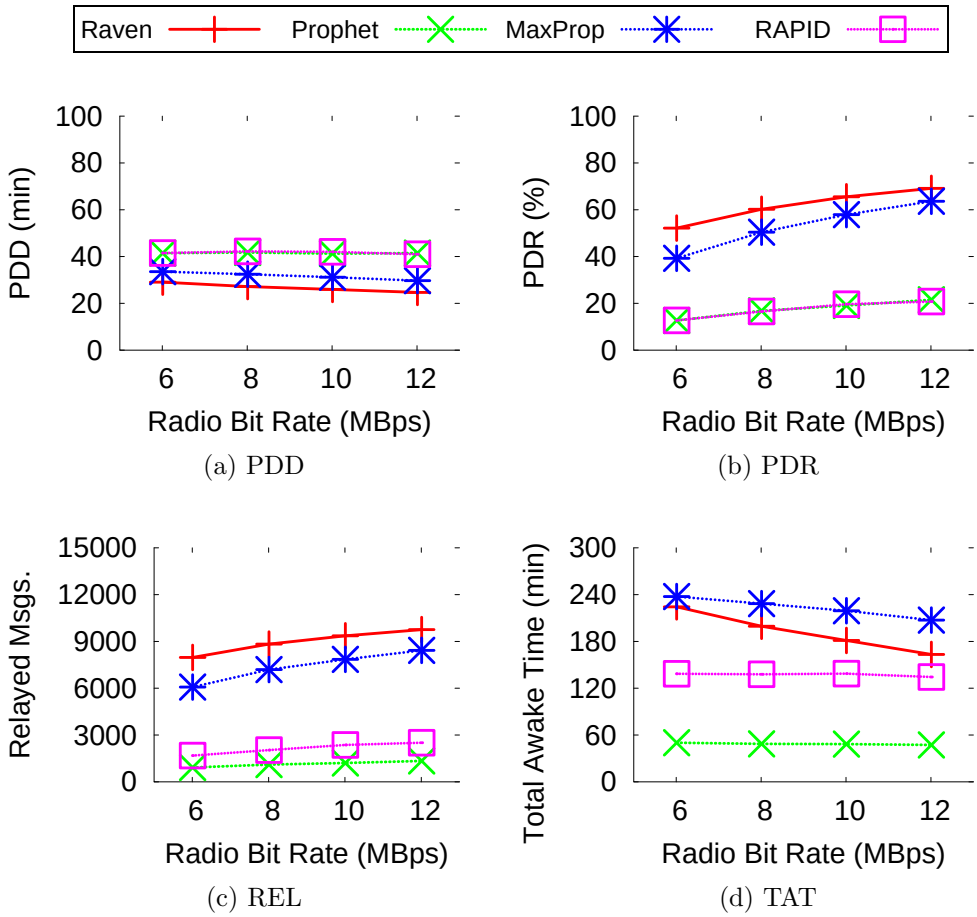


(a) PDD (b) PDR

(c) REL (d) TAT

Figure 8.4: Delay optimal Raven: effect of increasing workload per flow on the performance of several DRN routing protocols: (a) PDD (b) PDR (c) number of relayed messages and (d) total awake time. Raven operated at Pareto Point 1, bit rate was 8MBps.

calculated for delivered packets. Raven is able to deliver ~15% more packets than the nearest competitor. MaxProp replicates packets till an ACK is received - as a result it relays more packets than others (Figure 8.4b): this number grows with the data work load. However, due to excessive replication and redundant packet transfers, the capacity of the network is not used optimally and the PDR is lower as a result. Nodes need to stay awake for longer durations to relay more packets - MaxProp has the highest energy consumption (Figure 8.4d). Both Prophet and



(a) PDD (b) PDR

(c) REL (d) TAT

Figure 8.5: Delay optimal Raven: effect of increasing radio bitrate on the performance of several DRN routing protocols: (a) PDD (b) PDR (c) number of relayed messages and (d) total awake time. Raven operated at Pareto Point 1, workload was 300MB.

RAPID perform selective forwarding, relaying packets to only those nodes with a higher probability of reaching the destination (Prophet) or based on the marginal utility of relaying a packet (RAPID). Since RAPID makes a relaying decision based on the decreasing order of marginal utilities for each packet in the buffer, it relays more packets than Prophet. To summarize, Raven has the highest PDD, but it also delivers more packets and consuming less energy than MaxProp.

8.4.3.2 Effect of Bitrate

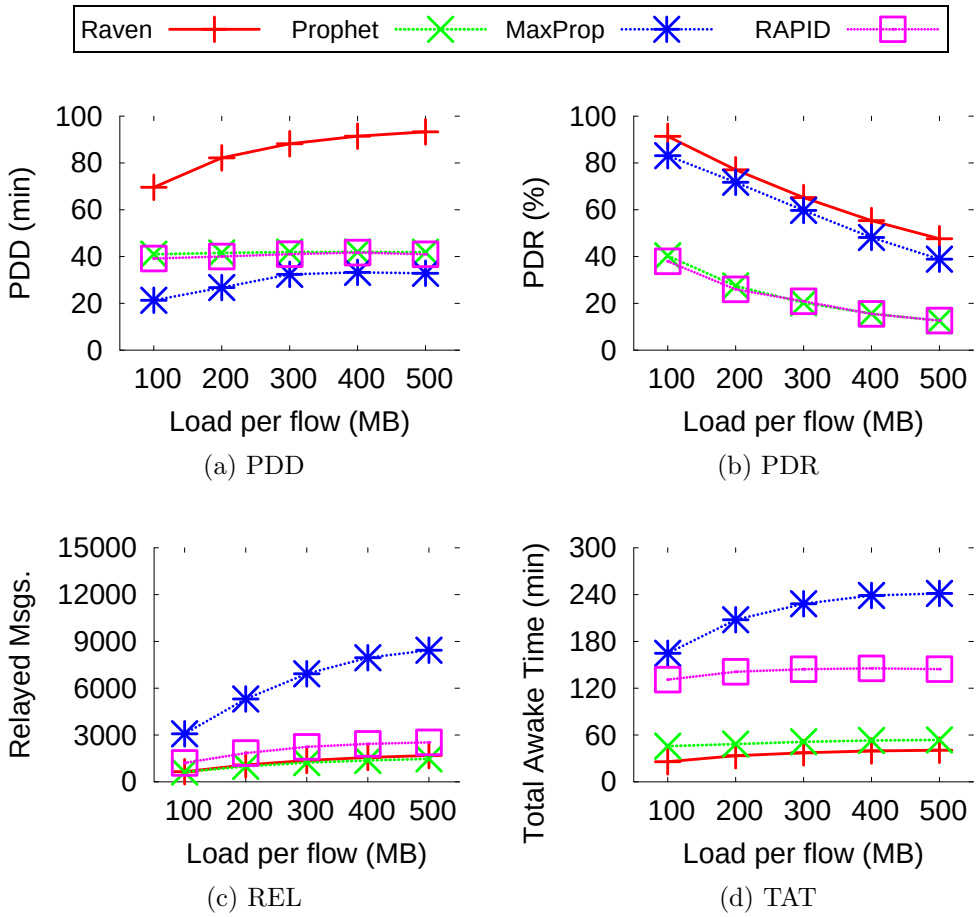
Increasing the radio bit rate allows more data to be transferred per contact, while keeping the total amount of contact time constant. Increasing this metric measures how effective a protocol is at prioritizing packets at each contact. In general, all protocols show reduced PDD (Figure 8.5a), increased PDR (Figure 8.5b) and increased number of relayed packets (Figure 8.5c). The decrease in TAT (Figure 8.5d) can be explained as follows. Each node only transfers those packets that the other node does not have. With increasing radio bit rate, nodes are able to disseminate all of the generated data (note that no new data is generated after $t = 0$). Therefore, towards the end of the deployment, the number of contacts which involve transfers of unique packets decreases - thus decreasing the TAT. To summarize, Raven has the highest PDD, but delivers 20% more packets than MaxProp, while showing a 50% *reduction* in system wide energy consumption.

8.4.4 Energy Optimal Raven

Next, we operate Raven at the Pareto point which guarantees the highest energy savings at the cost of PDD (Point 22 in Figure 8.2), and compare it with state of art protocols. Compared to the previous subsection in which Raven operated at Point 1, the endpoints chosen for conversion of potential flows to firm flows are different *but still satisfy the availability constraints*. State of art protocols are evaluated with the same final firm flows as Raven for fairness. Results are shown in Figure 8.6. The axes in each of the figures is identical to Figure 8.5 for ease of comparison.

8.4.4.1 Effect of Workload

Raven has the least energy consumption, both in terms of the number of relayed packets (Figure 8.6c), as well as the total awake time metric (Figure 8.6d), across *all*

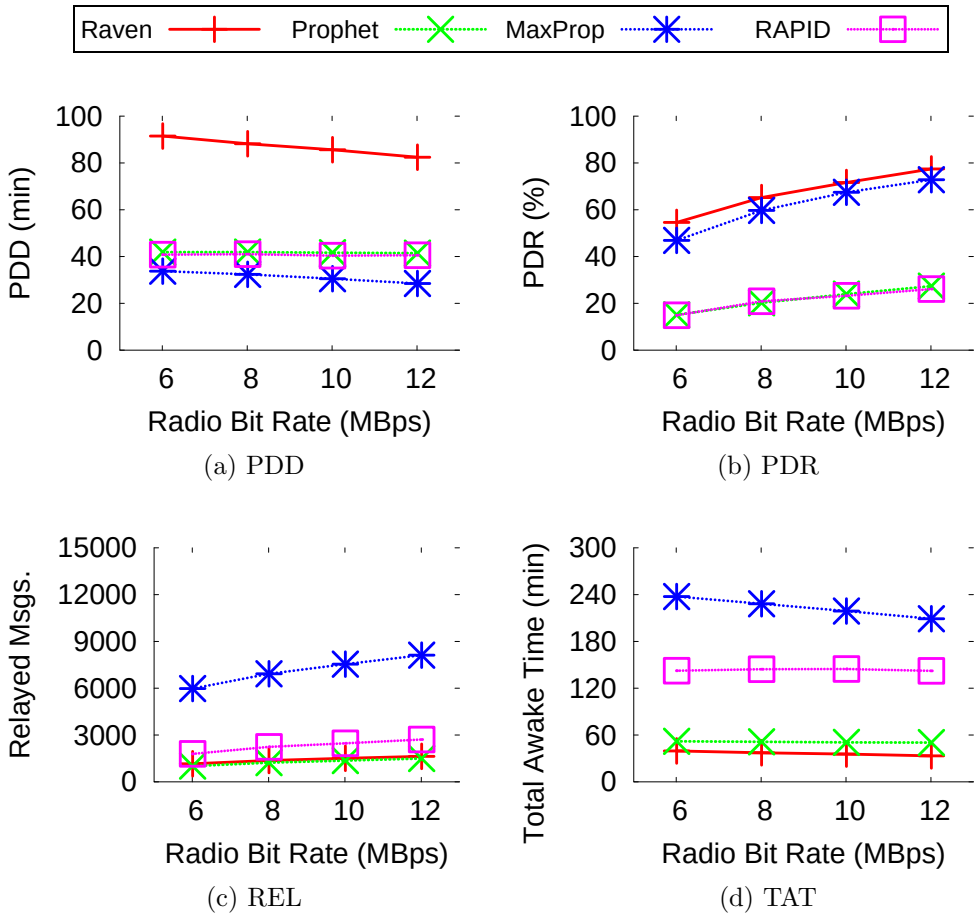


(a) PDD (b) PDR

(c) REL (d) TAT

Figure 8.6: Energy optimal Raven: effect of increasing workload per flow on the performance of several DRN routing protocols: (a) PDD (b) PDR (c) number of relayed messages and (d) total awake time. Raven operated at Pareto Point 22, bit rate was 8MBps.

data workloads. Additionally it delivers the most packets (Figure 8.6b) compared to other protocols. However, the delay is higher than the others (Figure 8.6a), but only because it delivers more packets while spending less energy. Compared to Figure 7.6a, the average delay for Raven is higher by a few minutes. This is a very small price to pay for having the least energy consumption. MaxProp, an epidemic-like routing protocol has the least delay and second best PDR, but at the cost of high energy consumption. For comparison, MaxProp relays 5x as many packets as Raven and



(a) PDD (b) PDR

(c) REL (d) TAT

Figure 8.7: Energy optimal Raven: effect of increasing radio bitrate on the performance of several DRN routing protocols: (a) PDD (b) PDR (c) number of relayed messages and (d) total awake time. Raven operated at Pareto Point 22, workload was 300MB.

based on TAT, has 5x the energy consumption. Even then, it does not deliver as many packets as Raven can.

8.4.4.2 Effect of Bitrate

Once again, compared to the delay optimal variant (Figure 8.7), Raven has the least energy consumption both in terms of the number of relayed packets (Figure 8.7c) as well as the total awake time of Centers (Figure 8.7d), while delivering the *most*

packets (Figure 8.7b). Similar trends are observed, such as increasing PDR, decreasing PDD, increasing REL and decreasing TAT with an increase in radio bitrate. Raven is found to consume almost 80% less energy than MaxProp, while delivering about 10% more packets.

8.5 Conclusions

We have presented a framework to characterize the Pareto front between system performance and energy consumption. Certain nodes in the network which only relay data but do not produce or consume it, can be excluded from the routing process to save energy - at the cost of performance. A dual objective non-linear program is formulated with Raven as the underlying routing protocol, and is then solved using NSGA-II. A set of Pareto optimal points is found, along with the respective network settings (potential flow endpoints and K values for each firm flow). For a 5 minute increase in delivery delay, a 3% decrease in energy consumption can be achieved. The setup is evaluated using TheONE simulator and compared against state of art protocols.

9. MOBILE SELF-LOCALIZATION IN FOG COMPUTING

In this section we present the working of the Localization App that was discussed earlier. As mentioned before, this app is designed for the Sensor class of devices since they typically do not have on board GPS owing to cost and energy constraints. When these Sensors are deployed indoors or in GPS-denied environments, there is a need for a self-localization technique that will allow each device to accurately determine its location. Self-localization refers to the process of a node estimating its own location in a distributed fashion, using a small number of nodes as "anchors" (nodes that already know their location). This Localization App can be effectively used in scenarios such as responders exploring a collapsed building, where a Sensor device is present on each responder as well as each of their tools, like cement saws. The location of each responder/tool, in addition to data generated by them, can be used as high quality informational data that can be used at the EOC for a variety of purposes.

Received signal strength (RSS) based ranging is a popular localization technique. Since the RSS is inversely proportional to the distance between transmitter and receiver, the RSS can theoretically be used to recover the distance. However, in complex environments such as a collapsed building, the RSS is inaccurate due to radio phenomenon like interference and reflections off cement surfaces. Therefore, since the RSS is inaccurate, the computed range also suffers, leading to inaccurate location estimation. FuzLoc is based on the premise that the location can be computed

more precisely *as an area* instead of an imprecise 2-dimensional point, when given imprecise RSS data. Fuzzy Logic is used to convert traditional variables like RSS into fuzzy variables called *bins* which represent a range of values. Then, learned intelligence is used to associate fuzzy RSS bins with fuzzy distance bins, which are subsequently used in a non-linear system of equations to compute an area instead of a 2-D point.

This work [127] improves upon our previous work [128] by considering 2-hop anchors. Using multi-hop anchors increases the localization accuracy, since using more anchors for a multi-lateration based localization method will increase accuracy. Network messaging overhead is minimal, and computing/storage overhead is almost non-existent. Compared to [128], all the simulations have been re-done considering 2-hop anchors instead of single hop anchors.

9.1 Introduction

Wireless sensor networks are increasingly a part of the modern landscape. Disciplines as diverse as volcanic eruption prediction [129] and disaster response [118] benefit from the addition of sensing and networking. A common requirement of many wireless sensor network (WSN) systems is localization, where deployed nodes in a network discover their positions. In some cases, localization is simple. For smaller networks covering small areas, fixed gateway devices and one-hop communications provide enough resolution. Larger networks may be provisioned with location information at the time of deployment [130].

However, in many common environments, localization is more difficult. GPS-based localization may be unreliable indoors, under forest canopies, or in natural and urban canyons. For example, GPS is used for high-precision asset tracking in [131] but fails indoors. Signal strength-based solutions similarly fail when there is a high

degree of RF multi-path or interference. [132] relies on accurate measurement of RF TDOA and distance traveled and quickly degrades as accuracy decreases. Radio interferometry localizes nodes to within centimeters in [106] but fails in multipath environments. Mobile beacons roam an outdoor environment in [133] but localization requires a dense network and assumes favorable conditions. All these solutions rely on stable environments with low multi-path, where measured or sensed ranges (which are typically obtained by time of arrival, angle of arrival or received signal strength techniques) reliably predict the actual distance between two nodes. For low multi-path environments, accurate models have been proposed for estimating time of arrival, angle of arrival and received signal strength [134].

Mobility complicates the localization problem since node to node distance variations *and environment changes* (e.g., due to node mobility or interference from an external source) introduce additional effects, such as small scale fading. Due to the relative motion between mobile nodes, each multipath wave experiences an apparent shift in frequency (i.e., the Doppler shift), directly proportional to the direction of arrival of the received multipath wave, and to the velocity/direction of motion of the mobile [135]. Due to environment changes (i.e., objects in the radio channel are in motion), a time varying Doppler shift is induced on multipath components. Consequently, in such environments affected by small scale fading, it is challenging to use simple connectivity (which itself can vary dramatically [136]) or Received Signal Strength (RSS) for accurate localization.

Fuzzy logic offers an inexpensive and robust way to deal with highly complex and variable models of noisy, uncertain environments. It provides a mechanism to learn about an environment in a way that treats variability consistently. In one well-established fuzzy system, the Sendai railroad [137], fuzzy logic allowed the integration of noisy data related to rail conditions, train weight, and weather into acceleration

and braking algorithms. Fuzzy logic can similarly be applied to localization. Empirical measurements are made between participating anchors in predictable encounters. These measurements are analyzed to produce rules that are used by the fuzzy inference systems, which interpret RSS input from unlocalized nodes and other anchors. The output of this process recovers the actual distance, compensated for variability in the local environment. This basic technique is employed in two constituent subsystems of FuzLoc - the Fuzzy Multilateration System (FMS) and the Fuzzy Grid Prediction System (FGPS).

9.2 Motivation

FuzLoc is motivated by our interest in a localization technique for a mobile sensor network, deployed in a harsh environment and a set of interesting/surprising results obtained from simulations of two state of the art localization techniques for mobile sensor networks, namely MCL [138] and MSL [139]. We define a harsh environment as one in which the distance between sender and receiver cannot be accurately determined from the RSS alone, due to environmental phenomena such as multipath propagation and interference.

For more complete *problem formulation* we mention that the aforementioned localization techniques assume that given a set of mobile sensor nodes, a subset of nodes, called anchors, know their location in a 2-dimensional plane. Also, nodes and anchors move randomly in the deployment area. Maximum velocity of a node is bounded but the actual velocity is unknown to nodes or anchors. Nodes do not have any knowledge of the mobility model. Anchors periodically broadcast their locations. All nodes are deployed in a noisy, harsh environment and they do not have any additional sensors except their radios. MCL gathers samples using Monte Carlo methods and filters them using a particle filter, with the criteria being that

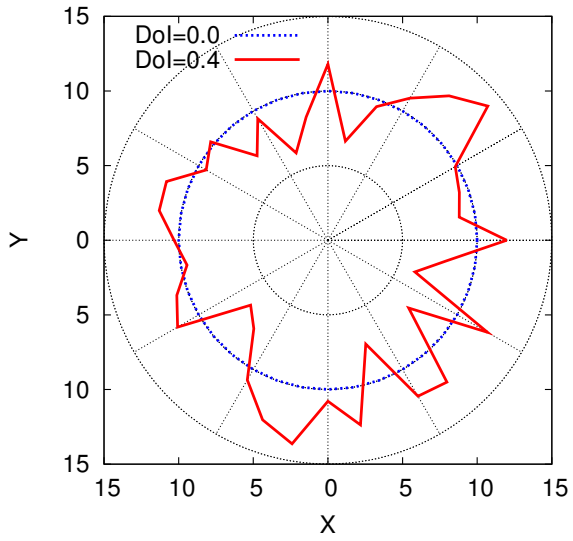


Figure 9.1: Illustration of radio patterns for two different degrees of radio irregularity (DoI)

each sample should be within range of a 1 hop anchor (with respect to itself) while at the same time, not being in range of a 2-hop anchor. Samples are assigned weights over successive iterations. MSL improves upon MCL by using criteria involving all neighbors and not just anchors. MSL is also adaptable to static scenarios if the nodes are allowed to exchange their samples and weights.

Using simulators developed by the authors of [138] [139], we developed a scenario with highly irregular radio ranges, typical of harsh indoor or extremely obstructed outdoor environments. The irregularity in the radio range is modeled in these simulators as a degree of irregularity (DoI) parameter [138]. The DoI represents the maximum radio range variation per unit degree change in direction. An example, depicted in Figure 9.1, when DoI=0.4 the actual communication range is randomly chosen from [0.6r, 1.4r].

Simulation results, for a network of 320 nodes, 32 anchors deployed in a 500×500

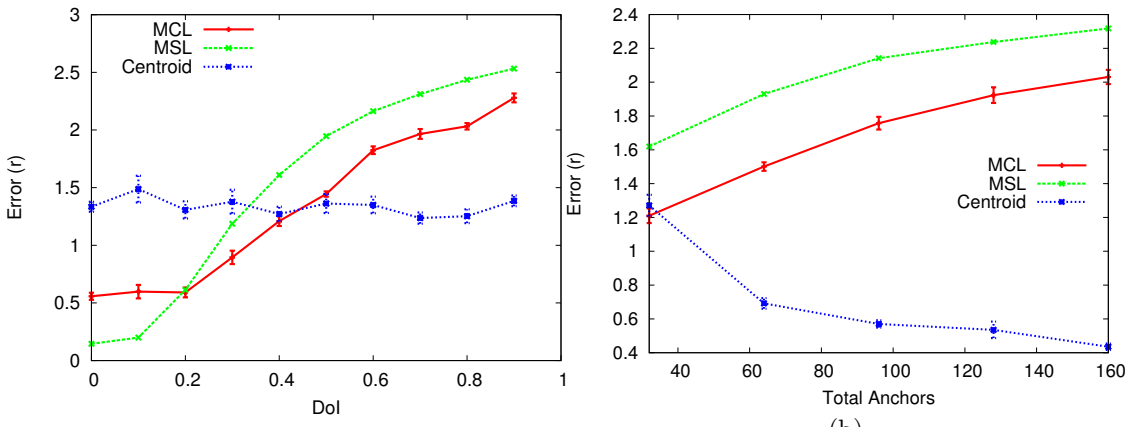


Figure 9.2: (a) The effect of DoI on localization error in MSL, MCL and Centroid; and (b) the effect of anchor density on localization error, at DoI=0.4, for MSL, MCL and Centroid.

grid and moving at 0.2r (r, the radio range) are shown in Figures 9.2a and 9.2b. Figure 9.2a demonstrates that the DoI parameter has a significant negative effect on the localization accuracy. At DoI=0, MCL and MSL achieve localization errors of 0.2r and 0.5r. With an increase in the DoI to 0.4, their localization error increases 400%. More surprisingly, as depicted in Figure 9.2b, at a high DoI value, an increase in the number of anchors has a detrimental effect on localization accuracy. This result is counter-intuitive since access to more anchors implies that nodes have more opportunities to receive accurate location information, as exemplified by the performance of Centroid (which computes the location as the average of the coordinates of all anchors in its vicinity), in the same figure. A similar observation is made in [133] although no further study was performed. Our results and also those of [140, 141] suggest that large errors are detrimental to the Monte Carlo method since the samples get successively polluted with time. In [141], a proposed Mixture-MCL method uses odometry to gather samples and then uses sensor data to assign weights, en-

abling it to recover quickly from such errors, while [140] does the same based on error correction based on learnt paths and topological constraints. In the specific case of MCL, the nodes used for filtering the samples may not be actual neighbors because of the non-uniformity in the radio range varies in every direction. The number of polluted samples increases with increasing anchor density. Simply increasing the size of the particle filter in MCL (to 1000 from the current value of 50) does not improve the accuracy significantly, as can be seen in Fig. 10 of [138].

9.3 A Fuzzy Logic-Based Node Localization Framework

The challenges identified above were partially addressed in recent work in sensor network node localization [142, 96]. The authors create hybrid localization mechanisms that make use of range-based localization primitives (e.g., RSSI) to validate and improve the accuracy of range-free techniques.

In a similar vein, we propose to formulate the localization problem as a fuzzy inference problem by using RSSI to obtain distance, in a fuzzy logic-based localization system where the concept of distance is very loose, such as "High", "Medium" or "Low". The core intuition is that accurate ranges can be determined by learning about the local RF environment and developing rules based on this knowledge. Fuzzy logic provides a simple and computationally inexpensive way to accomplish this learning. In other, similarly dynamic scenarios like rail transportation [137] and photovoltaic power generation [143], fuzzy logic provides mechanisms that allow simple systems to smartly adapt to rapidly changing environments.

In our proposed fuzzy logic-based localization system, distances between a mobile sensor node and anchor nodes are fuzzified, and used, subsequently in a Fuzzy Multilateration procedure to obtain a fuzzy location. In case two or more anchors are not available for performing localization using fuzzy multilateration, the sensor node

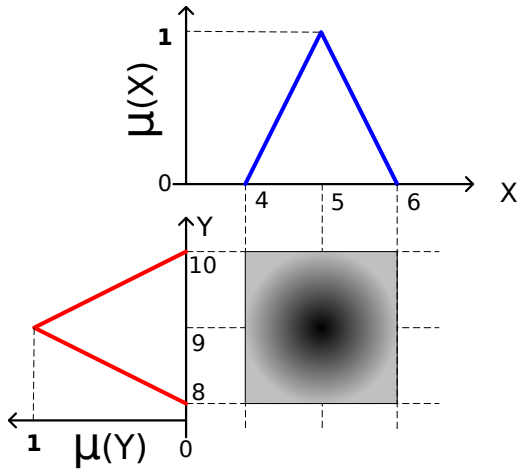


Figure 9.3: Representation of a fuzzy location, using two triangular membership functions.

employs a new technique, called fuzzy grid prediction, to obtain a location, albeit imprecise. In the Fuzzy Grid Prediction method, the node uses ranging information from any available anchor to compute distances to several fictitious "virtual anchors" which are assumed to be located in predetermined grids or quadrants. This allows the node to locate the grid/quadrant in which it is present.

In conventional localization schemes, the location of a node is typically represented by two coordinates that uniquely identify a single point within some two-dimensional area. Localization using fuzzy coordinates follows a similar convention. The two dimensional location of a node is represented as a pair (X, Y), where both X and Y are fuzzy numbers and explained below. However, instead of a single point, the fuzzy location represents an area where the probability of finding the node is highest, as depicted in Figure 9.3. This section develops the theoretical foundation behind the computation of this fuzzy location, using imprecise and noisy RSSI measurements.

9.3.1 Fuzzy Logic Preliminaries

Fuzzy logic revisits classical set theory and modifies it to have non-rigid, or fuzzy, set boundaries. Where classical set theory is concerned with collections of discrete objects, a *fuzzy set*, sometimes called a *fuzzy bin*, is defined by an associated *membership function* μ, which describes the degree of membership $0 \leq \mu(x) \leq 1$ of a *crisp (regular) number* x in the fuzzy set. The process of calculating the membership of a crisp number for many fuzzy sets is called the *fuzzification process*.

A *fuzzy number* is a special fuzzy bin where the membership is 1 at one and only one point. A fuzzy number represents a multi-valued, imprecise quantity unlike a single valued traditional number. One popular $\mu(x)$ function, is the *triangular membership function*:

$$\mu(x) = \begin{cases} 0 & \text{if } x < a \\ (x-a)/(b-a) & \text{if } a \leq x \leq b \\ (c-x)/(c-b) & \text{if } b \leq x \leq c \\ 0 & \text{if } x > c \end{cases} \tag{9.1}$$

where (a, b, c) defines a triangular bin. We chose a triangular membership function because, in addition to being a good substitute for the more widely used Gaussian function, it has linear components only and computing membership is less resource intensive, suitable for our resource constrained sensor nodes. Since not all triangular memberships are symmetric, we use the triangular function in its most general form.

As shown in Figure 9.4, the $WEAK$ fuzzy set can be represented as (-90, -70, -50) and $MEDIUM$ as (-70, -50, -30). A crisp number, RSSI = -55dBm has a membership of 0.25 in $WEAK$ and 0.75 in $MEDIUM$.

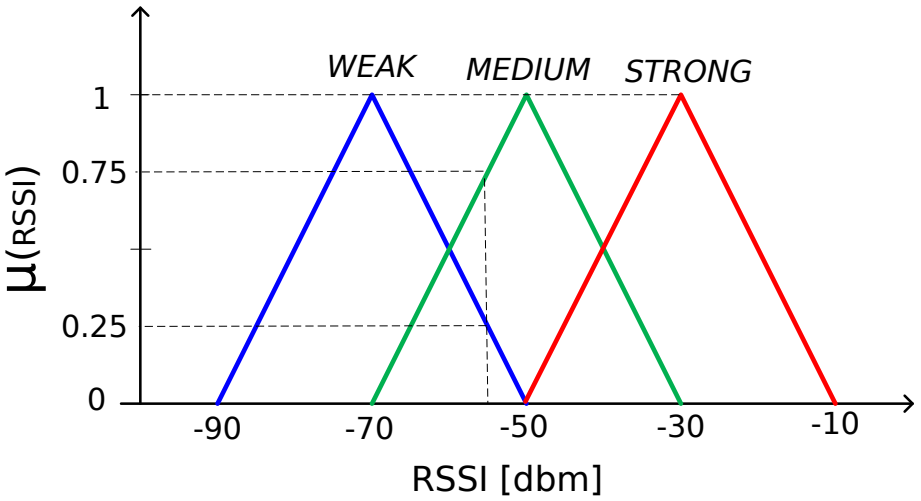


Figure 9.4: Fuzzification of a crisp value of -55dBm into fuzzy bins $WEAK$, $MEDIUM$ and $STRONG$ having triangular membership functions: $WEAK$, $MEDIUM$ and $STRONG$ with degrees of membership of 0.25, 0.75 and 0.0 respectively.

A *fuzzy system* translates a crisp input into a fuzzy output using a set of *fuzzy rules* which relate input and output variables in the form of an IF-THEN clause. Typically the IF clause contains the input linguistic variable (e.g., RSSI) and the THEN clause contains the output linguistic variable (e.g., DISTANCE). An example rule is:

IF RSSI is *WEAK* **THEN** DISTANCE is *LARGE*

9.3.2 Fuzzy Multilateration

As shown in Figure 9.5, consider a node S that wants to be localized, in the vicinity of three anchor nodes A_j ($j = \overline{1,3}$). Each anchor node is equipped with a set of fuzzy rules that map fuzzy RSSI values to fuzzy distance values:

Rule i: **IF** RSSI is $RSSI_i$ **THEN** DIST is $Dist_i$

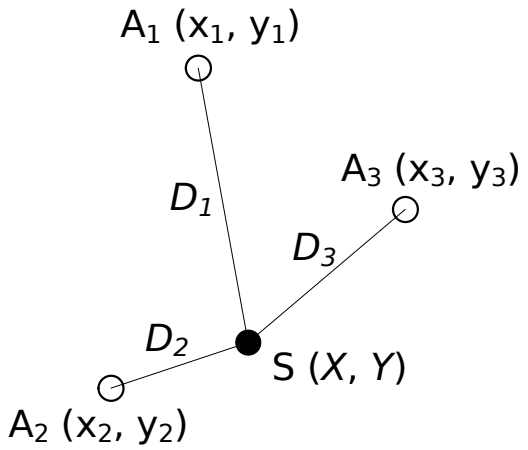


Figure 9.5: A sensor node S with fuzzy coordinates X and Y, to be located using three anchors at (x_1, y_1), (x_2, y_2) and (x_3, y_3).

where $RSSI_i$ and $Dist_i$ are fuzzy linguistic variables (e.g. $WEAK$, $MEDIUM$, $HIGH$).

A fuzzy rule is created when two anchors can communicate directly. Since anchors know their locations, they can find the distance between themselves and also measure the RSSI. The anchors then fuzzify the crisp RSSI and distance values into two fuzzy bins $RSSI_i$ and $Dist_i$ respectively, through the process of fuzzification. The chosen fuzzy bin is the one in which the crisp value will have the highest membership value.

For a more general case, when the node S is within radio range of n anchors, the node localization problem can be formulated as a fuzzy multilateration problem. The following:

$$\begin{aligned} F_1 &= (X - x_1)^2 + (Y - y_1)^2 - D_1^2 = 0 \\ F_2 &= (X - x_2)^2 + (Y - y_2)^2 - D_2^2 = 0 \\ &\dots \\ F_n &= (X - x_n)^2 + (Y - y_n)^2 - D_n^2 = 0 \end{aligned} \tag{9.2}$$

defines a non-linear system of equations describing the relation between the locations of the nodes and anchors and the distances among them. The variables X, Y and D_k ($k = \overline{1,n}$) are fuzzy numbers representing the location of the node and the distance to anchors respectively, while (x_k, y_k) ($k = \overline{1,n}$) are crisp numbers representing the crisp location of the anchors. The objective is to minimize the mean square error over all equations.

9.3.3 Fuzzy Inference

A definition of the process of obtaining the fuzzy distance D_k between node and anchor is needed before solving the system of equations. This process, called fuzzy inference, transforms a crisp RSSI value obtained from a packet sent by a node and received by an anchor into a fuzzy number D_k. Figure 9.6 depicts an example for the fuzzy inference process. As shown, an RSSI value of -62dBm has different membership values $\mu(RSSI)$ for the fuzzy bins $WEAK$ and $MEDIUM$. The two fuzzy bins, in this example, are mapped by a fuzzy rule base formed by two fuzzy rules:

Rule i: **IF** RSSI is $MEDIUM$ **THEN** DIST is $MEDIUM$

Rule j: **IF** RSSI is $WEAK$ **THEN** DIST is $LARGE$

These two fuzzy rules define the mapping from the RSSI fuzzy sets to the DIST fuzzy sets. As shown in Figure 9.6, the two fuzzy rules indicate the membership $\mu(DIST)$ in the DIST domain. P_i and P_j indicate the center of gravity of the trapezoid formed by the mapping of the RSSI into fuzzy bins $MEDIUM$ and $LARGE$, respectively.

Typically, a single RSSI value triggers multiple fuzzy rules (the membership value of the crisp value in the input bin of the fuzzy rule is non-zero), resulting in multiple distance bins. Assume that the fuzzy rule base maps an RSSI value to a set of

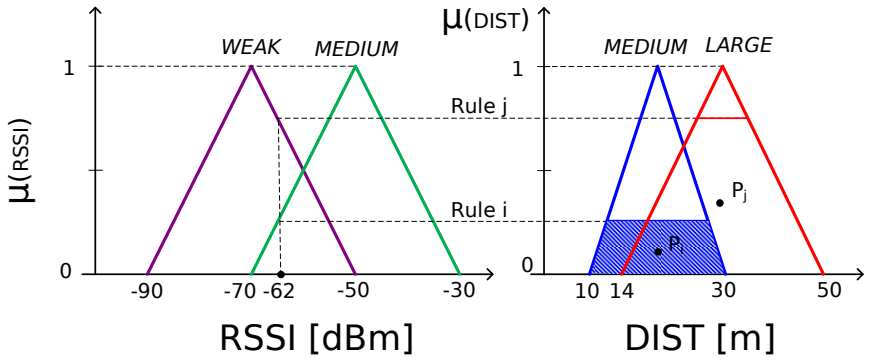


Figure 9.6: The fuzzy inference process for an input RSSI value of -62dBm. In this example, the fuzzy rule base maps this value through two rules: "Rule i" and "Rule j". The dotted lines represent fuzzy inference: finding the membership (vertical line on left), applying the same membership to the output bin (horizontal line towards right) and defuzzification (the lines intersect the triangles to form a trapezoid).

m fuzzy Dist bins. The set of centers of gravity P_l ($l = \overline{1,m}$) is denoted by $P = \{P_1, P_2 \ldots P_m\}$. The output fuzzy number D_k is calculated as follows: First, calculate the centroid of all points in P - call it P_c. Next, take the centroid of all points in P whose abscissa is less than that of P_c i.e., $L = \{P_n | x(P_n) \leq x(P_c)\}$. Similarly, $G = \{P_n | x(P_n) \geq x(P_c)\}$ is the set of points whose abscissa is greater than that of P. The abscissae of three points P, L and G represent the resulting fuzzy distance D_k, formally described as (subscript x denotes abscissa):

$$D_k = (a, b, c) = \left(\left(\frac{\sum L_n}{|L|} \right)_x, (P_c)_x, \left(\frac{\sum G_n}{|G|} \right)_x \right) \tag{9.3}$$

This definition of obtaining a fuzzy number through fuzzy inference produces a fuzzy number while giving more "weight" to the centroid by eliminating some possibilities at the edge. To truly represent the result one would need to compute a smooth and continuous function like the Gaussian membership function, but the triangular approximation has the advantage of reduced computation complexity.

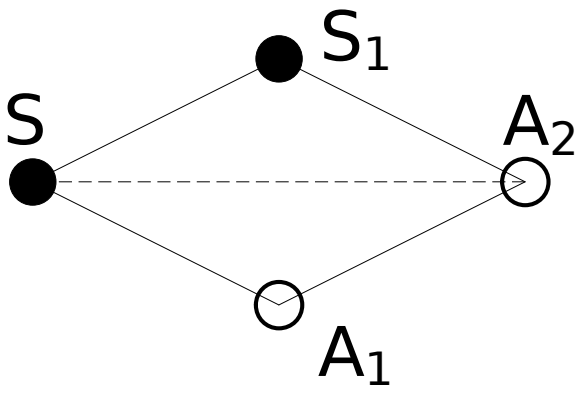


Figure 9.7: Illustrating the multi-hop case for fuzzy multilateration: a node S localizes itself using A_2 and A_1.

Equation 9.3 limits its analysis to situations where the anchors and the node desiring localization are one hop from each other. This constraint limits the degree of accuracy that can be achieved. Two hops provide a good trade off between messaging overheads and accuracy as explained later. Consider an anchor A_2 (Figure 9.7) which is 2 hops away from a node S. Suppose that a regular node S_1 and an anchor A_1 are neighbors of both S and A_2. The aim is now to find the distance $\overline{D_{SA_2}}$. In a 2-dimensional space, a straight line between two points is also the shortest possible; hence a good approximation is the minimum of all known distances between the 2 points. Applying this fact, we can now calculate:

$$\overline{D_{SA_2}} = \min(\overline{D_{SS_1}} + \overline{D_{S_1A_2}}, \overline{D_{SA_1}} + \overline{D_{A_1A_2}}) \tag{9.4}$$

The distances in Equation 9.4 are fuzzy values, as the result of defuzzification by either A_1 or A_2 depending on the sender. Addition of two triangular fuzzy numbers (a, b, c) and (d, e, f) is well known in fuzzy logic theory [144] to be the sum of their

individual parameters:

$$(a, b, c) + (d, e, f) = (a + d, b + e, c + f)$$

The smallest fuzzy number, to be computed in Equation 9.4 is simply the fuzzy number with the lowest center value [144]:

$$\min((a, b, c), (d, e, f)) = \begin{cases} (a, b, c) & \text{if } \min(b, e) = b \\ (d, e, f) & \text{if } \min(b, e) = e \end{cases} \tag{9.5}$$

The minimum of many fuzzy numbers can be recursively computed in the case of multi hop multilateration; it is beyond the scope of this section. In order to solve the non-linear system of Equations 9.2, in two fuzzy variables, the fuzzy variant of the iterative classical Newton method based on the Jacobian matrix [145] is used. To accomplish this, the fuzzy numbers are expressed in their parametric form $X = (\underline{X}, \overline{X})$ where $\underline{X}$ and $\overline{X}$ are continuous bounded non-decreasing and non-increasing, respectively, functions. These functions effectively represent the "left half" and "right half" of the membership function.

For a triangular membership function, such as defined in Equation 9.1, a parametric representation in $r \in [0, 1]$ is:

$$X = (a + (b - a)r, c - (c - b)r) \tag{9.6}$$

The system of Equations 9.2 is, therefore, represented in the parametric form. Without loss of generality, assume that X and Y are positive. Then, the system can

be split into:

$$\begin{aligned}
\underline{F_1} &= (\underline{X} - x_1)^2 + (\underline{Y} - y_1)^2 - \underline{D_1}^2 = 0 \\
\underline{F_2} &= (\underline{X} - x_2)^2 + (\underline{Y} - y_2)^2 - \underline{D_2}^2 = 0 \\
&\dots \\
\underline{F_n} &= (\underline{X} - x_n)^2 + (\underline{Y} - y_n)^2 - \underline{D_n}^2 = 0
\end{aligned} \tag{9.7}$$

and

$$\begin{aligned}
\overline{F_1} &= (\overline{X} - x_1)^2 + (\overline{Y} - y_1)^2 - \overline{D_1}^2 = 0 \\
\overline{F_2} &= (\overline{X} - x_2)^2 + (\overline{Y} - y_2)^2 - \overline{D_2}^2 = 0 \\
&\dots \\
\overline{F_n} &= (\overline{X} - x_n)^2 + (\overline{Y} - y_n)^2 - \overline{D_n}^2 = 0
\end{aligned} \tag{9.8}$$

The Jacobian J is constructed as:

$$J = \begin{bmatrix}
\underline{F_1}_{\underline{X}} & \underline{F_1}_{\overline{X}} & \underline{F_1}_{\underline{Y}} & \underline{F_1}_{\overline{Y}} \\
\overline{F_1}_{\underline{X}} & \overline{F_1}_{\overline{X}} & \overline{F_1}_{\underline{Y}} & \overline{F_1}_{\overline{Y}} \\
\underline{F_2}_{\underline{X}} & \underline{F_2}_{\overline{X}} & \underline{F_2}_{\underline{Y}} & \underline{F_2}_{\overline{Y}} \\
\overline{F_2}_{\underline{X}} & \overline{F_2}_{\overline{X}} & \overline{F_2}_{\underline{Y}} & \overline{F_2}_{\overline{Y}} \\
\dots & \dots & \dots & \dots \\
\underline{F_n}_{\underline{X}} & \underline{F_n}_{\overline{X}} & \underline{F_n}_{\underline{Y}} & \underline{F_n}_{\overline{Y}} \\
\overline{F_n}_{\underline{X}} & \overline{F_n}_{\overline{X}} & \overline{F_n}_{\underline{Y}} & \overline{F_n}_{\overline{Y}}
\end{bmatrix} \tag{9.9}$$

Simplifying,

$$J = \begin{bmatrix} 2(\underline{X} - x_1) & 0 & 2(\underline{Y} - y_1) & 0 \\ 0 & 2(\overline{X} - x_1) & 0 & 2(\overline{Y} - y_1) \\ 2(\underline{X} - x_2) & 0 & 2(\underline{Y} - y_2) & 0 \\ 0 & 2(\overline{X} - x_2) & 0 & 2(\overline{Y} - y_2) \\ \dots & \dots & \dots & \dots \\ 2(\underline{X} - x_n) & 0 & 2(\underline{Y} - y_n) & 0 \\ 0 & 2(\overline{X} - x_n) & 0 & 2(\overline{Y} - y_n) \end{bmatrix} \tag{9.10}$$

Initial guesses of X and Y can be updated as follows. For every iteration compute a matrix Δ:

$$\Delta = \begin{bmatrix} \underline{h}(r) & \overline{h}(r) & \underline{k}(r) & \overline{k}(r) \end{bmatrix}^T \tag{9.11}$$

where $\underline{h}$, $\overline{h}$, $\underline{k}$ and $\overline{k}$ are defined as incremental updates to the initial guess:

$$\begin{aligned} \underline{X}(r) &= \underline{X}(r) + \underline{h}(r) \\ \overline{X}(r) &= \overline{X}(r) + \overline{h}(r) \\ \underline{Y}(r) &= \underline{Y}(r) + \underline{k}(r) \\ \overline{Y}(r) &= \overline{Y}(r) + \overline{k}(r) \end{aligned} \tag{9.12}$$

The set of equations evaluated at the initial guess is:

$$F = \begin{bmatrix} \underline{F_1} & \overline{F_1} & \dots & \underline{F_n} & \overline{F_n} \end{bmatrix}^T \tag{9.13}$$

The equation that connects them is $\Delta = -J^{-1}F$. The initial guess (X_0, Y_0) is computed from the average of the coordinates of the anchors. Then, J and F are computed for this initial guess. The incremental update Δ is calculated and applied

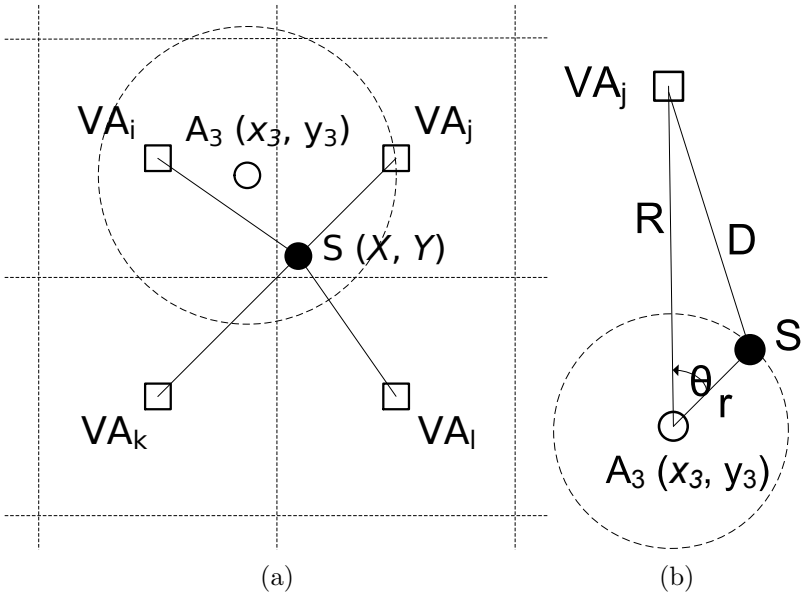


Figure 9.8: (a) A sensor S and the grid cells in its vicinity, is within radio range of anchor A_3; and (b) average distance between sensor S and virtual anchor VA_j.

to X and Y. J and F are computed for the new values and the process is repeated until Δ converges to 0 within ϵ.

9.3.4 Fuzzy Grid Prediction

The multilateration technique presented in the previous section assumes the presence of a sufficient number of anchors, typically three or more. However, in mobile sensor networks with low anchor densities, it might frequently be the case that a node does not have enough anchors for multilateration. To address this problem we extend our fuzzy logic-based localization framework to predict an area, e.g., a cell in a grid, where the node might be. The idea is inspired from cellular systems [146]. We propose to virtualize the anchors, so that a node is within a set of Virtual Anchors at any point in time. A Virtual Anchor is a fictitious anchor which is assumed to located at a known, fixed location in the field of deployment, the distance to which

can be found in an approximate way from the node. In FuzLoc, we place virtual anchors at the center of every square cell that the field is divided into, as described below. The key idea is that the nearer a node is to a virtual anchor, the more likely it is that the node can be found in that cell.

Consider the area in which the network is deployed to be subdivided into a grid of G cells, as depicted in Figure 9.8a. Denote the probability that a node S is in a cell j ($j = 1 \ldots G$) by p_j. To infer these probabilities, we construct a fuzzy system, whose input is the distance d_j between S and the center of cell j, and the output is a scalar $0 < p_j < 1$ for each j. A rule in our fuzzy system is as follows:

$$\text{Rule } i\text{: } \textbf{IF } (\text{DIST}_{grd1} \text{ is } D_{i1}) \text{ and } \ldots \text{ and } (\text{DIST}_{grdG} \text{ is } D_{iG}) \textbf{ THEN } (PROB_{grd1} \text{ is } P_{i1}) \text{ and } \ldots \text{ and } (PROB_{grdG} \text{ is } P_{iG})$$

where D_{ij} is the fuzzy bin representing the distance between the node and the center of cell j, and P_{ij} is the fuzzy bin representing the probability that node S is in cell j.

For each rule i, we calculate p_j by first fuzzifying d_j, applying it to the rule, and then defuzzifying the aggregate, as we described in Section 9.3.3. Once the most probable cell is found, the location of the node can be computed as the intersection between this cell and a circle with a radius of r around the anchor.

It is paramount to remark that we can obtain p_j only if the node S has at least one anchor in its vicinity, i.e., we can estimate D_{ij}. The technique we propose for estimating D_{ij} is described in Section 9.3.4.1.

Before proceeding with the description of how we compute D_{ij}, we describe how to update p_j when no anchor is in the vicinity of node S. Since there is a high correlation between the current and previous cell a node is in, we construct a Recursive Least Squares (RLS) filter which predicts the cell in which the node S might be. For each

cell j, we store a buffer $x_j(k) = [p_j(k)\ p_j(k-1)\ \ldots\ p_j(k-m)]^T$ of m previous samples. We then define an RLS filter, updated whenever a new sample $p(k+1)$ is available, as:

$$x_j(k+1) = w_j^T(k)x_j(k) \tag{9.14}$$

where $w_j(k) = w_j(k-1) + a_j(k)g_j(k)$ is a vector of coefficients, computed as follows:

$$\begin{aligned} a_j(k) &= x_j(k) - w_j^T(k-1)x_j(k) \\ g_j(k) &= P_j(k-1)x_j(k)\{\lambda + x_j^T(k)P_j(k-1)x_j^T(k)\}^{-1} \\ P_j(k) &= \lambda^{-1}P_j(k-1) - g_j(k)x_j^T(k)\lambda^{-1}P_j(k-1) \end{aligned} \tag{9.15}$$

where $0 \leq \lambda \leq 1$ is the forgetfulness factor, a design parameter. $P_j(0)$ is initialized to δI_j, where I is the identity matrix of size $(m+1) \times (m+1)$ and δ is a typically large value.

9.3.4.1 Calculation of D_{ij}

The fuzzy system requires that we calculate the distance from the node to the virtual anchor. We have to find the average distance instead, because we do not know the node's location. These average distances can be calculated only when at least one anchor is in the node's vicinity.

Consider a node and a sole anchor A_3 which is its neighbor, as illustrated in Figure 9.8b. Take the set of all virtual anchors and discard the ones which are at a distance of more than $2R$ from the anchor where R is the radio range of the anchor, since this is the most distant virtual anchor the node can hear in the limiting case where the node is between the anchor and the virtual anchor. To calculate the average distance $\overline{D}$ from the node to a virtual anchor VA_j in cell j, we estimate the average distance from VA_j to all points on the circumference of a hypothetical circle

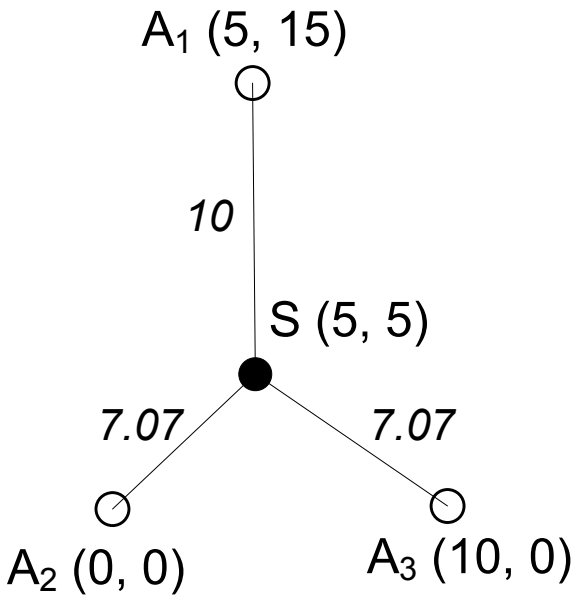


Figure 9.9: Three anchors and a node used in the numerical example.

around A_3. The radius r of this circle is the defuzzified distance obtained from the fuzzy multilateration system. If the distance between A_3 and VA_j is R, the average distance $\overline{D}$ can be calculated as follows:

$$\overline{D} = \frac{1}{2\pi}\int_0^{2\pi} \sqrt{(R\ -\ r\ \cos\theta)^2 + (r\ \sin\theta)^2}\ d\theta \tag{9.16}$$

$$= \frac{(R-r)}{\pi} E\big[\pi|\frac{-4Rr}{(R-r)^2}\big] \tag{9.17}$$

where $E[x|m]$ is the incomplete elliptic integral of the second kind [147].

This distance $\overline{D}$ to a virtual anchor in cell k for node S is nothing but d_j. When calculated for all j, it becomes the input to the fuzzy system. The output will be a set of probabilities p_j pertaining to each cell. The center of gravity of the lamina defined by the intersection of a circle around the anchor of radius r with the most likely cell is calculated, as explained above. Thus, a location is obtained.

9.3.5 Fuzzy Multilateration Numerical Example

A numerical example elaborates the concept of fuzzy multilateration. Consider three anchors forming an isosceles triangle (Figure 9.9) - $(0,0)$, $(10,0)$ and $(5,15)$. Let the node to be localized be at the centroid which is $(5,5)$. The actual distances to the anchors would be $5\sqrt{2}, 5\sqrt{2}, 10$ respectively. Assume that because of defuzzification, the node calculates the fuzzy distances as $(6,7,8),(6,7,8),(9,10,11)$ respectively. Then the system of equations can be expressed as:

$$\begin{aligned} F_1 &= (X-0)^2 + (Y-0)^2 - (6,7,8)^2 &&= 0 \\ F_2 &= (X-10)^2 + (Y-0)^2 - (6,7,8)^2 &&= 0 \\ F_3 &= (X-5)^2 + (Y-15)^2 - (9,10,11)^2 &&= 0 \end{aligned} \tag{9.18}$$

The actual fuzzy location (X,Y) of the node is $(5,5,5),(5,5,5)$. Assume the initial guess of (X,Y) to be $(5,6,7),(5,6,7)$ for the purposes of demonstration, which by the process of convergence should ideally yield the actual location. The parametric form would then be $(5+r, 7-r)$. Expanding only F_1 for brevity,

$$\begin{aligned} \underline{F_1} &= (\underline{X}-0)^2 + (\underline{Y}-0)^2 - \underline{(6,7,8)}^2 \\ \overline{F_1} &= (\overline{X}-0)^2 + (\overline{Y}-15)^2 - \overline{(6,7,8)}^2 \end{aligned} \tag{9.19}$$

which can then be simplified to

$$\begin{aligned} \underline{F_1} &= (5+r-0)^2 + (5+r-0)^2 - (6+r)^2 \\ \overline{F_1} &= (7-r-0)^2 + (7-r-15)^2 - (8-r)^2 \end{aligned} \tag{9.20}$$

Similarly the Jacobian can now be constructed as (first 2 rows only):

$$J = \begin{bmatrix} 2(5+r) & 0 & 2(5+r) & 0 \\ 0 & 2(7-r) & 0 & 2(7-r) \end{bmatrix} \tag{9.21}$$

The pseudo-inverse of a matrix with symbolic elements is computationally expensive, especially for embedded sensor nodes. Instead of inverting J which contains a symbolic element r, two non-symbolic inverses can be computed (for $r = 0$ and $r = 1$) and the results combined. This alternate computation incurs a loss of accuracy because the solution will be a perfect triangular fuzzy number and not a fuzzy number with little variation. However, the accuracy lost is extremely small.

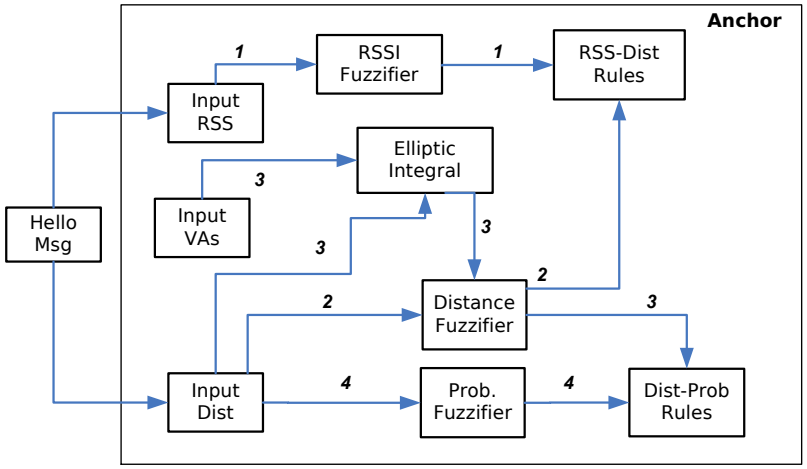
A simple substitution of $r = 0$ and $r = 1$ in J and F yields:

$$\Delta_0 = J_0^{-1} F_0 \text{ and } \Delta_1 = J_1^{-1} F_1 \tag{9.22}$$

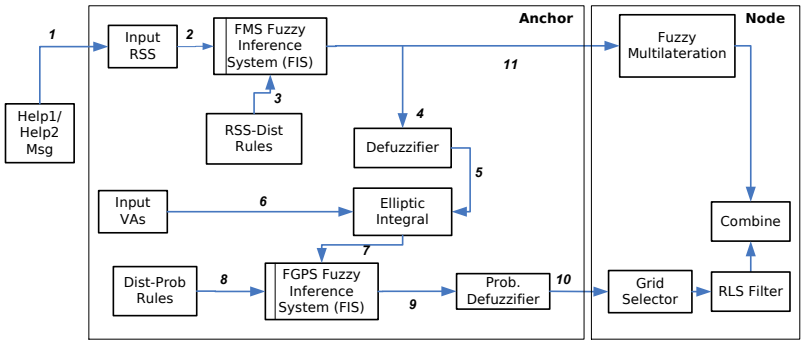
Given a solution (X, Y) expressed in simple form as (x_A, x_B, x_C) and (y_A, y_B, y_C), then:

$$\Delta_0 = \begin{bmatrix} \delta x_A \\ \delta x_C \\ \delta y_A \\ \delta y_C \end{bmatrix} \text{ and } \Delta_1 = \begin{bmatrix} \delta x_B \\ \delta x_B \\ \delta y_B \\ \delta y_B \end{bmatrix} \tag{9.23}$$

where δx_A is the incremental update to x_A. This is obvious since the left half of any fuzzy number in parametric form $(a + (b - a)r)$ evaluates to a when $r = 0$. The same argument holds for the right half. After this step, the new (x_A, x_B, x_C) and (y_A, y_B, y_C) are the input for the next iteration. The process is repeated until sufficient accuracy is obtained. Upon running our algorithm for 10 iterations with the above initial guess, the final location of the node was calculated as $(5.0, 5.0, 5.0)$,



(a)



(b)

Figure 9.10: The fuzzy logic based localization system design with the (a) training; and (b) localization phases.

$(4.72, 4.97, 5.14)$. The value that would have been used for comparison with algorithms that do not compute an area as a node's location will be the center values of the two fuzzy numbers, i.e, $(5, 4.97)$.

9.4 Localization System Design

The node localization system (called FuzLoc) that implements the proposed fuzzy logic-based localization framework is depicted in Figure 9.10. As shown, the localization system runs on both anchor and sensor nodes. The pseudocode for the localization protocol, as executed by anchors and sensor nodes, is shown in

Algorithm 8 and Algorithm 9, respectively.

Figure 9.10a depicts the training phase of Fuzloc while Figure 9.10b, the localization phase. Training happens with the participation of anchors only, while the localization phase involves both anchors and nodes. The components required for the fuzzy multilateration subsystem (FMS), as well as the fuzzy grid prediction subsystem (FGPS) are implemented on both anchors and nodes. The fuzzy rules required for these subsystems are created during the training phase. Anchors are assumed to have more computing power than ordinary nodes; they can then maintain these fuzzy rules.

The FuzLoc localization system uses two types of messages - a Hello-type message which anchors *use to train the localization system* (i.e., anchors broadcast their location and build rules), and a Help-type message which nodes *use for localization* (i.e., nodes notify 1-hop and 2-hop anchors and nodes that they need to localize).

The remaining part of this section describes the localization system training (and its use of Hello messages) and the localization protocol execution (and its use of Help-type messages).

9.4.1 Localization System Training

The training of the localization system takes place every time two anchors come within communication range with each other. The anchors know their locations, hence, an anchor can compute the distance between it and the other anchor. The key observation here is that since an anchor can also measure the RSSI of an incoming message, it can build the fuzzy rules required for both FMS and FGPS. Figure 9.10a depicts the training phase where a single Hello message is used to the build the rulesets for both FMS and FGPS.

9.4.1.1 FMS - Training

Anchors exchange HELLO messages (Algorithm 8, step 2). As shown in Figure 9.10a, the RSS of an incoming HELLO message ("Input RSS") is fuzzified by choosing the fuzzy set with the highest membership $\mu(RSSI)$ (Figure 9.10a, Path 1). The distance between anchors ("Input Dist") is fuzzified into a distance fuzzy set (Figure 9.10a, Path 2). The result of the training populates the rule base, i.e., "RSS-Dist Rules" (Figure 9.10a, and Algorithm 8, step 7).

9.4.1.2 FGPS - Training

When an anchor receives a HELLO message, it calculates the distances between the sender and each virtual anchor, using Equation 9.17 (Path 3). This calculation is shown in Algorithm 8, step 9. These distances are then fuzzified ("Distance Fuzzifier", Path 3). Additionally, the probabilities for the anchor being in each grid are updated, as shown in Algorithm 8, step 10. The probabilities are updated based on anchor's real movement, as presented in Section 9.3.4 (Figure 9.10a, Path 4). The computed probabilities are then fuzzified and used for populating (Algorithm 8, step 11) the rules set "Grid-Prob Rules" (Path 4).

9.4.2 Localization Protocol

The localization phase which runs on both anchors and nodes is shown in Figure 9.10b. In order to obtain its location, a node sends a HELP2 message (Algorithm 9, step 2). A HELP2 message is meant to trigger actions in nodes/anchors which are 1-hop away. These nodes/anchors perform some calculations (explained below), then rebroadcast a HELP1 message, meant to trigger actions in the 2-hop anchors.

When an anchor receives a HELP2 message (shown as "Help Msg" in Fig. 9.10b),

Algorithm 8 FuzLoc Protocol - Anchors

```
 1: [VA] ← FGPS.getVirtualAnchors                          ▷ VA of self
 2: BroadcastHello(VA)
 3: procedure RecvHello
 4:     rss ← Radio.getRSS()
 5:     loc ← Message.parseLocation()                      ▷ Loc of sender
 6:     dist ← Distance to sender
 7:     FMS.train(rss, dist)
 8:     [VA] ← Message.parseVA()                           ▷ VA of sender
 9:     [dist] ← Calculate distances to virtual anchors
10:     [prob] ← Calculate probabilities
11:     FGPS.train(dist, prob)
12: end procedure
13: procedure RecvHelp2                                   ▷ Intermediary anchor
14:     Check for cache
15:     rss ← Radio.getRSS()
16:     [ReplyMsg] ← BroadcastHelp1()                      ▷ Rebroadcast
17:     dist ← FMS.getDist(rss)
18:     [VA] ← FGPS.getVirtualAnchors
19:     [VA.dist] ← FPGS.getDists(VA)
20:     [VA.prob] ← FPGS.defuzzify(VA.dist)
21:     Radio.reply(VA.prob,dist,ReplyMsg)
22:     Cache result
23: end procedure
24: procedure ReceiveHelp1(rss1)                          ▷ 2 hop anchor
25:     rss2 ← Radio.getRSS()
26:     dist1 ← FMS.getDist(rss1)
27:     dist2 ← FMS.getDist(rss2)
28:     Radio.reply(dist1,dist2)
29: end procedure
```

it uses the RSSI of the packet in two ways. First, using Equation 9.3, the anchor computes the fuzzy distance ("FMS FIS") between itself and the node (Algorithm 8 step 17) (Paths 1, 2, 3, 11). This sequence of steps represents the anchor's implementation of the FMS subsystem. These fuzzy distances, from multiple anchors, are then used by the node to compute its location using the nonlinear system of equations ("Fuzzy Multilateration" box).

Secondly, the anchor defuzzifies the fuzzy distance (from Path 4) into a crisp

Algorithm 9 FuzLoc Protocol - Nodes

```
 1: procedure Localize
 2:     [info] ← BroadcastHelp2()                          ▷ 2 hop HELP
 3:     anchors ← Count(info)
 4:     if anchors = 0 then
 5:         [prob] ← Filter.predict()
 6:         grid ← Max(prob).index
 7:         loc ← center(grid)
 8:     else if anchors = 1 then                           ▷ FGPS
 9:         [prob] ← info[0].parseVAProb()
10:         TrainFilter(prob)
11:         grid ← Max(prob).index
12:         dist, center ← info[0].parseAnchorLoc()
13:         circle ← ConstructCircle(dist, center)
14:         loc ← SolveIntersection(grid,circle)
15:     else                                               ▷ FMS
16:         [dists] ← info.parseDistances()
17:         [centers] ← info.parseLocations()
18:         loc ← solveFMS(dists,centers)
19:     end if
20: end procedure
21: procedure ReceiveHelp2                                 ▷ Intermediary Node
22:     Check for cache
23:     rss ← Radio.getRSS()
24:     [ReplyMsg] ← BroadcastHelp1(rss)                   ▷ Rebroadcast
25:     Radio.reply(ReplyMsg)
26:     Cache result
27: end procedure
28: procedure ReceiveHelp1
29:     return                                             ▷ Only for anchors
30: end procedure
```

value by taking the center value of the fuzzy bin ("Defuzzifier", Path 4). This is needed since the elliptic integral method can handle crisp values only. Based on this crisp distance, the anchor calculates the distances between the node sending the Help message and (Section 9.3.4.1) its virtual anchors (step 19) using the incomplete elliptic integral method (Figure 9.10b, Paths 5, 6). This set of crisp distances serves as the input (Section 9.3.4) to the FGPS FIS (Algorithm 8 step 20) (Figure 9.10b,

Path 7). The FGPS FIS then computes a vector or grid probabilities using the Dist-Prob ruleset (Figure 9.10b, Paths 7, 8) obtained using FGPS training.

This vector of probabilities is then defuzzified to a crisp value (by defuzzifying the individual elements) and compiled into the reply message to be sent back to the node (Paths 10). In the reply to the HELP2 message, as mentioned above, the anchor also includes the fuzzy distance between it and the node (Figure 9.10b, Path 11). A vector containing the probabilities for the node being in each of the grids (Algorithm 8 step 21) (Figure 9.10b, Path 10), which is essentially the output of FGPS FIS and the reply for the HELP1 message that was broadcast.

The anchor then rebroadcasts a HELP1 message with an empty body (Algorithm 8 step 16). When an anchor receives a HELP1 message, it performs the same steps as before: it defuzzifies the RSSI of the received packet into a distance, and replies with the same (Algorithm 8 steps 25-28). A 2-hop anchor does not invoke its FGPS FIS. In case the HELP1 message contains an RSSI in the body (which happens when the intermediary node is a non-anchor), both the contained RSSI and the packet RSSI are defuzzified and included in the reply.

Once a node receives response(s) to its HELP2 message it decides to compute or predict its location (Algorithm 9 steps 4-18). If the node does not receive a response, it uses the RLS filter to predict the most probable grid it is in (Algorithm 9 steps 5-7). If the node receives a response from one anchor, it computes the center of gravity of the area obtained by intersection between: a) the grid with the maximum probability; and b) the circle with a center at the anchor location and with radius equal to the distance between the anchor and the node (Algorithm 9 steps 9-14). If the node receives two or more responses, it uses fuzzy multilateration to iteratively compute its location (Algorithm 9, steps 16-18).

A node can also be on the receiving end of a HELP2 message - when it is an

intermediary node. In this case, the node first detects the RSSI of the received message (Algorithm 9 step 23) and then packages into a HELP1 message and broadcasts it (Algorithm 9 step 24). Any response(s) to this message will be sent back to the sender as a reply to the original HELP2 message that was sent by the node intending to localize. A node ignores any HELP1 message it receives, since it is meant only for anchors (Algorithm 9 step 29).

9.5 Performance Evaluation With Two Hop Anchors

In this section, we first demonstrate that FUZLOC can be implemented and run on real mote hardware, then show FUZLOC's superior performance, when compared with state of art solutions like MCL [138], MSL [139] and Centroid [94]. Owing to the relatively few number of robots, the difficulty in implementing MCL and MSL on real hardware (please note that neither MSL, nor MCL have been implemented/evaluated on real hardware), controlling the anchor and seed density and the physical space constraints, we decided to compare performance of FUZLOC with state of art solutions, in simulations using both empirical and synthetic data.

In the remaining part of this section we present FUZLOC implementation on real-hardware, describe the empirical and synthetic RSSI-Distance mapping, and performance evaluation results.

9.5.1 System Implementation Validation

We implemented FUZLOC/FMS on EPIC motes running TinyOS 2.1.1. Since the matrices involved in FMS are not always square and hence they cannot be simply inverted, the fast and lightweight SVD based pseudo inverse method [147] was implemented on the motes. Relevant portions of the GNU Scientific Library (GSL) were ported to the MSP430 architecture in order to achieve this goal. The result was a fast method of inverting matrices, providing 4 digits of accuracy when compared

to a similar computation on a desktop PC. The 1,574 lines of code fit comfortably in 18,726B.

A Fuzzy Inference System (FIS) consisting of a triangular rule set and center-average defuzzification method was implemented in 19,932B of ROM (including the code required to send and receive messages in the radio) and 1,859B in RAM on EPIC motes running TinyOS 2.1.1. Whenever a packet was received on the onboard radio, the detected RSS was applied to the pre-built ruleset and then defuzzified into a fuzzy distance. The distance and RSSI binset consisted of 8 bins each. The defuzzified distance was equal to that produced by a similar computation on a desktop computer, within rounding errors. The execution time was less than 1 second. This proof of concept implementation of FuzLoc on motes demonstrates its feasibility of implementation on a mote.

9.5.2 Empirical and Synthetic RSSI-Distance Mapping

For our performance evaluation, we used RSSI-distance mappings obtained from a static sensor network, a small mobile sensor network and from a newly proposed DoI model. They are as follows.

9.5.2.1 Static Sensor Network

We used a static 42 node indoor testbed, in our lab. RSSI data was collected over 500 iterations with each node beaconing in each iteration. Since the nodes were static, inter-node distances could be calculated easily. This data was used to train and evaluate the FIS, as will be shown in Figure 9.18b. However, since only a finite number of unique distances are possible with a static testbed, we decided to use a small mobile testbed as well.

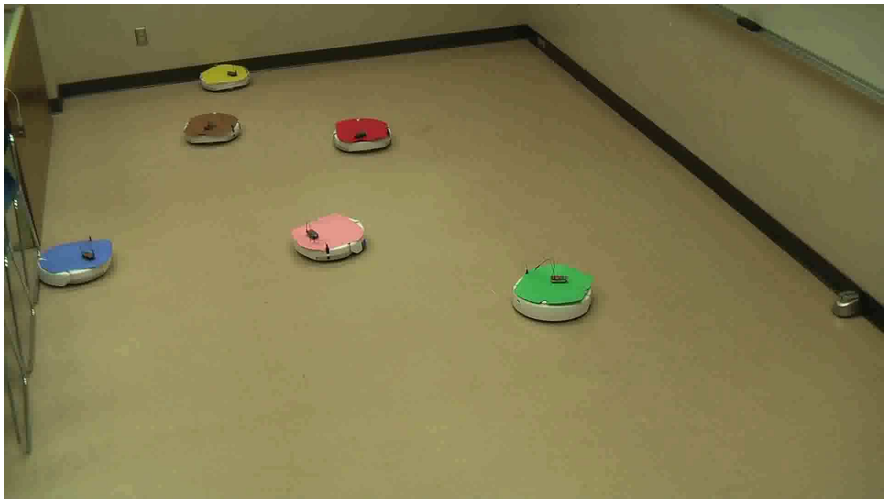


Figure 9.11: Experimental setup consisting of 6 iRobot Creates equipped with Epic motes.

9.5.2.2 Small Mobile Sensor Network

We collected data (RSSI-distance pairs) using a mobile testbed consisting of 6 "iRobot Creates" and EPIC motes (which interfaced using the serial bus) shown in Figure 9.11. Over a 125-iteration run, RSSI data was collected between pairs of neighbors at every iteration. In order to get the true locations of the robots (for calculating the distances between them) a digital video camera was used to film the entire experiment in 1080p HD. A small program was written in C and used OpenCV to infer the ground location of the robots using planar homography, since the camera was not in the same plane as the robots.

Each robot has a different color since this makes it easier to track them in the recorded video. Capturing the radio effects caused by mobility and the orientation of the antennae on the motes in real time was the main motivation behind the experiment. The ground locations of the robots at each step is then used to infer the actual distance between nodes for every measured RSS between nodes. These

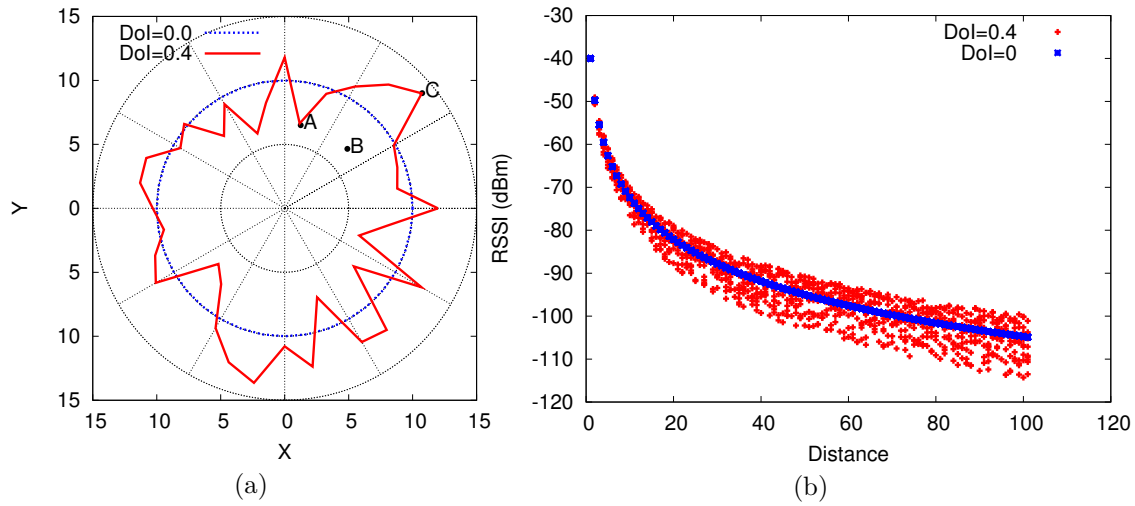


Figure 9.12: (a) The DoI model with three points of interest: although A and B are equally distant, their RSS values differ significantly in our EDoI model; and (b) RSSI vs. distance for the radio model used in the simulator, at DoI=0.4 and 0.

RSSI-distance pairs were then used to train and evaluate the FIS as will be shown in Figure 9.18c and described below.

9.5.2.3 EDoI Model

Since our fuzzy logic-based localization technique makes use of the RSSI, we extended the DoI model [95]. In order to adjust the simulated RSSI for both the actual radio range and log-normal fading, we developed the EDoI model. It combines the general log-normal fading model with the DoI model [95]. In Figure 9.12a, $\overline{OA}$ and $\overline{OC}$ are the radio ranges for the antenna situated at the origin O, in two different directions as evaluated by the DoI model. Assume that the receiver sensitivity is -94dBm i.e., if a transmitter with similar characteristics as the receiver is situated at A or C, then the RSSI at the origin will be -94dBm. To calculate the RSSI at a point B in the same direction as C where $\overline{OA} = \overline{OB}$, we apply a log-normal fading model with the reference distance as $\overline{OC}$, such that the RSSI at point C is -60dBm.

Note that the RSSI at A is -94dBm, whereas the RSSI at an equidistant point B in a different direction is -60dBm. On top of this, additive random noise (uniformly distributed, min = -20dB, max = 20dB) is applied to the calculated RSSI. This procedure is done every time a node uses this model to simulate an RSSI, ensuring randomness in both temporal and spatial randomness. Formally:

$$RSSI(d) = S_i \times \frac{\log_{10} d}{\log_{10}[r(1 + DoI \times rand())]} \tag{9.24}$$

where S_i is the receiver sensitivity, r is the ideal radio range, DoI is the radio degree of irregularity and $rand$ is a random number $\mathcal{U}[0, 1]$.

9.5.3 Simulation Setup

Through extensive simulations, we compare our solution with MCL [138], MSL [139] and Centroid [94], since we wanted to evaluate our solution against non-centralized solutions for non-static networks, both Monte Carlo based (MCL, MSL) and simple (Centroid). A theoretical "Perfect FuzLoc" method shows the theoretical optimum FuzLoc can reach, by simply bypassing the FIS and considering the actual distance between nodes. The problem of not having enough anchors in the vicinity of nodes causes non-zero error for Perfect FuzLoc. Data gathered from the the static and small mobile sensor network has been used to evaluate the FIS system. Thus, the FIS system is evaluated using simulated RSSI-Distance data as well as data from the two experiments described before.

We simulate a set (N) of 320 sensor nodes deployed in a 500×500 area. Of the 320 nodes deployed, 32 nodes are designated anchors (set S). The radio range (r) of a node is 50 and the default DoI is 0.4. We chose these simulation parameters for consistency with results reported in [138, 139]. The default receiver sensitivity (S_i) is -94dBm, and a plot depicting the predicted RSSI by our EDoI model, is shown

in Figure 9.12b. The default maximum node velocity is to 0.2r. This velocity has been reported in [138] to be optimal. We investigate the performance of all solutions for node velocities up to 0.5r. The node velocity is an important parameter since MCL and MSL use it as a filtering criterion in their particle filters. The default setup uses 10 fuzzy triangular bins and the defuzzification method is center-average. The fuzzy location is defuzzified into a crisp location by considering only the center values of the abscissa and the ordinate. The fuzzy bins for distance and RSSI are uniformly distributed between $(0, r)$ and $(-40, -100)$ respectively, with the width of each bin being twice the separation between peaks of two adjacent fuzzy bins. With 10 distance and RSS bins, there are 100 different combinations that can be seen in a RSS-Dist ruleset. Rules encountered more frequently tend to affect the output more than infrequent ones because the defuzzification method involves centroids corresponding to the output bin of each rule. A sample set of fuzzy RSS-Dist rules has been provided in the Supplemental Material, Section 3.

9.5.4 Radio Irregularity

We performed simulations for different DoI values with all other parameters kept constant. Figure 9.13 depicts our results, indicating the deterioration in localization accuracy of MCL, MSL and Centroid. The effect of compounded errors due to polluted samples has been investigated as the "kidnapped robot problem" [140] in robot localization. The kidnapped robot test verifies whether the localization algorithm is able to recover from localization failures, as signified by the sudden change in location due to "kidnapping". It has been shown [140] that such uncorrected algorithms collapse when the observed sample is far from the estimated sample. MSL demonstrates an even more pronounced effect, since it also uses non-anchor neighbors for filtering, thus leading to more pollution. Both FuzLoc and Perfect FuzLoc are unaffected by

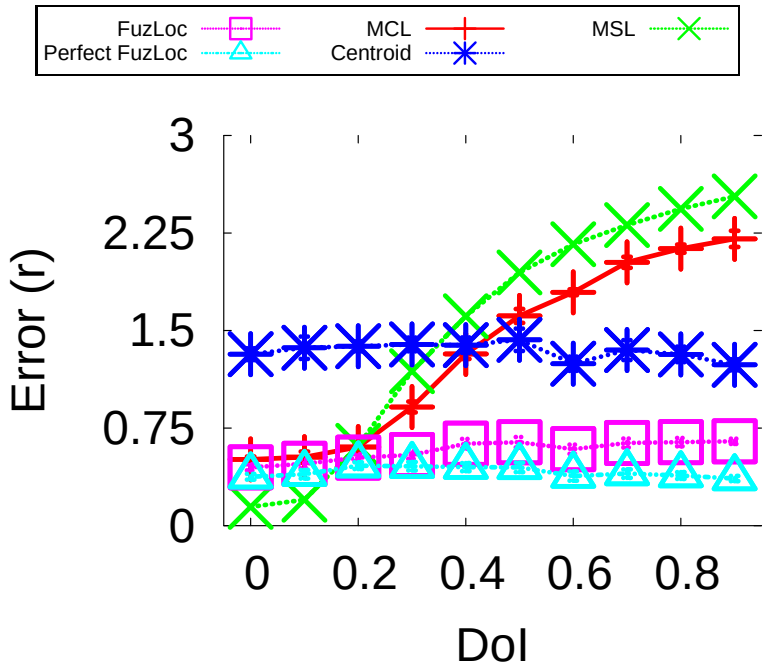


Figure 9.13: The effect of DoI on localization accuracy. (N=320, S=32, v=0.2r)

DoI, with the error of FuzLoc increasing by about 20% at maximum DoI compared to MCL's 300%.

9.5.5 Maximum Node Velocity

We investigate the effect of maximum node velocity on localization accuracy, for velocities up to 0.5r, a reasonably fast moving speed. The performance results are depicted in Figure 9.14. MCL and MSL assume that nodes know their maximum velocity. Hence, they use the velocity as a filtering condition, which improves their performance. Moreover, high velocity means having more anchors to filter against, leading to the freshening of samples at every instance. Figure 9.14 shows that MCL and MSL decrease their localization error from 1.4r to 0.9r, and 1.9r to 1.4r, respectively. Since Centroid and FuzLoc do not use the velocity, their performance is not expected to improve. Figure 9.14 indicates that their performance is not deteriorat-

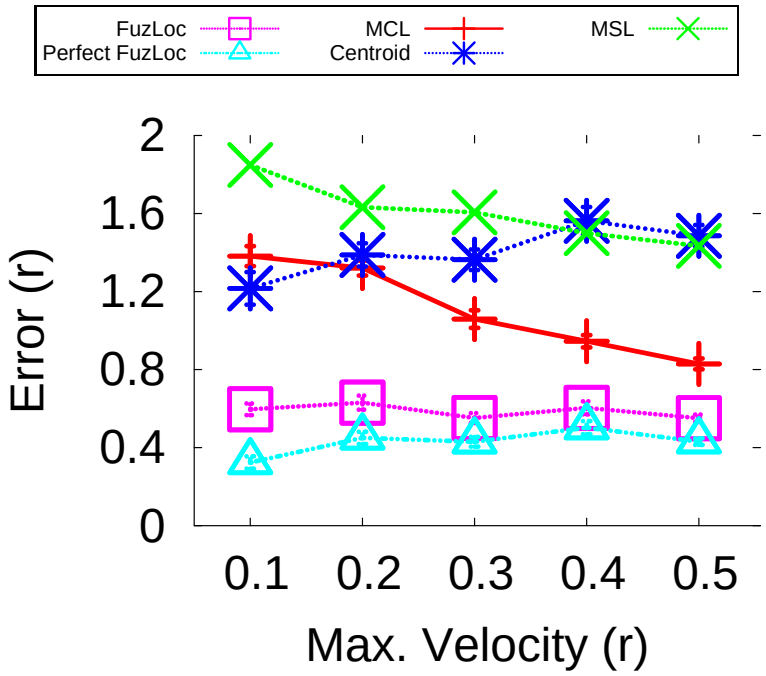


Figure 9.14: The effect maximum node velocity has on localization accuracy. (N=320, S=32, DoI=0.4)

ing.

9.5.6 Anchor Density

Anchor density is a critical parameter for anchor-based localization schemes. Figure 9.15 displays the impact of anchor density on the localization schemes where the number of anchors varies from 10% (32 anchors) to 50% (160 anchors), and the DoI is constant at 0.4. The accuracy of MCL and MSL deteriorates because an increase in anchor density is associated with an increase in the number of polluting sources. The mismatch of observed and actual radio ranges causes spurious anchors to appear as node's direct and indirect seeds. MSL considers non-anchor neighbors, hence it experiences higher pollution. Centroid performs better with increasing anchor density, as expected. FuzLoc also has a decrease in localization error, with a larger

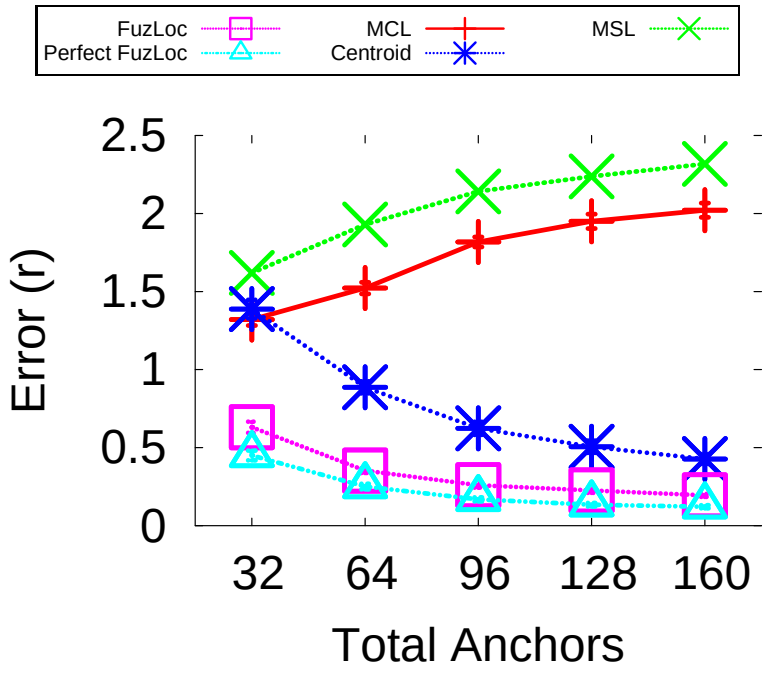


Figure 9.15: The effect of anchor density on localization accuracy at DoI=0.4 (N=320, v=0.2r)

number of anchors. We observe that FuzLoc is not significantly affected by DoI and ranging errors.

9.5.7 Node Density

For this performance evaluation scenario we maintained the percentage of anchors fixed at 10%. As shown in Figure 9.16, the evaluated algorithms either suffer or are unaffected. None of the localization algorithms benefits from an increase in the node density. As shown, Centroid and FuzLoc are not substantially affected, except by the inherent randomness in simulation. MCL considers indirect seeds for sampling, hence a high node density means more anchors are misreported as indirect seeds. MSL considers non-anchor neighbors, hence at high node densities, it experiences a huge amount of sample pollution. While non-anchor neighbors help MSL to improve

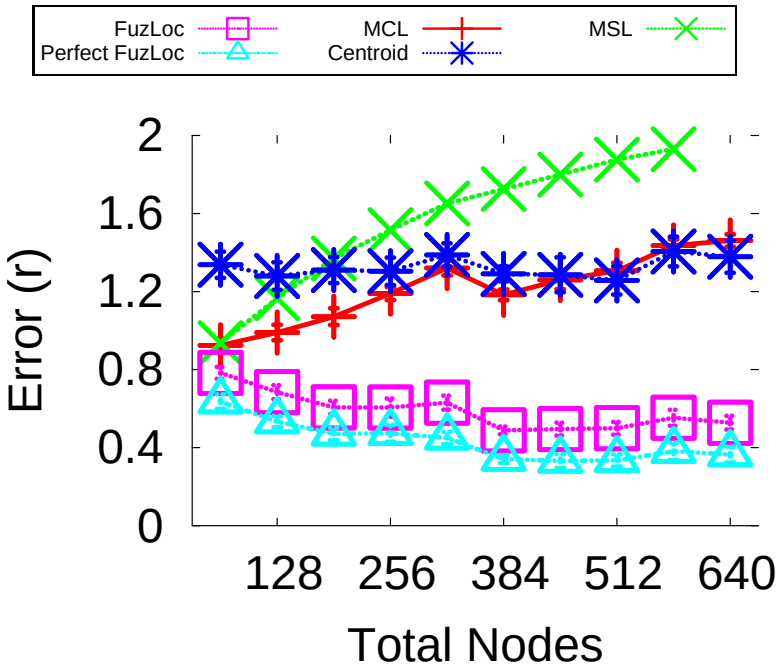


Figure 9.16: The effect of node density on localization accuracy. (S=32, v=0.2r, DoI=0.4)

accuracy at low DoI, they become harmful at higher DoI values.

9.5.8 Number of Bins

The number of bins in the fuzzy system is a design parameter - the greater the number of bins, the higher the accuracy of the system. Our evaluation of the influence of the number of bins is depicted in Figure 9.17. As shown, as the number of bins increases, the localization error of FuzLoc decreases. This is because more and more RSSs find a bin with high membership. The change in the number of bins, is expected to not affect MCL, MSL, Centroid, or even Perfect FuzLoc. Figure 9.17 shows that the aforementioned schemes remain invariant whereas FuzLoc experiences decreasing error with an increase in the number of bins.

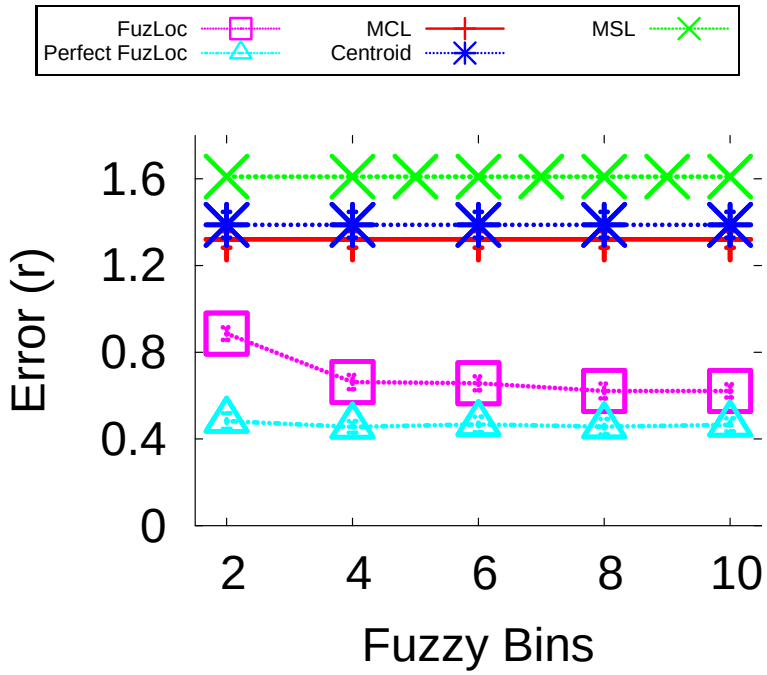


Figure 9.17: The effect of the number of fuzzy bins on localization accuracy. (N=320, S=32, v=0.2r, DoI=0.4)

9.5.9 Fuzzy Inference System Performance

Figure 9.18a shows the performance of the FIS engine evaluated using RSSI-distance data generated by the EDoI model, while Figure 9.18b does the same using data from our static testbed and finally, Figure 9.18c uses data from the small mobile testbed. Input distance is on the X axis while the Y axis marks the center value of the defuzzified output distance. After training the system with 30 random RSS-Distance pairs, RSS values deduced from distances were fed into the system so that a distance could be inferred. The straight line shows the ideal case. In order to quantify the accuracy, the root mean squared (RMS) error was calculated and normalized to the radio range. The values are remarkably similar: for the EDoI dataset it was 0.156, for the static network dataset it was 0.182 and for the small mobile testbed it was

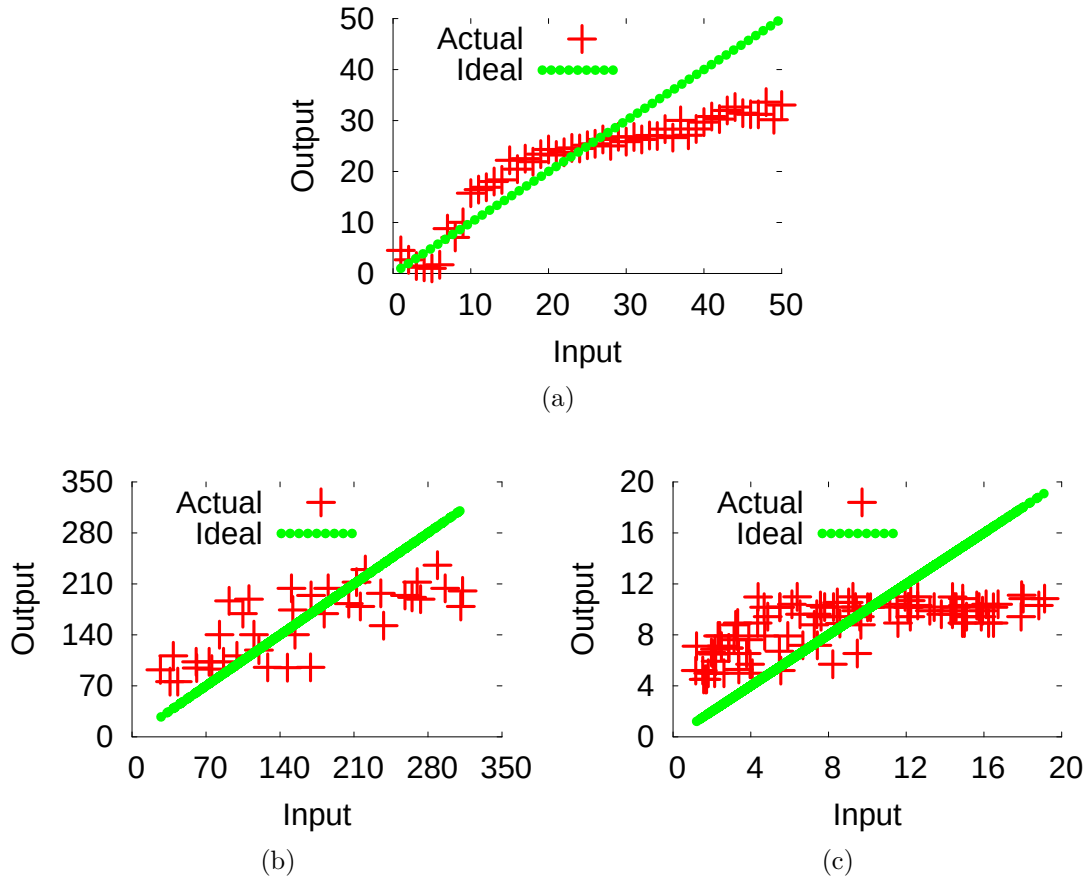


Figure 9.18: (a) Performance of the FMS FIS subsystem, with the test input on the X axis and the inferred distances on the Y axis; (b) performance based on real data gathered from our indoor testbed; (c) performance based on real data gathered from our mobile testbed consisting of 6 iRobots.

0.166. In a way, these numbers reinforce the equivalence of the simulated and real indoor static/mobile radio models, while proving the effectiveness of the EDoI model.

9.5.10 Overhead

A typical FIS does not require much storage capacity. If there were 8 bins, for example, a single byte could represent a bin. Hence, each FMS rule requires just 2B of storage. Typically, an anchor creates approximately 30 rules during the period

of deployment which translates to 60B of storage. The FGPS FIS however, requires 50B for each rule (25 bins in the input, 25 in the output). Note that regular nodes do not store rules, only the anchors store rules. Moreover, due to the nature of the triangular bin shapes, simple calculations are required in order to fuzzify and defuzzify. The only caveat is the inversion of matrices that is required. As for the filter, the node does not construct a filter for all possible cells, since it usually visits a maximum of 4 cells per iteration. Hence, the storage required by a 3-order filter on each non-anchor node will be (288×4×4×3) = 13,824B. MCL requires at least 50 samples for low localization error. Each sample requires a weight. Centroid does not store any history and thus has the smallest storage requirement. Amorphous stores announcements made by the anchors which are flooded throughout the network. If there are 320 nodes, 32 of which are anchors, MCL requires each node to store 50 samples. Each sample has an abscissa and an ordinate, each of at least 4B. Hence, MCL requires around (50×4×2×320) = 128,000B. Fuzzy on the other hand requires around 1,500B for FGPS and around 60 for FMS, with 13,824B for the filters, which sums up to (1,560×32 + 13,824) = 63,744B which is roughly 50% of the storage MCL requires, and even less than what MSL requires, since MSL mains closeness values.

The communication overhead for 2-hop anchor discovery is the same as that of MCL, and less that of MSL (since MSL needs to exchange samples in addition to anchor discovery). When FuzLoc uses only 1-hop anchors, the communication overhead required is significantly lower since all that is needed is a simple broadcast. Still, FuzLoc performs better than MCL as can be seen in [128]. Therefore, systems desiring lesser communication overhead should use anchors within 1 hop only, while those desiring higher accuracy need to consider anchors within 2 hops. In no state will the communication overhead required by FuzLoc exceed that of MCL or MSL.

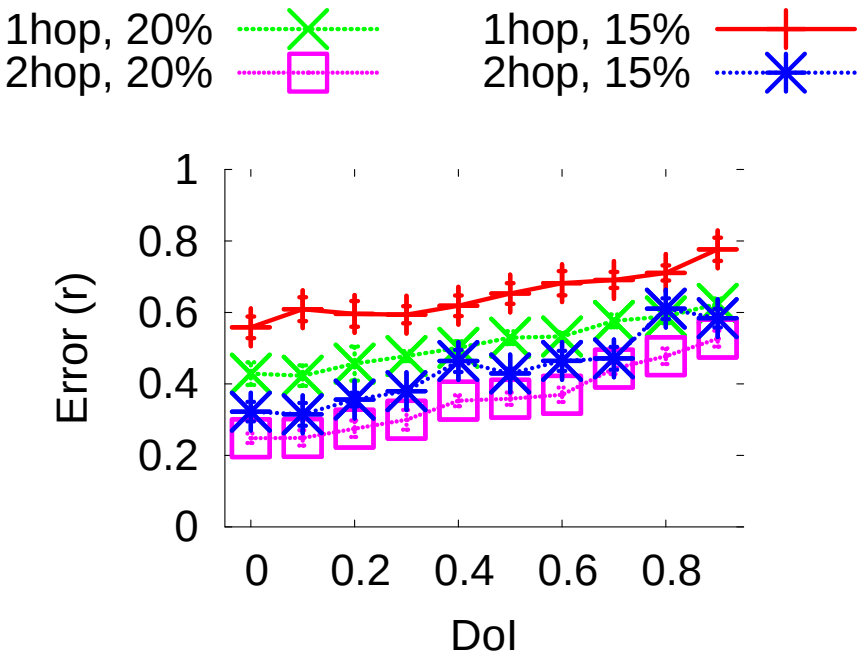


Figure 9.19: Comparison of 1 and 2 hop FuzLoc variants at seed densities of 15% (S=48) and 20% (S=64) in a 320 node (N) network across multiple DoIs.

9.5.11 Single Hop and Dual Hop FMS

Figure 9.19 compares the 1-hop and 2-hop variants of FuzLoc. Being a multilateration based method, the presence of a sufficient number of anchors in a node's vicinity is crucial to reducing the error in location estimation. A simple way to ensure this is to increase the percentage of anchors in the network. However, the addition of anchors may be cost prohibitive. A simpler way and less costly solution is to consider anchors which are two hops away. The additional cost incurred for this solution is higher messaging overhead. Instead of traveling over a single hop, localization request broadcasts must take two hops to reach the outer anchors. Replies are consolidated, so no additional messaging is incurred in the reply phase. The number of additional transmissions required vary based on node and anchor density. Figure 9.19 shows that merely considering the 2-hop anchors results in a much lower

error due to the increased number of anchors, than introducing more anchors, across all DoIs. Note that although the number of messages increases, the error is more than halved.

9.6 Conclusions

We have proposed FuzLoc, a fuzzy logic based localization method suitable for wireless sensor nodes that are mobile in noisy, harsh environments. The constituent systems use fuzzy multilateration and a grid predictor to compute the location of a node as an area. The RSS is cast into bins which encode the imprecision; these bins are subsequently used in our mathematical framework. We remark here that the case of static anchors, considered by neither MCL, nor MSL, will be investigated in future work.

Our method has been evaluated based on a variety of metrics. They prove that our method is resistant to high DoI environments while providing a low localization error without any extra hardware. Only anchors need to have a slightly higher storage requirement. A deployment with more anchors at high DoI decreases the error. The ability to localize using both single-hop and two-hop anchors greatly increases the variety of topologies where localization succeeds. The system implementation proves that the algorithm functions well on resource constrained devices.

10. CONCLUSIONS AND FUTURE WORK

This dissertation has proposed Fog Computing, a mobile cloud architecture for disaster response networks. Starting with a motivating scenario consisting of a natural disaster that occurs over a large geographical area, the needs of Urban Search and Rescue workers in such a situation were analyzed. Based on these requirements, as well as certain design principles, a high level system architecture was presented. The research challenges in designing and implementing this architecture were enumerated. Heterogeneous COTS devices, ranging from low power wireless sensors to enterprise grade wireless routers were used to build a proof of concept implementation. Some responder requirements were implemented as user level apps, while others comprised the middleware that runs in the background.

Additional hardware were placed over the area of deployment to artificially increase the contact opportunities in the underlying DRN, thereby increasing the capacity of the network. A reduced version of the Post Disaster Mobility Model was used to formulate a problem that can be cast as a binary integer programming problem. As a result of optimal placement of devices, the aggregate throughput was found to increase based on experiments involving multiple data flows and devices. The impact of MTU size at the bundle and packet layers upon the total amount of data transferred per contact was analyzed experimentally.

The Raven framework provides the user control over various QoS metrics. More importantly, it allows the user to control the variance of the packet delivery delay, using risk aversion techniques. A multi-graph whose edge weights are not scalars but distributions was used to compute multiple paths between source and destination, based on the enhanced version Post Disaster Mobility Model. Results using simula-

tion were presented. The effect of simultaneous control of multiple QoS metrics was analyzed mathematically.

Energy consumption profiles of fog computing devices were analyzed, and it was determined that energy can be saved on certain routers deployed at Centers by excluding them from the routing process. Such an approach has marked effect on the system's performance, and this energy-performance tradeoff was investigated. A Pareto front between energy and performance was discovered using a problem formulation that involved dual non-linear objectives. The system could be made to operate at each of these Pareto points by carefully choosing end points for potential flows, as well as assigning a K value to each of these flows. Simulation results confirmed that the system could be highly energy efficient at the cost of high PDD, or have a very low PDD at the cost of high energy consumption. This scheme can outperform state of art techniques in terms of both performance or energy consumption, but not simultaneously due to the inherent tradeoff present.

Indoor localization for mobile sensors was one of the requirements since GPS may not available in indoor environments due to radio shadow areas. FuzLoc, a distributed algorithm that performs range-based localization over multiple hops was presented. It recovers the distance between nodes using only the RSS value, and this association is learned over time. The location of the node is computed as an area instead of a 2-D point, using a system of non-linear equations where the variables are not scalars but are fuzzy bins. Using 2-hop anchors, the localization accuracy was shown to be much better than when using single hop anchors. The network messaging overhead for two hop localization was analyzed.

10.1 Future Work

10.1.1 Hardware Implementation

This dissertation is an academic effort that proposes many new sensing modalities as well as mobile cloud paradigms. The scalability of this system was not completely analyzed owing to the manpower required to conduct experiments that involve hundreds of devices and tens of vehicles. System-wide software integration of various middleware and apps is not a trivial task, and is left as future work. Newer PHY layer protocols such as Bluetooth LE and ANT can be integrated into the network architecture. As PHY layer diversity increases, so does the complexity. When the Router class of devices is required to support more and more PHY layers, the power consumption increases. An optimal duty cycling scheme that takes into account the number of nearby devices with the same PHY layer is a research question that can be answered. The protocols used and proposed in this dissertation can be implemented in silicon for lower energy consumption as well as a better form factor, leading to possible commercialization of fog computing. A new computer architecture at the hardware level can be proposed for the fog computing network architecture. The increasing use of SoCs and specialized hardware such as Apple's motion co-processor that allows continuous low power sensing should provide sufficient motivation for this direction.

10.1.2 Security

The topic of security has not been addressed in this dissertation due to its wide scope. Any good networking system should be designed with security in mind. DRNs in particular handle sensitive information that could emotionally affect victims - such information should be contained within the system. At the PHY layer, securing 802.11 IBSS mode will lead to an increase in overhead as well as a possible increase

in setup time for inter mobile node links. This will in turn have repercussions on the amount of data that can be transferred in a contact (i.e., the DTC). The fog computing API can be secured at the application layer. Enterprise-level standards compliant security enables third party developers to confidently integrate their apps with the system.

10.1.3 Theory and Algorithms

A large complex system that implements fog computing has to function in the face of uncertainty, due to the febrile and fast paced environment of disaster recovery. The input provided to the various algorithms may not be precise - thus, the output of these algorithms may have some degree of error. This error propagates throughout the system over time, as the computed results provided by sub-systems is used as input in other sub-systems. Theoretical analysis of error propagation in a large complex system can help improve the performance. Certain algorithms used in the dissertation, such as the Bellman-Ford algorithm, may not return good quality paths. Addressing whether edge disjoint paths improve Raven's performance is a topic for future research.

REFERENCES

[1] H. Chenji, W. Zhang, M. Won, R. Stoleru, and C. Arnett, "A wireless system for reducing response time in urban search & rescue," in *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*, Austin, TX, USA, 2012. ©2012 IEEE. Reprinted, with permission.

[2] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. M. Ni, "Recognizing exponential inter-contact time in VANETs," in *Proceedings of the 29th Conference on Information Communications*, INFOCOM'10, San Diego, California, USA, 2010.

[3] A. Balasubramanian, B. Levine, and A. Venkataramani, "Dtn routing as a resource allocation problem," in *Proceedings of the 2007 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, SIGCOMM '07, Kyoto, Japan, 2007.

[4] M. Uddin, D. Nicol, T. Abdelzaher, and R. Kravets, "A post-disaster mobility model for delay tolerant networking," in *Simulation Conference (WSC), Proceedings of the 2009 Winter*, Austin, TX, USA, 2009.

[5] Missouri DPS, "Joplin tornado situation report 6 A.M. May 24," 2011. [Online, last accessed 15 Aug 2013]. Available: http://sema.dps.mo.gov/newspubs/SRTemplate.asp?ID=N09110049

[6] FEMA, "US&R task force equipment list," 2008. [Online, last accessed 15 Aug 2013]. Available: http://www.fema.gov/emergency/usr/equipment.shtm

[7] A. L. Hughes and L. Palen, "Twitter adoption and use in mass convergence and emergency events," *Int. J. of Emergency Management*, vol. 6, no. 3/4,

2009.

[8] J. Sutton, L. Palen, and I. Shklovski, "Backchannels on the front lines: emergent uses of social media in the 2007 southern california wildfires," in *Proceedings of the 5th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, Washington, DC, USA, 2008.

[9] B. Wiebusch, "Device 'listens' for sounds of life at WTC," 2001. [Online, last accessed 15 Aug 2013]. Available: http://www.designnews.com/document.asp?doc_id=213077

[10] USGS, "M7.0 - Haiti region," 2010. [Online, last accessed 15 Aug 2013]. Available: http://comcat.cr.usgs.gov/earthquakes/eventpage/pde20100112215310060_13

[11] British Broascasting Corporation, "Japan hit by tsunami after massive earthquake," 2011. [Online, last accessed 15 Aug 2013]. Available: http://www.bbc.co.uk/news/world-asia-pacific-12709850

[12] FEMA, "Federal emergency management agency," 2014. [Online, last accessed 15 Aug 2013]. Available: http://www.fema.gov

[13] ——, "Unit 4: NIMS communications and information management," 2014. [Online, last accessed 15 Aug 2013]. Available: http://training.fema.gov/EMIWeb/is/IS700a/SM%20files/IS700A_StudentManual_L4.pdf

[14] PTIG, "Project 25 technology interest group," 2014. [Online, last accessed 15 Aug 2013]. Available: http://www.project25.org/

[15] Motorola, "Project 25 interoperable communications for public safety agencies," 2011. [Online, last accessed 15 Aug 2013]. Available: http://www.motorolasolutions.com/web/Business/Solutions/Business%

20Solutions/Mission%20Critical%20Communications/ASTRO%2025%20Trunked%20Solutions/_Document/Project%2025%20Whitepaper.pdf

[16] Mesh Dynamics, "Network-centric warfare and wireless communications," 2008. [Online, last accessed 15 Aug 2013]. Available: http://www.meshdynamics.com/documents/MD_MILITARY_MESH.pdf

[17] L. A. Lenert, D. Kirsh, W. G. Griswold, C. Buono, J. Lyon, R. Rao, and T. C. Chan, "Design and evaluation of a wireless electronic health records system for field care in mass casualty settings," *Journal of American Medical Informatics Association*, vol. 18, no. 6, 2011.

[18] S. W. Brown, M. William, G. Griswold, B. Demchak, B. Leslie, and A. Lenert, "Middleware for reliable mobile medical workflow support in disaster settings," *AMIA Annu Symp Proc.*, vol. 2006, pp. 309–313, 2006.

[19] O. Chipara, W. G. Griswold, A. N. Plymoth, R. Huang, F. Liu, P. Johansson, R. Rao, T. Chan, and C. Buono, "Wiisard: A measurement study of network properties and protocol reliability during an emergency response," in *Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services*, MobiSys '12, Low Wood Bay, Lake District, UK, 2012.

[20] T. Gao, T. Massey, L. Selavo, D. Crawford, B. rong Chen, K. Lorincz, V. Shnayder, L. Hauenstein, F. Dabiri, J. Jeng, A. Chanmugam, D. White, M. Sarrafzadeh, and M. Welsh, "The advanced health and disaster aid network: a light-weight wireless medical system for triage," *Biomedical Circuits and Systems, IEEE Transactions on*, vol. 1, no. 3, pp. 203–216, 2007.

[21] M. Arisoylu, R. Mishra, R. Rao, and L. A. Lenert, "802.11 wireless infrastructure to enhance medical response to disasters," *AMIA Annu Symp*

Proc, vol. 2005, pp. 1–5, 2005.

[22] J. P. Killeen, T. C. Chan, C. Buono, W. G. Griswold, and L. A. Lenert, "A wireless first responder handheld device for rapid triage, patient assessment and documentation during mass casualty incidents," *AMIA Annu Symp*, vol. 2006, 2006.

[23] L. A. Lenert, D. A. Palmer, T. C. Chan, and R. Rao, "An intelligent 802.11 triage tag for medical response to disasters," *AMIA Annu Symp Proc*, vol. 2005, 2005.

[24] S. Mehrotra, C. T. Butts, D. Kalashnikov, N. Venkatasubramanian, R. R. Rao, G. Chockalingam, R. Eguchi, B. J. Adams, and C. Huyck, "Project rescue: challenges in responding to the unexpected," *Proc. SPIE*, vol. 5304, pp. 179–192, 2003.

[25] R. Dilmaghani and R. Rao, "An ad hoc network infrastructure: communication and information sharing for emergency response," in *Proceedings of the 2008 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob '08)*, Avignon, France, 2008.

[26] B. Manoj and A. H. Baker, "Communication challenges in emergency response," *Commun. ACM*, vol. 50, no. 3, pp. 51–53, Mar. 2007.

[27] R. B. Dilmaghani, B. S. Manoj, B. Jafarian, and R. R. Rao, "Performance evaluation of rescue mesh: a metro-scale hybrid wireless network," in *Proceedings of the first IEEE Workshop on Wireless Mesh Networks (WiMesh)*, Santa Clara, CA, USA, 2005.

[28] B. Xing, S. Mehrotra, and N. Venkatasubramanian, "Radcast: enabling reliability guarantees for content dissemination in ad hoc networks," in *Proceedings of the 28th IEEE International Conference on Computer Communications (INFOCOM)*, Rio de Janeiro, Brazil, 2009.

[29] S. George, W. Zhou, H. Chenji, M. Won, Y. O. Lee, A. Pazarloglou, R. Stoleru, and P. Barooah, "Distressnet: a wireless ad hoc and sensor network architecture for situation management in disaster response," *Communications Magazine, IEEE*, vol. 48, no. 3, pp. 128–136, March 2010. ©2010 IEEE. Reprinted, with permission.

[30] F. Dong, Y. Hu, M. Tong, and X. Ran, "Supporting emergency service by retasking delay-tolerant network architecture," in *Proceedings of the 2009 Fifth International Conference on Mobile Ad-hoc and Sensor Networks*, MSN '09, Wu Yi Mountain, China, 2009.

[31] D. Bradler, B. Schiller, E. Aitenbichler, and N. Liebau, "Towards a distributed crisis response communication system," in *Proceedings of the 6th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, Gothenburg, Sweden, May 2009.

[32] R. Bruno, M. Conti, and A. Passarella, "Opportunistic networking overlays for ICT services in crisis management," in *Proceedings of the 5th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, Washington, DC, USA, 2008.

[33] C. Boldrini, M. Conti, J. Jacopini, and A. Passarella, "Hibop: a history based routing protocol for opportunistic networks," in *World of Wireless, Mobile and Multimedia Networks, 2007. WoWMoM 2007. IEEE International Symposium on a*, Helsinki, Finland, 2007.

[34] G. Werner-Allen, J. Johnson, M. Ruiz, J. Lees, and M. Welsh, "Monitoring volcanic eruptions with a wireless sensor network," in *Wireless Sensor Networks, 2005. Proceeedings of the Second European Workshop on*, Jan 2005.

[35] S. Iyengar, P. Varshney, and T. Damarla, "On the detection of footsteps based on acoustic and seismic sensing," in *Conference Record of the Forty-First Asilomar Conference on Signals, Systems and Computers, 2007. ACSSC 2007.*, Asilomar Grounds, Pacific Grove, California, Nov 2007.

[36] G. Mazarakis and J. Avaritsiotis, "A prototype sensor node for footstep detection," in *Proceeedings of the Second European Workshop on Wireless Sensor Networks (EWSN)*, Istanbul, Turkey, Jan 2005.

[37] A. Pakhomov and T. Goldburt, "Seismic signal and noise assessment for footstep detection range estimation in different environments," *Proc. SPIE*, vol. 5417, pp. 87–98, 2004.

[38] M. Ceriotti, L. Mottola, G. Picco, A. Murphy, S. Guna, M. Corra, M. Pozzi, D. Zonta, and P. Zanon, "Monitoring heritage buildings with wireless sensor networks: The torre aquila deployment," in *Information Processing in Sensor Networks, 2009. IPSN 2009. International Conference on*, San Francisco, CA, USA, April 2009.

[39] F. Peña Mora, A. Chen, Z. Aziz, L. Soibelman, L. Liu, K. El-Rayes, C. Arboleda, T. Lantz, A. Plans, S. Lakhera, and S. Mathur, "Mobile ad hoc network-enabled collaboration framework supporting civil engineering emergency response operations," *Journal of Computing in Civil Engineering*, vol. 24, no. 3, 2010.

[40] W. Zhao, Y. Chen, M. Ammar, M. Corner, B. Levine, and E. Zegura, "Capacity enhancement using throwboxes in DTNs," in *Mobile Adhoc and*

Sensor Systems (MASS), 2006 IEEE International Conference on, Vancouver, BC, Canada, Oct 2006.

[41] N. Banerjee, M. Corner, and B. Levine, "An energy-efficient architecture for DTN throwboxes," in *INFOCOM 2007. 26th IEEE International Conference on Computer Communications. IEEE*, Anchorage, Alaska, USA, 2007.

[42] N. Banerjee, M. D. Corner, D. Towsley, and B. N. Levine, "Relays, base stations, and meshes: enhancing mobile networks with infrastructure," in *Proceedings of the 14th ACM International Conference on Mobile Computing and Networking*, MobiCom '08, San Francisco, California, USA, 2008.

[43] M. Ibrahim, P. Nain, and I. Carreras, "Analysis of relay protocols for throwbox-equipped DTNs," in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks, 2009. WiOPT 2009. 7th International Symposium on*, Seoul, Korea, June 2009.

[44] K. Fall, G. Iannaccone, J. Kannan, F. Silveira, and N. Taft, "A disruption-tolerant architecture for secure and efficient disaster response communications," in *Proceedings of the 7th International Conference on Information Systems for Crisis Response and Management (ISCRAM)*, Seattle, WA, USA, 2010.

[45] H. Soroush, N. Banerjee, A. Balasubramanian, M. D. Corner, B. N. Levine, and B. Lynn, "Dome: a diverse outdoor mobile testbed," in *Proceedings of the 1st ACM International Workshop on Hot Topics of Planet-Scale Mobility Measurements*, HotPlanet '09, Kraków, Poland, 2009.

[46] V. Dyo, S. A. Ellwood, D. W. Macdonald, A. Markham, C. Mascolo, B. Pásztor, S. Scellato, N. Trigoni, R. Wohlers, and K. Yousef, "Evolution and sustainability of a wildlife monitoring sensor network," in *Proceedings of*

the 8th ACM Conference on Embedded Networked Sensor Systems, SenSys '10, Zürich, Switzerland, 2010.

[47] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," in *Tech. Report CS-200006, Duke Univ., Durham, NC, USA*, 2000.

[48] M. Khouzani, S. Eshghi, S. Sarkar, N. B. Shroff, and S. S. Venkatesh, "Optimal energy-aware epidemic routing in DTNs," in *Proceedings of the Thirteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '12, Hilton Head, South Carolina, USA, 2012.

[49] T. Spyropoulos, K. Psounis, and C. S. Raghavendra, "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, Philadelphia, Pennsylvania, USA, 2005.

[50] ——, "Efficient routing in intermittently connected mobile networks: the single-copy case," *IEEE/ACM Trans. Netw.*, vol. 16, no. 1, pp. 63–76, Feb. 2008.

[51] A. Lindgren, A. Doria, and O. Schelén, "Probabilistic routing in intermittently connected networks," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 7, no. 3, pp. 19–20, Jul. 2003.

[52] J. Burgess, B. Gallagher, D. Jensen, and B. Levine, "Maxprop: routing for vehicle-based disruption-tolerant networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, Barcelona, Spain, April 2006.

[53] A. Picu and T. Spyropoulos, "Distributed stochastic optimization in opportunistic networks: the case of optimal relay selection," in *Proceedings of*

the 5th ACM Workshop on Challenged Networks, CHANTS '10, Chicago, Illinois, USA, 2010.

[54] D. Gunawardena, T. Karagiannis, A. Proutiere, E. Santos-Neto, and M. Vojnovic, "Scoop: decentralized and opportunistic multicasting of information streams," in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, Las Vegas, Nevada, USA, 2011.

[55] K. Chen and H. Shen, "Smart: lightweight distributed social map based routing in delay tolerant networks," in *Network Protocols (ICNP), 2012 20th IEEE International Conference on*, Austin, TX, USA, Oct 2012.

[56] X. Tie, A. Venkataramani, and A. Balasubramanian, "R3: robust replication routing in wireless networks with diverse connectivity characteristics," in *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking*, MobiCom '11, Las Vegas, Nevada, USA, 2011.

[57] J. Alonso and K. Fall, "A linear programming formulation of flows over time with piecewise constant capacity and transit times," Intel Research Berkley, Tech. Rep., 2003.

[58] M. Al-Siyabi, H. Cruickshank, and Z. Sun, "Dtn qos metrics and fair resources management model," in *Electrical and Computer Engineering (CCECE), 2011 24th Canadian Conference on*, Niagara Falls, Ontario, 2011.

[59] D. P. Bertsekas and J. N. Tsitsiklis, "An analysis of stochastic shortest path problems," *Math. Oper. Res.*, vol. 16, no. 3, Aug 1991.

[60] S. Lim, C. Sommer, E. Nikolova, and D. Rus, "Practical route planning under delay uncertainty: stochastic shortest path queries," in *Proceedings of*

Robotics: Science and Systems, Sydney, Australia, July 2012.

[61] P. Gupta and P. Kumar, "A system and traffic dependent adaptive routing algorithm for ad hoc networks," in *Decision and Control, 1997., Proceedings of the 36th IEEE Conference on*, San Diego, CA, USA, Dec 1997.

[62] M. Rubinstein, F. Ben Abdesslem, M. Dias de Amorim, S. Cavalcanti, R. Dos Santos Alves, L. Costa, O. Duarte, and M. Campista, "Measuring the capacity of in-car to in-car vehicular networks," *Communications Magazine, IEEE*, vol. 47, no. 11, pp. 128–136, November 2009.

[63] M. Uddin, H. Ahmadi, T. Abdelzaher, and R. Kravets, "Intercontact routing for energy constrained disaster response networks," *Mobile Computing, IEEE Transactions on*, vol. 12, no. 10, pp. 1986–1998, 2013.

[64] W. Wang, M. Motani, and V. Srinivasan, "Opportunistic energy-efficient contact probing in delay-tolerant applications," *IEEE/ACM Trans. Netw.*, vol. 17, no. 5, pp. 1592–1605, Oct. 2009.

[65] X. Lu and P. Hui, "An energy-efficient n-epidemic routing protocol for delay tolerant networks," in *Networking, Architecture and Storage (NAS), 2010 IEEE Fifth International Conference on*, University of Macau, Macau SAR, China, July 2010.

[66] Y. Li, Y. Jiang, D. Jin, L. Su, L. Zeng, and D. Wu, "Energy-efficient optimal opportunistic forwarding for delay-tolerant networks," *Vehicular Technology, IEEE Transactions on*, vol. 59, no. 9, pp. 4500–4512, 2010.

[67] J. Widmer and J.-Y. Le Boudec, "Network coding for efficient communication in extreme networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop*

on Delay-tolerant Networking, WDTN '05, Philadelphia, Pennsylvania, USA, 2005.

[68] Y. Wang, S. Jain, M. Martonosi, and K. Fall, "Erasure-coding based routing for opportunistic networks," in *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking*, WDTN '05, Philadelphia, Pennsylvania, USA, 2005.

[69] Y. Liao, K. Tan, Z. Zhang, and L. Gao, "Estimation based erasure-coding routing in delay tolerant networks," in *Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*, IWCMC '06, Vancouver, British Columbia, Canada, 2006.

[70] C. Chilipirea, A. Petre, and C. Dobre, "Energy-aware social-based routing in opportunistic networks," in *Advanced Information Networking and Applications Workshops (WAINA), 2013 27th International Conference on*, Barcelona, Spain, March 2013.

[71] N. Banerjee, M. Corner, and B. Levine, "Design and field experimentation of an energy-efficient architecture for DTN throwboxes," *Networking, IEEE/ACM Transactions on*, vol. 18, no. 2, pp. 554–567, April 2010.

[72] H. Jun, M. H. Ammar, M. D. Corner, and E. W. Zegura, "Hierarchical power management in disruption tolerant networks with traffic-aware optimization," in *Proceedings of the 2006 SIGCOMM Workshop on Challenged Networks*, CHANTS '06, Pisa, Italy, 2006.

[73] W. Zhao, M. Ammar, and E. Zegura, "Controlling the mobility of multiple data transport ferries in a delay-tolerant network," in *INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings IEEE*, Miami, FL, USA, March 2005.

[74] F. De Pellegrini, E. Altman, and T. Başçar, "Optimal monotone forwarding policies in delay tolerant mobile ad hoc networks with multiple classes of nodes," in *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2010 Proceedings of the 8th International Symposium on*, Avignon, France, May 2010.

[75] Y. Shang, W. Ruml, Y. Zhang, and M. P. J. Fromherz, "Localization from mere connectivity," in *Proceedings of the 4th ACM International Symposium on Mobile Ad Hoc Networking & Computing*, MobiHoc '03, Annapolis, Maryland, USA, 2003.

[76] J. A. Costa, N. Patwari, and A. O. Hero III, "Distributed multidimensional scaling with adaptive weighting for node localization in sensor networks," *ACM J. Sensor Networking*, vol. 2, 2004.

[77] M. Maróti, P. Völgyesi, S. Dóra, B. Kusý, A. Nádas, A. Lédeczi, G. Balogh, and K. Molnár, "Radio interferometric geolocation," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, San Diego, California, USA, 2005.

[78] H.-l. Chang, J.-b. Tian, T.-T. Lai, H.-H. Chu, and P. Huang, "Spinning beacons for precise indoor localization," in *Proceedings of the 6th ACM Conference on Embedded Network Sensor Systems*, SenSys '08, Raleigh, NC, USA, 2008.

[79] R. Stoleru, P. Vicaire, T. He, and J. A. Stankovic, "Stardust: A flexible architecture for passive localization in wireless sensor networks," in *Proceedings of the 4th International Conference on Embedded Networked Sensor Systems*, SenSys '06, Boulder, Colorado, USA, 2006.

[80] R. Stoleru, T. He, J. A. Stankovic, and D. Luebke, "A high-accuracy, low-cost localization system for wireless sensor networks," in *Proceedings of the 3rd International Conference on Embedded Networked Sensor Systems*, SenSys '05, San Diego, California, USA, 2005.

[81] A. Savvides, C.-C. Han, and M. B. Srivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, MobiCom '01, Rome, Italy, 2001.

[82] N. B. Priyantha, A. Chakraborty, and H. Balakrishnan, "The cricket location-support system," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, MobiCom '00, Boston, Massachusetts, USA, 2000.

[83] K. Whitehouse and D. Culler, "Calibration as parameter estimation in sensor networks," in *Proceedings of the 1st ACM International Workshop on Wireless Sensor Networks and Applications*, WSNA '02, Atlanta, Georgia, USA, 2002.

[84] N. Patwari and A. O. Hero III, "Using proximity and quantized rss for sensor localization in wireless networks," in *Proceedings of the 2Nd ACM International Conference on Wireless Sensor Networks and Applications*, WSNA '03, San Diego, CA, USA, 2003.

[85] P. Bahl and V. Padmanabhan, "RADAR: an in-building RF-based user location and tracking system," in *INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings. IEEE*, Tel Aviv, Israel, 2000.

[86] Y. Ji, S. Biaz, S. Pandey, and P. Agrawal, "ARIADNE: A Dynamic Indoor Signal Map Construction and Localization System," in *Proceedings of the 4th*

International Conference on Mobile Systems, Applications and Services, MobiSys '06, Uppsala, Sweden, 2006.

[87] A. Dharne, J. Lee, and S. Jayasuriya, "Using fuzzy logic for localization in mobile sensor networks: simulations and experiments," in *American Control Conference (ACC)*, Minneapolis, MN, USA, June 2006.

[88] S.-Y. Chiang and J.-L. Wang, "Localization in wireless sensor networks by fuzzy logic system," in *Knowledge-Based and Intelligent Information and Engineering Systems*, Lecture Notes in Computer Science, J. Velásquez, S. Ros, R. Howlett, and L. Jain, Eds. Springer Berlin Heidelberg, 2009, vol. 5712, pp. 721–728.

[89] D. Shao-Long, W. Jian-Ming, X. Tao, and L. Hai-tao, "Constraint-based fuzzy optimization data fusion for sensor network localization," in *Semantics, Knowledge and Grid, 2006. SKG '06. Second International Conference on*, Guilin, Guangxi, China, Nov 2006.

[90] X. Shen, J. W. Mark, and J. Ye, "Mobile location estimation in cellular networks using fuzzy logic," in *Vehicular Technology Conference, 2000. IEEE-VTS Fall VTC 2000. 52nd*, Boston, MA, USA, 2000.

[91] D. Niculescu and B. Nath, "Dv based positioning in ad hoc networks," *Telecommunication Systems*, vol. 22, no. 1-4, pp. 267–280, 2003.

[92] C. Wang and L. Xiao, "Sensor localization in concave environments," *ACM Trans. Sen. Netw.*, vol. 4, no. 1, pp. 3:1–3:31, Feb. 2008.

[93] S. Lederer, Y. Wang, and J. Gao, "Connectivity-based localization of large-scale sensor networks with complex shape," *ACM Trans. Sen. Netw.*, vol. 5, no. 4, pp. 31:1–31:32, Nov. 2009.

[94] N. Bulusu, J. Heidemann, and D. Estrin, "Gps-less low-cost outdoor localization for very small devices," *Personal Communications, IEEE*, vol. 7, no. 5, pp. 28–34, Oct 2000.

[95] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *Proceedings of the 9th Annual International Conference on Mobile Computing and Networking*, MobiCom '03, San Diego, CA, USA, 2003.

[96] Z. Zhong and T. He, "Achieving range-free localization beyond connectivity," in *Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems*, SenSys '09, Berkeley, California, 2009.

[97] Z. Yang and Y. Liu, "Quality of trilateration: Confidence-based iterative localization," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 21, no. 5, pp. 631–640, May 2010.

[98] A. Basu, J. Gao, J. S. B. Mitchell, and G. Sabhnani, "Distributed localization using noisy distance and angle information," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '06, Florence, Italy, 2006.

[99] M. Youssef, A. Youssef, C. Rieger, U. Shankar, and A. Agrawala, "Pinpoint: An asynchronous time-based location determination system," in *Proceedings of the 4th International Conference on Mobile Systems, Applications and Services*, MobiSys '06, Uppsala, Sweden, 2006.

[100] J. Liu, Y. Zhang, and F. Zhao, "Robust distributed node localization with error management," in *Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, MobiHoc '06, Florence, Italy, 2006.

[101] W. Xi, J. Zhao, X. Liu, X.-Y. Li, and Y. Qi, "EUL: An efficient and universal localization method for wireless sensor network," in *Distributed Computing Systems, 2009. ICDCS '09. 29th IEEE International Conference on*, Montreal, Quebec, Canada, June 2009.

[102] J.-g. Park, E. D. Demaine, and S. Teller, "Moving-baseline localization," in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks*, IPSN '08, Washington, DC, USA, 2008.

[103] J. N. Ash and L. C. Potter, "Robust system multiangulation using subspace methods," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, IPSN '07, Cambridge, Massachusetts, USA, 2007.

[104] P. Volgyesi, G. Balogh, A. Nadas, C. B. Nash, and A. Ledeczi, "Shooter localization and weapon classification with soldier-wearable networked sensors," in *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, San Juan, Puerto Rico, 2007.

[105] J. Sallai, P. Volgyesi, and A. Ledeczi, "Radio interferometric quasi doppler bearing estimation," in *Proceedings of the 2009 International Conference on Information Processing in Sensor Networks*, IPSN '09, Washington, DC, USA, 2009.

[106] B. Kusy, J. Sallai, G. Balogh, A. Ledeczi, V. Protopopescu, J. Tolliver, F. DeNap, and M. Parang, "Radio interferometric tracking of mobile wireless nodes," in *Proceedings of the 5th International Conference on Mobile Systems, Applications and Services*, MobiSys '07, San Juan, Puerto Rico, 2007.

[107] L. Klingbeil and T. Wark, "A wireless sensor network for real-time indoor localisation and motion monitoring," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, St. Louis, MO, USA, April 2008.

[108] Y. Kwon and G. Agha, "Passive localization: Large size sensor network localization based on environmental events," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, St. Louis, MO, USA, April 2008.

[109] V. Cevher and L. Kaplan, "Pareto frontiers of sensor networks for localization," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, St. Louis, MO, USA, April 2008.

[110] I. Paschalidis, K. Li, and D. Guo, "Model-free probabilistic localization of wireless sensor network nodes in indoor environments," in *Mobile Entity Localization and Tracking in GPS-less Environnments*, Lecture Notes in Computer Science, 2009.

[111] A. Bahill and R. Botta, "Fundamental principles of good system design," *EMJ - Engineering Management Journal*, vol. 20, no. 4, pp. 9–17, 2008.

[112] P. Barooah, H. Chenji, R. Stoleru, and T. Kalmar-Nagy, "Cut detection in wireless sensor networks," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 23, no. 3, pp. 483–490, 2012.

[113] G. Guven and E. Ergen, "Identification of local information items needed during search and rescue following an earthquake," *Disaster Prevention and Management*, vol. 20, no. 5, 2011.

[114] P. Dutta and D. Culler, "Epic: An open mote platform for application-driven design," in *Information Processing in Sensor Networks, 2008. IPSN '08. International Conference on*, St. Louis, MO, USA, April 2008.

[115] T. Winter, P. Thubert, A. Brandt, J. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J. Vasseur, and R. Alexander, "RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks," Internet Engineering Task Force, Mar. 2012. [Online, last accessed 15 Aug 2013]. Available: RFC6550(ProposedStandard)

[116] OpenWRT Dev Team, "Openwrt wireless freedom," 2014. [Online, last accessed 15 Aug 2013]. Available: https://openwrt.org/

[117] S. Schildt, J. Morgenroth, W.-B. Pöttner, and L. Wolf, "IBR-DTN: A lightweight, modular and highly portable bundle protocol implementation," *Electronic Communications of the EASST*, vol. 37, pp. 1–11, Jan 2011.

[118] H. Chenji, W. Zhang, R. Stoleru, and C. Arnett, "Distressnet: A disaster response system providing constant availability cloud-like services," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2440 – 2460, 2013.

[119] H. Chenji, L. Smith, R. Stoleru, and E. Nikolova, "Raven: Energy aware QoS control for DRNs," in *2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, Lyon, France, 2013. ©2013 IEEE. Reprinted, with permission.

[120] R. P. Loui, "Optimal paths in graphs with stochastic or multidimensional weights," *Commun. ACM*, vol. 26, no. 9, pp. 670–676, Sep. 1983.

[121] E. Nikolova, "Approximation algorithms for reliable stochastic combinatorial optimization," in *Proceedings of the 13th International Conference on*

Approximation, and 14 the International Conference on Randomization, and Combinatorial Optimization: Algorithms and Techniques, APPROX/RANDOM'10, Barcelona, Spain, 2010.

[122] J. Y. Yen, "Finding the K shortest loopless paths in a network," *Management Science*, vol. 17, no. 11, jul 1971.

[123] O. Barndorff-Nielsen, "On the limit behaviour of extreme order statistics," *The Annals of Mathematical Statistics*, vol. 34, no. 3, pp. 992–1002, sep 1963.

[124] A. Ker, "On the maximum of bivariate normal random variables," *Extremes*, vol. 4, no. 2, pp. 185–190, 2001.

[125] H. Chenji and R. Stoleru, "Pareto optimal cross layer lifetime optimization for disaster response networks," in *2014 Sixth International Conference on Communication Systems and Networks (COMSNETS)*, Bangalore, India, 2014. ©2014 IEEE. Reprinted, with permission.

[126] A. Balasubramanian, B. N. Levine, and A. Venkataramani, "Replication routing in DTNs: A resource allocation approach," *IEEE/ACM Trans. Netw.*, vol. 18, no. 2, pp. 596–609, Apr. 2010.

[127] H. Chenji and R. Stoleru, "Toward accurate mobile sensor network localization in noisy environments," *Mobile Computing, IEEE Transactions on*, vol. 12, no. 6, pp. 1094–1106, June 2013.

[128] ——, "Mobile sensor network localization in harsh environments," in *Proceedings of the 6th IEEE International Conference on Distributed Computing in Sensor Systems*, DCOSS'10, Santa Barbara, CA, 2010. ©2010 IEEE. Reprinted, with permission.

[129] G. Werner-Allen, K. Lorincz, M. Welsh, O. Marcillo, J. Johnson, M. Ruiz, and J. Lees, "Deploying a wireless sensor network on an active volcano," *IEEE Internet Computing*, vol. 10, no. 2, pp. 18–25, Mar. 2006.

[130] T. He, S. Krishnamurthy, L. Luo, T. Yan, L. Gu, R. Stoleru, G. Zhou, Q. Cao, P. Vicaire, J. A. Stankovic, T. F. Abdelzaher, J. Hui, and B. Krogh, "Vigilnet: An integrated sensor network system for energy-efficient surveillance," *ACM Trans. Sen. Netw.*, vol. 2, no. 1, pp. 1–38, Feb. 2006.

[131] D. Balakrishnan, A. Nayak, P. Dhar, and S. Kaul, "Efficient geo-tracking and adaptive routing of mobile assets," in *High Performance Computing and Communications, 2009. HPCC '09. 11th IEEE International Conference on*, Seoul, Korea, June 2009.

[132] H. Akcan, V. Kriakov, H. Brönnimann, and A. Delis, "Gps-free node localization in mobile wireless sensor networks," in *Proceedings of the 5th ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '06, Chicago, Illinois, USA, 2006.

[133] A. Baggio and K. Langendoen, "Monte carlo localization for mobile wireless sensor networks," *Ad Hoc Networks*, vol. 6, no. 5, pp. 718 – 733, 2008.

[134] D. Puccinelli and M. Haenggi, "Multipath fading in wireless sensor networks: Measurements and interpretation," in *Proceedings of the 2006 International Conference on Wireless Communications and Mobile Computing*, IWCMC '06, Vancouver, British Columbia, Canada, 2006.

[135] T. Rappaport, *Wireless Communications: Principles and Practice*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 2001.

[136] K. Whitehouse, C. Karlof, and D. Culler, "A practical evaluation of radio signal strength for ranging-based localization," *SIGMOBILE Mob. Comput. Commun. Rev.*, vol. 11, no. 1, pp. 41–52, Jan. 2007.

[137] H. Oshima, S. Yasunobu, and S.-i. Sekino, "Automatic train operation system based on predictive fuzzy control," in *Artificial Intelligence for Industrial Applications, 1988. IEEE AI '88., Proceedings of the International Workshop on*, Hitachi City, Japan, May 1988.

[138] L. Hu and D. Evans, "Localization for mobile sensor networks," in *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*, MobiCom '04, Philadelphia, PA, USA, 2004.

[139] M. Rudafshani and S. Datta, "Localization in wireless sensor networks," in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks*, IPSN '07, Cambridge, Massachusetts, USA, 2007.

[140] S. Engelson and D. McDermott, "Error correction in mobile robot map learning," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*, Nice, France, May 1992.

[141] S. Thrun, D. Fox, W. Burgard, and F. Dellaert, "Robust monte carlo localization for mobile robots," *Artificial Intelligence*, vol. 128, no. 1-2, pp. 99–141, 2000.

[142] K. Yedavalli and B. Krishnamachari, "Sequence-based localization in wireless sensor networks," *Mobile Computing, IEEE Transactions on*, vol. 7, no. 1, pp. 81–94, Jan 2008.

[143] J. Li and H. Wang, "Maximum power point tracking of photovoltaic generation based on the fuzzy control method," in *Sustainable Power*

Generation and Supply, 2009. SUPERGEN '09. International Conference on, Nanjing, China, April 2009.

[144] A. Kaufman, M. Gupta, and B. Esposito, *Introduction to Fuzzy Arithmetic: Theory and Applications*. New York, NY, USA: Van Nostrand Reinhold Company, 1991.

[145] J. Shokri, "On systems of fuzzy nonlinear equations." *Appl. Math. Sci.*, vol. 2, no. 25, pp. 1205–1217, 2008.

[146] X. Shen, J. W. Mark, and J. Ye, "User mobility profile prediction: An adaptive fuzzy inference approach," *Wirel. Netw.*, vol. 6, no. 5, pp. 363–374, Nov. 2000.

[147] M. Abramowitz, *Handbook of Mathematical Functions, With Formulas, Graphs, and Mathematical Tables*. Mineola, New York: Dover Publications, Incorporated, 1974.