3-21-2013

Airborne Network Data Availability Using Peer to Peer Database Replication on a Distributed Hash Table

Trevor J. Vranicar

AIRBORNE NETWORK DATA AVAILABILITY USING PEER TO PEER DATABASE REPLICATION ON A DISTRIBUTED HASH TABLE

THESIS

Trevor J. Vranicar, Second Lieutenant, USAF

AFIT-ENG-13-M-48

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT-ENG-13-M-48

AIRBORNE NETWORK DATA AVAILABILITY USING PEER TO PEER DATABASE REPLICATION ON A DISTRIBUTED HASH TABLE

THESIS

Presented to the Faculty

Department of Electrical and Computer Engineering

Graduate School of Engineering and Management

Air Force Institute of Technology

Air University

Air Education and Training Command

in Partial Fulfillment of the Requirements for the

Degree of Master of Science in Computer Engineering

Trevor J. Vranicar, B.S.C.E.

Second Lieutenant, USAF

March 2013

AFIT-ENG-13-M-48

AIRBORNE NETWORK DATA AVAILABILITY USING PEER TO PEER DATABASE REPLICATION ON A DISTRIBUTED HASH TABLE

Trevor J. Vranicar, B.S.C.E.
Second Lieutenant, USAF

Approved:

Gilbert L. Peterson, PhD (Chairman)	Date
Lt Col Robert J. McTasney, PhD (Member)	Date
Rusty O. Baldwin, PhD (Member)	Date

AFIT-ENG-13-M-48

Abstract

The concept of distributing one complex task to several smaller, simpler Unmanned Aerial Vehicles (UAVs) as opposed to one complex UAV is the way of the future for a vast number of surveillance and data collection tasks. One objective for this type of application is to be able to maintain an operational picture of the overall environment. Due to high bandwidth costs, centralizing all data may not be possible, necessitating a distributed storage system such as mobile Distributed Hash Table (DHT). A difficulty with this maintenance is that for an Airborne Network (AN), nodes are vehicles and travel at high rates of speed. Since the nodes travel at high speeds they may be out of contact with other nodes and their data becomes unavailable. To address this the DHT must include a data replication strategy to ensure data availability. This research investigates the percentage of data available throughout the network by balancing data replication and network bandwidth. The DHT used is Pastry with data replication using Beehive, running over an 802.11 wireless environment, simulated in Network Simulator 3. Results show that high levels of replication perform well until nodes are too tightly packed inside a given area which results in too much contention for limited bandwidth.

To My Bear

Acknowledgments

I express my deepest gratitude to all those whom continue to support me and sacrificed time and effort to help me through a difficult time.

Professor Peterson provided the patience, encouragement, and knowledge making this effort possible. He has dedicated extraordinary time in my education and success. I am indebted to him for my growth as a student, officer, and engineer. My gratitude goes out as well to all the professors in the Computer Engineering department for their advice and teachings.

I would not have contemplated this road if not for my parents, who instilled within me a love of creative pursuits, science, and language, all of which finds a place in this thesis. To my parents, thank you. My siblings have also been great friends throughout this process. My best friend stayed in touch weekly while completing Pilot Training and deserves my recognition; as do many other friends from Ohio and the Academy.

I thank my classmates for their insights both academically and professionally.

I owe uncountably infinite thanks my Maker.

Trevor J. Vranicar

Table of Contents

Page

List of Figures

List of Tables

List of Acronyms

Acronym	Definition
AN	Airborne Network
AODV	Ad-hoc On-Demand Distance Vector
CAN	Content Addressable Network
CHR	Cell Hash Routing
DHT	Distributed Hash Table
CIDR	Classless Inter-Domain Routing
DSR	Dynamic Source Routing
DSSS	Direct-sequence spread spectrum
GUID	Globally Unique ID
MANET	Mobile Ad-hoc Network
NS3	Network Simulator 3
OLSR	Optimized Link State Routing
OSI	Open Systems Interconnection
P2P	Peer-to-Peer
PDP	Peer Database Protocol
SAR	Spatially Aware Routing
UAV	Unmanned Aerial Vehicle
VANET	Vehicular Ad-Hoc Network

AIRBORNE NETWORK DATA AVAILABILITY USING PEER TO PEER DATABASE REPLICATION ON A DISTRIBUTED HASH TABLE

I. Introduction

A wide variety of military and civilian applications of airborne networking have resulted in growing research efforts in Airborne Networks (ANs) over the past few years [8, 60]. Supported by the advances in sensing and wireless communication technologies, ANs hold promise in providing effective, wide-applicable, low-cost, and secure information exchange among airborne vehicles[60]. For instance, the in-flight communication among commercial airlines can allow the sharing of adverse weather conditions and emergency situations, which are of significant value, especially, when the flights are in areas outside the reach of ground control stations. Similarly, Unmanned Aerial Vehicles (UAVs) may need to rely on reliable communication and networking schemes for safe maneuvering and data communication.

Imagine needing to maintain an operational picture of an overall environment using a decentralized storage system. The operational picture is being maintain by a group of UAVs that are searching an environment. However, due to bandwidth limitations not all data can be backed up to a single point, or UAV because of concern for failure and bandwidth limitations. Yet, there is still a need to be able to call up any piece of data at any time to obtain an operational picture. This, means that even when a node appears to be offline when queried for information, the best available information should still be available. This scenario needs a model that is reliable, decentralized, and secure. A MANET overlayed with a P2P storage application with redundancy satisfies most of those requirements.

1.1 Research Goal

Evaluate a potential data sharing system to use a P2P DHT to maintain data availability in AN. The evaluation measures are bandwidth usage and successful query answers.

1.2 Methodology

Employ a MANET running a DHT to provide a reliable and decentralized database. Determine the availability of data throughout the network by querying nodes for given data. Determine the amount of replication that is allowable before network bandwidth is exceeded.

1.3 Limitations of the Study

The mobility model used for nodal movement is a random way-point model. The nodes are designed as vehicles doing a search, the mobility model is not an optimal model for searching. The lookup model for nodes to request data is a globally known variable that is accessible by nodes in NS3. In reality, nodes would not know what data is known at each nodes with out asking the node first. In NS3 the antenna range is an ideal model that has a very specific cutoff and interference range. Once again in reality, this range is constantly changing depending on power levels and environment. Also, the frequency interference range is much further than effective communication range of a signal [19].

1.4 Thesis Organization

This thesis is organized as follows: Chapter II contains is an overview of current research into DHTs, including Chord, Pastry, Tapestry, and CAN. The chapter discusses some of the deficiencies of mobile DHTs, and some possible optimizations. Next, several data replication architectures are examined, ending with an overview of several current VANET projects. Chapter III presents a testing plan for the model, including system setup, procedures, workload, parameters, and design. Chapter IV discusses the research findings

in network utilization and replication. Finally, Chapter V presents the research conclusions and presents recommendations for future work.

II. Related Works

A mobile P2P database is a database that is stored in the peers of a mobile P2P network. A common storage structure for the P2P database is a DHT. A DHT provides a lookup service similar to a hash table, that operates on a P2P network in a decentralized distributed manner. Data in a DHT consists of key and value pairs with the goal that any participating node can efficiently retrieve the value associated with a given key. A DHT is able to scale to extremely large numbers of nodes and to handle continual node arrivals, departures, and failures [37]. A DHT can provide a high level of fault tolerance, and several degrees of lookup speed.

There are several reasons to choose a P2P overlay for implementing a database. P2P networks can be self maintaining, resilient, and require limited infrastructure and control [36]. A P2P network can be decentralized, allowing for high reliability in the sense that failure at a central site will not render the system unavailable [39]. P2P networks can be configured to allow for dynamic member arrival and departure. All of these reasons support the goal of creating a data sharing system that is dependable, distributed and decentralized.

2.1 Distributed Hash Table (DHT)

There are two big considerations when designing a DHT: availability of data and scalability from a few nodes to many hundreds [28]. There are also four core requirements that must be satisfied for a DHT. The first requirement is that every node can find the answer to a query. The second is that keys are load-balanced among nodes. Load balancing distributes a workload across multiple computers, nodes, network links, central processing units, disk drives, or other resources, to achieve optimal resource utilization, maximize throughput, minimize response time, and avoid overload [46]. With a DHT, this means ensuring that one node does not contain too many key and value pairs and become a pseudo-

central server that could become overloaded with data requests. The third requirement is that routing tables must adapt to node failures and arrivals. This does not refer to the network layer of the Open Systems Interconnection (OSI) model that routes packets from node to node. This refers to the DHT's routing tables which are commonly referred to as finger tables. Finger tables store a node's view of where the requested data is stored. The fourth requirement is actually more of an optimization for a DHT; it concerns how many hops lookups must take. This can be as simple as *O(n)* where every node is sequentially queried to *O log(n)*, or *O(1)*. These schemes will be discussed with each individual DHT. Four of the original DHTs are Chord [52], Pastry [48], Tapestry [62], and Content Addressable Network (CAN) [45].

2.1.1 Chord.

Chord [4] supports just one main operation: given a key, it maps it to a node. Chord uses what is called consistent hashing to assign keys to nodes. Consistent hashing is designed to facilitate joining and leaving the network. The consistent hashing algorithm assigns each node and key a *m* bit ID (128 bits for Chord). Chord uses the SHA-1 hashing function to generate ID's. The algorithm hashes the IP address of the node to generate a unique Node ID. Generating a unique node ID is done to avoid a naming structure that needs to rely on another application such as DNS. The key ID is hashed from the total file. The algorithm then maps keys to the correct successor node. That node is then responsible for that key. A small example can be seen in Figure 2.1. A file is hashed to a key-K1. K1 would attempt to map to N1. Since there is no N1, K1 sequentially counts up looking for the next node. It sees that N3 is the next node and the file K1 is stored on N3 in the figure. in the figure, N5 is responsible for no files, N12 is responsible for files K6 and K12, N24 is responsible for storing K13, K15, K18, and K21. SHA-1 is used to try to avoid the unbalanced loading as depicted on N24.

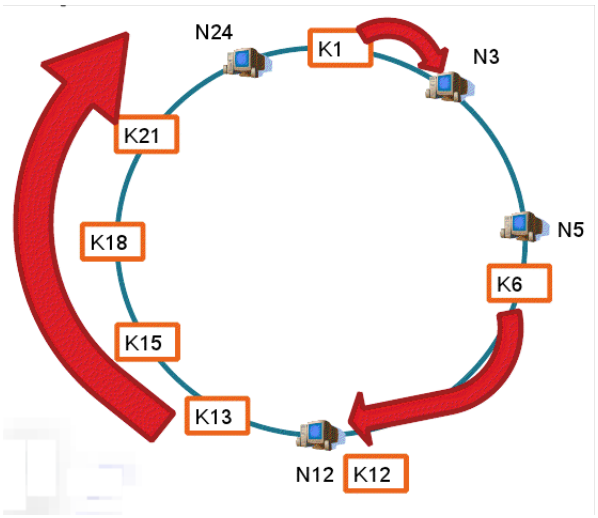

Figure 2.1: Chord Node/Key Example [4].

Chord uses only a minimal amount of routing information. Each node is responsible for tracking *O log(n)* nodes in what is called a finger table. The finger table allows a node to halve the distance each time it forwards a request for data. The finger table uses Equation (2.1) to calculate the successor nodes. Where n is a node's numerical ID number, set from SHA-1, i is the index number in the finger table (this number is iterated up from 0 to i), and m is the number of bits used for the ID (usually 128 bits from SHA-1).

$$S = (n + 2^{i-1}) \mod 2^m \tag{2.1}$$

When a node in Chord locates data, it requires only *O(log N)* time to lookup and find a node. This is because the finger table allows a node to halve the distance each time it forwards a request for data. An example lookup is shown in Figure 2.2.

During Chord initialization, each node generates a hash ID. After a nodes ID has been hashed, it searches for the next higher and previous lower IDs. It then populates its finger table and predecessor node (pointer). Next, the node must tell its predecessor nodes to

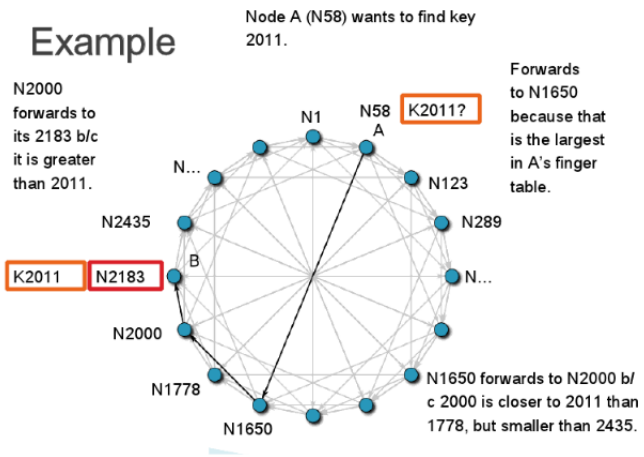

Figure 2.2: Chord Key Lookup Example [4].

update its fingers, and notify the higher layer software to transfer files associated with keys that the new node is responsible for.

2.1.2 Tapestry.

Tapestry [62] is a P2P overlay network which provides a DHT, routing, and multicasting infrastructure for distributed applications. The Tapestry P2P system offers efficient, scalable, self-repairing, location-aware routing to nearby resources. Each node is assigned a unique node ID uniformly distributed in a large identifier space. Tapestry uses SHA-1 to produce a 160-bit identifier space represented by a 40 digit hex key. Application-specific endpoints, called Globally Unique IDs (GUIDs), are similarly assigned unique identifiers. Node IDs and GUIDs are roughly evenly distributed in the overlay network with each node storing several different IDs. From experiments it is shown that Tapestry efficiency increases with network size, so multiple applications sharing the same overlay network increases efficiency. To differentiate between applications, a unique application identifier is used.

Tapestry uses best-effort to publish and route objects. Using Plaxton [56] routing uses local routing maps at each node, called *neighbor maps* to incrementally route over-lay

messages to the destination ID digit by digit (e.g., ***8 ⇒ **98 ⇒ *598 ⇒ 4598 where *'s represent wildcards). This approach is similar to longest-prefix routing in the Classless Inter-Domain Routing (CIDR) IP address allocation architecture. In Plaxton, each node or machine can take on the roles of servers (where objects are stored), routers (which forward messages), and clients (origins of requests). A server S publishes that it has an object O by routing a message to the root node of O. The root node is a unique node in the network used to place the root of the embedded tree for object O. The publishing process consists of sending a message toward the root node. At each hop along the way, the publish message stores location information in the form of a mapping <O.ID, S.ID>.

The Plaxton location and routing system provides several desirable properties for both routing and location:

- *Simple Fault Handling* Because routing only requires nodes match a certain suffix, there is potential to route around any single link or server failure by choosing another node with a similar suffix.

- *Scalable* It is inherently decentralized, and all routing is done using locally available data. Without a point of centralization, the only possible bottleneck exists at the root node.

- *Exploiting Locality* Queries for local objects are likely to quickly run into a router with a pointer to the objects location

- *Proportional Route Distance* Plaxton has proven that the total network distance traveled by a message during both location and routing phases is proportional to the underlying network distance, assuring us that routing on the Plaxton overlay incurs a reasonable overhead.

2.1.3 Pastry.

Pastry [48] is a scalable, distributed object location and routing substrate for wide-area peer-to-peer applications. Pastry is completely decentralized, fault-resilient, and reliable. Pastry shares many similarities with Chord. Each node in Pastry is assigned a 128-bit node ID. Similarly to Chord, the node ID is used to indicate a node's position in a circular node ID space, which ranges from 0 to $2^{128} - 1$. The node ID is assigned randomly when a node joins the system. It is assumed that node IDs are generated such that the resulting set of node IDs is uniformly distributed in the 128-bit node ID space. For instance, a node ID could be generated by computing a cryptographic hash of the node's public key or its IP address.

Pastry can route to the numerically closest node for a given key in *O(log N)* time under normal operation. Pastry like Tapestry adopts the Plaxton routing algorithm. For the purpose of routing, node IDs and keys are treated of as a sequence of digits with base 2^b. Pastry routes messages to the node whose node ID is numerically closest to the given key. This is accomplished as follows. In each routing step, a node normally forwards the message to a node whose node ID shares with the key a prefix that is at least one digit (or bit) longer than the prefix that the key shares with the present node's ID. If no such node is known, the message is forwarded to a node whose node ID shares a prefix with the key as long as the current node, but is numerically closer to the key than the present node's ID. To support this routing procedure, each node maintains some routing state.

Each Pastry node maintains a *routing table*, a *neighborhood set* and a *leaf set*. Each entry in the routing table contains the IP address of one of potentially many nodes whose node ID has the appropriate prefix; in practice, a node is chosen that is close to the present node, according to a proximity metric.

The neighborhood set *M* contains the node IDs and IP addresses of the *M* nodes that are closest (according to the proximity metric) to the local node. The neighborhood set is not normally used in routing messages; it is useful in maintaining locality properties.

The leaf set *L* is the set of nodes with the *L/2* numerically closest larger node IDs, and the *L/2* nodes with numerically closest smaller node IDs, relative to the present node's node ID. The leaf set is used during the message routing.

2.1.4 Content Addressable Network.

Similarly to the previous DHTs, CAN [45] is designed to be scalable, fault tolerant, and self-organizing. CAN's design centers around a virtual *d*-dimensional Cartesian coordinate space on a *d*-torus (similar to the circular topology of the other DHTs). This coordinate space is completely logical and bears no relation to any physical coordinate system. At any point in time, the entire coordinate space is dynamically partitioned among all the nodes in the system such that every node owns its individual, distinct zone within the overall space.

A node learns and maintains the IP addresses of those nodes that hold coordinate zones adjoining its own zone. This set of immediate neighbors in the coordinate space serves as a coordinate routing table that enables routing between arbitrary points in this space. Routing in CAN works by following the straight line path through the Cartesian space from source to destination coordinates. Note that many different paths exist between two points in the space and so, even if one or more of a node's neighbors were to crash, a node can automatically route along the next best available path.

To allow the CAN to grow, a new node that joins the system must be allocated its own portion of the coordinate space. This is done by an existing node splitting its allocated zone in half, retaining half and handing the other half to the new node. The process takes three steps:

1. Find a node already in the network.

2. Identify a zone that can be split.

3. Update the routing tables of nodes neighboring the newly split zone.

2.2 Mobile Distributed Hash Table

The previously discussed DHTs were designed to work on wired networks. Transitioning to a wireless domain adds several complexities. Heer, et al. [24] concludes that one major aspect is the interaction between two routing systems: the ad-hoc routing protocol and the DHT routing algorithms. Since both systems attempt to handle topology changes and node failures concurrently, their interactions often result in suboptimal routing at high costs in terms of consumed network and host resources. Also, wireless ad-hoc routing of messages between two arbitrary nodes is an extremely expensive operation, that may involve flooding of the network to find a path or global state updates when the topology changes due to node mobility. Furthermore, each additional hop consumes significant resources at intermediate nodes. Another aspect is the high churn (nodes joining and leaving) rate DHTs experience in MANETs, leading to DHTs being separated and reunited network behavior atypical of infrastructure-based networks and hence rarely addressed in the original DHT algorithms.

These implications of running DHTs on MANETs make it necessary to adapt DHTs for efficiency and stability. Heer, et al. [24] propose that the integration of the DHTs and ad-hoc routing layers is crucial to achieve efficiency, as it significantly reduces the duplication of routing information and maintenance efforts. Before moving on to optimizations and enhancements for mobile DHTs Hoschek [26] presents an abstract Peer Database Protocol (PDP) Messaging Model for a unified P2P database protocol that each mobile database should implement at a basic level.

2.2.1 Peer Database Protocol Messaging Model.

The abstract PDP messaging model employs four request messages (QUERY, RECEIVE, INVITE, CLOSE) and a response message (SEND). A *transaction* is defined

as one or more message exchanges between two nodes for a given query. An example transaction is a QUERY-RECEIVE-SEND-RECEIVE-SEND-CLOSE sequence.

- QUERY - A QUERY message is forwarded along node hops through the P2P node topology. The message contains the query itself as well as a transaction identifier. The QUERY message also contains scope hints such as a loop timeout, abort timeout, radius and a neighbor selection query. A node accepting a QUERY message returns immediately without any results. Results are explicitly requested via a subsequent RECEIVE message.

- RECEIVE- A RECEIVE message is used by a client to request query results from another node. A client can successively issue multiple RECEIVE messages until the result set is exhausted.

- SEND- When a node accepts a RECEIVE message, it responds with a SEND message.

- CLOSE- A client may issue a CLOSE message to inform a node that the remaining results (if any) are no longer needed and can safely be discarded. A CLOSE message responds immediately with an acknowledgement. At the same time, the node asynchronously forwards the CLOSE to neighbors involved in result set delivery, which in turn forward the CLOSE to their neighbors, and so on. Being informed of a CLOSE allows a node to release resources as early as possible.

- INVITE- INVITE messages only apply to Direct Response, which are not covered here.

Now that a basic protocol model has been established, there are two deficiencies in mobile DHTs. The first is multi-hop searching for data, this adds a larger latency for lookup. The second is the overlap between what is needed by mobile DHTs and

network layer routing protocols, and integrating the two. Below is the current research and optimizations to overcome these deficiencies.

2.2.2 Decreased Hop Count of a Distributed Hash Table.

One optimization is to increase the size of a node's routing table, this decreases the latency of multiple hops for a query to arrive at a destination node [33]. Ignoring the network layer routing, if a node has a large enough routing table it could directly query the node for data allowing for constant lookup. This may help at the network layer because a node could perform a complete *transaction* before a node moves out of range or is otherwise lost.

2.2.2.1 UnoHop.

UnoHop [51] is a DHT that reduces data lookups to *O(1)*. UnoHop propagates membership changes through a distribution mechanism that has each node in the P2P network maintain routing tables with complete membership information. Unohop works most effectively with a low churn rate. Sitepu, et al. [51] the creators of Unohop, argue that unlike the traditional DHTs (Chord, Pastry, etc) where each node maintains minimal information about its neighbors in the routing table. When the churn rate is low it is more efficient for each node to maintain complete membership information to allow very fast *O(1)* lookup time. Testing of UnoHop showed that for a large MANET constant lookup time and one hop routing tables is likely not plausible, and therefore a hop count latency optimization is also not plausible.

2.2.2.2 Mobile Chord.

Cramer and Fuhrmann [14] performed an evaluation of Chord in a MANET. They determined the main issue of deploying Chord in a MANET not to be its overhead, but rather the protocols timeout and failover strategy. This strategy enables fast lookup resolution in spite of highly dynamic node membership, which is a significant problem in the Internet context. However, with the inherently higher packet loss rate in a MANET, a

failover strategy results in lookups being inconsistently forwarded even if node membership does not change.

Cramer and Fuhrmann's [14] model consisted of nodes using the IEEE 802.11 MAC with RTS/CTS extension, and a 2.4-GHz radio interface with a transmission speed of 2 Mbps. Radio propagation follows the two-ray model with a nominal transmission range of 250 meters. Dynamic Source Routing (DSR), Optimized Link State Routing (OLSR), and Ad-hoc On-Demand Distance Vector (AODV) were used as the three routing protocols. All routing protocols were configured with the default values of their parameters.

The metrics used were:

- *Request success ratio* The total fraction of requests for which the originator successfully receives an answer.
- *Overlay consistency* The percentage of all nodes that from a global point of view have selected their correct successor. Consistency is a function of time.
- *End-to-end delay* The cumulated delay between issuing a request and receiving the reply. This includes both the delay of performing the Chord lookup and of the request/response cycle.
- *Total network load* The total load exerted on the network. In contrast to the offered load, it comprises application and control traffic, and each transmission on a multi-hop route is separately counted.

Cramer and Fuhrmann [14] concluded that in the majority of all tested scenarios, which varied in network size, node mobility, and offered application load, they found that Chord's ability to consistently resolve lookups was significantly impaired. Notwithstanding, the observed performance issues unexpectedly were not a result of congestion. Upon packet loss due to mobility or erroneous transmission, the protocol's

failover strategy prematurely assumes that the packets destination departed from the network. It disposes of the corresponding state information and thereby lowers network consistency. Lookups are inconsistently resolved, resulting in incorrect application behavior.

2.2.2.3 P4P Optimization for P2P Pastry.

The P4P [22] Pastry routing algorithm, performs peer clustering to the same resources and the proximity of physical location of nodes. When routing, the node routes to the nodes which are physical adjacent to it and have the lowest cost of communication. Theory analysis and the experimental results show that the algorithm proposed has realized localization download and greatly raised the data transfer rate, reduced the load in backbone network and enhanced the network performance.

2.2.2.4 NN-Chord.

NN-Chord [10] is Neighbors Neighbor Chord, and the algorithm is named BNN-Chord (Bidirectional Neighbors Neighbor Chord). The routing lookup algorithm of classical Chord is unidirectional; it only can look up key through the clockwise direction of Chord circle. However, the structure of Chord is circular in logic, so the lookup direction can be clockwise or anticlockwise, lookup efficiency can not be high only through clockwise direction. The NN-Chord algorithm extends the finger table using neighbors neighbors link, which is named a learn table. The routing table maintains the information of the successors successor node, which can reasonably increase finger density of the routing table to find the neighbor node, that is closer to the key. The learn table of NN-Chord algorithm only covers approximately three quarters of Chord circle, and the finger table suffers serious information redundancy, meanwhile, this algorithm only can process unidirectional lookup. For solving these issues and with the idea of bidirectional Chord, BNN-Chord add a converse routing table (including converse finger table and converse learn table) for every node in the NN-Chord, by which the routing table can cover

whole Chord circle and lookup direction can be clockwise or anticlockwise. Meanwhile, redundant information is delete in the routing table to simplify it. Simulation results show that the algorithm effectively reduces the search path length and can improve the system performance

2.2.3 Intergrated DHTs in a MANET.

Das, et al. [16] explain the difference between a layered approach and integrated approach to implementing a MANET DHT using Pastry. The layered design is similar to implementing a DHT in the Internet. It leverages the existing routing infrastructure for MANETs to the full extent. This approach, while consistent with the layered principle of the OSI model of networking, makes it difficult to exploit many optimization opportunities from the interactions between the DHT protocol and the underlying multi-hop routing protocol. For example, it is difficult for the routing structures of the DHT and the route cache of DSR to coordinate with each other to optimally discover and maintain source routes. Instead of stacking one protocol on the top of the other, with no information sharing, more work can be done to make both P2P file sharing protocol and MANET routing protocol interact with each other [53]. Experiments in NS2 shows that this strategy performs better than the layered approach in terms of traffic, average query delay and packet delivery ratio.

The NS2 implementation follows a simulation area with a grid of 1500 meters by 320 meters. The area is divided into 10 sub-grids of 150 meters by 320 meters with a super-node placed at the center of each sub-grid. The random waypoint movement model is used in which 50 nodes move at a speed uniformly distributed between 0-20 m/s. We assume the wireless bandwidth is 2 Mbps and the transmission range is 250m. The run time of the experiment is kept as 500 second. Three metrics are measured: average delay, message over- head and packet delivery ratio. The integration of P2P file sharing and AODV routing is done in current version (2.27) of NS2. There are 100 data items randomly distributed

among all 50 nodes. In the P2P model, each peer can play the role of both a client and server.

2.2.3.1 Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile ad-hoc Networks.

In Ekta [16], [43], the DHT abstraction is supported by integrating Pastry and DSR at the network layer of MANETs via a one-to-one mapping between the IP addresses of the mobile nodes and their node IDs in the namespace. With this integration, the routing structures of a DHT and of a multi-hop routing protocol, DSR, are integrated into one structure which can maximally exploit the interactions between the two protocols to optimize the routing performance.

Ekta explores two disparate design options: the simple approach of directly overlaying a DHT on top of a MANET multi-hop routing protocol, and the Ekta approach which integrates a DHT with a multi-hop routing protocol at the network layer. Second, Ekta examines the efficiency of DHT substrates in supporting applications in MANETs by examining the performance of a resource discovery application built on top of Ekta with one that directly uses physical layer broadcast. Ekta's implementation uses mobility scenarios that are generated using a modified waypoint model. In the model, 50 nodes move at a speed uniformly distributed between 1-19 m/s in an area of 1500m x 300m. A wireless radio with 2 Mbps bit rate and 250m transmission range is used. DSR is used as the routing protocol. The simulation duration chosen is 900s. The communication pattern consists of 40 trafc sources, each initiating packets at the rate of 3 packets/second. Each packet has a 48-byte message body, prepended with a 128- bit key generated from hashing the message body. Thus, the effective packet payload is 64 bytes. This communication pattern models the trafc in a DHT-based storage system such as PAST.

2.2.3.2 Cell Hash Routing.

Cell Hash Routing (CHR)[3] is a DHT specifically designed for wireless ad-hoc networks. CHR was built to be an integrated design. CHR uses an inexpensive localized cell-based clustering method that groups nodes according to their location. This clustering method groups nodes inside cells of predefined and globally known shape. As a consequence, messages are not addressed to an individual node destination, because routing works at the cell level. Such an approach is particularly well-suited to the world of small and simple wireless devices or embedded systems, where nodes may look for specific contents and not for peers.

The advantage of clustering is twofold. First, it creates a very structured and sparsely populated network of clusters, where a lightweight routing scheme can be applied. Second, the efficiency of the routing scheme is not affected by increasing node density. Furthermore, routing scales well with increasing network sizes, because it is localized. Hence, by using a location-based clustered approach, routing in CHR is scalable with respect to both, network size and node density. Although simple, this scheme is powerful and enables a DHT to be implement in a straightforward and efficient way. The DHT implemented by CHR fits wireless ad-hoc environments and can be used as a component of more complex architectures.

2.3 Data Replication

In certain P2P systems[40] such as Gnutella, only nodes that request an object make copies of the object. Other P2P systems such as Freenet allow for more proactive replications of objects, where an object may be replicated at a node even though the node has not requested the object. For such systems, how many copies of each object should there be so that the search overhead for the object is minimized, assuming that the total amount of storage for objects in the network is fixed? Answers to this question have implications to non-proactive replication systems as well, because the information of an

objects location could be proactively replicated to expedite the searches. There are two replication strategies that are easily implementable. One is "owner replication", where, when a search is successful, the object is stored at the requester node only. The other is "path replication", where, when a search succeeds, the object is stored at all nodes along the path from the requester node to the provider node. Owner replication is used in systems such as Gnutella. Path replication is used in systems such as Freenet.

2.3.1 Beehive.

Beehive [44] is a general replication framework that operates on top of any DHT that uses prefix-routing. Beehive controls the extent of replication in the system by assigning a replication level to each object. An object at level i is replicated on all nodes that have at least i matching prefixes with the object. Queries to objects replicated at level i incur a lookup latency of at most i hops. Beehive's implementation is structured as a transparent layer on top of FreePastry 1.3, and supports an insert/modify/delete/query DHT interface for applications, and requires no modifications to the underlying Pastry implementation.

Beehive's replication protocol proactively makes copies of the desired objects around the network. Initially, each object is replicated only at the home node at a level $l = 0$. The highest level available k, where $k = log_b N$, where N is the number of nodes in the system and b is the base of the DHT, and shares k prefixes with the object. If an object needs to be replicated at the next level $l + 1$, the home node pushes the object to all nodes that share one less prefix with the home node. Each of the level $l + 1$ nodes at which the object is currently replicated may independently decide to replicate the object further, and push the object to other nodes that share one less prefix with it. Nodes continue the process of independent and distributed replication until all the objects are replicated at appropriate levels. An example can be seen in Figure 2.3.

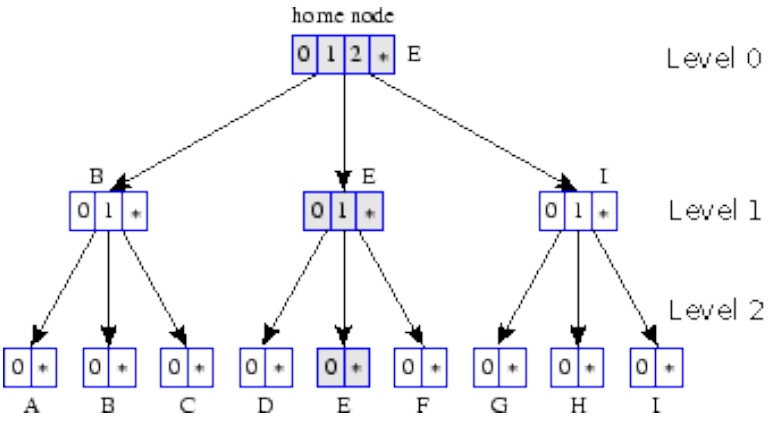

Figure 2.3: Beehive Replication Level [44]

2.3.2 Tempo.

Tempo [50] explores a proactive approach to data replication that creates additional copies not in response to failures, but periodically at a fixed low rate. This way a failure does not spike and overwhelm application traffic and make it difficult to provision bandwidth. In simulations based on PlanetLab traces, Tempo is able to provide the same level of durability as traditional reactive systems using a comparable amount of bandwidth and without significant fluctuations in bandwidth usage.

2.3.3 Max-Cap, practical load balancing for content requests in peer-to-peer networks.

Max-Cap [47] is decentralized algorithm, based on the maximum inherent capacities of the replica nodes. Unlike previous algorithms, it is not tied to the timeliness or frequency of updates, and consequently requires significantly less update overhead. Yet, Max-Cap can handle the heterogeneity of a peer-to-peer environment without suffering from load oscillations. This paper studies the problem of balancing the demand for content in a peer-to-peer network across heterogeneous peer nodes that hold replicas of the content.

Max-Cap assume a peer-to-peer overlay network of widely distributed nodes. The peers store and share content with other peers and are heterogeneous in their capacity to

serve content. The placement of a file on a particular peer is decided by the owner of the peer, not by a global placement policy. That is, there is no control over where replicas of a particular file are placed. Finally, the set of participating peers in the system is dynamic as peers enter and leave the system continuously and content availability at a peer can last for as little as a few minutes.

2.3.4 Chord.

Data redundancy has the benefit of improving data availability in Chord, but taking more memory to store. To increase data availability Kapellko [30] advocates the direct union of two copies of Chord, which he claims is a very soft modification of the original Chord protocol. Unexpected departures of $k < O(\sqrt{n})$ nodes from the system has a high probability that no information disappears. Only after $\sqrt{n} = O(k)$ unexpected departures, does the loss of some data occur with high probability.

2.4 Vehicular Ad-Hoc Network

A VANET [31], [55] is a technology that uses moving cars or other vehicles as nodes in a network to create a mobile network. Rather than moving at random, vehicles tend to move in an organized fashion. Cars that move in the opposite direction increases the data dissemination performance significantly [42]. VANETs have some common properties with MANETs, but there are some distinctive differences [38]: 1) Vehicles move at very high speed but stop at intersections when traffic lights turn red. 2) Since the route is restricted by the roads, the mobility is more predictable than the to random movement of nodes in a MANET.

2.4.1 VANET Chord.

For vehicular mobile Chord, Liu, et al. [38] propose several mechanisms to overcome problems caused by node mobility and topological changes.

1. Aggressive table update: try to use any available information to update finger table (for Chord) and knowledge table (for Mobile Chord).

2. Overlay table broadcasting: broadcast P2P overlay information to neighboring Mobile Chord nodes instead of using unicast ping for keep-alive mechanism in Chord

3. Passive bootstrapping: a new node learns the P2P overlay network information by listening to overlay table broadcasting rather than joining P2P overlay through hook nodes.

Liu, et al. [38] concluded Chord works fine in low-mobility, low traffic load, and low P2P node count scenarios, and modified Mobile Chord scheme performs well in the challenging VANET environment. The P2P overlay consistency was significantly higher than the baseline Chord. The proposed bootstrapping technique reduced control signaling cost and improved P2P network convergence time in highly dynamic vehicular environments. In summary, Mobile Chord outperforms Chord in terms of overlay consistency, the number of application layer forwarding steps, query response, correct query response, and the average query delay.

2.4.2 CarTALK 2000.

CarTALK 2000 [54] is a European project that focuses on new driver assistance systems which are based upon inter-vehicle communication. The main objectives are the development of co-operative driver assistance systems and the development of a self-organizing ad-hoc radio network as a communication basis with the aim of preparing a future standard.

Based on the current position and its driving destination, a vehicle movement is predictable with the help of a digital road map. For this reason CarTALK 2000 uses Spatially Aware Routing (SAR). SAR extends traditional position based routing by making use of the spatial environment model. The spatial model is constructed based on road

topology information extracted from digital road maps, which can be internally represented as a graph *G(E,V)* consists of a set *V* of vertices referring to road intersections together with a set *E* of edges denoting road segments. The weight of edges can be used to represent different characteristics, such as the road length, average speed, etc. SAR then relies heavily on GPS coordinates to find the optimal route for a packet to take. More in depth investigations are still needed in order to proceed with specific technical choices and to reach the final routing design for CarTALK 2000.

2.4.3 Fleet Net.

Fleet Net [23] is a project to bring the internet to vehicles on the road by using a VANET. Fleet Net is based on cellular communication systems with centralized organization adapted to an ad-hoc environment. The typical Fleet Net network topology will have a stationary gateway node, such as a gas station or restaurant, that is connected to the internet. Then, vehicles will then forward packets away from the gateway node to out of range nodes. An example can be seen in Figure 2.4. Fleet Net's network topology is similar to Data Mules [49], except in Data Mules the nodes buffer sensor data, then move to a wired access point to transmit data.

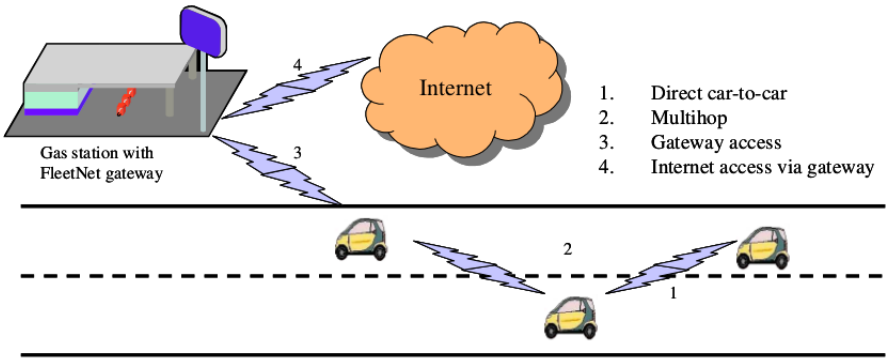

Figure 2.4: Fleet Net Scenario [23]

2.4.4 Grassroots.

Grassroots [20] is a program in which each vehicle contributes a small piece of traffic information to the network based on the P2P paradigm, and each vehicle aggregates pieces of the information into a useful picture of the local traffic information. Grassroots is different from the usual sensory environment since it relies on mobile sensors (dataflies) rather than on a fixed predefined infrastructure. Two important characteristics of Grassroots environment: redundancy, which imparts robustness, and dynamic nature of coverage, which changes with the location of its dataflies. The Grassroots architecture has four entities: information producers, information consumers, information aggregators, and information relayers. Producers are vehicles (dataflies) that collect raw sensory data. Consumers are objects (stationary or mobile) that query data (e.g., vehicles). Aggregators collect and aggregate data from multiple producers. An aggregator is a logical entity that can reside on a stand-alone server or can even reside on a consumer. Relayers participate in forwarding messages to their intended destination. Simulation results show promise in the Grassroots system.

2.5 Summary

This chapter contained a broad overview of the big four DHTs, Chord, Pastry, Tapestry, and CAN. Each DHT has advantages and disadvantages associated with it. Mobile DHTs have deficiencies that can cause slower lookup or redundant work, but there are optimizations and integrations available to mitigate inefficiencies. There are several data replication architectures available for DHTs. Once again each strategy has advantages and disadvantages associated with it. VANETs create their own intrinsic problems with generally fast moving nodes and movement patterns.

For this research Pastry is chosen as the DHT because of its availability and completeness in Network Simulator 3 (NS3). Chord and Pastry already have an implementation in NS3. The Chord implementation is poorly documented and difficult to

send a message. The Pastry implementation is much more detailed and easier to understand. Also, the Beehive replication strategy is chosen because Beehive is a general replication framework that operates on top of any DHT that uses prefix-routing, such as Chord, Pastry, and Tapestry. Therefore, Beehive could be implemented on top of either of the DHTs in NS3. The problem with Beehive is that its source code is not publicly available. However, there are papers that describe Beehive and a dedicated webpage.

III. Methodology

The goal of highly mobile ad-hoc networks is to investigate the applicability of P2P database replication to maintain data, testing the network bandwidth breaking point, and the effects of node density, churn, and area. This chapter describes the testing setup and procedures for the experiment, the system services, performance measures, and expected workload, system parameters, and experimental design. The testing setup describes the model used for the experiment. The testing procedures describes the implementation in NS3. The system services explains the what the system does. The performance measures detail the metrics used to evaluate the system. The expected workload describes the expected queries of the DHT. The system parameters list the parameters and factors of the system. Finally, the experimental design explains the final system overview.

3.1 Testing Setup Used

The approach tests the applicability of an ad-hoc P2P self replicating DHT network. The evaluation technique used is simulation, using NS3. The simulation uses the IEEE 802.11 standard for wireless communication. Nodes are mobile and move at a fast pace of 60 mph, to simulate slow moving helo type vehicles such as the (A160T) Hummingbird [7]. The nodes move using a random way-point model [5]. The vehicles travel for a random distance, pause, then travel in another direction. Evidence suggests [60] that random way points are not optimal for vehicle searching as there is usually much overlap. Still, this model comes readily available, and provides a baseline for research.

The P2P network uses Pastry with Beehive. A node replicates data and sends the data to other nodes in the network. The known data is then queried for availability. Chord has already shown that with no data replication, linear node failure results in linear data loss [52].

3.2 Test Procedure

The simulation starts with a designated number of nodes. The nodes use the NS3 implementation of standard IEEE 802.11b protocol with a constant data rate of 1 Mbps. Nodes have a constant speed propagation model applied to them which means all packets will travel at the same speed. By default, the channel model is set with a propagation delay equal to a constant, the speed of light. Nodes have a range dependent propagation loss model meaning that nodal contact is dependent on distance, with an antenna range of 100 meters [2]. In NS3, the range propagation loss model uses one attribute ”Max Range”. The single ”Max Range” attribute(units of meters), synonymous with the antenna range, determines path loss. Receivers at or within ”Max Range” meters receive the transmission at the transmit power level. Receivers beyond MaxRange receive at power -1000 dBm (effectively zero).

The nodes move to random way points, with a speed of 60 mph (26.8224 m/s), and a constant pause of 5 seconds. Optimized Link State Routing (OLSR) is the underlying network routing protocol. Each node is sequentially assigned a static IPv4 address starting at 10.1.0.1. The underlining database and DHT used for this experiment is Pastry. A Pastry application is installed on every node. Pastry considers a neighbor dead after two missed “Hello” messages. After sending a request, Pastry makes two attempts before considering the neighbor dead as well. Request are set to time out after 1,000 ms, and have a time to live of 255 hops.

Each node’s DHT database initializes with 10 entries. Each entry is a random number from 0 to 32,767. This means that it is possible for a duplicate data entry on two or more nodes. Assuming 500 nodes choosing 10 entries, 5,000 randomly chosen numbers, according to the birthday paradox it is almost 100% certain that there will be matching numbers, but since the data is duplicated to nodes after the first data replication, the impact of this scenario is minimal. When a node duplicates its data entry over the network. The

node sends its data identifier, along with data padding to have a data object that appears to be of a size equivalent to sensor data of interest going over the network.

For testing purposes, all entries are stored in a global table that tracks the data and which node has this data. This is done so that nodes know what data to request. Although it is possible to have scenarios where nodes request data that has never existed, that is not the case for this experiment. However, it is possible in this experiment for a node to request data that no longer exist. This occurs when one node has the only copy of a data item, and goes offline before replicating its data. This means that the data is in the global table, but no longer exists. This is a highly unlikely scenario.

Data is replicated following the Beehive replication strategy [44] with the level of prefix matching specified from 1 - 4, with 1 as the lowest replication rate and 4 the highest. For this implementation, prefix matching is done with the nodes number. Assume the first node, although it is node one, in binary it is node zero. At replication level one, node one would then prefix match with node two, because all except for their last binary digit matches. At replication level two, nodes one, two, three, and four would prefix match, because all except for their last two binary digits prefix match. This means when a node is replicating at level one it replicates to one other node. Level two to three other nodes, level three to seven other nodes, and level four to fifteen other nodes.

Each node is scheduled to replicate a piece of its own data once every second. Additionally, all nodes send out a query request for data once every ten seconds. When replying, a node only responds with an acknowledgment through the network that it has the requested data. It does not send another full 1024 byte packet.

3.3 System Boundaries

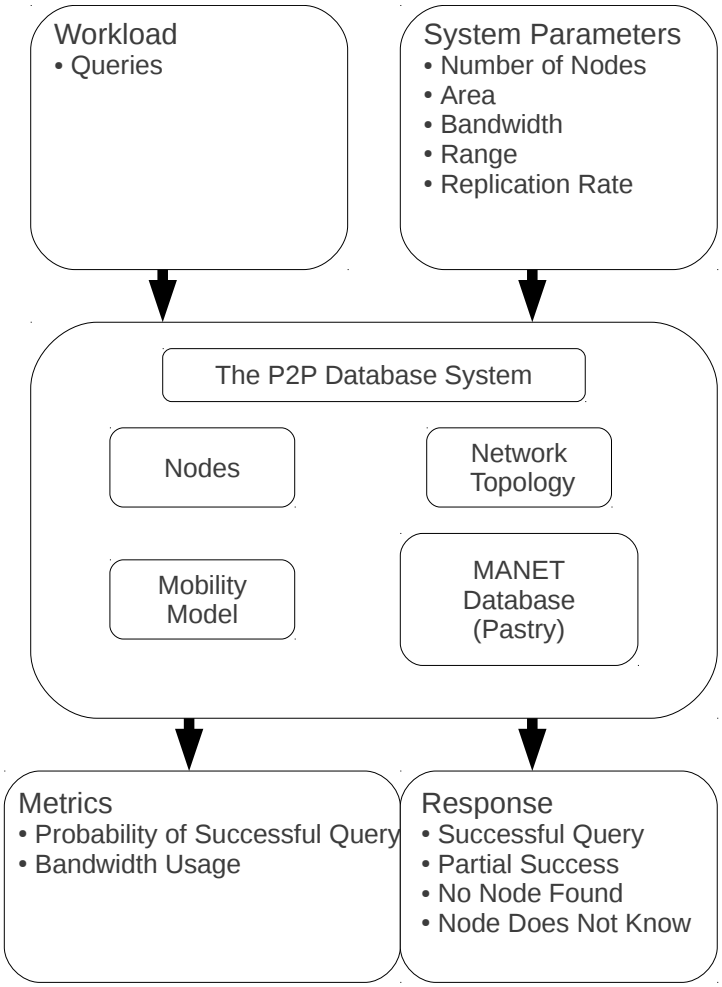

Figure 3.1: The P2P Database System Under Test.

3.4 System Services

The system offers one service: a distributed database. A successful outcome is defined as receiving the correct response to a database query. The response can come from either the original data producing node or another node that has replicated data.

There are several ways the system can fail. First, the requesting node cannot find a node containing the data in the network. Second, the network can lose the data request before being received by the destination node. Third, the network can lose the data response before the querying node receives the data.

3.5 Performance Measures

The first measure is the amount of bandwidth usage. For this experiment, *Bandwidth usage* is defined as all packets sent across the network including the data packets with headers. As well as lower layer protocol and control packets, such as routing. The total available bandwidth is 1 Mbps. This is measured to determine the amount of bandwidth nodes use to replicate data.

The second measure, *Successful Query Answer* is defined as the requested node receives the requested data. This measure defines the data availability of the system. For a given configuration, this provides a base performance of the system.

3.6 Workload

The expected workload for the database consists of request queries. A query is defined as a node requesting a piece of data from somewhere in the network. A query request is made by each node every ten seconds. The request is for random piece of data from a globally known table. The request is sent throughout the network.

3.7 System Parameters

Table 3.1 and Table 3.2 lists the parameters, test values, and a short description for the system. Table 3.1 presents parameters for the overall scenario including number of nodes,

area, churn rate, and the mobility model. Table 3.2 presents the nodal model parameters including the replication rate, bandwidth, packet size, and antenna range.

The first parameter is the number of nodes. The number of nodes refers to the total number of nodes per simulation. For this experiment the number of nodes ranges from 10 to 500 nodes in steps of 10. Table 3.3 shows the anticipated outcomes for the number of nodes in the system.

Table 3.1: Scenario Parameters.

System Parameter	**Tested Values**	**Description**
Number of Nodes	10-500	The total number of nodes in the network for a simulation.
Area of Simulation	Squares with lengths of 200m to 2,500m.	This is the total area nodes have to move.
Churn Rate	Lose and gain of 1 node every 30 mins.	The rate at which nodes are entering and leaving the network
Mobility Model	60 mph; 5 sec pause	The speed that a node will travel, and the time it will pause at that location before starting to travel again.

Table 3.2: Nodal Model Parameters.

System Parameter	**Tested Values**	**Description**
Replication Rate	1-4	The rate at which data is replicated to other nodes.
Bandwidth	1 Mbps	The rate at which a node's link can transmit and receive data.
Size of Data Packets	1024 bytes	This is the size of a replication data packet.
Antenna Range	100 m	The wifi antenna range of a node.

Table 3.3: Number of Nodes.

Number of Nodes	**Anticipated Outcome**
Low (10 - 20)	If the area is small, nodes should be able to maintain good connectivity, and there should not be much interference over the 2.4 GHz spectrum. On the other hand, a large area there may be minimal network connectivity, a higher replication rate should help relieve this problem.
Medium (30 - 100)	May produce some of the best results depending on an experiments configuration. Nodes will usually be in range of at least 1 other node, but not an overbearing number of nodes as to cause great interference. A node will likely be able to query and receive a response. A medium number of nodes may perform well for the amount of resources it cost to build.
High (110- 500)	If the area is large, nodes should be able to maintain good network connectivity with minimal interference over the 2.4 GHz spectrum. However, a small area may cause great interference with many nodes attempting to transmit at once.

Besides the size of a query response, the database replication rate has the largest impact on available bandwidth. A higher rate means more replication of data, but also more bandwidth use. The more replication, the greater the chance a node finds the data, being queried. The replication rate and anticipated outcome appear in Table 3.4.

The nodes move in a square area ranging from side lengths of 200 m to 2,500 m. Thus, the effective area of the simulation ranges from $40 km^2$ to $6,250 km^2$. The side lengths increase by 300 m, and then in 500 m steps until reaching 2,500 m.

One constant parameter is the amount of bandwidth available in the network. This parameter effects several elements. The first is the amount of data that can be traveling over the network. The second is how long nodes have to be in range of one another. The higher the bandwidth, the more data that nodes can push to each other in a given amount of time. For this experiment the bandwidth is set to a constant rate of 1 Mbps over a Direct-

Table 3.4: Replication Level.

Replications Level	**Anticipated Outcome**
4 Low	There will be minimal data replication. It is likely that duplicate data will not be found easily. However, bandwidth usage will be minimal for a constrained network.
3 Medium-Low & 2 Medium-High	The medium amount of network replication may produce a good outcome by having a moderate amount replication and bandwidth usage.
1 High	There will be lots of data replication. It is likely that duplicate data is found easily. However, bandwidth usage will be extremely high and will constrain the network. A query should be answered most every time.

sequence spread spectrum (DSSS) for both control and data packets. The bandwidth and network utilization will be measured by the built in NS3 flow-probe class.

3.8 Experimental Design

Each experiment simulates for five hours. Multiple trials with the same configuration are run with a different seed value at least ten times to obtain a viable 95% confidence interval. The overall network usage and number of satisfied queries, is then graphed.

3.9 Summary

This chapter details the highly mobile ad-hoc network and DHT used for this experiment. The system model is an ad-hoc P2P self replicating DHT network. The system replicates data to nodes and then queries all nodes for available data. The system is evaluated in NS3 running Pastry with a Beehive replication strategy.

IV. Results

If an environment is an active battlefield the space may include airborne assets, tanks, ground personnel, UAVs, missiles, satellites, and commanders all communicating on Internet Protocol based links. With such a high quantity of communication it is important to not only prevent enemy jamming, but to avoid self-inflicted jamming by crowding to many devices. Therefore, determining a network's bandwidth breaking point becomes important. Also, in the likely event that any member of the network goes offline for any reason, replication becomes the only way of retrieving what would be lost data.

The results of this experiment are broken down into to two categories, the *Bandwidth Usage* and *Successful Query Answer* of the system. The bandwidth utilization is an overview of the usage of the network. The replication rate is number of successful queries per all queries sent.

4.1 Bandwidth Usage

The following figures show graphs for a 95% confidence interval network utilization for a given node number, replication level and area of simulation. Starting with Figure 4.1, this figure shows the network usage on the vertical axis and the simulation area (square's side length) on the horizontal axis. Each of the four replication levels are shown as well. Replication level one is shown as a circle with a solid line connecting each circle. Level two is shown as a triangle connected with a dashed line between. Level three is a plus connected by a dotted line. Level four is a x connected by a dotted and dashed line. The 95% confidence interval is shown by dashed lines above and below each symbol.

Figure 4.1 shows Pastry running with ten nodes. Starting at 200 m by 200 m and replication level one, Figure 4.1 shows that on average approximately nine percent of the network is being utilized. Dissecting into what is happening, at level one each node

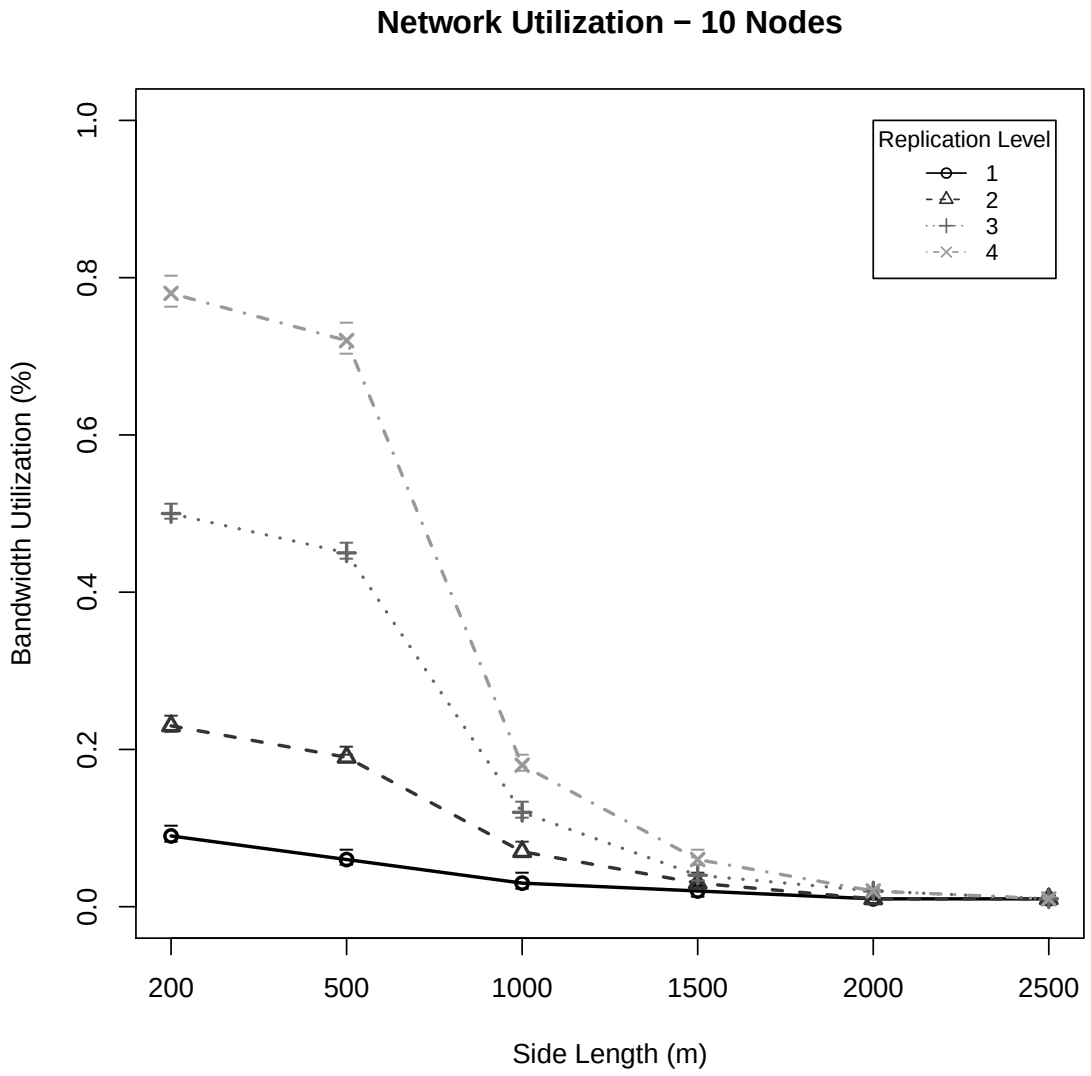

Figure 4.1: Bandwidth Utilization - 10 Nodes.

replicates to one other node that is a 1024 bytes being sent to one other node ten times approximates to 81,920 bits needing to be sent in one second. The simulator is set to send 1 Mbps, in an ideal scenario where all nodes were in range of each other, each send was perfectly timed, and there was no other overhead data or control packets the network would use 8.192% ($1024bits * 8 * 1 * 10/1,000,000bps = 0.08192$).

At level two each node replicates to three other nodes. That is 1024 bytes being sent to three other nodes ten times, approximates to 245,760 bits needing to be sent. However, due to the replication prefix matching nodes nine and ten only match to each other. This leaves two nodes that are only prefix match to each other. This same problem arises at a replication level three. The binary prefix of nodes one though eight match each other, but nine and ten only match to each other. Finally at level one all ten nodes binary prefix match each other, and attempt to replicate 1024 bytes (8192 bits) of data to all other nodes. Estimating 1024 bytes being sent to nine other nodes ten times approximates to over 737,280 bits needing to be sent in one second. Once again with the simulator set to send 1 Mbps, and an ideal scenario where all nodes were in range of each other, each send was perfectly timed, and there was no other overhead data or control packets the network would use 73.7%. But the simulation is not an ideal situation. Once again the network utilization accounts for all types of packets, and not all node's can reach one another directly in a 200 m by 200 m square with an antenna range of 100 m. Nodes not directly being able to reach one another affect several aspects of network utilization. First, it actually decreases a node's view of the network's utilization because a node that is 101 meters away from another node does not see any of the frequency interference caused by that node, because of the range propagation loss model used. The communication range uses an ideal model and interference would likely occur in a real world test. Second, nodes not able to see each other also increases data needed to flow through the network because a third node must now route the packet to the destination node. Third, the node may need additional routing packets to establish and maintain a viable route.

In Figure 4.1 the area of simulation has a large impact on the network utilization. The area of simulation increases each time. For example, the area of 500 m by 500 m is a quarter of a square kilometer. But at 1000 m by 1000 m the area is a one square kilometer, four times the area of the previous simulation area. Simply put, ten nodes with only a 100

m radius communication range are not at a high density enough inside a square kilometer to maintain contact with each other which is why network utilization drops.

In Figure 4.2 the number of nodes in the simulation has increased to twenty nodes. This increase increases the bandwidth usage of replication levels three and four. There are two reasons for this increase. The first is there is now twice as many nodes as there were previously. The second is that replication level four can now match the first sixteen nodes. This means that one node may be expected to replicate 131,072 bits itself. In a small area there is little doubt why twenty nodes are unable to effectively try and communicate.

With 100 nodes in a 200 m by 200 m space replication level one maximizes the network utilization (Figure 4.3). However, following Figure 4.3 a more constant rise in the larger areas in the can be seen as opposed to the larger jumps that happened at smaller areas. There are several reasons for this. First, the nodes are much more spread out and packets are not colliding as often. Second, each node is not solely responsible for only routing its payload. At the greater distances neighbor nodes must route the original nodes 1024 bytes of information to destination nodes, increasing the strain on the network. Third, one of the big users of network bandwidth is the OLSR routing protocol. OLSR is a proactive protocol meaning that it is continually searching and trying to maintain routes to other nodes. As more nodes are included in the network the more the network topology is ever changing especially as nodes are moving relatively fast. This causes more routing control packets to be sent, increasing the bandwidth being used for the network. But, this also let nodes move in range of other nodes very quickly. After 250 nodes, regardless of the replication rate, each simulation was running at full network usage.

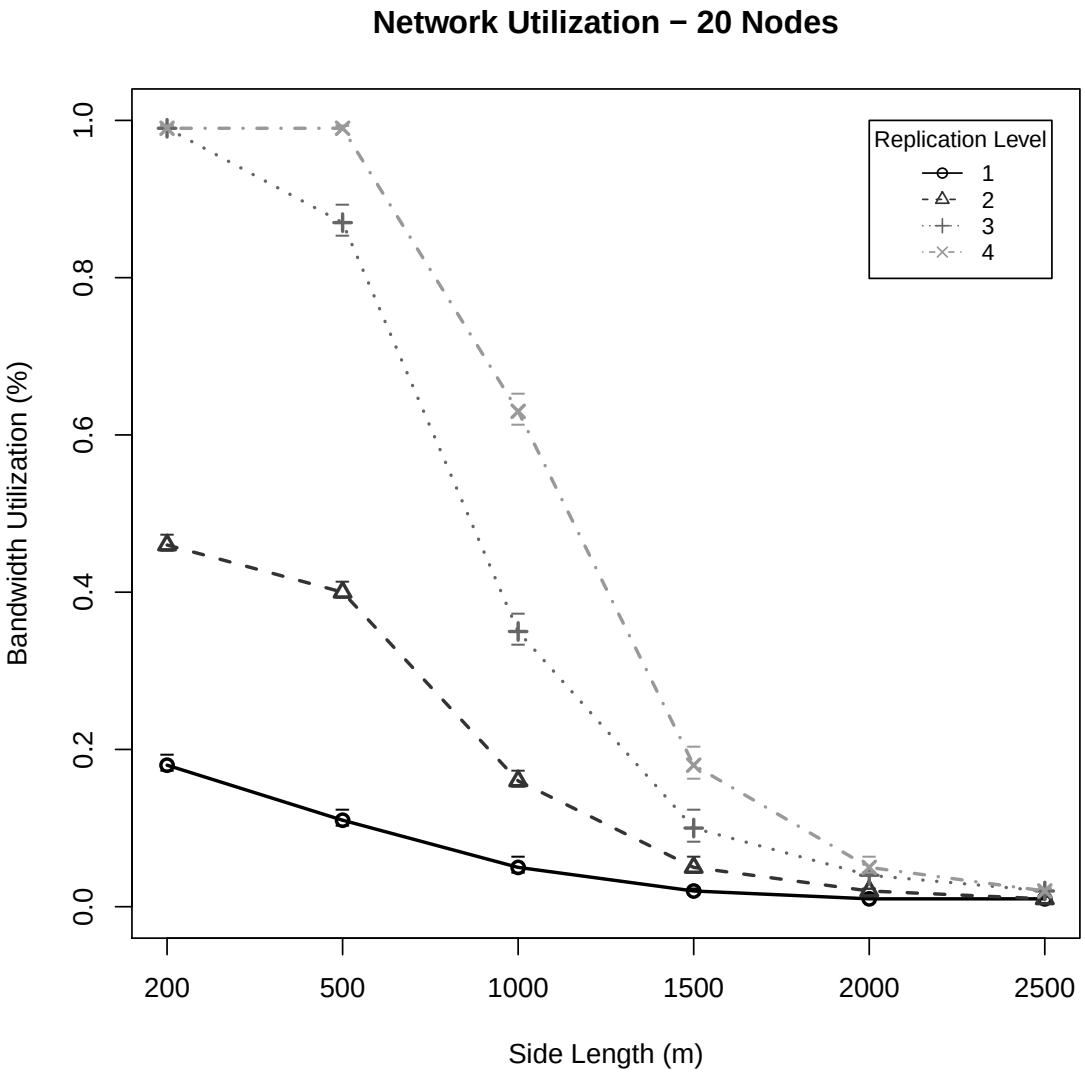

Figure 4.2: Bandwidth Utilization - 20 Nodes.

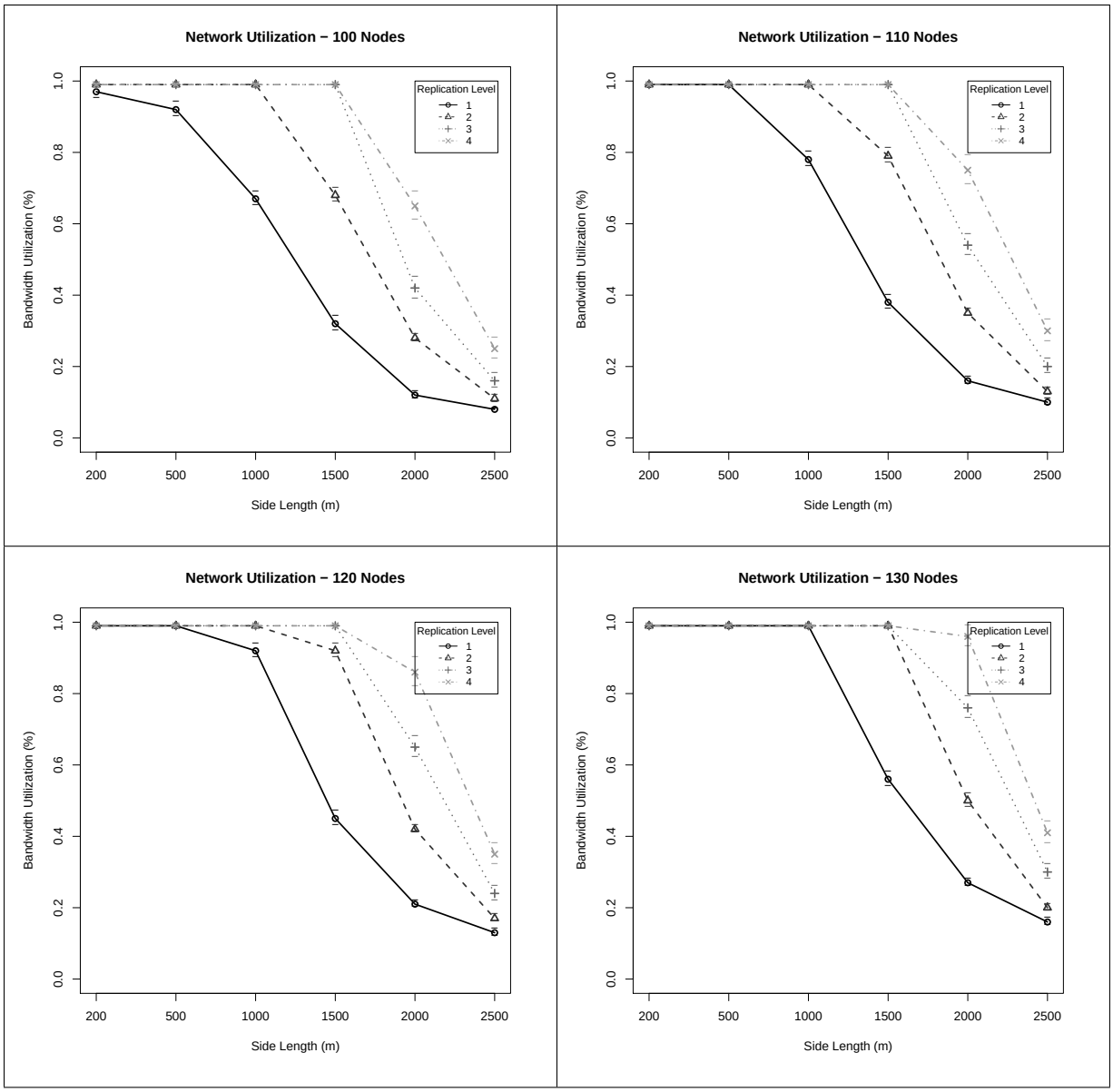

Figure 4.3: Bandwidth Utilization for 100-130 Nodes.

4.2 Successful Query Answer

Figure 4.4 shows the percentage of successful queries for ten nodes. The layout of the graph is the same as the graph for the bandwidth utilization. As Figure 4.4 shows all levels have near 100% success at 200 m by 200 m. As the area increase the system continues to have a high level of success until 1 km^2. At that point there is a drastic decrease in the percentage of successful queries. This is because nodes are not in range of each other. Query fails occur due to a lack of a response because of distances. Yet, at as they spread out as 2,500 m by 2,500 m all levels appear to maintain a 10% response. This percentage is artificially high because of the way the program was written. This is because nodes are randomly requesting data based off of the globally known data table. With 10 nodes, the chance of a node requesting data that it has is 10%. This is also why in Figure 4.5 for twenty nodes the percentage of successful queries at 2,500 m by 2,500 drops to 5%.

With the number of nodes set at 20, replication levels four and three resulted in full network utilization. Figure 4.5 shows that when the network is at full utilization the number of successful queries drops to zero. This is because the network is over capacity and packets cannot replicate due to the high drop rate of packets with such high usage.

For comparison to the bandwidth utilization at 100 - 130 nodes, Figure 4.6 is shown. As the figures show the percentage of successful queries rises as more nodes are introduced into the system. This is because the nodes have greater connectivity, and percentage chance that a closer node has a replication of the data requested. The random way-point mobility model also works in favor of nodes coming into range of one another. With the random way-point model, nodes tend to move back to the middle of the system area [5]. This implies that nodes are continually criss-crossing, and thus able to find routes to key destination nodes.

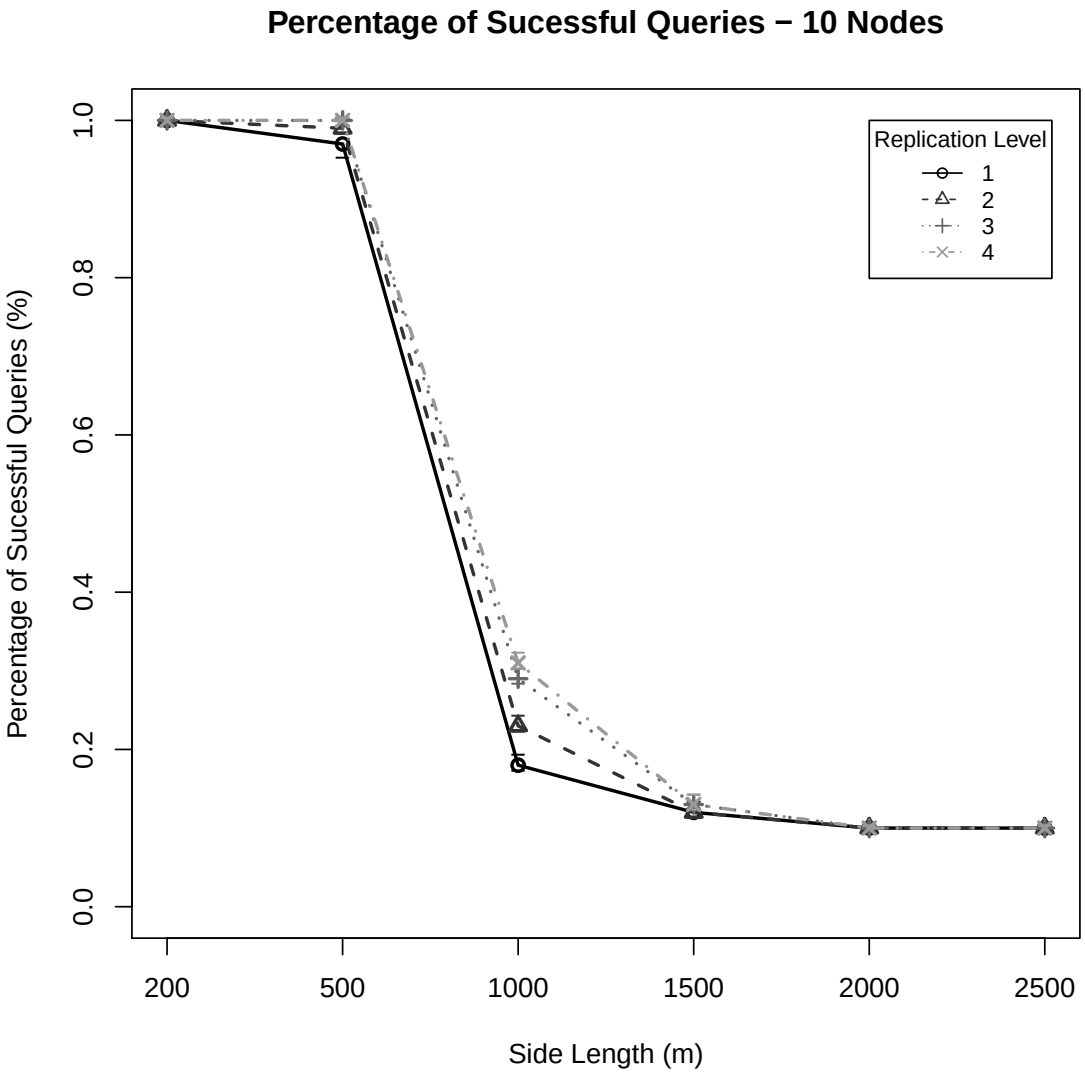

Figure 4.4: Percentage of Successful Queries - 10 Nodes.

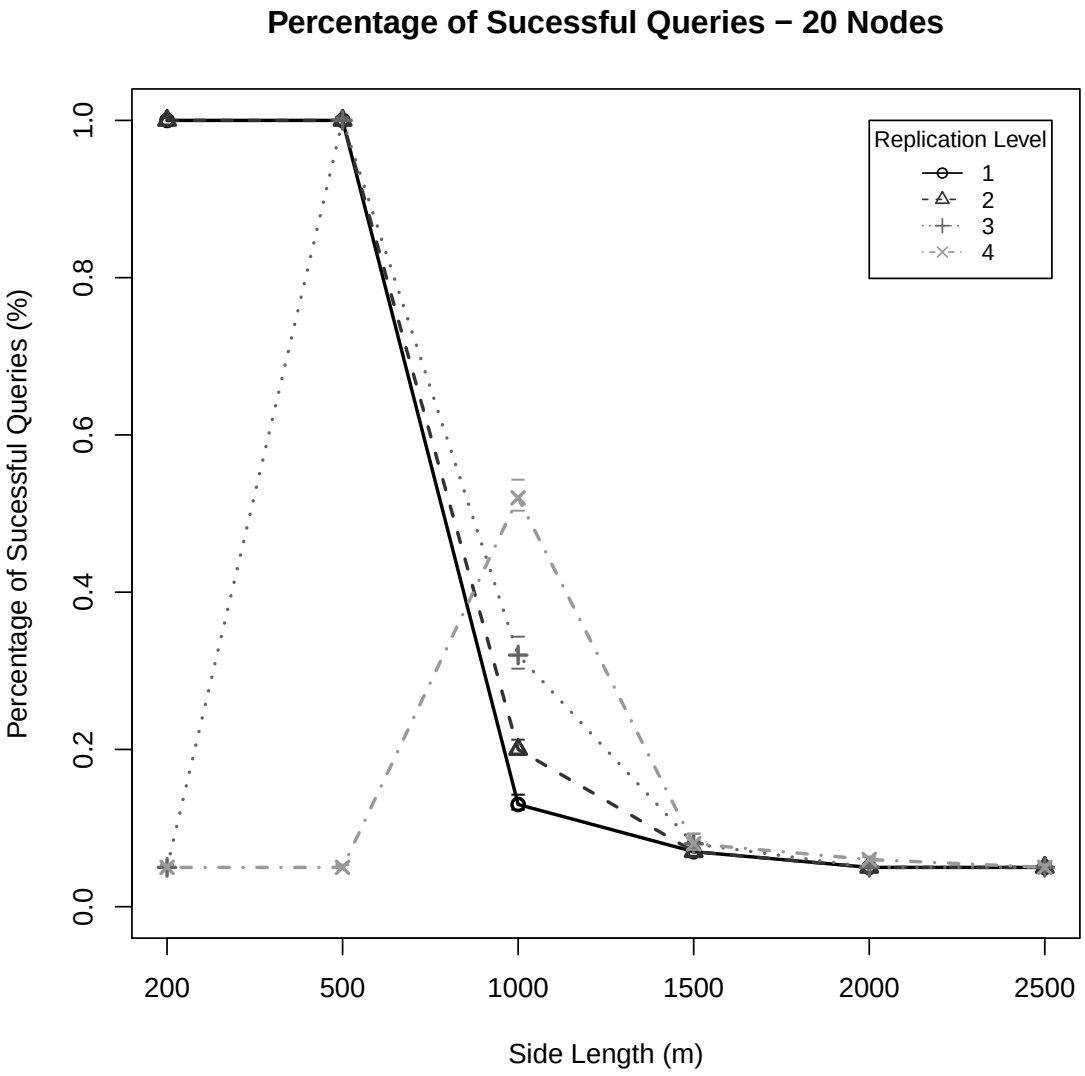

Figure 4.5: Percentage of Successful Queries - 20 Nodes.

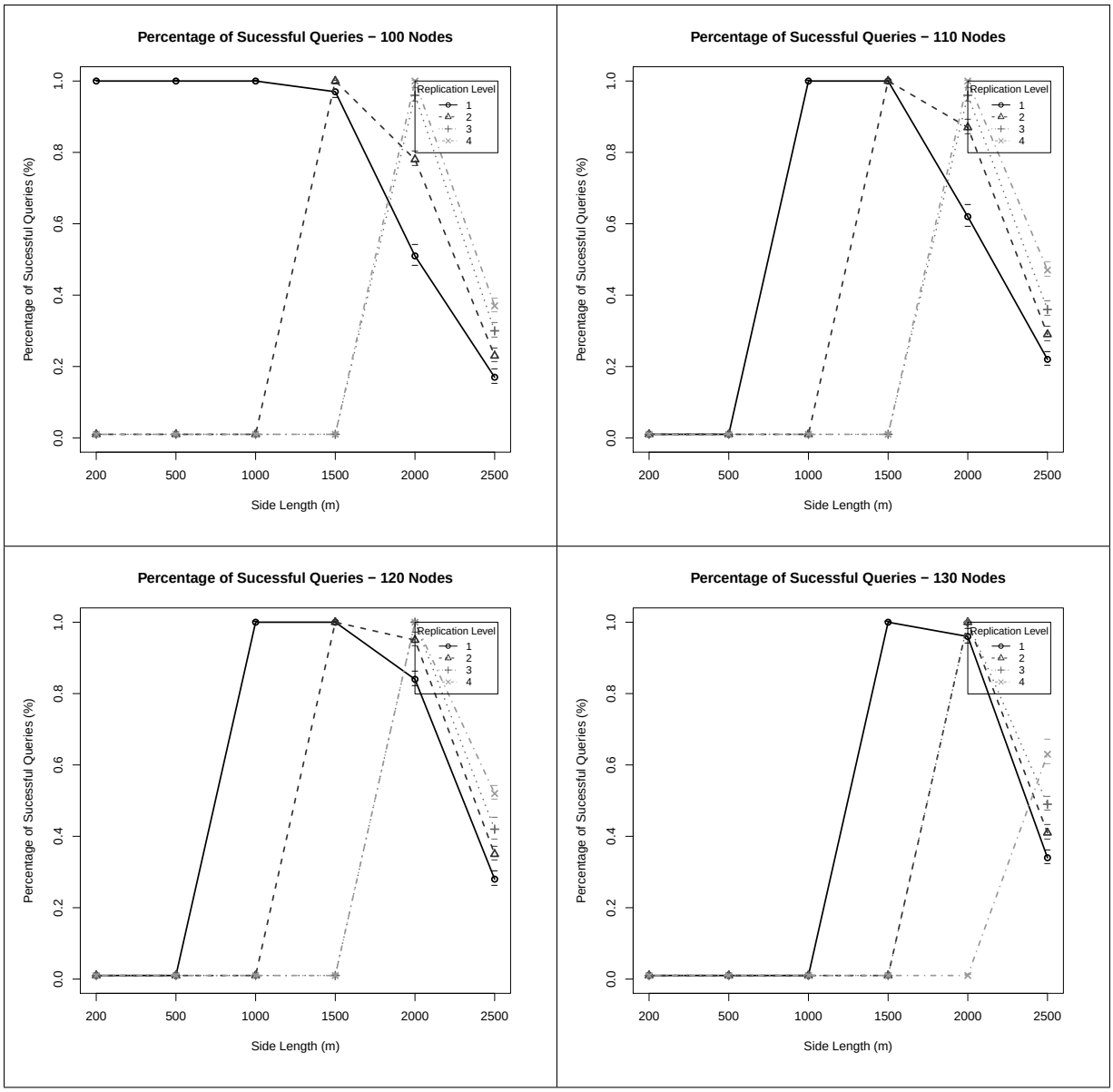

Figure 4.6: Percentage of Successful Queries for 100-130 Nodes.

4.3 Summary

The results of this experiment show that a P2P DHT with data replication for an AN performs best when there is a high level of replication, but nodes are not crammed on top of one another. For an AN this is important to remember to maintain some space between vehicles while searching, and to try and minimize overlap of search patterns.

V. Conclusions

Decentralized, distributed, self replicating systems used in certain situations provide a significant benefit. They can provide redundancy, have a high level of fault tolerance, and survive multiple failures. They can usually scale to large numbers while maintaining robustness. But, there are some drawbacks to large amounts of replication. On bandwidth constrained networks, data replication can quickly erode all communication.

This research explored data replication for highly mobile Ad-Hoc DHT for Airborne Networks. Results show that the densities of nodes in a given area should be limited if large amount of data transfer is going to occur. However, with real world ANs it is unlikely their search pattern is a random waypoint. Also, in real world Airborne Networks the data transfer rate will not be set at constant rate. It is likely with the advances in 4G and WiMax network the data transfer rate will be much quicker.

5.1 Research Conclusions

In determining the amount of replication that is allowable before network bandwidth is choked. The system performed best when nodes have a little distance between one another, and are not trying to replicate large amounts data on top each other. As the number of nodes and amount of replication increases, the number of satisfied queries increases until the network utilization caps. As the number of nodes increases the overhead bandwidth increased as well from the routing protocol. The fast changing network topology allowed nodes quickly move in and out of range on one another. With regards to determine the availability of data throughout the network by querying nodes for given data. The increase the replication rate does increase the change the that a node will find requested data.

5.2 Future Work

There are many other configurations and potential optimizations that could be investigated into possibly improving a vehicular based MANET employing a replicating DHT.

- Determine the effects a vehicle based mobility model. In random way-point model nodes tend to move back to the middle of the system area [5]. If the mobility model was better suited for vehicular search following a designated pattern, with less bunching and overlap of nodes. Nodes may be able to better utilize the network and achieve better replication percentages.

- Determine the effects of a reactive network layer routing protocol. Since, OLSR is a proactive protocol it is continually using bandwidth to try and find a route, while DSR waits until packet is ready to be send to find a route. However, since the topology of this network changed quickly and data was sent out often is is unlikely this would make a significant difference.

- Implement an Ekta type system with integration between protocol layer. This may cut down on some bandwidth usage.

- Implement this system on a larger scale using a larger area as well as what is considered a longer range protocol such as cellular 3G or 4G.

Bibliography

[1] Akbarinia, Reza, Mounir Tlili, Esther Pacitti, Patrick Valduriez, and Alexandre A. B. Lima. "Transactions on Large-scale Data and Knowledge Centered Systems III". chapter Replication in DHTs using dynamic groups, 1–19. Springer-Verlag, Berlin, Heidelberg, 2011. ISBN 978-3-642-23073-8. URL http://dl.acm.org/citation.cfm?id=2028190.2028191.

[2] Anastasi, Giuseppe, Eleonora Borgia, Marco Conti, and Enrico Gregori. "IEEE 802.11b Ad Hoc Networks: Performance Measurements". *Cluster Computing*, 8:135–145, 2005. ISSN 1386-7857. URL http://dx.doi.org/10.1007/s10586-005-6179-3. 10.1007/s10586-005-6179-3.

[3] Araujo, Filipe, Luus Rodrigues, Jorg Kaiser, Changling Liu, and Carlos Mitidieri. "CHR: A Distributed Hash Table for Wireless Ad Hoc Networks". *Proceedings of the Fourth International Workshop on Distributed Event-Based Systems (DEBS) (ICDCSW'05) - Volume 04*, ICDCSW '05, 407–413. IEEE Computer Society, Washington, DC, USA, 2005. ISBN 0-7695-2328-5-04. URL http://dx.doi.org/10.1109/ICDCSW.2005.48.

[4] Aung, A.M. and M. Pwint. "A Framework for Supporting Consistent Lookup in Distributed Hash Table". *Computer Design and Applications (ICCDA), 2010 International Conference on*, volume 5, V5–344 –V5–348. june 2010.

[5] Bettstetter, C., H. Hartenstein, and X. Pérez-Costa. "Stochastic Properties of the Random Waypoint Mobility Model". *Wireless Networks*, 10(5):555–567, 2004.

[6] Bianchi, S., S. Serbu, P. Felber, and P. Kropf. "Adaptive Load Balancing for DHT Lookups". *Computer Communications and Networks, 2006. ICCCN 2006. Proceedings. 15th International Conference on*, 411–418. IEEE, 2006.

[7] Boeing. "A160 Hummingbird". URL http://www.boeing.com/bds/phantom_works/hummingbird.html.

[8] Buford, J.F. and H. Yu. "Peer-to-peer Networking and Applications: Synopsis and Research Directions". *Handbook of Peer-to-Peer Networking*, 3–45, 2010.

[9] Chang, Robert W. "Synthesis of Band-Limited Orthogonal Signals for Multichannel Data Transmission". *Bell Systems Technical Journal*, 45, December 1966.

[10] Chao, Fan, Hongqi Zhang, Xuehui Du, and Chuanfu Zhang. "Improvement of Structured P2P Routing Algorithm Based on NN-CHORD". *Wireless Communications, Networking and Mobile Computing (WiCOM), 2011 7th International Conference on*, 1 –5. sept. 2011. ISSN 2161-9646.

[11] Chen, Dong, Zhenhua Tan, Guiran Chang, and Xingwei Wang. "An Improvement to the Chord-Based P2P Routing Algorithm". *Proceedings of the 2009 Fifth International Conference on Semantics, Knowledge and Grid*, SKG '09, 266–269. IEEE Computer Society, Washington, DC, USA, 2009. ISBN 978-0-7695-3810-5. URL http://dx.doi.org/10.1109/SKG.2009.32.

[12] Cheng, L. "Bridging Distributed Hash Tables in Wireless Ad-Hoc Networks". *Global Telecommunications Conference, 2007. GLOBECOM '07. IEEE*, 5159 –5163. nov. 2007.

[13] Chung, Yoo. "Efficient Batch Update of Unique Identifiers in a Distributed Hash Table for Resources in a Mobile Host". *Proceedings of the International Symposium on Parallel and Distributed Processing with Applications*, ISPA '10, 625–630. IEEE Computer Society, Washington, DC, USA, 2010. ISBN 978-0-7695-4190-7. URL http://dx.doi.org/10.1109/ISPA.2010.73.

[14] Cramer, Curt and Thomas Fuhrmann. "Performance Evaluation of Chord in Mobile Ad Hoc Networks". *Proceedings of the 1st international workshop on Decentralized resource sharing in mobile computing and networking*, MobiShare '06, 48–53. ACM, New York, NY, USA, 2006. ISBN 1-59593-558-4. URL http://doi.acm.org/10.1145/1161252.1161264.

[15] Dabek, Frank, M. Frans Kaashoek, and C. Smith. *A Distributed Hash Table*. Technical report, Technical report, 2005.

[16] Das, Himabindu, Saumitra M.and Pucha and Y. Charlie Hu. "How to Implement DHTs in Mobile Ad Hoc Networks?" *Student poster, the 10th ACM International Conference on Mobile Computing and Network (MobiCom 2004), September-October*. 2004.

[17] Dudek, Gregory, Michael R. M. Jenkin, Evangelos Milios, and David Wilkes. "A taxonomy for multi-agent robotics". *AUTONOMOUS ROBOTS*, 3:375–397, 1996.

[18] Frank, H. Fitzek. *Mobile Peer to Peer*. Wiley, 2009.

[19] Gast, Matthew S. *802.11 Wireless Networks: The Definitive Guide, Second Edition*. O'Reilly Media, Inc., 2005. ISBN 0596100523.

[20] Goel, S., T. Imielinski, K. Ozbay, and B. Nath. "Grassroots: A Scalable and Robust Information Architecture". *Dept. of Computer Science, Rutgers University, Tech. Rep. DCS-TR-523*, 2003.

[21] Goel, Samir and Tomasz Imielinski. "Prediction-based Monitoring in Sensor Networks: Taking Lessons from MPEG". *SIGCOMM Comput. Commun. Rev.*, 31(5):82–98, October 2001. ISSN 0146-4833. URL http://doi.acm.org/10.1145/1037107.1037117.

[22] Guo, Zhengwei, Lin Min, Shuai Yang, and Huaipo Yang. "An Enhanced P4P-Based Pastry Routing Algorithm for P2P Network". *Proceedings of the 2010 IEEE International Conference on Granular Computing*, GRC '10, 687–691. IEEE Computer Society, Washington, DC, USA, 2010. ISBN 978-0-7695-4161-7. URL http://dx.doi.org/10.1109/GrC.2010.111.

[23] Hartenstein, H., B. Bochow, A. Ebner, M. Lott, M. Radimirsch, and D. Vollmer. "Position-aware Ad Hoc Wireless Networks for Inter-vehicle Communications: The Fleetnet Project". *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking & computing*, 259–262. ACM, 2001.

[24] Heer, Tobias, Stefan Gotz, Simon Rieche, and Klaus Wehrle. "Adapting Distributed Hash Tables for Mobile Ad Hoc Networks". *Proceedings of the 4th annual IEEE international conference on Pervasive Computing and Communications Workshops*, PERCOMW '06, 173–. IEEE Computer Society, Washington, DC, USA, 2006. ISBN 0-7695-2520-2. URL http://dx.doi.org/10.1109/PERCOMW.2006.16.

[25] Hooper, Daylond James. "A Hybrid Multi-Robot Control Architecture". March 2007.

[26] Hoschek, Wolfgang. "A Unified Peer-to-Peer Database Protocol". *European Organization for Nuclear Research*, 16:8–37, 2004.

[27] Hossain, Ekram. *Introduction to Network Simulator NS2*. Springer, 2009.

[28] Jiang, Yi, Guangtao Xue, Zhaoqing Jia, and Jinyuan You. "DTuples: A Distributed Hash Table based Tuple Space Service for Distributed Coordination". *Proceedings of the Fifth International Conference on Grid and Cooperative Computing*, GCC '06, 101–106. IEEE Computer Society, Washington, DC, USA, 2006. ISBN 0-7695-2694-2. URL http://dx.doi.org/10.1109/GCC.2006.41.

[29] Jin-Lin, Wang, Wang Ling-Fang, and Song Lei. "LandHouse: a Network Storage Structure Combining Regular Graph with Distributed Hash Table and its Data Accessing Methods". *Proceedings of the 3rd international conference on Intelligent information technology application*, IITA'09, 360–365. IEEE Press, Piscataway, NJ, USA, 2009. ISBN 978-1-4244-5212-5. URL http://dl.acm.org/citation.cfm?id=1794914.1795004.

[30] Kapelko, Rafal. "Improving Data Availability in Chord P2P System". *Proceedings of the Second international conference on Information Computing and Applications*, ICICA'11, 208–215. Springer-Verlag, Berlin, Heidelberg, 2011. ISBN 978-3-642-25254-9. URL http://dx.doi.org/10.1007/978-3-642-25255-6_27.

[31] Karagiannis, G., O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil. "Vehicular Networking: A Survey and Tutorial on Requirements, Architectures, Challenges, Standards and Solutions". *Communications Surveys & Tutorials, IEEE*, 13(4):584–616, 2011.

[32] Karrels, Daniel R., Gilbert L. Peterson, and Barry E. Mullins. "Survey of Structured Peer-to-Peer Overlay Networks". 2008.

[33] Kato, Daishi. "Latency Model of a Distributed Hash Table with Big Routing Table". *Proceedings of the Fourth International Conference on Peer-to-Peer Computing*, P2P '04, 274–275. IEEE Computer Society, Washington, DC, USA, 2004. ISBN 0-7695-2156-8. URL http://dx.doi.org/10.1109/P2P.2004.27.

[34] Kummer, R., P. Kropf, and P. Felber. "Distributed Lookup in Structured P2P Ad Hoc Networks". *On the Move to Meaningful Internet Systems 2006: CoopIS, DOA, GADA, and ODBASE*, 1541–1554, 2006.

[35] Kurose, J.F. and K.W. Ross. *Computer Networking: A Top-Down Approach*. Pearson Education Canada, 2012. ISBN 9780132856201.

[36] Landsiedel, Olaf, Stefan Gotz, and Klaus Wehrle. "Towards Scalable Mobility in Distributed Hash Tables". *Proceedings of the Sixth IEEE International Conference on Peer-to-Peer Computing*, P2P '06, 203–209. IEEE Computer Society, Washington, DC, USA, 2006. ISBN 0-7695-2679-9. URL http://dx.doi.org/10.1109/P2P.2006.46.

[37] Landsiedel, Olaf, Katharina Anna Lehmann, and Klaus Wehrle. "T-DHT: Topology-based Distributed Hash Tables". *Proceedings of the Fifth IEEE International Conference on Peer-to-Peer Computing*, P2P '05, 143–144. IEEE Computer Society, Washington, DC, USA, 2005. ISBN 0-7695-2376-5. URL http://dx.doi.org/10.1109/P2P.2005.36.

[38] Liu, Che-Liang, Chih-Yu Wang, and Hung-Yu Wei. "Mobile Chord: Enhancing P2P Application Performance over Vehicular Ad Hoc Network". *GLOBECOM Workshops, 2008 IEEE*, 1 –8. 30 2008-dec. 4 2008.

[39] Luo, Yan and Ouri Wolfson. "Mobile P2P Databases". University of Illinois at Chicago, 2007. URL http://www.cs.uic.edu/~boxu/domino/encyc-MP2PDB-revised.pdf.

[40] Lv, Q., P. Cao, E. Cohen, K. Li, and S. Shenker. "Search and Replication in Unstructured P2P Networks". *Proceedings of the 16th international conference on Supercomputing*, 84–95. ACM, 2002.

[41] Monnerat, L.R. and C.L. Amorim. "D1HT: a Distributed One Hop Hash Table". *Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International*, 10–pp. IEEE, 2006.

[42] Nadeem, T., P. Shankar, and L. Iftode. "A Comparative Study of Data Dissemination Models for VANETs". *Mobile and Ubiquitous Systems-Workshops, 2006. 3rd Annual International Conference on*, 1–10. IEEE, 2006.

[43] Pucha, H., S.M. Das, and Y.C. Hu. "Ekta: An Efficient DHT Substrate for Distributed Applications in Mobile Ad Hoc Networks". *Mobile Computing Systems and Applications, 2004. WMCSA 2004. Sixth IEEE Workshop on*, 163–173. IEEE, 2004.

[44] Ramasubramanian, Venugopalan and Emin Gün Sirer. "Beehive: O(1)lookup Performance for Power-law Query Distributions in P2P Overlays". *Proceedings of the 1st conference on Symposium on Networked Systems Design and Implementation - Volume 1*, NSDI'04, 8–8. USENIX Association, Berkeley, CA, USA, 2004. URL http://dl.acm.org/citation.cfm?id=1251175.1251183.

[45] Ratnasamy, Sylvia, Paul Francis, Mark Handley, Richard Karp, and Scott Shenker. "A Scalable Content Addressable Network". *SIGCOMM Comput. Commun. Rev.*, 31(4):161–172, August 2001. ISSN 0146-4833. URL http://doi.acm.org/10.1145/964723.383072.

[46] Robertazzi, T. G. *Computer Networks and Systems: Queueing Theory and Performance Evaluation*. Telecommunication networks and computer systems. Springer-Verlag, 1994. ISBN 9780387941707.

[47] Roussopoulos, M. and M. Baker. "Practical Load Balancing for Content Requests in P2P Networks". *Distributed Computing*, 18(6):421–434, 2006.

[48] Rowstron, Antony and Peter Druschel. "Pastry: Scalable, Decentralized Object Location, and Routing for Large-scale Peer-to-Peer Systems". *IN: MIDDLEWARE*, 329–350, 2001.

[49] Shah, R.C., S. Roy, S. Jain, and W. Brunette. "Data mules: Modeling and Analysis of a Three-tier Architecture for Sparse Sensor Networks". *Ad Hoc Networks*, 1(2):215–233, 2003.

[50] Sit, E., A. Haeberlen, F. Dabek, B.G. Chun, H. Weatherspoon, R. Morris, M.F. Kaashoek, and J. Kubiatowicz. "Proactive Replication for Data Durability". *5rd International Workshop on Peer-to-Peer Systems (IPTPS 2006)*. 2006.

[51] Sitepu, Herry Imanta, Carmadi Machbub, Armein Z. R. Langi, and Suhono Harso Supangkat. "UnoHop: Efficient Distributed Hash Table with O(1) Lookup Performance". *Proceedings of the 2008 Third International Conference on Broadband Communications, Information Technology & Biomedical Applications*, BROADCOM '08, 76–81. IEEE Computer Society, Washington, DC, USA, 2008. ISBN 978-0-7695-3453-4. URL http://dx.doi.org/10.1109/BROADCOM.2008.72.

[52] Stoica, Ion, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. "Chord: A Scalable Peer-to-Peer Lookup Service for Internet Applications". *SIGCOMM Comput. Commun. Rev.*, 31(4):149–160, August 2001. ISSN 0146-4833. URL http://doi.acm.org/10.1145/964723.383071.

[53] Tang, B., Z. Zhou, A. Kashyap, and T. Chiueh. "An Integrated Approach for P2P File Sharing on Multi-hop Wireless Networks". *Wireless And Mobile Computing, Networking And Communications, 2005.(WiMob'2005), IEEE International Conference on*, volume 3, 268–274. IEEE, 2005.

[54] Tian, J. and L. Coletti. "Routing Approach in CarTALK 2000 Project". *proceedings of the IST Mobile & Wireless Communications Summit 2003*, volume 2. Citeseer, 2003.

[55] Toor, Yasser, Paul Mhlethaler, Anis Laouiti, and Arnaud de La Fortelle. "Vehicle Ad Hoc Networks: Applications and Related Technical Issue". *IEEE Communications Surveys and Tutorials*, 74–88, 2008.

[56] Wederhake, Lars P. "Plaxton Routing From a Peer-to-Peer-Network Point of View". 2008.

[57] Winter, S., M. Sester, O. Wolfson, and G. Geers. "Towards a Computational Transportation Science". *Journal of Spatial Information Science*, (2):119–126, 2012.

[58] Wolfson, O., A.P. Sistla, and B. Xu. "The TranQuyl language for Data Management in Intelligent Transportation". *Transportation Research Part C: Emerging Technologies*, 2012.

[59] Wu, Yi-Chun, Chuan-Ming Liu, and Jenq-Haur Wang. "Enhancing the Performance of Locating Data in Chord-Based P2P Systems". *Proceedings of the 2008 14th IEEE International Conference on Parallel and Distributed Systems*, ICPADS '08, 841–846. IEEE Computer Society, Washington, DC, USA, 2008. ISBN 978-0-7695-3434-3. URL http://dx.doi.org/10.1109/ICPADS.2008.88.

[60] Yan Wan, Yi Zhou Shengli Fu, Kamesh Namuduri. "A Smooth-turn Mobility Model for Airborne Networks". *University of North Texas*. 2012.

[61] Yousefi, S., T. Chahed, S.M.M. Langari, and K. Zayer. "Comfort Applications in Vehicular Ad Hoc Networks Based on Fountain Coding". *Vehicular Technology Conference (VTC 2010-Spring), 2010 IEEE 71st*, 1–5. IEEE, 2010.

[62] Zhao, Ben Y., John D. Kubiatowicz, and Anthony D. Joseph. *Tapestry: An Infrastructure for Fault-tolerant Wide-area Location and*. Technical report, Berkeley, CA, USA, 2001.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704–0188

1. REPORT DATE *(DD–MM–YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From — To)*
21–03–2013	Master's Thesis	Oct 2011–Mar 2013

4. TITLE AND SUBTITLE

Airborne Network Data Availability Using Peer to Peer Database Replication On A Distributed Hash Table

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Vranicar, Trevor J., Second Lieutenant, USAF

5d. PROJECT NUMBER

JON 12G194

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Air Force Institute of Technology
Graduate School of Engineering and Management (AFIT/EN)
2950 Hobson Way
WPAFB, OH 45433-7765

8. PERFORMING ORGANIZATION REPORT NUMBER

AFIT-ENG-13-M-48

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Jacob Campbell
AFRL/RYWN
2241 Avionics Circle
WPAFB, OH 45433-7301

10. SPONSOR/MONITOR'S ACRONYM(S)

AFRL/RYWN

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

DISTRIBUTION STATEMENT A.
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

The concept of distributing one complex task to several smaller, simpler Unmanned Aerial Vehicles (UAVs) as opposed to one complex UAV is the way of the future for a vast number of surveillance and data collection tasks. One objective for this type of application is to be able to maintain an operational picture of the overall environment. Due to high bandwidth costs, centralizing all data may not be possible, necessitating a distributed storage system such as mobile Distributed Hash Table (DHT). A difficulty with this maintenance is that for an Airborne Network (AN), nodes are vehicles and travel at high rates of speed. Since the nodes travel at high speeds they may be out of contact with other nodes and their data becomes unavailable. To address this the DHT must include a data replication strategy to ensure data availability. This research investigates the percentage of data available throughout the network by balancing data replication and network bandwidth. The DHT used is Pastry with data replication using Beehive, running over an 802.11 wireless environment, simulated in Network Simulator 3. Results show that high levels of replication perform well until nodes are too tightly packed inside a given area which results in too much contention for limited bandwidth.

15. SUBJECT TERMS

Network, Ad-hoc, DHT, Replication

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Dr. Gilbert L. Peterson, AFIT/ENG
U	U	U	UU	65	19b. TELEPHONE NUMBER *(include area code)* (937) 255-3636 ext. 4281