LOCATION-DEPENDENT DATA CACHING WITH HANDOVER AND REPLACEMENT FOR MOBILE AD HOC NETWORKS

QIAO YUNHAI

(B.Eng(Hons.), NUS)

A THESIS SUBMITTED
FOR THE DEGREE OF MASTER OF ENGINEERING
DEPARTMENT OF ELECTRICAL AND COMPUTER
ENGINEERING
NATIONAL UNIVERSITY OF SINGAPORE
2006

Acknowledgements

The first thank you should be given to my supervisors Associate Professor Bharadwaj Veeravalli and Professor Lawrence Wong WaiChoong, whose help, stimulating suggestions and encouragement helped me in all the time of research for and writing of this thesis. They are not only great scientist with deep vision but also and most importantly kind persons. Their trust and scientific excitement inspired me in the most important moments of making right decisions and I am glad to work with them.

I would like to thank Assistant Professor Vikram Srinivasan, whom I am deeply indebted to. I owe him lots of gratitude for having shown me this way of research. He could not even realize how much I have learned from him.

I would like to express my sincere appreciation to Miss Xia Li for her invaluable discussions, suggestions and technical supports on this work.

My thanks also go out to the Department of Electrical and Computer Engineering for giving me permission to commence this thesis in the first instance, to do the necessary research work and to use all kind of resources for my research.

I feel a deep sense of gratitude for my parents who formed part of my vision and taught me the good things that really matter in life. I wish to thank my beloved girlfriend for her love and patience during the past 10 years.

Finally, I would like to express my gratitude to all the others who gave me the possibility to complete this thesis.

Qiao Yunhai

January 2006

Contents

Summary

Over the last few years, the field of wireless ad hoc networks has attracted tremendous interest from the research community. The primary attraction of a wireless ad hoc network is the fact that networks can form spontaneously without the need for any fixed infrastructure. The critical design issues for mobile ad hoc networks include provisioning of seamless communication with Quality-of-Service (QoS) guarantees, high data accessibility, reliable data transfer, low energy consumption, and high communication performance. However, limited bandwidth and battery power, mobility of nodes, and frequent change of network topology add several new dimensions to this problem.

Due to the mobility characteristic of a mobile ad hoc network, many routing techniques have been developed to route messages. Although routing protocols play an important role in mobile ad hoc networks, other issues such as data access are also important to achieve the ultimate goal of setting up a network, which is to communicate with each other and exchange information. Since wireless resources in a mobile ad hoc network are rather limited, data requests must be satisfied in a very efficient way. Usually those data requests, that cannot be satisfied within a period of time, are considered as failed/blocked. Therefore, it is a challenging task to retain data accessibility over a mobile ad hoc network. The overall objective of this research is to design and develop data caching schemes in order to retain data accessibility. In particular, a single-server network model is considered and a location-dependent data access pattern is addressed. A selfish cache technique is introduced as the underlying reference. Two caching schemes are proposed, Simple Cache and Relay Cache. Location-dependent cache handover and replacement schemes are introduced to further enhance data accessibility.

Most of previous works use Random Waypoint Mobility Model, which is not realistic enough in most situations. In order to verify the performance of the proposed schemes and to recommend the most relevant caching policy in different cases, various mobility models are examined in this research, including Random Waypoint, Random Direction, Gauss-Markov, Manhattan Grid and Reference Point Group Mobility (RPGM).

The performance is evaluated by examining the impact of memory size, request generating time, and the maximum moving speed of mobile nodes on data accessibility. Furthermore, energy consumption is considered in this research. Hence, a reasonable recommendation could be made by balancing energy consumption and data accessibility.

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Introduction to Ad Hoc Networks

The last decade has seen the rapid convergence of two pervasive technologies: wireless communication and the Internet [1]. Today, many people carry numerous portable devices, such as laptops, mobile phones, PDAs and iPods or other mp3 players, for use in their professional and private lives. As a result, people may store their data in different devices according to their own preferences. Most of the time, it is very difficult to exchange data between different types of devices without the aid of a network. Furthermore, it is not always possible to make use of the Internet as their underlying networking platform due to physical/geographical constraints. With the development of technology in wireless communication, more and more mobile devices are integrated with wireless communication capacity. Therefore, a technique allowing a group of mobile devices to build a network among themselves anytime and anywhere becomes interesting to the research community.

Mobile ad hoc network is the outcome of this demand as mentioned before. Basically, a mobile ad hoc network is an autonomous collection of mobile nodes [2].

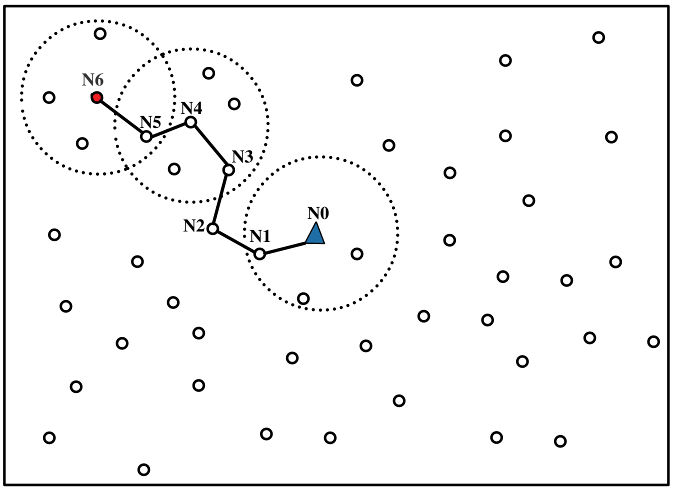

Figure 1.1: A Mobile Ad Hoc Network

Mobile nodes can communicate with each other by creating a multi-hop wireless connection and maintaining connectivity without any special infrastructure. Each mobile node plays the role of a router which handles the communications among mobile nodes. A mobile node can communicate with another node that is immediately within its radio range. If a node outside its radio range need to be accessed, one or more intermediate nodes will be needed to relay the data between the source and the destination. Figure 1.1 shows an example of a typical mobile ad hoc network. In this example, N4 can communicate with N5 and N3 directly as they are located within the transmission range of N4. However, N0 is outside of the radio range of N6, intermediate nodes N1, N2, N3, N4 and N5 then act as routers while establishing the connection between N0 and N6.

1.1.1 Advantages & Applications of Ad Hoc Networks

The major advantage of a mobile ad hoc network is that it does not need any base station as is required in either wired network or regular mobile networks, such as

GSM, GPRS or even 3G [2]. With further advances in technology, mobile ad hoc networks will be implemented in various situations. A mobile ad hoc network can be formed in any place as required immediately which makes it indispensable in battlefield and disaster relief/rescue situations. It is useful in some places that are not covered by fixed network with Internet connectivity. In this situation, the mobile nodes in the newly established ad hoc network can be used to provide the coverage. It also can be used in areas where the available network has been destroyed. As mobile devices are driven by battery, mobile ad hoc networks can be used in the situation of electricity failure, which lead the traditional Internet or cellular network out of order because they are both dependent on the line power.

1.1.2 Challenges Faced by Ad Hoc Networks

As any conventional wired/wireless network, there are some common challenges that need to be faced while setting up a new mobile ad hoc network. Similar to other wireless networks, the boundaries of the network are not well defined and hence it is possible for any node to enter and leave the network at any time while they are moving. It is also possible for a mobile ad hoc network with a large number of nodes to split into two or more networks either because these groups are physically apart from one another or due to disfunction of some key joint mobile nodes. Hidden-terminal and exposed-terminal problems are also faced by mobile ad hoc networks. In a mobile ad hoc network, mobile nodes have both power and bandwidth constraints, which will lead to power failure or channel congestions and both will decrease the QoS of the mobile ad hoc network. Furthermore, a mobile ad hoc network may be constructed by all kinds of mobile devices, which may have different capacity, functionality and protocols. Hence it is necessary to find a solution where all these devices can operate together.

1.1.3 Routing Schemes for Ad Hoc Networks

Both the advantages and the challenges of mobile ad hoc networks are due to the mobility of nodes, which makes the network topology change frequently. Therefore, routing in such networks is an important issue and meanwhile is a challenging task. Because of the importance of routing in mobile ad hoc networks, a lot of research have been done on this topic and many routing schemes have been proposed. Most of the proposed routing schemes use the information about the links that exist in the network to perform data forwarding. Those routing protocols can be roughly divided into three categories: *proactive (table-driven)*, *reactive (on-demand)* and *hybrid*.

1. Proactive routing algorithms employ classical routing schemes such as distance-vector routing or link-state routing. They maintain routing information about the available paths in the network even if these paths are not currently used.

2. Reactive routing protocols maintain only the routes that are currently in use, thereby reducing the burden on the network when only a small subset of all available routes is in use at any time.

3. Hybrid routing protocols combine local proactive routing and global reactive routing in order to achieve a higher level of efficiency and scalability.

Examples of routing protocols belonging to these three categories are shown in Table 1.1 respectively. Several more sophisticated routing protocols have been proposed by employing route caching schemes [15, 16].

Table 1.1: Routing Protocols

Proactive	1. Destination Sequenced Distance Vector (DSDV) [8] 2. Fisheye State Routing (FSR) [5] 3. Optimized Link State Routing Protocol (OLSR) [6] 4. Source Tree Adaptive Routing (STAR) [4] 5. Topology Dissemination Based on Reverse-Path Forwarding (TBRPF) [3] 6. Wireless Routing Protocol (WRP) [7]
Reactive	7. Associativity Based Routing Protocol (ABR) [9] 8. Ad Hoc on Demand Distance Vector Routing (AODV)[12] 9. Dynamic Source Routing (DSR) [11] 10. Temporary Ordered Routing Algorithm (TORA) [10]
Hybrid	11. Zone Routing Protocol (ZRP)[13]

1.2 Overview of Data Caching

In traditional wired networks, the network topology seldom changes once the network is set up properly. The servers usually have very high computation capacity and storage space, which allow them to implement complicated algorithms to serve various applications in the network. On the other hand, the bandwidth and other resources are abundant, which ensure that data requests are not to be blocked due to lack of resources within a short period of time. However, in mobile ad hoc networks, disconnection and network division occur frequently as mobile nodes move arbitrarily, and the wireless resources are very sparse. As a result, data request may be easily blocked when no route exist between requesting node and the data server or when the wireless bandwidth is used up. Thus, data accessibility in mobile ad hoc networks is lower than in the conventional wired networks, where the data accessibility is defined as the ratio of successfully served data requests, R_{suc}, over all data requests in a network, R_{tot}, as shown by the equation below:

$$P_a = \frac{R_{suc}}{R_{tot}} \tag{1.1}$$

Caching was first introduced by Wilkes [17] in 1965, and is popularly employed in many systems, such as distributed file systems, database systems, and traditional wired network systems, etc.

In the past few decades, data caching has been widely studied and used in distributed file systems and traditional wired networks [19, 20, 21, 22, 23, 24, 33, 34]. In such systems, nodes that host the database are more reliable and system failures do not occur as frequently as in mobile ad hoc networks. Therefore, it is usually sufficient to create a few replicas of a database, which can be used to provide higher accessibility.

Data caching has been extensively studied in a Web environment as well [30, 31]. The goal is to place some replicas of web servers among a number of possible locations so that the query delay is minimized. In the Web environment, links and nodes are stable. Therefore, the performance is measured by the query delay, and data accessibility is not a big issue. Energy and memory constraints are not considered either.

Hara [38] proposed some replica allocating methods to improve data accessibility on mobile ad hoc networks by replicating the original data and distributing the replicas over the network beforehand. Those methods assume that all mobile nodes are aware of the overall access probabilities to every data item in the network and the access pattern is static throughout the life of the network.

Another group of researchers addressed the cached data discovery problems. Takaaki [39] proposed a "self-resolver" paradigm as a cached data discovery method in his paper, which took into account the stability of a multi-hop route and derived two types of link model: neighbor-dependent link model and neighbor-independent link model. Instead of developing a complicated caching algorithm, Lim [40] integrated a simple search algorithm into an aggregated caching scheme so as to access the cached data more effectively. Yin and Cao [25] proposed a set of cooperative-caching algorithms, *CachePath* and *CacheData*. In *CachePath*, the path to each cached data item is stored and the cached data path will be used to redirect further requests to nearby caching nodes. *CacheData* allows multiple nodes to cache the data along the path established between the requesting node and the data server.

There are several advantages of using data caching:

1. Data caching reduces bandwidth consumption, thereby decreasing network traffic and lessens network congestion.
2. Data caching reduces access latency due to two reasons:
   (a) Frequently accessed data are fetched from nearby caching nodes instead of faraway data servers, thus the transmission delay is minimized.
   (b) Because of the reduction in network traffic, those data not cached can also be retrieved relatively faster than without caching due to less congestion along the path and less workload at the server.
3. Data caching reduces the workload of the data server by disseminating data among the mobile nodes over the ad hoc network.
4. If the data server is not available due to physical failure of the server or network partitioning, the requesting node can obtain a cached copy at the caching nodes. Thus, data accessibility is enhanced.

5. Data caching reduces the battery energy consumption as some requests are served either locally or by some nearby caching nodes.

1.3 Basic Cache Replacement Policies

The objective of cache replacement algorithms is to minimize the miss count in finite-sized storage systems. Some of the cache replacement policies have been studied for Web caching [41, 42]. A replacement policy can be generally defined by a comparison rule that compares two cached items. Once such a rule is known, all objects in the cache can be sorted in an increasing order, and this is sufficient to apply a replacement policy: the cache will remove the object of lowest value with respect to the given comparison rule. Each cached item has several attributes, such as access time (the last time when the object was accessed), item size or access frequency. These attributes are used to define the replacement policies. Least Recently Used (LRU)[36], Least Frequently Used (LFU)[36] and Minimum Size (MINS)[37] are three such policies.

1.4 Motivation of This Thesis

Because of the amount of efforts have been put by the researchers over the years, nowadays, the routing protocols for mobile ad hoc networks are more mature than any other research topic in the area of mobile ad hoc networks. With the currently available routing schemes, it is not difficult to establish effective routes between sources and destinations in a mobile ad hoc network. However, the ultimate goal of setting up a mobile ad hoc network is not to establish routes, but to provide a means to accomplish information exchange. Therefore, besides developing high-performance routing protocols, more efforts should be put in improving data accessibility among mobile nodes in mobile ad hoc networks. In order to address this objective, the idea of data caching is employed in mobile ad hoc networks, whereby intermediate nodes hold the data requested by other nodes in their cache memories and those cached data will be used to serve further requests generated among mobile nodes in the network.

So far in the literature there is little attention devoted to the study of caching schemes with location-dependent data access being taken into account. Data caching schemes with location-dependent data replacement and data handover policies are even rarer. However, the scenario is common that nodes have similar sets of desired data while they are traveling in the same location. For example, people are more likely to ask for information about animals/birds when they are visiting a zoo, but people who are shopping at a downtown area hardly have the interests to know anything about a tiger or a fox. Therefore, the type of information people access is related to their location, in this thesis, we call it location-dependent data access pattern. On the other hand, as mobile nodes only have limited storage space, it is impossible for one node to hold all the data available in the network due to these physical limitations. Due to the limited bandwidth and energy, it is also not a good idea to have all requests served by the data server because the wireless channels will be very congested near the data server and those mobile nodes close to the data server have to consume their energy to relay data for others, which makes it easy for them to drain out their batteries and the whole network will be affected. However, if any mobile node could contribute part of its memory space to hold data for others, the whole network will benefit from its contributions.

However, when a node only holds part of the data, there will be a tradeoff between the query delay (which may be in terms of hops to traverse or time to spend) and data accessibility. For example, in a mobile ad hoc network, a node caches the received data in its own memory whenever its request is successfully served. As a result, the cached data are mainly for its own benefit, and the query delay may be reduced dramatically since most of requests will be served locally with zero or very small delay. However, when the location-dependent access pattern is considered, mobile nodes in the same location will request for a similar set of data, which will end up with an extreme scenario, that is every mobile node caches similar data locally, and the rest of the data are not cached by anyone. Therefore, if a node suddenly requests data, which is not cached by anyone nearby, the request will be relayed to the data server. The probability to have the request successfully served by the data server far away from the requesting node is much lower than the success probability if a node nearby has the data in its memory. In this scenario, in order to increase data accessibility, neighboring nodes should avoid caching too many copies of same data by some means (For example, by disallowing caching the same data that neighboring nodes already have). However, this solution may increase the hops needed to travel in order to fetch the data since some nodes may not be able to cache the most frequently accessed data locally, and have to access it from other caching nodes or data server. Traversing more hops will end up with longer query delay and higher energy consumption.

In this thesis, we focus on enhancing data accessibility. A location-dependent data access pattern is studied. Several data caching schemes are proposed to address data accessibility. The impact on the energy consumption will be considered with the various data caching protocols employed. Location-dependent data handover and data replacement techniques are introduced to further improve performance.

Research on mobile ad hoc networks are mostly simulation-based. NS-2 [61] and Glomosim [62] are the two most popularly used simulators. With the lack of mobility model support, many researchers adopt the Random Waypoint mobility model

[50] as their underlying mobility model in most of their simulation experiments. However, mobility is the most important characteristic for mobile ad hoc networks. Therefore, in order to evaluate the performance of our proposed caching schemes under different mobility models, different sets of simulation experiments have been done over a network with different mobility models, such as the Gauss-Markov mobility model, the Manhattan Grid mobility model and the Group mobility model.

1.5 Thesis Organization

The remainder of this thesis is organized as follows. In Chapter 2, we introduce the required notations, definitions, and formulate the problem. In Chapter 3, we present the proposed data caching schemes, then the location-dependent handover and replacement policies are introduced. In Chapter 4, various mobility models are presented. In Chapter 5, the detailed simulation testbed is described, then the experimental results are presented and discussions are made for the results. Finally, Chapter 6 concludes this thesis and discusses future works.

Chapter 2

Contributions & Problem Formulation

In this chapter, we shall introduce the problem we target to solve in this thesis. The required notations, definitions and terminologies that will be used throughout this thesis will be presented.

2.1 The Problem

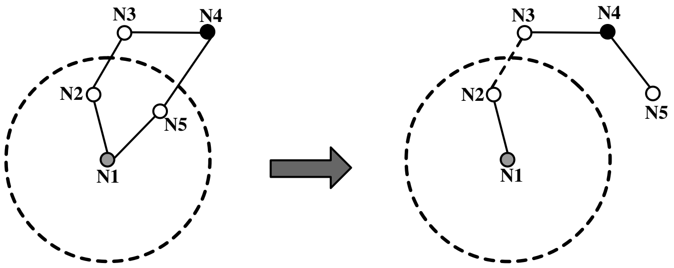

Figure 2.1: Topology Change in Ad Hoc Networks

In wireless ad hoc networks, network disconnections are common because of the movements and the drain of limited battery resources in mobile devices, which make it difficult to retain data accessibility. Furthermore, the data traverses many hops from source to destination, which could result in a very large access latency. Figure 2.1 shows such an example: initially, node N1 and N5 have a direct link between them. When N5 moves out of N1's radio range, the link between them is broken. However, they can still communicate with each other through the intermediate nodes N2, N3, and N4. As a result, after the location change of N5, data has to travel through 4 hops (N1-N2-N3-N4-N5) to serve a single request generated by N5 (assuming N1 is the source); in contrast, the data need to traverse only one hop before the link broke. If N4 and N5 keep generating data requests with a small time interval between 2 requests, since bandwidth is a scarce and expensive resource in mobile ad hoc networks, congestion may occur at the links from N1 to N4. In a more serious case, if the link between N2 and N3 is also broken, the network will be divided into two partitions. Then, the requests for data by N1 from N3, N4 and N5 will all be blocked because no routes are able to be established to make the communication successful.

From the example described in the previous paragraph, it is apparent that there are several difficulties we may encounter while designing a mobile ad hoc network. These difficulties are network disconnections due to mobility of nodes, channel congestions near the data source because of high demand of the data and limited wireless resources, long transmission delay and high energy consumption caused by multiple hop communication, etc. To address these problems, data caching is a very effective technique. Assume that N4 keeps a copy of the data in its local cache memory after receiving it from N1, then this cached data could be used to serve requests generated by both itself and nearby mobile nodes (e.g. N3,N5). If N5 migrates out of N1's radio range, it is able to fetch data from N4 within one hop instead of fetching it from N1, that is 4 hops away. The transmission delay is reduced and channel usage is also saved. In the case of a link break between N2 and N3, although the network is divided and N3, N4 and N5 are not able to get data from N1 directly, their requests would be served by N4. Hence data accessibility is retained.

2.2 Problem Formulation

The following notations are used in this thesis.

- n: the total number of mobile nodes.
- N_i: mobile node i.
- M: the total number of data items available in the network.
- D_i: data item i.
- s_i: the size of D_i.
- C_i: the cache memory size of N_i.
- f_{ij}: the link failure probability between N_i and N_j.
- t: the time if a request cannot be served within which it will be considered as blocked.
- P_b: the data blocking ratio.

Consider an ad hoc network with n mobile nodes, $N_1, N_2, ..., N_n$ with M data items, $D_1, D_2, ..., D_M$, available in the network. At any given time, the link between N_i and N_j has a probability of f_{ij} to fail, which indicates the disconnection of the network. In this thesis, f_{ij} is equal to f_{ji} since only symmetric link is considered. The link failure is caused only by physical partition of two mobile nodes, which means there is no route found between them. Furthermore, all requests generated when link failure occurs will be considered as blocked. For data access, there is no restriction for any mobile node to access any data item. Every mobile node has some memory space to be used to cache data locally for self usage or for others. However, as the cache memory size is limited, mobile node N_i can only keep a limited number of data items in its local memory, and the number is determined by C_i and s_i. When a mobile node N_i needs to access a data item D_j, N_i will first search its own local memory for D_j. If D_j is found, the request is served locally. The energy consumption and access delay latency are both very low in this case. However, if N_i cannot find a copy of D_j in its local memory, a request for D_j will be broadcasted until some nodes respond to this request. If there is no acknowledgement received for a request within t, the request will be treated as blocked and the request will be dropped by all mobile nodes in the network. Hence, the data blocking ratio is defined as the number of blocked data requests, $R_{blocked}$, over the total number of data requests generated all over the network, R_{tot}.

$$P_b = \frac{R_{blocked}}{R_{tot}} \tag{2.1}$$

2.3 Complexity Analysis

It is a hard problem to optimize the performance of an ad hoc network as the performance could be influenced by various conditions, such as the mobility pattern, the bandwidth resources, the strength of wireless signal, the physical failure of a mobile node, etc. However, it is still a very hard problem even if only one performance metric needs to be optimized, such as data accessibility, energy consumption, access delay. The computational complexity is so high that to find the optimal solution for this problem is not practical at all.

In this complexity analysis, we take the optimization of energy consumption as an example, which is similar to minimizing the average number of hops traversed to get a data. Although the data items are not same sized in our experiments, which will be discussed in later chapters, in order to further simplify the problem in our analysis, let us assume all the data items are of the same size and the cache memory in each node is the same. Therefore, each node is able to store the same number of data items locally. Furthermore, instead of applying a reactive caching scheme, a proactive caching scheme is discussed here, which needs much stronger assumptions and those assumptions may not be realistic all the time. For example, Hara [38] assumes that all mobile nodes are aware of the overall access probabilities to every data item in the network and the access pattern is static throughout the life of the network.

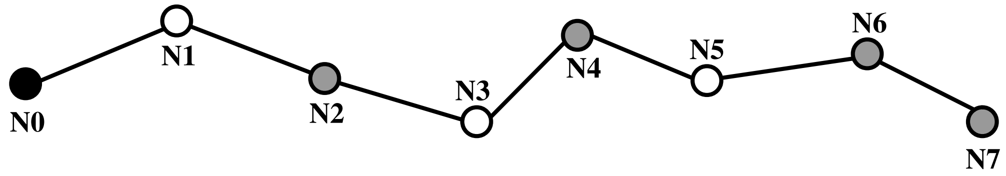

Figure 2.2: Example of Data Caching

The energy consumption will be decreased by caching each data at some vantage nodes since a request will be served by nearby caching nodes instead of a faraway data server. In this analysis, we define the benefits made by caching data D_i at node N_j as the performance improvement of energy consumption. An expression of the benefit is shown as follows, where E_{nc} is the energy consumed without the caching scheme and E_c is the energy consumed with caching scheme.

$$Benefit = E_{nc} - E_c \tag{2.2}$$

However, the benefit of caching data D_i at node N_j could be affected by network topology, data access pattern, wireless communication technique, etc. Therefore, the benefit of caching D_i at different nodes will be different. Furthermore, when D_i is cached in multiple nodes, the benefit will be affected by a previous cached copy of D_i in the network also. For example in Figure 2.2, N_0 is the data server with data D_i in its memory. If N_7 has a copy of D_i, the overall network will benefit from it and the benefit is represented as b_1. N_4 may request for D_i and get it from either N_0 or N_7 and cache D_i in its own memory. Then the network will benefit from the cache of D_i at N_4 by an amount of b_2. Here, b_2 is affected by b_1 because if N_7 did not cache D_i, the cached copy of D_i at N_4 will give more benefit to the whole network. Therefore, the benefit of caching a data item is influenced by the expansion of the number of caching nodes. That is, the benefit of allocating D_i at a node with no other nodes caching the data yet is different from allocating D_i at a node with the data item already having been replicated once, twice, or more times at some other mobile nodes. Therefore, we can see that at one given node, the benefit of a data item D_i may have $(n-1)!$ values, where n is the number of mobile nodes in the network, depending on the distribution of replicas of D_i in the network.

To further simplify this problem, let us assume that the benefit of caching D_i in different mobile nodes are mutually exclusive with one another; that is, the benefit of caching D_i in N_j is independent of previous copies of D_i in the network. Therefore, we could model our analysis as a *Generalized Assignment Problem (GAP)* [43], which is described as:

INSTANCE: A pair (B, S), where B is a set of m bins, and S is a set of n items. Each bin $j \in B$ has a capacity of $c(j)$, and for each item i and bin j, we are given a benefit $b(i, j)$ and a size $s(i, j)$.

OBJECTIVE: Find a subset $U \subseteq S$ of maximum benefit such that U has a feasible packing in B. Here feasible packing means a method to distribute items using bins with capacities restricted to be $c(j)$ so as to minimize the sum of the capacities of the bins used.

In our analysis, D_i is fixed in size and the bin size is identical, which is C. Chekuri and Khanna [43] proved that the *GAP* problem is an APX-hard problem (approximable hard) [44] even for a very special and simple case, where:

- each data item takes only two distinct benefit values,
- each data item has an identical size across all bins and there are only two distinct item sizes, and
- all bin capacities are identical.

That means there exists some constant $\epsilon > 0$ such that it is NP-hard to approximate the problem within a factor of $(1 + \epsilon)$

Therefore, even the simplified version of finding an optimal solution to distribute data over the network at vantage nodes is an APX-hard problem. In this thesis, the data items in the network are not fixed sized like what was done by Chekuri and Khanna [43], which will further increase the computational complexity to find the optimal solution. Instead of evaluating one performance metric, we examine both data accessibility and energy consumption in this thesis, which makes the caching problem even harder. We can conclude that it is an APX-hard problem to find a caching schemes to optimize data accessibility and energy consumption. Therefore, instead of trying to design a complicated mathematical model, we will present some simple approaches which are able to enhance the overall performance of the network with small overhead.

2.4 Data Access Models

Most likely, the data access models can be divided into two groups, location-dependent and location-independent. An example of these two data access models

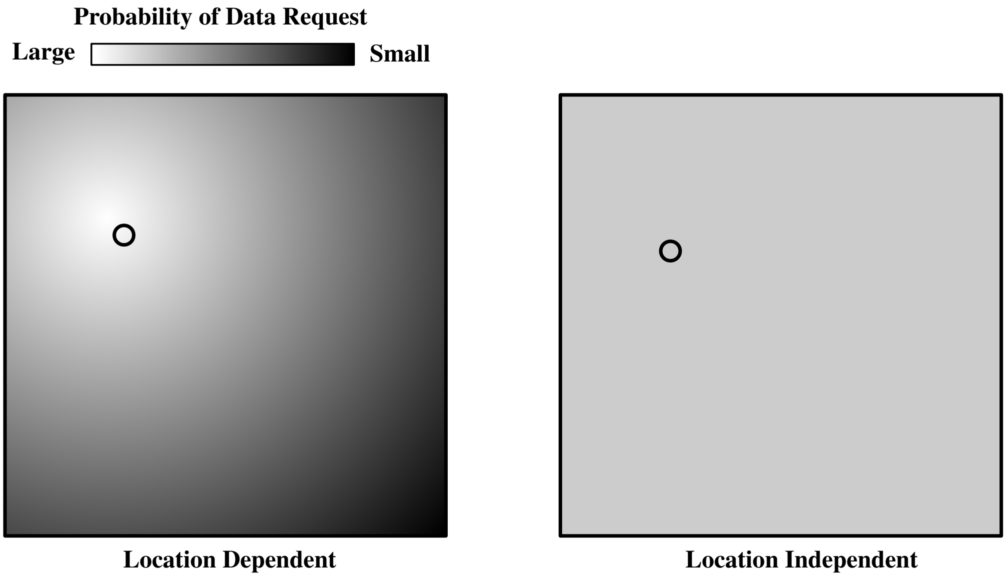

Figure 2.3: Data Access Models

is shown in Figure 2.3. In the location-independent model, the probability to access each individual data item is equal:

$$P_{D_i} = 1/M \tag{2.3}$$

where M is total number of data items available in the entire network.

In this thesis, we model the whole system as a location-dependent system. We classify the data into several categories:

- Global Hot Data(GH): data which is of high interest to mobile nodes anywhere in the network.
- Local Hot Data(LH): data which is of high interest to mobile nodes moving around a particular location.
- Local Cool Data(LC): data which is of low interest to mobile nodes moving around a particular location.

- Global Cool Data(GC): data which is seldom accessed by any node in the network.

The data server keeps a set of data items $D_1, D_2, ..., D_M$, where M is total number of data items. The entire network described in the previous section is divided into K local grids ($LG_1, LG_2, ..., LG_K$) based on their coordinates. These M data items is classified into four categories, GH, GC, LH and LC.Data items $D_{GH_1}, D_{GH_2}, ..., D_{GH_u}$ (GH) are of general interest, and can be requested uniformly from every mobile device in the network with probability P_{GH}. Here $(GH_1, GH_2, ..., GH_u) \subset (1, 2, ...M)$, and u is the number of GH data items. Similarly, $D_{GC_1}, D_{GC_2}, ..., D_{GC_v}$ (GC) can be requested uniformly from every mobile device in the network with probability P_{GC}. Here $(GC_1, GC_2, ..., GC_v) \subset (1, 2, ...M)$, where v is the number of GC data items. In LG_i $(i \in (1, 2, ..., K))$, $D_{i_1}, D_{i_2}, ..., D_{i_w}$ (LH) are those data items that are potentially accessed by every mobile nodes within the locality of local grid LG_i with a high access probability P_{LH_i}, where $(i_1, i_2, ..., i_w) \subset (1, 2, ...M)$ and w is the number of LH data items in LG_i . The rest of data (LC) are likely to be accessed by mobile nodes in LG_i with very low probability P_{LL_i}. A local grid LG_i shares local interests with its neighboring grid LG_j, which can be presented by $(D_{i_1}, D_{i_2}, ..., D_{i_w}) \cap (D_{j_1}, D_{j_2}, ..., D_{j_w}) \neq \phi$. Generally $P_H \geq P_L$ and $P_{LH} \geq P_{LL}$. If $P_H = P_L = P_{LH} = P_{LL}$, the location-dependent access pattern is identical to a location-independent access pattern.

2.5 Assumptions & Properties

Here, we list the assumptions we made in this thesis.

- The data server is the only wireless device generating all the original data items.

- The data server is not power limited so that the lifetime of a data server is not shorter than the lifetime of any other mobile nodes in the network.
- Mobile nodes are power limited. Once a node drains out its battery, it will stop all its functionalities.
- Omni-directional antennas are used in all wireless devices including data server, and the transmission radius (R) are all the same.
- Each node has a unique ID and a mechanism to discover its one-hop neighbors.
- Each node is able to get its location information, e.g. via GPS, and be able to know the relative position of another node through interaction between them.
- Mobility is characterized by a maximum node moving speed v_{max}.
- Each node has a same size of memory used as cache storage.
- A mobile node has only limited cache memory which is only sufficient to store part of the data items available in data server.
- Mobile nodes are able to establish broadcast or unicast connection with one another depending on the routing information and the requirements of communication.
- Data are not updated, therefore data consistency is not considered in this thesis.
- All mobile nodes must cooperate with each other.

Chapter 3

Data Caching with & without Handover and Replacement

In this section, we shall introduce a set of caching schemes. *Selfish Cache* is a typical data caching technique in the literature. *Simple Cache* (SC) is developed on top of Selfish Cache, and *Relay Cache* (RC) is a further advanced caching scheme allowing multiple caching nodes. Further, a location-dependent handover policy and replacement policy will be presented.

3.1 Proposed Data Caching Schemes

3.1.1 Selfish Cache Scheme

Selfish Cache is developed by migrating the idea of web-cache over the Internet into ad hoc wireless networks domain. Web-cache allows the requesting device to cache the received data in its own memory for its own usage in the near future. The reason we call this scheme as selfish cache is that the local cached data is used to serve its own purposes only. In Selfish Cache, whenever a request is generated, the node checks its local memory first. The request is served locally if the data item is found, otherwise, a query is sent to the data server and the server responds to the query by sending back the requested data item. If the requesting node does not receive any reply within a "timeout" period, the request will be blocked.

3.1.2 Simple Cache (SC) Scheme

Selfish cache is the first step of employing a cache technique in ad hoc networks, but more benefits can be achieved by adopting the same kind of caching scheme, but with a better query serving protocol. Therefore, we propose our SC scheme based on Selfish Cache. Same as Selfish Cache, we only allow the requesting nodes to cache the received data. However, these cached data will be used to serve not only its own queries, but also queries from other mobile nodes. The SC scheme is able to gain better performance compared with Selfish Cache in two aspects. If a requesting node is able to set a path with the data server and there is a caching node along the path, it will respond to the request by sending back the data item to the requesting node directly. By allowing intermediate nodes to serve the requests, queries could be served within fewer hops, the query delay is reduced, energy is saved and the number of requests coming to the data server is reduced, which in turn will reduce the probability of congestion happening at the server.

3.1.3 Relay Cache (RC) Scheme

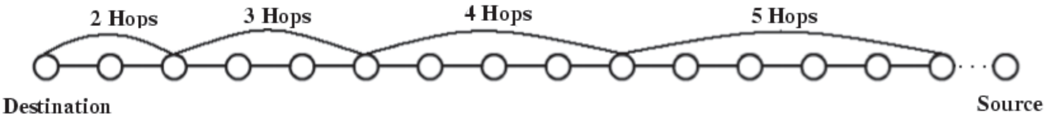

Figure 3.1: Relay Cache Example

In a human network, if one gets something done through another person, a "trust" relationship will be built. The more successful interaction between them will strengthen their relationship and enhance their "trust" level. Everybody has a list of parties with names, their strength and the rate of "trust". Once he has something to do, firstly he will try it by himself. However, if he fails to do so, he will probably ask the most trustable person among those capable ones to help him. Similarly, when we set up a network, mobile nodes will help each other to get data from the data server. Therefore, the probability for a node to route along the same path or choose a path across the same set of nodes around similar positions is high if it accesses a data item successfully through this path. In real world, another scenario is that users are more likely to access similar information when they are close to each other physically. Hence, when a requesting node receives its requesting data item through a path to data server, it is meaningful for some nodes along the path to cache the data item, which could be used to serve future requests from the same node or other nodes around the same location.

In this thesis, we consider that data are categorized as "hot" or "cool". When a path is set up between a requesting node and the data server, it is expected that a node along the path caches the data if the data is "hot"; however, if the data is "cool", a selfish node may choose not to cache the data in order to save its memory space. Since two neighboring grids may share some common interests, nodes close to the requesting nodes along the path may also have a high interest in the data passing-by, therefore, the nodes closer to the requesting nodes should have a higher priority of caching the data. Its interest is less when the relaying nodes are further from the requesting node. In order to make use of the limited memory effectively to gain higher performance improvement, we propose a relay cache scheme, in which we allow more nodes to cache the data where the nodes are nearer to the requesting node. On the other hand, fewer nodes will cache the data when they are further away from the requesting node.

For example in Figure 3.1, the destination is requesting for a data item, therefore it broadcasts a request packet to the network, a flag is attached with the request packet to track the hop number away from the destination. When the path is set up between the destination and the source, the intermediate nodes along the path will selectively cache the data based on the tracked hop number. The requesting node (destination) will cache the data locally if there is no one-hop neighbor caching it (if the flag number is not 1). By doing this, redundancy of too many replicas is prevented. Therefore, the first two caching nodes are two hops away from each other. The next caching node will be selected $H_n = H_p + \Gamma$ hops away from the latest one, where $H_p = H_i$. H_i and Γ are system parameters in terms of hop number. H_i is used to determine the distance between the destination and the next intermediate node to cache the data along the path. Γ is used to determine the increment of hop number between two adjacent caching nodes for the same data along the path. This process will continue until the path ends at the source node. When the intermediate node is far away from the requesting node, the data items relayed may not be "hot" to them. However, the storage size spent to cache those "cool" data is small because the probability to be selected as the caching node is small when the node is far away from the requesting node. On the contrary, a node far from the requesting node is near to the source node. As a result, spending part of its memory to cache "cool" data for others will help the source node to serve the requests before queries arrive at the source node. Since the requests may be served within fewer hops, energy consumption will be reduced. Although caching "cool" data may reduce the memory space to caching more interested "hot" data locally, and the data accessibility of "hot" data may be reduced in a small scale, it is still worth doing so, in order to achieve a balance between energy consumption and data accessibility.

In the example shown in Figure 3.1, we set $H_i = 2$ and $\Gamma = 1$. If we set $H_i = 1$ and $\Gamma = 0$, the RC scheme is identical to Greedy cache, which means all the intermediate nodes from the destination to the source will cache the data. If $H_i > 1$ and $\Gamma = 0$, the RC scheme becomes a uniform distribution of replicas, which means the data is cached every H_i hops from the destination to the source. In this thesis, we focus on the location-dependent data access pattern, therefore we adjust H_i and Γ in order that the RC scheme is a non-uniform caching scheme which may achieve better performance.

3.2 Location-Dependent Cache Handover Policy

Since the topology of the mobile ad hoc network keeps changing, data accessibility may become poor when the caching nodes move away from the location where the cached data are highly desired. Therefore, we propose a cache handover scheme in order to retain the "local" data accessibility by handing over the cached data to some nearby nodes when the caching nodes move out from that neighborhood area. When we apply this handover policy in every "local" area of this network, we are able to achieve "globally" high data accessibility. The nodes will also have better usages of their cache memory since after they hand over those cached data to their neighbors, they will have free spaces to store more useful data when they move into another location with different sets of interested data. There are several scenarios to trigger a node to handover its cached data items to other nodes.

Firstly, in order to optimize the performance of our caching schemes, we need to redistribute the cached data among a group of mobile nodes close to the current caching nodes. Therefore, every node needs to keep track of arriving requests information so as to be aware of the data access frequency. As mentioned before, each request package will be attached with a flag to indicate the hops traversed from the requesting node. Every node keeps a list of its one-hop neighbors by examining the flag. If the flag is 1, it means the query is from a node one hop away, which means the requesting node is its one-hop neighbor. By doing so, a node will know which data is requested by its neighbors more frequently. The location-dependent data category will be determined by the data access frequency collected over time. A node needs to store a list of "hot" data IDs and a table of its one-hop neighbors. The extra memory spent here and the computation power used to update the "hot" data list and one-hop neighbor table are considered as overhead. The overhead will increase if the network becomes dense or the query generating rate increases.

The second situation happens when a caching node moves to another grid and changes its interests. Based on the assumptions made in Section 2.4, a caching node may cache four types of data items: GH, GC, LH, LC. The majority of cached data items would belong to LH category, which are likely to be requested by others in the same grid later. Therefore, it is reasonable for this caching node to handover those "local hot" data items to some neighbors in that grid in order to retain the data availability for its neighbors when the caching node leaves that grid. Furthermore, when the node enters another grid, its interests will change with the locality. Therefore, the LH data items in a previous grid may not be "hot" anymore. Hence, the node should change its preference accordingly in order to cache the most useful data items with limited memory space. In this thesis, SC-H is used to indicate Simple Cache scheme with handover policy, and RC-H is used to indicate Relay Cache scheme with handover policy.

Finally, a node may cache some "locally cool" data after relaying it for others by being selected using system parameters H_i and Γ. A reasonable assumption is that a node, that caches "local cool" data for others, usually is relatively far away from the area where the cached data is highly demanded. As a result, the relative displacement will be small between the caching node and the area with a high interest in that data. Therefore, further requests from those mobile nodes in that grid with a high interest in that data may send a query packet to nodes at the same location with "local cool" data cached; but there will be no data available there anymore if the node goes too far from its original position. As discussed in Section 3.1.3, the probability for a node to route the same path or choose a path across same set of nodes around similar positions is high if it accessed a data item successfully through this path. Therefore, in order to retain the service along the path across same set of nodes around similar positions, the caching node should pass the data item to its neighbor in order for those requesting nodes to find the data along the similar path. By adopting these handover policies, data accessibility may be increased significantly. However, the energy consumption may be high if the mobile nodes are moving at very high speeds, which will cause frequent handovers, and the handovers consume energy, too.

3.3 Location-Dependent Cache Replacement Policy

The cache memory in a mobile device is limited, therefore a cache replacement policy is required in designing a caching scheme in order to determine which item to victimize when the cache memory is full. A new incoming data will only be able to replace the "cool" data item first when the cache memory is full. If all the cached data items are "hot", a least frequently used (LFU) scheme is used to discard the less frequently accessed data item from the cache memory in order to free up space for the new incoming data item. This replacement policy may trigger a handover event if a "hot" data item is going to be discarded from its cache memory. In such a case, the node will interact with its neighbors within the same grid, and the data will be dropped if there is a node within one or H_i hops caching the same data depending whether the caching scheme is SC or RC. Otherwise, it will find one neighboring node not caching this data and with enough free space, then hand it over to the newly found neighbor. In this thesis, the primary idea is that, in order to increase accessibility, we try to cache as many data items as possible while trying to avoid too many duplications. Therefore, we give the smaller data items higher priority because caching many smaller data items will improve the performance of the network to a higher degree compared with caching bigger data items. By giving priority to the smaller sized data, nodes are able to cache more data items with the same size of cache memory, and then data accessibility may be further enhanced. Therefore, while a replacement is needed, the biggest items will be removed first as other conditions are satisfied.

Chapter 4

Mobility Models

As we mentioned in Chapter 1, most of the researchers adopt Random Waypoint Mobility Model as their underlying mobility model while evaluating their schemes on mobile ad hoc networks. However, in order to examine the effectiveness of our caching schemes in different situations, five mobility models are introduced in this chapter. They are the Random Waypoint Mobility Model, the Random Direction Mobility Model, the Gauss-Markov Mobility Model, the Manhattan Grid Mobility Model and the Reference Point Group Mobility Model. The performance of our proposed caching schemes over different mobility models are addressed accordingly in Chapter 5.

4.1 Random Waypoint

In the Random Waypoint Mobility Model [50], mobile nodes are randomly located in the simulation area. The simulation area is usually described as a rectangle of $w_x \times w_y$, where w_x is the length of the area and w_y is the width of the area. At the beginning of the simulation, each node stays at its initial position for a period of pause time(t_p). Upon the expiry of the pause time, the mobile node

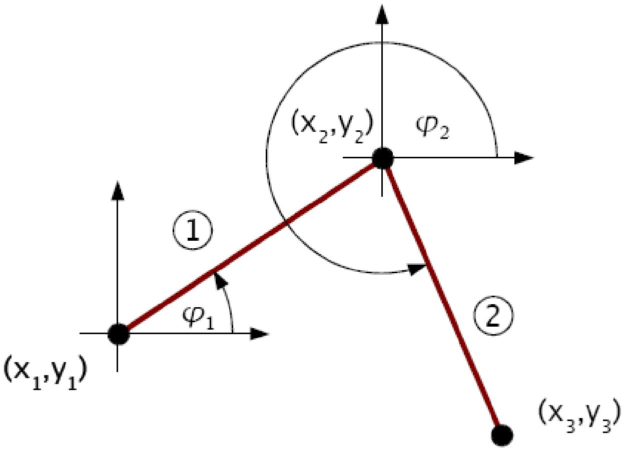

Figure 4.1: Random Mobility Model

randomly chooses a destination (waypoint) in the simulation area and a constant speed uniformly chosen between maximum(v_{max}) and minimum speed(v_{min}). The mobile node then travels on a straight line towards the destination at the selected speed. Once it arrives at the destination, it will stay for t_p before starting the process again. As each destination is chosen within the simulation area, mobile nodes will never hit any of the boundaries. With a fixed simulation area, the two mobility parameters, speed($[v_{min}, v_{max}]$) and the pause time(t_p) will determine the mobility pattern.

For example in Figure 4.1, mobile node N_i is located at (x_1, y_1) at time t_1, then a destination (x_2, y_2) is selected to be the waypoint. N_i travels towards (x_2, y_2) with a constant speed v, where $v_{min} \leq v \leq v_{max}$. When N_i arrives at (x_2, y_2), it will pause there for t_p, then a new destination (x_3, y_3) is selected. The process continues until the simulation is over. The parameters are chosen with the following uniform distributions:

- $x_i = uniform[0, X]$, where X is the width of the simulation area.

- $y_i = uniform[0, Y]$, where Y is the length of the simulation area.
- $v_i = uniform[v_{min}, v_{max}]$

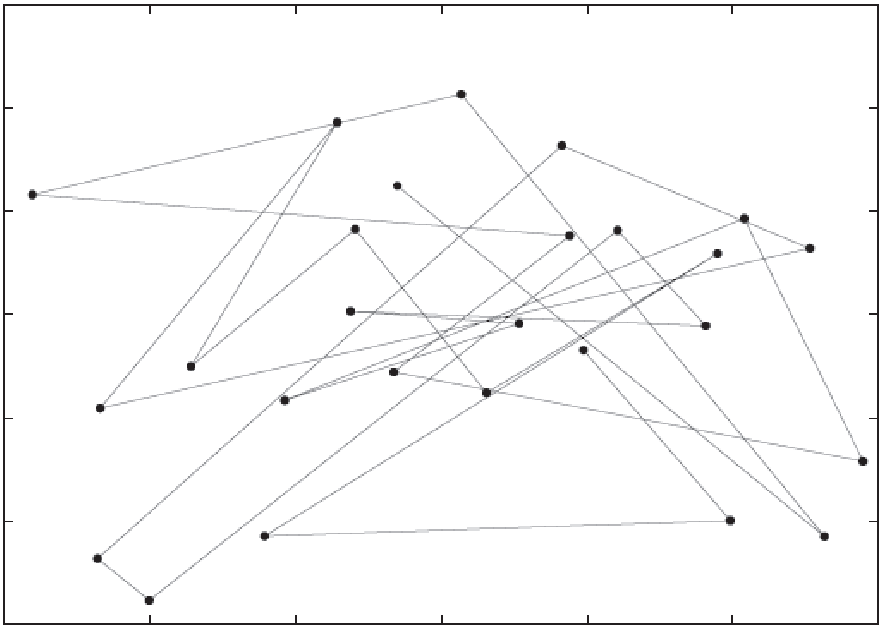

Figure 4.2: Example of Random Waypoint Mobility Models

Figure 4.2 shows a sample traveling pattern of a mobile node using the Random Waypoint Mobility Model starting at a randomly chosen point in the simulation area; the speed of the mobile node in the figure is uniformly chosen between 0 and $20 m/s$. The Random Waypoint Mobility Model is a widely used mobility model.

4.2 Random Direction

Besides the random waypoint model, the Random Direction Mobility Model (also called Random Walk Mobility Model) is probably the most widely used synthetic

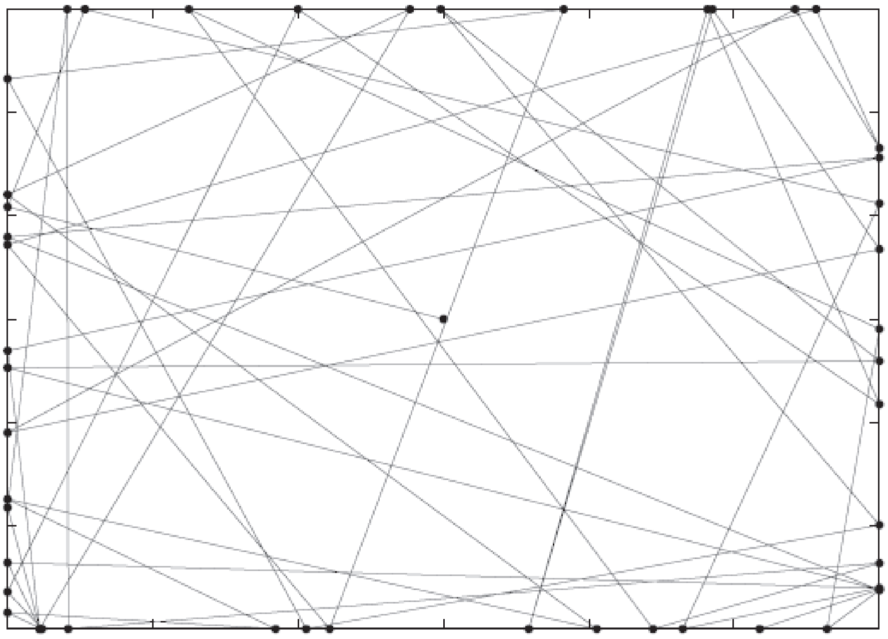

Figure 4.3: Example of Random Direction Mobility Models

mobility model for mobile communications research. Similar to the Random Waypoint Mobility Model, this model considers mobile nodes moving on straight walk segments with constant speed and optional pauses. There are several flavors of the random direction model which slightly differ in the way they obtain the next walk segment. Hong and Rappaport [51] propose a model that is built on top of a cell structure and apply walkers that pass those cells on straight lines and choose new directions at cell borders. Guerin [52] extends this model in a way that direction changes can be performed anywhere in a walk area. Some approaches model the direction choice with absolute angles while others like the one proposed by Zanoozi [53] calculate with relative changes to the current direction.

For example in Figure 4.1, the horizontal x-axis is chosen as the reference direction. A node N_i at (x_1, y_1) selects a direction with an angle φ_1 counterclockwise from the reference direction. N_i travels towards the selected direction until it hits the boundary at point (x_2, y_2). N_i pauses there for t_p, then a new direction φ_2 is selected. The process continues until the simulation is over. This is the traditional model proposed by Hong and Rappaport [51]. If (x_2, y_2) and (x_3, y_3) are not located on the boundary, then the example shown is identical to the model proposed by Guerin [52]. If the new direction is calculated with relative changes to the current direction ($\varphi_{new} = \varphi_2 - \varphi_1$), the example then becomes a model proposed by Zanoozi [53]. The parameters are chosen with the following uniform distributions:

- $l_i = uniform[l_{min}, l_{max}]$, where l_i is the length between the current position and the destination point if we allow the node to select a destination along the chosen direction. l_{min} and l_{max} are the minimum and maximum length the node could select.
- $\varphi_i = uniform[0, 2\pi]$
- $v_i = uniform[v_{min}, v_{max}]$

Figure 4.3 shows an example path of one mobile node, which begins in the center of the simulation area, using the Random Direction Mobility Model. The dots in the figure illustrate when a mobile node reaches a border, pauses, and then choses a new direction. By allowing a node to travel to the boundary and then selecting a new direction, we achieve the scenario that a node passes the boundary, while a new node may enter the simulation area at that location traveling in a different direction some time later.

Since the mobile nodes travel to, and usually pause at the border of the simulation area, the average hop count for data packets using the Random Direction Mobility Model will be much higher than the average hop count of most other mobility models (e.g., Random Waypoint Mobility Model). In addition, network partitions will be more likely with the Random Direction Mobility Model compared to other mobility models. In this thesis, we adopt the Random Direction Mobility Model presented by Guerin [52], in which a node is allowed to select its destination along the direction randomly selected.

4.3 Gauss-Markov

The Gauss-Markov Mobility Model was first introduced by Liang and Haas [55] in order to adapt to different levels of randomness via one tuning parameter. Initially each mobile node is assigned a current speed and direction. At fixed intervals of time, n movements occur, and the speed and direction of each mobile node is updated accordingly. Specifically, the values of speed and direction at the n^{th} instance is calculated based on the values of speed and direction at the $(n-1)^{th}$ instant and a random variable using the following equations:

$$S_n = \alpha S_{n-1} + (1-\alpha)\overline{S} + \sqrt{(1-\alpha^2)}S_{x_{n-1}} \tag{4.1}$$

$$D_n = \alpha D_{n-1} + (1-\alpha)\overline{D} + \sqrt{(1-\alpha^2)}D_{x_{n-1}} \tag{4.2}$$

S_n and D_n are the new speed and direction of the mobile node at time interval n; α, where $0 \leq \alpha \geq 1$, is the tuning parameter used to vary the randomness; $\overline{S}$ and $\overline{D}$ are constants representing the mean values of speed and direction as $n \to \infty$; and $S_{x_{n-1}}$ and $D_{x_{n-1}}$ are random variables from a Gaussian distribution.

To ensure that an mobile node does not remain near an edge of the grid for a long period of time, the mobile nodes are forced away from an edge when they move within a certain distance of the edge. This is done by modifying the mean direction variable $\overline{D}$ in Equation 4.2. For example, when an mobile node is near the right edge of the simulation grid, the value $\overline{D}$ is changed to 180 degrees. Thus, the mobile node's new direction is away from the right edge of the simulation grid.

4.4 Manhattan Grid

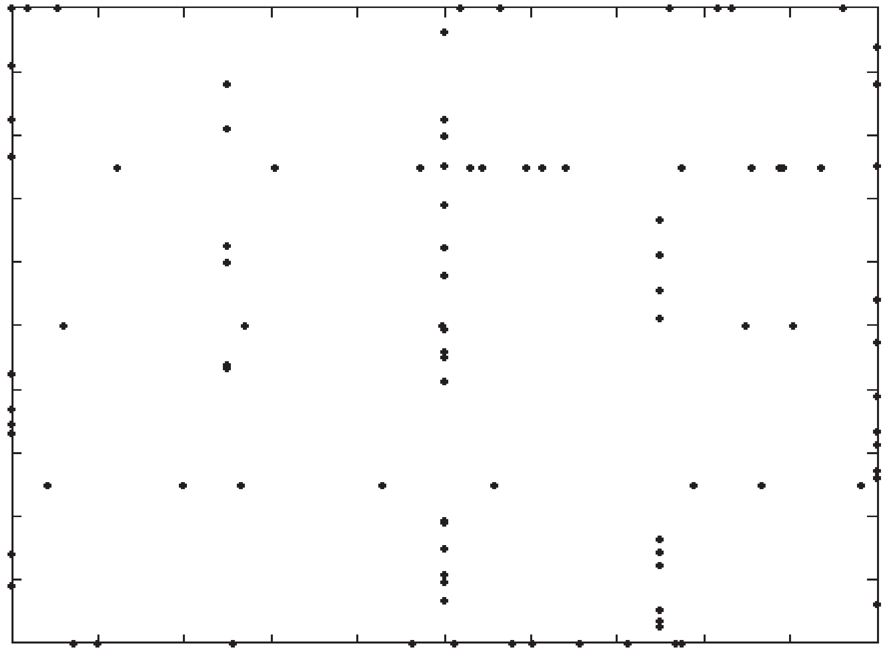

Figure 4.4: Example of Manhattan Grid Mobility Models

In the Manhattan Grid Mobility Model, the simulation area is a street network that represents a section of a city where the mobile ad hoc network exists [57]. The streets usually cross each other perpendicularly and form a grid in the downtown area of the city. Each node begins the simulation at a defined point on some street. A node then randomly chooses a destination, also represented by a point on some street. The movement algorithm from the current destination to the new destination locates a path corresponding to the shortest travel time between the two points, which is identical to the shortest distance between these two points along the streets. Upon reaching the destination, the node pauses for a specified time and then randomly chooses another destination (i.e., a point on some street) and repeats the process. Figure 4.4 shows an example of a Manhattan Grid Mobility Model. In this example, the whole area is divided into a 4×4 grid, and the nodes are allowed to move along the "streets" only.

The Manhattan Grid Mobility Model provides realistic movements for a section of a city since it severely restricts the traveling behavior of mobile nodes. In other words, all mobile nodes must follow predefined paths and behavior guidelines (e.g. traffic laws). In the real world, streets are usually separated by obstacles (e.g. buildings) in downtown area, and the wireless signals are hardly go through those obstacles. Therefore, mobile nodes can only interact with other nodes along the same streets without difficulties. If a node would like to communicate with another node in different streets, nodes, which is passing through the cross, must be the intermediaries to relay data for them.

The Manhattan Grid Mobility Model may be improved to simulate more realistic scenario in real world. For example, speed limits may be set to each individual streets and the direction of traveling may also be pre-defined. A node may have to pause for different time at the cross with respect to the direction it wants to turn. In addition, the model should be expanded to include a larger simulation area, an increased number of streets, a high-speed road along the border of the simulation area, and other novel path-finding algorithms. However, we are not going to expand the mobility model in this thesis.

4.5 Reference Point Group Mobility (RPGM)

The Reference Point Group Mobility (RPGM) model represents the random motion of a group of nodes as well as the random motion of each individual node

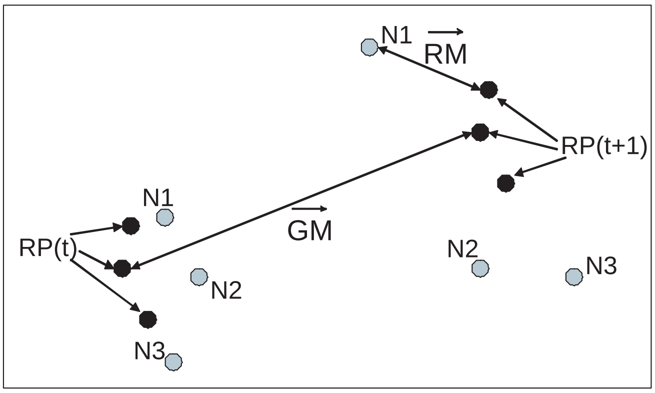

Figure 4.5: RPGM Mobility Models

within the group [58]. Group movements are based upon the path traveled by a logical center for the group. The logical center for the group is used to calculate group motion via a group motion vector, $\overrightarrow{GM}$. The motion of the group center completely characterizes the movement of its corresponding group of nodes, including their direction and speed. Individual nodes randomly move about their own pre-defined reference points, whose movements depend on the group movement. As the individual reference points move from time t to $t+1$, their locations are updated according to the groups logical center. Once the update reference points, $RP(t+1)$, are calculated, they are combined with a random motion vector, $\overrightarrow{RM}$, to represent the random motion of each node about its individual reference point.

Figure 4.5 gives an illustration of three nodes, $N_1, N2$ and N_3, moving with the RPGM model. The figure illustrates that, at time t, three black dots exist to represent the reference points, $RP(t)$, for the three nodes. As shown, the RPGM model uses a group motion vector $\overrightarrow{GM}$ to calculate each node's new reference point, $RP(t+1)$, at time $t+1$; as stated, $\overrightarrow{GM}$ may be randomly chosen or predefined. The new position for each node is then calculated by summing a random motion vector, $\overrightarrow{RM}$, with the new reference point. The length of $\overrightarrow{RM}$ is uniformly distributed within a specified radius centered at $RP(t+1)$ and its direction is uniformly distributed between 0 and 2π.

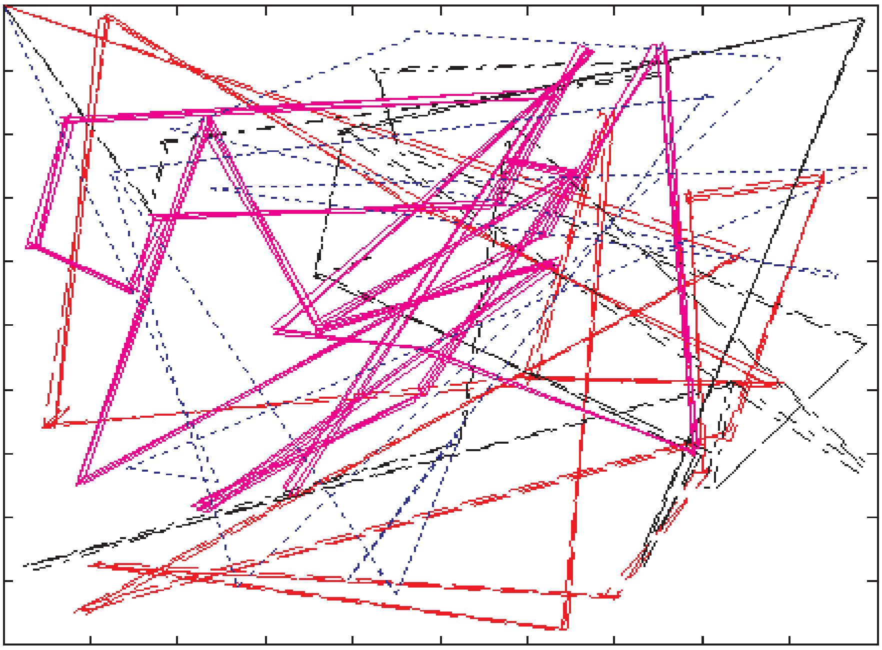

Figure 4.6: Example of RPGM Mobility Model

Figure 4.6 shows an example of RPGM. In this example, the mobile nodes are teamed into four groups. The movement of each group is indicated by a particular color. There are different number of mobile nodes in different groups. Both the movement of the logical center for each group, and the random motion of each individual node within the group, are implemented via the Random Waypoint Mobility Model. The only difference is that individual nodes do not use pause times while the group is moving. Pause times are only used when the group reference point reaches a destination and all group nodes pause for the same period of time. The RPGM model is designed to simulate scenarios such as an avalanche rescue. During an avalanche rescue, the responding team consisting of human and canine members work cooperatively. The human guides tend to set a general path for the dogs to follow, since they usually know the approximate location of victims. The dogs each create their own “random” paths around the general area chosen by their human counterparts.

4.6 Discussions of Mobility Models

In this chapter, we introduce five mobility models and our cache schemes are evaluated for each mobility models by simulations. In this section, the differences and similarity of these five mobility models will be discussed briefly.

The Random Waypoint Mobility Model is the most popularly used model within mobile ad hoc network research community. This is mainly because the model is easy to design and it is supported by NS-2 and Glomosim. It is an excellent model to simulate full randomness of nodes in mobile ad hoc networks, such as their destination to go and their speed of traveling. Based on the full randomness characteristic, a node may travel anywhere and most likely to a location with totally different interests. Therefore, if a node only caches data according to its own interest, the cached data may be not useful at all when it travels to another “local” grid as defined in Section 2.4. Therefore, if a node may spend part of its memory to store data for others, even when it leaves its current grid, the cached data may be “hot” to other nodes in another grid. Finally, the whole network will benefit from the contribution of each individual.

Handover and data replacement policies will be significant in a network with Random Waypoint Mobility Model. As discussed in the previous paragraph, a node may leave its current grid and it may not visit the same grid shortly. If the node may handover those cached “hot” items to its neighbors, who are still traveling nearby or in the same grid, the data handed-over will keep helping others to improve their data accessibility. The data handed-over may be dropped to make more memory space to cache other useful data when it enters another grid. Therefore, we can predict that our proposed RC and RC-H will be useful in this scenario.

Similar to Random Waypoint Mobility Model, the Random Direction Mobility Model and Gauss-Markov Mobility Model are random mobility models with different level of randomness. Therefore, the caching schemes will perform similarly. RC and RC-H will outperform other schemes in a network with the Random Direction Mobility Model or Gauss-Markov Mobility Model.

The Manhattan Grid Mobility Model is similar to Random Waypoint Mobility Model. The difference is that the destinations and the traveling routes must be on the pre-defined streets. Also due to the limits of space to travel (all alone the street), the chance for a node to re-enter its previous grid is high. Therefore, the handover scheme may not be that useful like in a network with Random Waypoint Mobility Model. But we still believe that RC will outperform SC and selfish caching scheme.

The Reference Point Group Mobility (RPGM) model represents the case where nodes are divided into groups and travel as teams. On one hand, as nodes in one group travels together, their interests will be similar wherever they go. On the other hand, nodes in different groups hardly communicate with each other because the time they come across each other is short unless the two groups are traveling with similar routes. However, the chance for two groups to travel with similar routes is very small. Therefore, it may not be very useful to cache data for others since everybody in the same group may have the same kind of interest. A node caching its own interested data will be good enough to serve the requests among its own group. The SC and RC will achieve similar performance in such a network.

Chapter 5

Performance Evaluation

Simulations are carried out to evaluate the performance of the proposed schemes. The experiments presented are designed with two objectives. First, we study the performance of our caching schemes over a typical ad hoc network with Random Waypoint Mobility Model. Second, we evaluate the impact of mobility models and find the most suitable caching scheme for the network with different mobility models.

5.1 Simulation Model and system parameters

All simulation experiments are done on a discrete-event network simulation model built on top of Parsec [63]. In our simulations, a mobile ad hoc network with a group of N mobile nodes is constructed over an $X \times Y$ geographical square-area. A single data server is located at the center, which initially contains M data items. N mobile nodes are randomly located in the network with a transmission range R. Their movements follow a specific pattern according to the mobility model applied. The mobility scenarios are generated with a tool called the BonnMotion package [60]. This package includes Random Waypoint, Gauss-Markov, Manhattan Grid and

RPGM. Since Random Direction Mobility Model is not included in BonnMotion package, we built a stand-alone mobility generator for it.

Each mobility model may have a set of parameters to determine the mobility scenarios generated. For example, the pause time (t_p) and speed domain ($[v_{min}, v_{max}]$) will determine the results of a Random Waypoint mobility model generation process. Similarly, for the Random Direction mobility model, the direction (φ), the pause time (t_p) and the length a node may travel along the chosen direction are significant parameters. However, in this thesis, we focus on the impact of our caching schemes with different mobility models with similar scenario generating configurations. The effect of our caching schemes on a single mobility model with different parameter settings are not discussed in detail. More research work may be done in the future.

The simulations are run for T time units and the request generation process follows a poisson distribution with the average inter-arrival time varying from 1 to 100 time units. Each node has a cache memory size of s_i to store received data, and the data items in the network are sized uniformly between o_{min} and o_{max} in terms of KB. The routing protocol employed in this thesis is rather simple – the shortest route with Dijkstra's algorithm. The wireless interface has a link capacity of 2 Mb/s, which is the setting used in [25]. A timeout T_{out} is implemented to drop a request if it is not served successfully within T_{out}.

The whole $X \times Y$ simulation area is divided into G grids and the M data items are classified into four categories – GH, GC, LH and LC. Among the M data items, 10% of them belongs to GH, and 10% of them belongs to GC. The remaining 80% are categorized as "local" data, and they are set to LH and LC according to the id of each local grid. Each local grid randomly selects 20% of that remaining 80% data items as their local interest and each grid shares 30% of its LH with its adjacent grid. For each node, the chance to access GH and LH is 70%, and leaves 30% chance to those GC and LC data.

The requests are broadcasted if a node cannot find a replica in its own memory. A flag is patched at the tail of the query packet with the requesting node's id, the query data id and number of hops traveled. A node receiving a request first checks its memory. If the data is not found. the hop number is increased by 1. If a request for the same data from the same requesting node with smaller hop number has been relayed out, the intermediate node drops the request received. If the data is found by an intermediate node, it will send back the data to its neighbor, from whom it received the request. If any node receives the reply data item and finds the same data has been sent to the same node, the received data item is dropped. The requests are dropped by all the relay nodes after T_{out}.

Some system parameters are common in every mobility model, such as the dimension of the simulation network, the number of data items and so on. We call these parameters global simulation parameters, and the default settings for them are shown in Table 5.1.

There are some other parameters, that are particular to only one mobility model, such as the direction φ in Random Direction Mobility Model. These parameters are shown in Table 5.2 with the relevant mobility model indicated.

In Figure 5.1, we describe the main process of event generation in all the simulation experiments. The mobility models are represented by the movement event files, which are pre-generated by the Bonnmotion package [60]. Therefore, the event generation process shown here is a general approach with pre-defined configuration files and movement event files. The symbols used in this algorithms are listed below:

- T : Total Simulation Time

Table 5.1: Global Simulation Parameters

Parameter	**Description**	**Default Value**	**Range**
N	Number of Mobile Nodes	2000	
X [m]	Simulation Area Width	2000	
Y [m]	Simulation Area Height	2000	
G	Number of Local grids in the network	25	
T [s]	Simulation Duration	60000	
R [m]	Radio Range	50	
λ [query/100s]	Mean Query Rate	10	1 to 100
M	Number of Data Items	1000	
o_{min} [KB]	Minimum Size of Objects	1	
o_{max} [KB]	Maximum Size of Objects	10	
s_i [KB]	Size of Cache Memory in Node i	800	10 to 2000
t_p [s]	Pause Time	300	
v_{min} [m/s]	Minimum Movement Speed	1	
v_{max} [m/s]	Maximum Movement Speed	20	1 to 50
T_{out}	Timeout period	1000	

Table 5.2: Specific Simulation Parameters for different mobility models

Parameter	**Description**	**Value**	**Mobility Model**
φ	Direction	[0 to 2π]	Random Direction
l_i	distance to destination	$[l_{min}, l_{max}]$	Random Direction
l_{min}	minimum distance to destination	100	Random Direction
l_{max}	maximum distance to destination	2500	Random Direction
mg_x	Number of grids in x-axis	10	Manhattan Grid
mg_y	Number of grids in y-axis	10	Manhattan Grid
ng	Number of groups	20	RPGM
$R_{\overrightarrow{RM}}$	Radius of any reference point	100	RPGM

- t_c : Current Event Time
- t_n : Next Event Time
- t_m : Next Movement Event Time
- t_l : Next Link Change Event Time
- t_r : Next Request Event Time
- t_h : Next Handover Event Time
- (P_x, P_y) : Current Position of a node
- (D_x, D_y) : Destination of the node's next movement
- t_a : Time to get the destination
- v_i : Speed of next movement

- $R_{blocked}$: Number of Blocked Request

When a movement event is received, the related node updates its position and destination. The time to get the destination is calculated with respect to the speed v_i. When a link change event is received, the network map is updated and routes are calculated based on the updated map later when a request generated. The number of blocked request is increased by 1 if a timeout event is received. When a data request is served successfully, a set of nodes caches the received data in their memory according to the caching scheme employed. In Selfish Cache, only the requesting node caches the data, as well as SC and SC-H. In RC and RC-H, the caching nodes are selected by comparing the hop number away from the requesting node with λ and H_i.

5.2 Performance Metrics

Since we focus on improving data accessibility in this thesis. The performance of our proposed caching schemes are examined in terms of Data Blocking Ratio and Energy Consumption. The details of these metrics are defined as the following.

$$P_b = \frac{R_{blocked}}{R_{tot}} \tag{5.1}$$

Where P_b is the data blocking ratio; $R_{blocked}$ is the number of blocked requests and R_{tot} is the total number of requests.

Since the energy consumed while the network is operating is mainly by forwarding data packets to one another, the storage cost and other insignificant energy cost are neglected. We adopt the linear model proposed by Freeney [59] in our calculation of energy cost. Each time a data transmission event is triggered, a certain amount of power is consumed from each node along the route from the source to the

```
Step 1 Load Configuration File & Movement Event File.

Step 2 Generate Location-dependent data access events.

Step 3 Start

    While Current Time < Simulation Time ($t_c < T$)
        If (Received Event = Movement Event )
            Update Node: [Pos($P_x, P_y$), Dest($D_x, D_y$), Arrival Time $t_a$; Velocity $v_i$];
            $movementIndex = movementIndex + 1$;
            Get $t_n$
        End if
        If (Received Event = Link Change )
            Update the Connection Map;
            $linkChangeIndex = linkChangeIndex + 1$;
            Get $t_l$
        End if
        If (Received Event = Request )
            Get $t_r$
        End if
        If (Received Event = Timeout )
            The request will be dropped.
            $R_{blocked} = R_{blocked} + 1$
        End if
        $T_n = min(t_m, t_l, t_r, t_h)$;
        $T_c = T_n$
    End while
```

Figure 5.1: Event Generation Process

destination. The following equation shows the cost calculation method:

$$E = C_u \times S \times H \tag{5.2}$$

where, S is the size of data packet needed to relay. C_u is the energy cost to relay one unit (e.g. 1KB) of data. E is the total energy cost for this particular transmission. H is the number of hops between source and destination. Energy Consumption is used to show the impact of changes in network characteristics on the energy spending to serve the requests. Energy consumption level is also an indicator to the average number of Hops.

5.3 Results and Discussions

A set of experiments for the same network are carried out. The mobile nodes randomly generate different requests in each run according to the same access generating mechanism. A number of simulation experiments has been conducted and the simulation results are close to one another if the experiment time is 50000 seconds or more. Since the results in the first 10000 seconds are not taken into account so that the effect of initial state is avoided. In our simulations, each experiment lasts for 60000 seconds in simulation time in order to make sure that the results collected are reasonable and have converged.

5.3.1 Tuning H_i & Γ

As discussed in section 3.1.3, H_i and Γ are used to determine the caching pattern in the RC scheme. If H_i is bigger, the distance between the destination and the first caching relay node is bigger in terms of hop number. When Γ is bigger, the distance between subsequent caching nodes along the path will become bigger and bigger. Therefore, it is necessary to find the most suitable values for H_i and Γ

before checking the impact of other parameters, such as memory size, query rate, etc. Random Waypoint Mobility Model is used in this section.

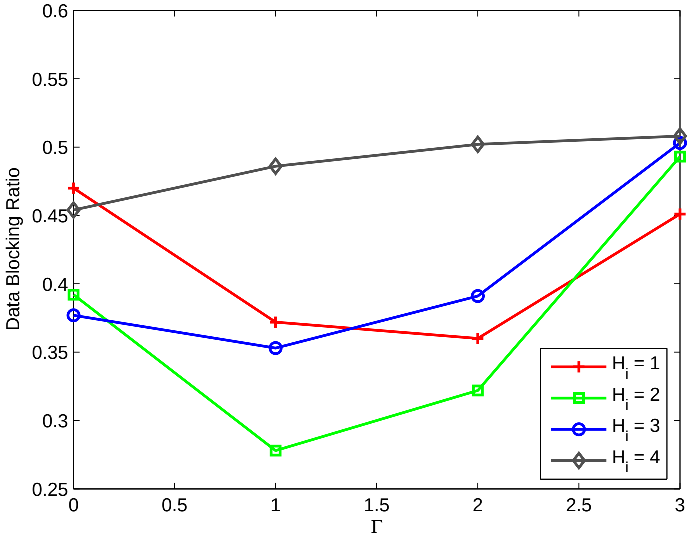

Figure 5.2: Data Blocking Ratio vs. Gamma (Γ)

Figure 5.2 shows the function of "Data Blocking Ratio" and Γ. We verified the performance with $H_i = 1$,$H_i = 2$, $H_i = 3$ and $H_i = 4$, respectively. When H_i is small, the data blocking ratio first decreases then increases again as Γ is varied from 0 to 3. But when $H_i = 4$, the data blocking ratio does not change very much as the value of Γ varies. This can be explained in the following way. When H_i is big, the two adjacent caching nodes are far from each other and they are most likely lying on different grids with little common interests. Therefore, the cached data are not able to help serve the requests from the requesting node or nodes close to the requesting nodes. Since the caching node may be far from the requesting node, the chance for the same request reach the caching node is low. The data blocking ratio is not influenced by the caching nodes very much. As Γ increases, the data blocking ratio are approaching each other with different settings of H_i.

This may be because when Γ is big, the distribution of caching node will become similar regardless of H_i.

When Γ is small, the data blocking ratios of RC with $H_i = 2$ and $H_i = 3$ are smaller than H_1. This is because when Γ is small, more replicas may be cached by neighboring nodes if $H_i = 1$. If more than one replicas are cached by one-hop neighbors, those replicas will not help to improve the data accessibility. On the other hand, the memory spent to store the same replicas within one-hop neighbors are considered as waste, and in turn fewer useful data are cached. The data blocking ratio becomes higher in this situation. Therefore, it is good to skip caching the data in one-hop neighbors if one has a copy of it.

By adjusting the settings of H_i and Γ, we are able to achieve uniform and non-uniform distribution of replicas over the network. For example, by setting $\Gamma = 0$, we achieve a uniform distribution of replicas with each replica being H_i hops away from each other. By setting $\Gamma > 0$, a non-uniform distribution is achieved. The first caching node is H_i hops away from the requesting node and the subsequent caching nodes are separated from the previous one by $H_i + \Gamma \times h$, where h is number of caching nodes from the requesting node along the serving path, except the requesting node itself. We can easily observe that the non-uniform distribution is better than the uniform distribution when Γ is small, which indicates that the increment of hop distance between two adjacent caching node is small. However, when Γ is bigger than H_i, uniform distribution outperforms non-uniform distribution in terms of data blocking ratio. This is because the distance between two adjacent caching nodes becomes too big to help serving the requests. For example, $H_i = 2$ and $\Gamma = 3$, if a route is 8 hops length, there will be 4 caching nodes with a uniform distribution. However, there will be only 2 caching nodes with a non-uniform distribution. It is very obvious that uniform distribution is better in this case. From this discussion and observation from the simulation results, we would like to choose a small Γ with non-zero value and a H_i with a value bigger than 1.

From the results, we can easily see that the network with $H_i = 2$ and $\Gamma = 1$ gives the lowest data blocking ratio, which stands for the best performance. Furthermore, by setting $H_i = 2$, we achieve the objective to avoid caching the same data in two adjacent nodes, and the caching memory is used more effectively. Hence, in the following sections, we use $H_i = 2$ and $\Gamma = 1$ in the RC and the RC-H schemes where the latter denotes a scheme with cache handover.

5.3.2 Effects of Giving Priority to Smaller Size File

As mentioned in section 3.3, we give higher priority to smaller data items to be cached by mobile nodes and bigger files will be replaced first when the memory is full. In this section, the benefit of giving priority to smaller data is shown.

Figure 5.3 shows the function between data blocking ratio and the memory size for the SC-H and RC-H schemes with/without giving priority to smaller data items, respectively. The mobility model employed is the Random Waypoint Mobility Model. By giving priority to smaller data items, more effective data items could be cached in local memory. As a result, more requests are able to be served locally or by some nearby caching nodes, then the overall data blocking ratio is reduced and the performance of the whole network is improved. The significance of giving priority to smaller data items is more apparent as the memory size increases. The reason is that with bigger memory, the increase in number of cached data items is faster for caching smaller size data than caching bigger size data. Therefore, more requests can be served locally and the data blocking probability is improved.

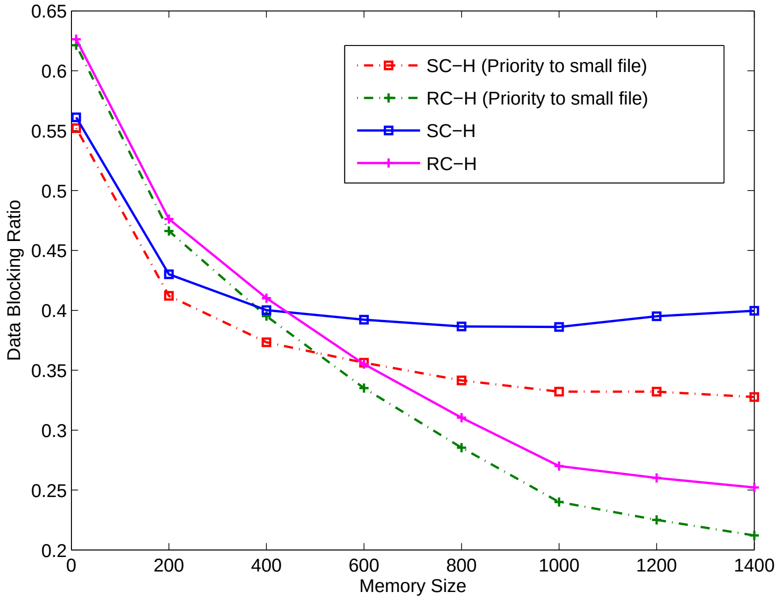

Figure 5.3: Data Blocking Ratio vs. Memory Size (Effects of Giving Priority to Smaller Size File)

5.3.3 Results for Random Mobility Models

The Random Waypoint Mobility Model, Random Direction Mobility Model and Gauss Markov Mobility Model are similar to each other except the minor difference in terms of randomness. Our proposed caching schemes gain similar performance improvements in the network with these three Random Mobility Models as mentioned before. Without loss of the specificity, we will present the experiments results for the network with Random Waypoint Mobility Models as a representative of the performance of our caching schemes in a network with Random Mobility Models. In the following sections, we present the effects of memory size, request generating time and the node moving speed on the performance of our caching schemes.

Effects of Memory Size

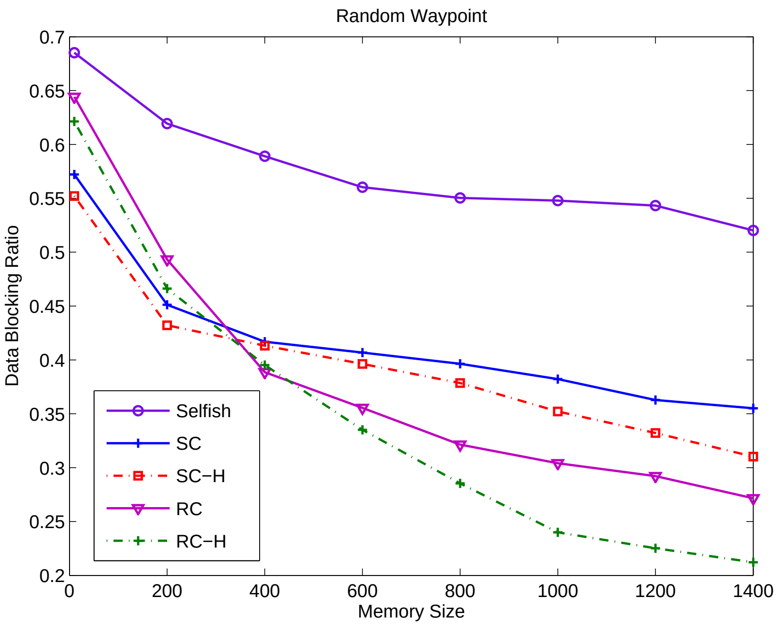

Figure 5.4: Data Blocking Ratio vs. Memory Size (Random Waypoint)

Figure 5.4 shows performance in terms of data blocking ratio against the cache memory size. By employing our proposed caching schemes, the cached data can be used, not only by the caching nodes themselves, but also by other mobile nodes in the network. In contrast, in the Selfish Caching scheme, the cached data is used to serve its own requests only. Because of the cooperation of mobile nodes to serve requests among each other in our proposed caching schemes, the data blocking ratio for our proposed caching schemes are much lower than that for Selfish Caching, which is easily observed in Figure 5.4. In the SC and the SC-H schemes, we only encourage the requesting nodes to cache the received data if the source is at least H_i hops away. However, in the RC and the RC-H schemes, mobile nodes along the path between the source and destination make use of part of their memory to store data for others. Therefore, when the cache memory is small, the SC and the

SC-H schemes are able to use their cache memory more effectively. The RC and the RC-H schemes show significant improvement as the cache memory becomes big. The RC scheme gains more than 30% performance improvement compared to the SC scheme when the cache memory is 800KB and above. Handover policy is always able to help both the SC and RC schemes to gain better performance. The performance improvement is small when the cache memory size is small. However, as the cache memory size increases, the performance improvement becomes more significant. From Figure 5.4, we can observe that the handover policy increases the performance by more than 20%. The RC-H scheme outperforms others when cache memory size is more than about 400KB.

Effects of Request Generating Time

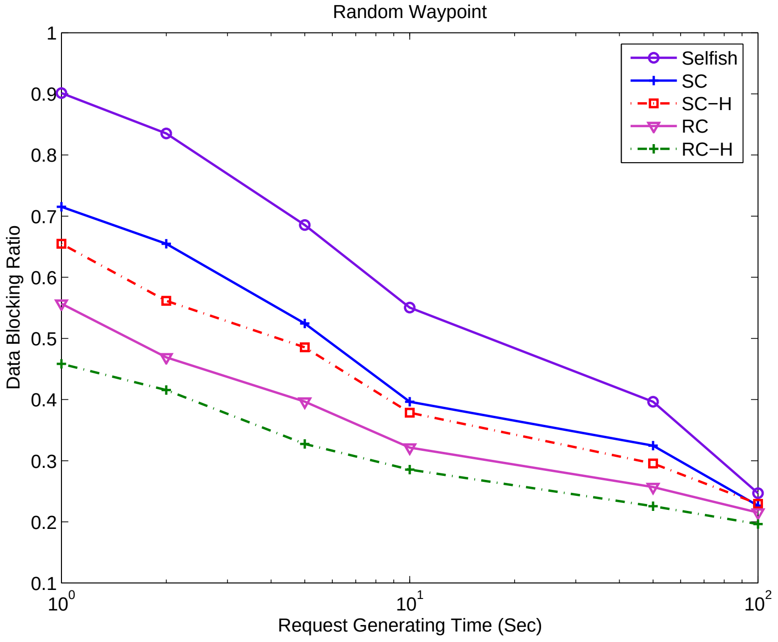

Figure 5.5: Data Blocking Ratio vs. Request Generating Time (Sec) (Random Waypoint)

The request generating rate is a fundamental challenge to the capacity of a mobile ad hoc network. In this section, we examine the effects of request generating time to our proposed caching schemes. Figure 5.5 shows data blocking ratio performance against the request generating time. As the request generating time decreases, the request generating rate increases. The data blocking ratio decreases as the request generating time increases. This is because when the request generating time is large, fewer requests are generated within a given time period. The wireless channels are not congested and the data requests maybe served by data server or other caching nodes. When the request generating time decreases, more requests are generated within same period of time. The wireless channel becomes more congested and more requests will be pending in the queue. Since the timeout period is fixed in our simulation experiments, more requests will not able to be served within the timeout period. Therefore, the data blocking ratio increases as the request generating time decreases.

The RC-H scheme always outperforms other caching schemes for the entire range of request generating time. The RC scheme achieves the second best performance, followed by the SC-H and SC schemes. Selfish caching is the worst since mobile nodes only use their cached data to serve their own interests. When the request generating time is small, which means a huge number of requests generated within a short period, our proposed relay caching schemes (RC & RC-H) are much better than the simple caching schemes and selfish scheme. This is because in the RC and RC-H schemes, mobile nodes share part of their cache memory to store data for others. As a result, within a group of mobile nodes, the chance for them to find a particular data item nearby is much higher, which in turn reduces the data blocking ratio. The handover scheme helps to retain the data at a location when the caching node leaves, which increases the service success probability in a local perspective. When we make this achievement in different locations, we can gain performance improvement from a global perspective. This difference between the performance gains using different caching schemes becomes smaller when the request generating time increases. The reason is that with lower request rate, fewer mobile nodes will access the same data within a period of time. The chance for a request served locally or by some nearby caching nodes becomes lesser. The data server will take more responsibility to serve the requests. Therefore, the data blocking ratios for different caching schemes become closer to one another.

Effects of Speed

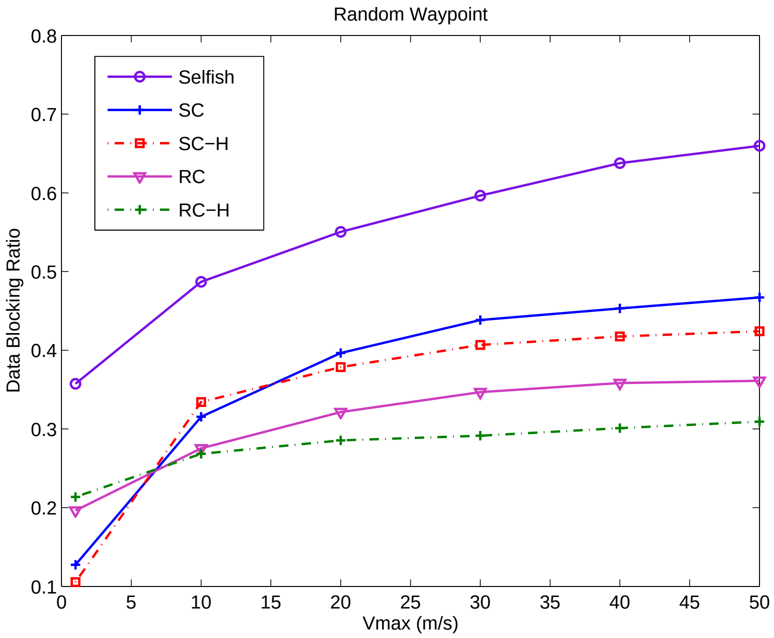

Figure 5.6: Data Blocking Ratio vs. Vmax (m/s) (Random Waypoint)

In this section, we examine the effects of moving speed of mobile nodes on the performance of our caching schemes. Besides data blocking ratio, we present the impact of moving speed on the total energy consumption throughout the simulation. Figure 5.6 shows performance in terms of data blocking ratio against the

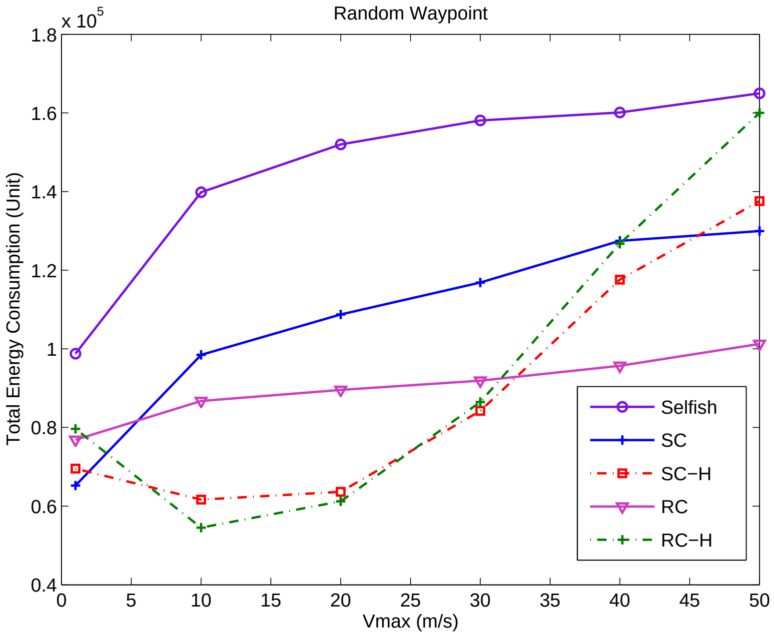

Figure 5.7: Total Energy Consumption vs. Vmax (m/s) (Random Waypoint)

maximum moving speed of mobile nodes. Figure 5.7 shows performance in terms of the total energy consumption against the maximum moving speed.

From Figure 5.6, we can observe that when V_{max} is small, the SC and SC-H schemes outperform the RC and RC-H schemes in terms of data blocking ratio. The reason is that when V_{max} is small, mobile nodes hardly move far away from their initial location. Because of the location-dependent data access pattern we employed in this thesis, the interested data set for a mobile node is attached with the location. As a result, mobile nodes seldom change their interests and the requesting nodes will cache the received data and these cached data are most likely accessed by themselves or some nodes nearby for the rest of simulation time. However, since we allow some intermediate nodes along the path between source and destination to cache the data in the RC and RC-H schemes, which is seldom accessed later and this leads to a situation that the cache memory is not used effectively. As a result, the data blocking ratios using the RC and RC-H schemes are higher than those for the SC or SC-H schemes. On the other hand, when moving speed increases, the chance for a mobile node to move out of its current grid is much higher, which results in changes in the data items of interest to a mobile node in different locations/grids. In this situation, the significance of RC and RC-H becomes more apparent. From Figure 5.6, the RC scheme outperforms the SC scheme by more than 30% when V_{max} is greater than 20 m/s. Handover policy further improves performance by around 20%.

The total energy consumption increases as V_{max} increases and it will not increase much while the speed is fast except for the SC-H and RC-H schemes. When moving speed is small, a big portion of requests are served locally or by some nearby caching nodes. Therefore, the total energy consumption is low. As the moving speed increases, the local serving probability decreases, as a result, the total energy consumption increases. However, in the SC-H and RC-H schemes, while the caching nodes leave their original grid, they relocate the cached data to their neighbors in that grid, thus maintaining the local data accessibility and the local serving probability. The total energy consumption decreases instead of increases. However, when the moving speed keeps increasing, handover events will be more frequent and it costs energy to hand data over to neighbors. Therefore, the caching schemes with handover (SC-H and RC-H) expend more energy than those schemes without handover policy. Since energy is an important resource for a mobile nodes, we suggest to disable the handover function while mobile nodes are moving at a very high speeds, but this will be at the expense of improvements elsewhere, such as data blocking ratio.

5.3.4 Results for Manhattan Grid Mobility Model

In a network with Manhattan Grid Mobility Model, all mobile nodes are traveling along the horizontal or vertical streets. Due to the specific node location arrangement, all communications are done through the paths established along the streets, too. Without employing any caching scheme, the server will take the responsibility of serving every request generated within the network. This may be a huge burden to both the data server and the mobile nodes near to the server. Since the data server is the only source node, the channels are easily used up when many requests are sent to it. The energy consumption of the data server and the surrounding nodes are extremely large too. Therefore, caching schemes are very useful in such a network with Manhattan Grid Mobility Model. In this section, we are going to present the experimental results to evaluate the performance of our proposed caching schemes.

Effects of Memory Size

Figure 5.8 shows data blocking ratio performance against the cache memory size. Similar to Figure 5.4, the data blocking ratio decreases as the cache memory size increases. However, the data blocking ratio approaches its steady state at a faster pace. To be more specific, the data blocking rate does not change very much as the memory size approaches 400KB or more. A similar steady state is achieved in the network with Random Waypoint Mobility Model as memory size approaches 800KB to 1000KB and more. Therefore, less memory size is required to gain competitive performance in a network with Manhattan Grid Mobility Model. Both the RC and SC schemes outperform the Selfish Cache scheme by a large margin. The system gains 30% more performance improvement using the RC scheme than using the SC scheme. Different from Figure 5.4, the benefit of handover is not significant. This is because of the characteristic of Manhattan Grid mobility pattern. The chance

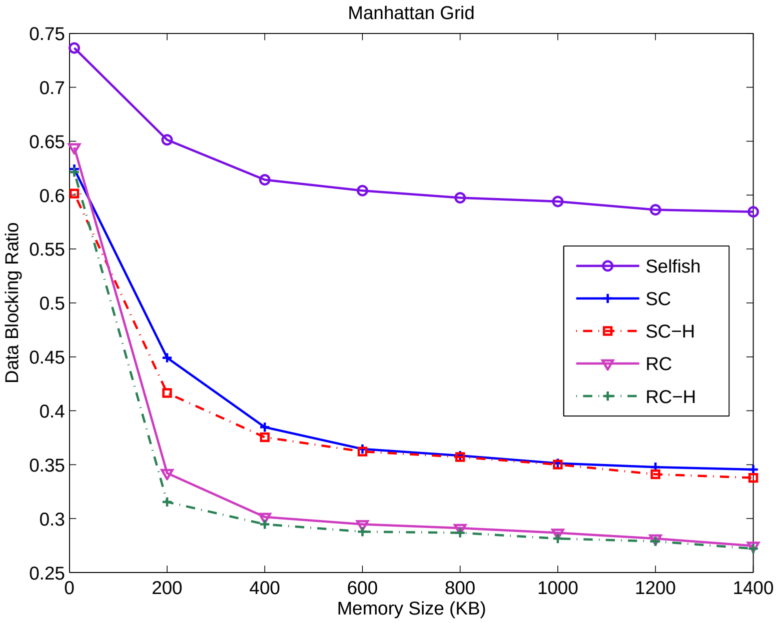

Figure 5.8: Data Blocking Ratio vs. Memory Size (Manhattan Grid)

for a node that leaves its previous grid to re-enter that grid is very high. Those data handed over to others shortly before, becomes “hot” again. As a result, the handover will not help very much in this kind of network. Therefore, mobile users may be highly encouraged to disable the handover function when they are traveling in a network with Manhattan Grid Movement Pattern.

Effects of Request Generating Time

Figure 5.9 shows data blocking ratio performance against the request generating time. The data blocking ratio decreases as the request generating time increases. This is because less requests are generated within the same period and more available wireless channels can be used to serve the requests. As the request generating time decreases, more requests are generated, the wireless channels become congested and more requests are blocked because they cannot be served within the

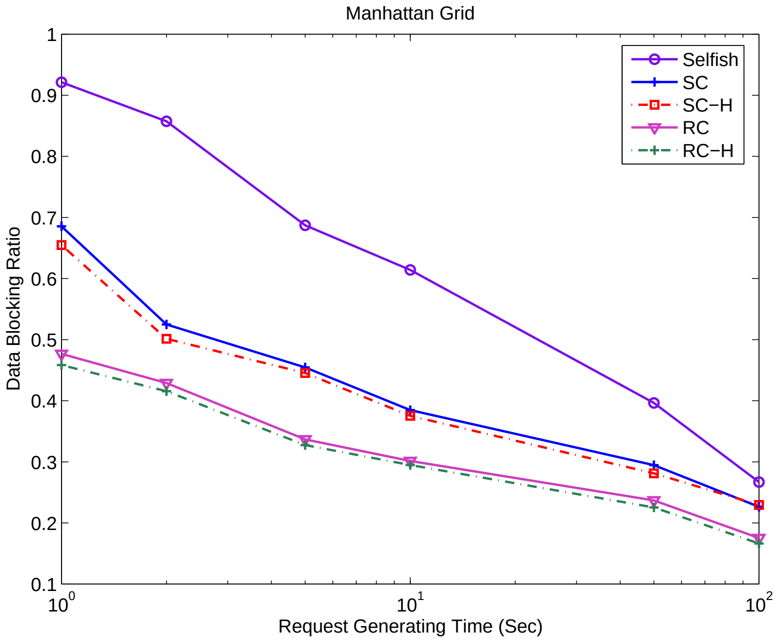

Figure 5.9: Data Blocking Ratio vs. Request Generating Time (Sec) (Manhattan Grid)

timeout period. This is similar to the results in the network with the Random Waypoint Mobility Model. The RC-H scheme outperforms others all the time and both the RC and SC schemes outperform the Selfish Cache scheme by a significant amounts. In a dense request situation, the network using the RC scheme is able to gain about 30% performance improvement over the SC scheme. The performance gain becomes smaller as the request generating time increases. Similar in the previous section, the benefit of handover policy is not significant and the reason has been discussed in the previous section. Hence, in an ad hoc network with Manhattan Grid Mobility Model, the RC scheme is a good choice to achieve satisfactory performance.

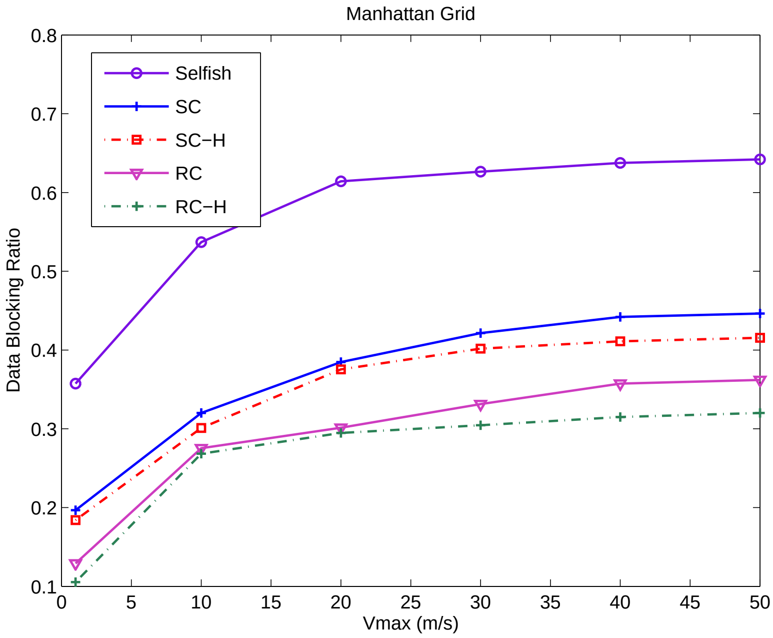

Figure 5.10: Data Blocking Ratio vs. Vmax (m/s) (Manhattan Grid)

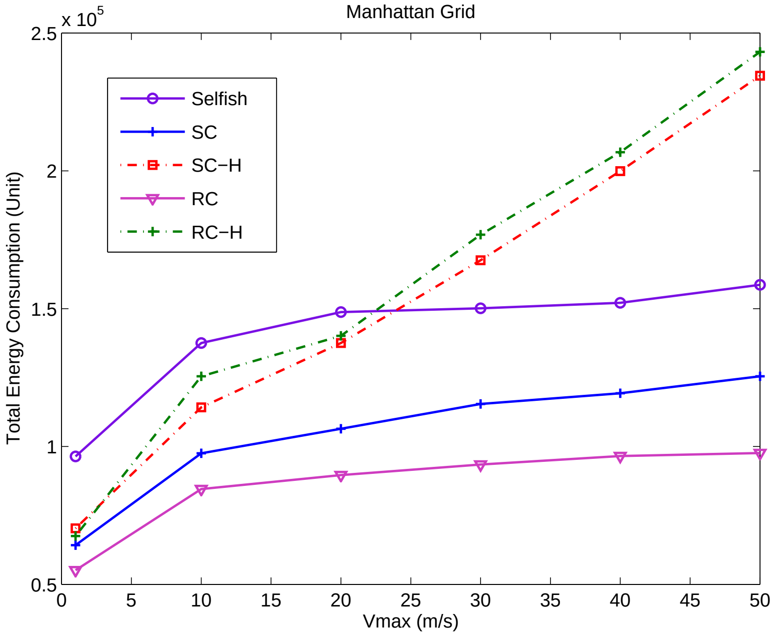

Figure 5.11: Total Energy Consumption vs. Vmax (m/s) (Manhattan Grid)

Effects of Speed

Figure 5.10 shows data blocking ratio performance against the maximum moving speed of mobile nodes. Figure 5.11 shows total energy consumption against the maximum moving speed.

As shown in Figure 5.10, the data blocking ratio increases as the moving speed of mobile nodes increases. At any speed, the SC and RC schemes are much better than the Selfish Cache scheme. The RC scheme outperforms the SC scheme by about 30%. Handover policy is able to gain about 10% performance improvement in a fast moving network. Similar to the data blocking ratio, total energy consumption increases as the moving speed increases as shown in Figure 5.11. At slow speeds, the SC-H and RC-H schemes consume similar energy like the SC and RC schemes. However, the energy consumption in the SC-H and RC-H schemes increases as the speed of mobile nodes increase. This is due to additional energy consumed when handing data over to neighboring nodes. The faster a node travels, the more frequent for it leaves a grid, which may cause it to hand over cached data items to its neighboring nodes. Therefore, in order to save energy and achieve competitive performance, we would like to employ the RC scheme in a network with the Manhattan Grid Mobility Model.

5.3.5 Results for RPGM Mobility Model

In a network with RPGM Mobility Model, mobile nodes are divided into several groups and nodes belonging to one group move together all the time with very small relative displacement. Therefore, the impact of our proposed caching scheme may be very different from a network with other mobility models. The details are presented in the following sections.

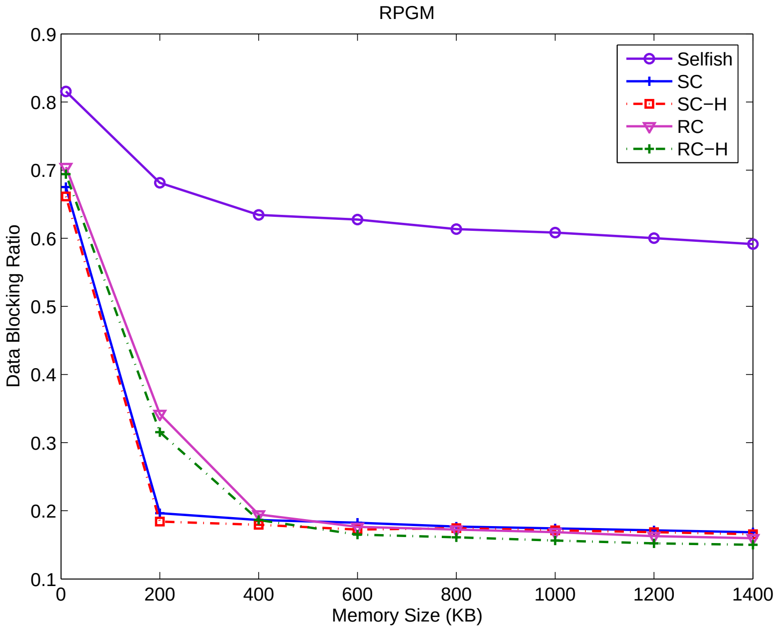

Figure 5.12: Data Blocking Ratio vs. Memory Size (RPGM)

Effects of Memory Size

Figure 5.12 shows data blocking ratio against the memory size. The SC and RC schemes always outperform the Selfish Cache scheme. The SC scheme outperforms the RC scheme when memory size is small. This is because nodes share part of their memory to store data for others while in the RC scheme more than one node may store the same data within the same group, which may reduce the percentage of important data in the memory. As the memory size increase, the RC scheme achieves similar performance to the SC scheme, and the data blocking ratio for both schemes are very low. The reason that the RC scheme is not able to outperform the SC scheme by a large margin is that network topology does not change frequently when mobile nodes are moving as a group. Because of the lack of connection with the data server, mobile nodes in any group hardly access data from the data server directly. Therefore, in the beginning of a simulation, the data blocking events happen very frequently. However, as long as a data is received from the server, the cached node will be able to serve the requests from nodes in the same group. In our simulations, we ignore the events in the first $\frac{1}{6}$ of simulation time, which significantly reduces the effect of the initial state problem. The number of copies of the same data in a group does not affect the data blocking ratio very much. Therefore, the RC scheme is only able to achieve similar performance improvement as the SC scheme. Since mobile nodes are moving as groups, the nodes in one group may leave a grid together. Therefore, handover policy does not help very much in this situation. Hence, the SC scheme is the best recommended caching scheme in an ad hoc network with RPGM Mobility Model.

Effects of Request Generating Time

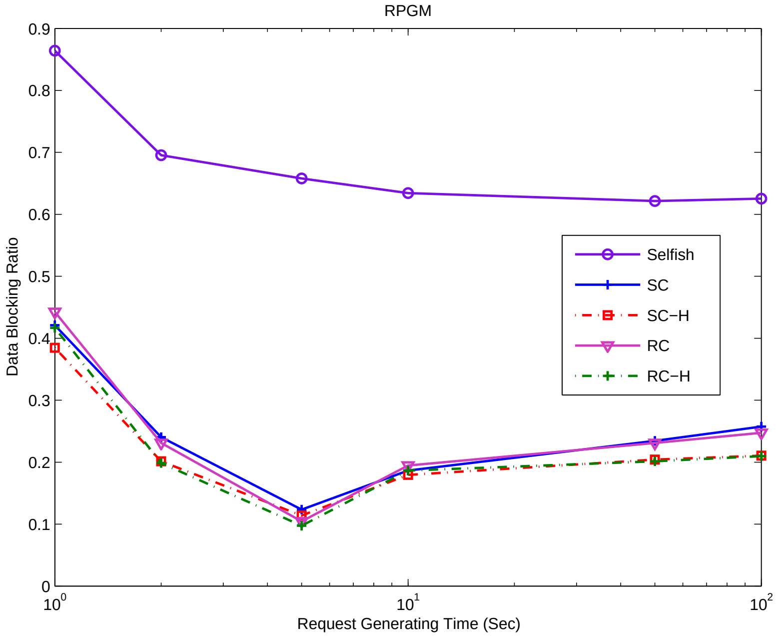

Figure 5.13: Data Blocking Ratio vs. Request Generating Time (Sec) (RPGM)

Figure 5.13 shows data blocking ratio against the request generating time. The data blocking ratio decreases when the request generating time decreases until some level. Then the data blocking ratio increases as the request generating time decreases further. This can be explained as follows: When the request generating time is large, fewer requests are generated within a given time period and most of the requests are served locally or by some caching nodes within the same group. When the request generating time decreases, the chance for the same data being requested becomes higher and the portion of successful requests will be more, which in turn will decrease the data blocking ratio. However, when the request generating time is too small, many requests will result in channel congestion in a group, which will reduce the performance in terms of data blocking ratio again. Similar to the previous section, the SC and RC schemes significantly outperform the Selfish Cache scheme and the SC, SC-H, RC and RC-H schemes are close to one another. Hence, SC is the best choice in such a network.

Effects of Speed

Figure 5.14 shows data blocking ratio against the maximum moving speed of mobile nodes. Figure 5.15 shows total energy consumption against the maximum moving speed. Unlike the other mobility models, the performance of our caching schemes are not influenced by moving speed very much. The data blocking ratio remains the same as the moving speed increases. This is because mobile nodes are divided into groups and they move together with mobile nodes in the same group. The connection within the same group is maintained as the moving speed changes. Therefore, data accessibility is also assured. The SC and RC schemes outperform the Selfish Cache scheme significantly because in the latter, mobile nodes use their cached data for themselves only. When considering the total energy consumption, since nodes are grouped together, handover does not happen frequently since it is not difficult to find a copy of the same data within H_i hops. Hence, the energy

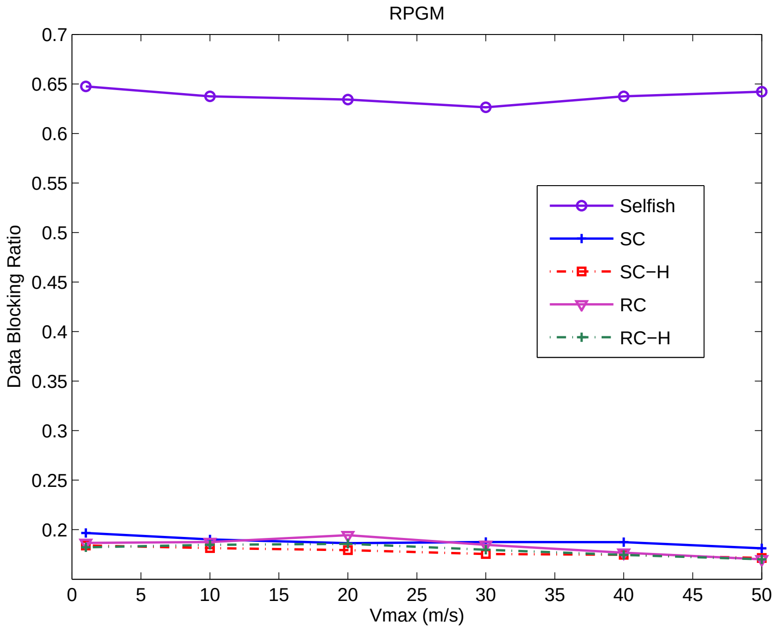

Figure 5.14: Data Blocking Ratio vs. Vmax (m/s) (RPGM)

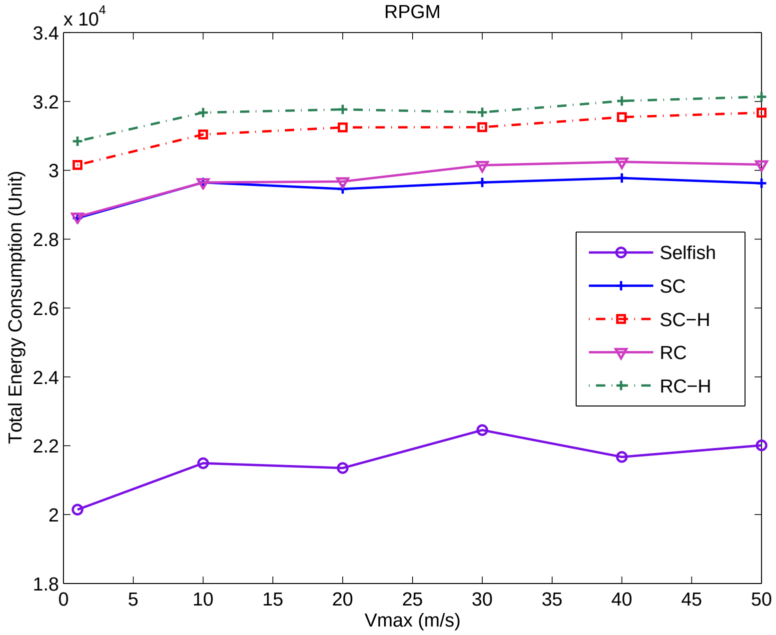

Figure 5.15: Total Energy Consumption vs. Vmax (m/s) (RPGM)

consumption in this kind of network is smaller than a network with Random Waypoint or Manhattan Grid Mobility Model. However, in a network with the RPGM model, the data blocking ratio using the Selfish Cache scheme is much higher than that using other caching schemes. This is because in Selfish Cache, a node only caches its own data of interest regardless of whether any neighboring node has cached the data or not. As a result, there are a lot of replicas of the same data within the group and memory is wasted because a lot of data cannot be cached and carried when the group moves together. In our proposed caching schemes, nodes will spend part of their memory to store data for others. There will be only one or two replicas of the same data within the group of nodes, then more space could be used to cache other different data items. These will significantly improve the performance.

As explained before, if Selfish Cache is employed in the network with RPGM mobility model, the data blocking ratio is high. Therefore, less data are successfully relayed over the network, which makes the energy consumption much lower than other caching schemes in this case.

Chapter 6

Conclusions

We focused on the design and development of suitable data caching schemes to improve data accessibility performance in mobile ad hoc networks in this thesis. Although a lot of challenging issues are involved in this topic, it is still a promising area for investigation. In this thesis, we only explored a the tip of the iceberg to develop suitable data caching mechanisms so that the overall data accessibility in a mobile ad hoc network is enhanced.

In this thesis, we investigated the demands of the applications of mobile ad hoc networks with the advances of technology in wireless communication and networking. We discussed the challenges faced by ad hoc networks and which are the key points for researchers to conquer. We reviewed the routing techniques available so far in the literatures for mobile ad hoc networks.

A simple complexity analysis was done to prove that it is a NP-hard problem to find an optimal data caching scheme in a mobile ad hoc network. Therefore, we proposed several simple caching schemes instead.

We examined location-dependent data access schemes in this thesis. Our proposed Relay Caching schemes allowed more intermediate nodes to cache interested data when they are nearer to the destination, and less intermediate nodes to cache the data when the distance is far from the destination. By adopting this non-uniform distribution of replicas, performance enhancement is achieved with less overhead.

We further improved our proposed caching schemes by integrating a location-dependent handover and replacement policies. In handing the cached data over to some neighboring nodes, local data accessibility is maintained when a caching node leaves a particular location. Furthermore, smaller sized data items were given higher priority to be cached in the memory, so that more data items could be cached with the same size of memory. Therefore, higher data success ratio is achieved.

In this thesis, we verified our proposed caching schemes under networks with five mobility models, namely, the Random Waypoint, Random Direction, Gauss-Markov, Manhattan Grid and Reference Point Group Mobility (RPGM) models.

The Random Waypoint, Random Direction and Gauss-Markov mobility models are similar to one another and there are only minor differences in terms of randomness. Similar performances were obtained for our proposed schemes in the network with these three mobility models. The SC and SC-H schemes performed well at low memory levels. However, the RC and RC-H schemes outperformed the SC and SC-H schemes by over 30%, and the schemes with handover outperform those without handover by about 20%. The drawback of employing handover scheme is that it consumes more energy. The energy consumption of the SC-H and RC-H schemes in a fast moving network overtook the others. Therefore, we encourage to disable the handover facility while a node is moving rapidly.

The performance of our caching schemes under Manhattan Grid Mobility Model and RPGM Model are very different compared to that under Random Mobility Models. Less cache memory is needed to reach stable data accessibility under

Manhattan Grid Mobility Model and RPGM Model. In a network with Manhattan Grid Mobility Model, the RC and RC-H schemes outperformed the SC and SC-H schemes by a large margin. However, the benefits of handover policy is not obvious in this case. Therefore, the RC scheme might be the best choice in a network with Manhattan Grid Mobility Model. On the other hand, the SC, SC-H, RC and RC-H schemes have similar performances in a network with RPGM model. Thus, the SC scheme is recommended in this situation because of its simplicity.

There are a number of related research activities which could be carried out further. Several examples are listed below:

- In this thesis, the network model we used was a single-server model. Further research could be done on a network without any server. Mobile nodes could function as data generators and/or requestors at the same time.
- More complicated mathematical models could be developed to distribute the replicas in a more effective way.
- Security problem could be addressed to make sure the data obtained from the caching nodes is the right copy. And further challenges could be made by allowing some mobile nodes to be selfish.
- Data consistency problem could be studied further by allowing updates of data.
- The cache scheme may be employed on network with different timeout values. The impact of timeout delay on the data blocking ratio and energy consumption may be addressed in the future.

Bibliography

[1] C. Bennett and D. Orchard, "Practical WAP : developing applications for the wireless web," *Cambridge University Press,* 2000.

[2] C. E. Perkins, "Ad Hoc Networks," *Addison-Wesley,* 2001.

[3] B. Bellur and R. G. Ogier, "A Reliable, Efficient Topology Broadcast Protocol for Dynamic Networks," In *Proceedings of IEEE INFOCOM 1999, the 18th Annual Joint Conference of the IEEE Computer and Communications Societies,* vol 1, pp. 178-186, Mar. 1999.

[4] J. J. Garcia-Luna-Aceves and M. Spohn, "Source-Tree Routing in Wireless Networks," In *Proceedings of IEEE ICNP 1999, the 7th International Conference On Network Protocols,* pp. 273-282, Oct. 1999.

[5] A. Iwata, C. C. Chiang, G. Pei, M. Gerla, and T. W. Chen, "Scalable Routing Strategies for Ad Hoc Wireless Networks," *IEEE Journals. Selected Areas in Communications., Special Issue on Wireless Ad Hoc Networks,* vol. 17, no. 8, pp. 1369-1379, Aug. 1999.

[6] P. Jacquet, P. Muhlethaler, A. Qayyum, A. Lanouiti, L. Viennot, and T. Clausen, MANET Draft "draft-ietf-manet-olsr-02.txt," Jul. 2000.

[7] S. Murthy and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," *ACM Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communication Networks,* pp. 183-197, Oct. 1996.

[8] C. E. Perkins and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM: Computer Communications Review,* vol. 24, no. 4, pp. 234-244, Oct. 1994.

[9] C. K. Toh, "Wireless ATM and Ad Hoc Networks: Protocols and Architectures," Kluwer Academic Press, 1997.

[10] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," In *Proceedings of IEEE INFOCOM 1997, the 16th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol 3, pp. 1405-1413, Apr. 1997.

[11] D. B. Johnson and D. A. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks," In *Mobile Computing,* pp. 153-181, Kluwer Academic Publishers, 1996.

[12] C. E. Perkins and E. M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," In *Proceedings of IEEE WMCSA 1999, the 2nd IEEE Workshop on Mobile Computing Systems and Applications,* pp. 90-100, Feb. 1999.

[13] Z. J. Haas, "A New Routing Protocol for the Reconfigurable Wireless Network," In *Proceedings of IEEE ICUPC 1997, the 6th International Conference on Universal Person Communications Record. Bridging the Way to the 21st Century,* vol 2, pp. 562-566, Oct. 1997.

[14] Y. B. Ko and N. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks,” In *Proceedings of MobiCom 1998, the 4th International Conference on Mobile Computing and Networking,* pp. 66-75, Oct. 1998.

[15] Y. C. Hu and D. B. Johnson, “Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks,” In *Proceedings of MobiCom 2000, the 6th Annual IEEE/ACM International Conference on Mobile Computing and Networking,* pp. 231-242, Aug. 2000.

[16] R. Beraldi and R. Baldoni, “A Caching Scheme for Routing in Mobile Ad Hoc Networks and Its Application to ZRP,” *IEEE Transactions on Computers,* vol. 52, no. 8, pp. 1051-1062, Aug. 2003.

[17] M. V. Wilkes, “Slave Memories and Dynamic Storage Allocation,” *IEEE Transactions on Electronic Computers,* vol. 2, pp. 270-271, Apr. 1965.

[18] J. E. Wieselthier, G. D. Nguyen and A. Ephremides, “On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks,” In *Proceedings of IEEE INFOCOM 2000, the 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, no. 1, pp. 585-594, Mar. 2000.

[19] M. J. Carey and M. Livny, “Distributed Concurrency Control Performance: A Study of Algorithm, Distribution, and Replication,” In *Proceedings of the 14th Internatianal Conference on Very Large Data Bases,* pp. 13-25, Aug. 1988.

[20] N. Krishnakumar and A. J. Bernstein, “High Throughput Escrow Algorithms for Replicated Databases,” In *Proceedings of the 18th Internatianal Conference on Very Large Data Bases,* pp. 175- 186, Aug. 1992.

[21] M. Stonebraker, “Concurrency cCntrol and Consistency in Multiple Copies of Data in Distributed INGRES,” *IEEE Transactions on Software Engineering,* vol. 3, no. 3, pp. 188-194, May. 1979.

[22] R. H. Thomas, “A Majority Consensus Approach to Concurrency Control for Multiple Copy Databases,” *ACM Transaction on Database Systems,* vol. 4, no. 2, pp. 180-209, Jun. 1979.

[23] O. Wolfson and A. Milo, “The Multicast Policy and Its Relationship to Replicated Data Placement,” *ACM Transaction on Database Systems,* vol. 16, no. 1, pp. 181-205, Mar. 1991.

[24] O. Wolfson and S. Jajodia, “Distributed Algorithms for Dynamic Replication of Data,” In *Proceedings of the 11th ACM PODS,* pp. 149-163, Jun. 1992.

[25] L. Z. Yin and G. H. Cao, “Supporting Cooperative Caching in Ad Hoc Networks,” In *Proceedings of IEEE INFOCOM 2004, the Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 4, pp. 2537-2547, Mar. 2004.

[26] E. Papapetrou and F. N. Pavlidou, “A Novel Approach to Source Routing for Multi-Hop Ad Hoc Networks,” *IEEE Communications Letters,* vol. 7, no. 10, pp. 472-474, Oct. 2003.

[27] E. M. Royer and C. K. Toh, “A Review of Current Routing Protocols for Ad Hoc Mobile Wireless Networks,” *IEEE Personal Communications,* vol. 6, no. 2, pp. 46-55, Apr. 1999.

[28] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, “Algorithms for Energy-Efficient Multicasting in Ad Hoc Wireless Networks,” In *Proceedings of IEEE MILCOM 1999, IEEE Military Communications Conference,* no. 1, pp. 1414-1418, Oct. 1999.

[29] S. Banerjee, A. Misra, J. Yeo, and A. Agrawala, “Energy-Efficient Broadcast and Multicast Trees for Reliable Wireless Communication,” In *Proceedings of IEEE WCNC 2003, IEEE Wireless Communications and Networking Conference,* vol. 4, no. 1, pp. 660-667, Mar. 2003.

[30] L. Qiu, V. N. Padmanabhan, and G. M. Voelker, “On the Placement of Web Server Replicas,” In *Proceedings of IEEE INFOCOM 2001, the 20th Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, pp. 1587-1596, Apr. 2001.

[31] B. Li, M. J. Golin, G. F. Ialiano, and X. Deng, “On the Optimal Placement of Web Proxies in the Internet,” In *Proceedings of IEEE INFOCOM 1999, the 18th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1282-1290, Mar. 1999.

[32] D. Y. Li, H. Liu, and X. H. Jia, “Minimum Energy-Cost Broadcast Routing in Ad Hoc Wireless Networks,” In *Proceedings of IEEE GLOBECOM 2003, IEEE Global Telecommunications Conference*, vol. 22, no. 1, pp. 367-371, Dec. 2003.

[33] A. W. Fu and D. W. L. Cheung, “A Transaction Replication Scheme for a Replicated Database with Node Autonomy,” In *Proceedings of the 20th Internatianal Conference on Very Large Data Bases,* pp. 214-225, Sep. 1994.

[34] T. Hara, K. Harumoto, M. Tsukamoto, and S. Nishio, “Dynamic Replica Allocation Using Database Migration in Broadband Networks,” In *Proceedings of IEEE ICDCS 2000, the 20th International Conference on Distributed Computing Systems*, pp. 376-384, Apr. 2000.

[35] G. H. Cao, "Proactive Power-Aware Cache Management for Mobile Computing Systems," *IEEE Transactions on Computers,* vol. 51, iss. 6, pp. 608-621, Jun. 2002.

[36] E. J. O'Neil, P. E. O'Neil, and G. Weikum, "The LRU-K Page Replacement Algorithm for Database Disk Buffering," In *Proceedings of ACM SIGMOD 1993, International Conference on Management of Data,* pp. 297-306, May. 1993.

[37] P. Cao and S. Irani, "Cost-Aware WWW Proxy Caching Algorithms," In *Proceedings of the USENIX Symposium on Internet Technologies and Systems (ITS-97),* pp. 193-206, Dec. 1997.

[38] T. Hara, "Effective Replica Allocation in Ad Hoc Networks for Improving Data Accessibility," In *Proceedings of IEEE INFOCOM 2001, the 20th Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, pp. 1568-1576, Apr. 2001.

[39] T. Moriya and H. Aida, "Cache Data Access System in Ad Hoc Networks," In *Proceedings of IEEE VTC 2003, the 57th IEEE Semiannual on Vehicular Technology Conference,* vol. 2, pp. 1228-1332, Apr. 2003.

[40] S. H. Lim, W. C. Lee, G. H. Cao and C. R. Das, "A Novel Caching Scheme for Internet Based Mobile Ad Hoc Networks," In *Proceedings of IEEE ICCCN 2003, the 12th International Conference on Computer Communications and Networks,* pp. 38-43, Oct. 2003.

[41] G. Barish, and K. Obraczka, "World Wide Web Caching: Trends and Techniques," *IEEE Communications Magazine Internet Technology Series,* vol. 38, iss. 5, pp. 178-184, May. 2000.

[42] J. Wang, “A Survey of Web Caching Schemes for the Internet,” *ACM Computer Communication Review,* vol. 25, no. 9, pp. 36-46, Oct. 1999.

[43] C. Chekuri and S. Khanna, “A PTAS for the Multiple Knapsack Problem,” In *Proceedings of the 11th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA),* pp. 213-222, Jan. 2000.

[44] D. B. Shmoys and E. Tardos, “An approximation algorithm for the generalized assignment problem,” *Mathematical Programming A,* 62:461C74, 1993.

[45] J. Cho, S. Oh, J. Kim, H. H. Lee, and J. Lee, “Neighbor Caching in Multi-Hop Wireless Ad Hoc Networks,” *IEEE Communications Letters,* vol. 7, iss. 11, pp. 525-527, Nov. 2003.

[46] V. Gianuzzi, “File Distribution and Caching in MANET,” *Technical Report,* University of Genova, 2003.

[47] Y. C. Hu and D. B. Johnson, “Caching Strategies in On-Demand Routing Protocols for Wireless Ad Hoc Networks,” In *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking,* pp. 231-242, Aug. 2000.

[48] C. H. Chuan and S. Y. Kuo, “Cache Management of Dynamic Source Routing for Fault Tolerance in Mobile Ad Hoc Networks,” In *Proceedings of the Pacific Rim International Symposium on Dependable Computing*, pp. 17-19, Dec. 2001.

[49] B. Liang and Z. J. Haas, “Optimizing Route-Cache Lifetime in Ad Hoc Networks,” In *Proceedings of IEEE INFOCOM 2003, the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 1, pp. 281-291, Apr. 2003.

[50] G. Resta and P. Santi, "An Analysis of the Node Spatial Distribution of the Random Waypoint Model for Ad Hoc Networks," In *Proceedings of ACM Workshop on Principles of Mobile Computing (POMC),* pp. 44-50, Oct. 2002.

[51] D. Hong and S. S. Rappaport, "Traffic Model and Performance Analysis for Cellular Mobile Radio Telephone Systems with Prioritized and Nonprioritized Handoff Procedures," *IEEE Transactions on Vehicular Technology,* vol. 35, no. 3, pp. 77-92, August 1986.

[52] R. A. Guerin, "Channel Occupancy Time Distribution in a Cellular Radio System," *IEEE Transactions on Vehicular Technology,* vol. 35, no. 3, pp. 89-99, Aug. 1987.

[53] M. M. Zanoozi, P. Dassanayake, and M. Faulkner, "Mobility Modelling and Channel Holding Time Distribution in Cellular Mobile Communication Systems," In *Proceedings of IEEE GLOBECOM95, Singapore,* pp. 12C16, Nov. 1995.

[54] E. Royer, P. M. Melliar-Smith, and L. Moser, "An Analysis of the Optimum Node Density for Ad Hoc Mobile Networks," In *Proceedings of IEEE ICC 2001, the IEEE International Conference on Communications,* vol. 3, pp. 857-861, Jun. 2001.

[55] B. Liang and Z. Haas. "Predictive Distance-Based Mobility Management for PCS Networks," In *Proceedings of IEEE INFOCOM 1999, the 18th Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, pp. 1377-1384, Mar. 1999.

[56] V. Tolety. "Load Reduction in Ad Hoc Networks Using Mobile Servers," *Masters Thesis,* Colorado School of Mines, 1999.

[57] Universal Mobile Telecommunications System (UMTS), "Selection Procedures for the Choice of Radio Transmission Technologies of the UMTS," ver. 3.2.0, Apr. 1998.

[58] X. Hong, M. Gerla, G. Pei, and C. Chiang. "A Group Mobility Model for Ad Hoc Wireless Networks," In *Proceedings of the ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems (MSWiM),* pp. 53-60, Aug. 1999.

[59] L. M. Freeney, and M. Nilsson, "Investigating the Energy Consumption of a Wireless Network Interface in an Ad Hoc Networking Environment," In *Proceedings of IEEE INFOCOMM 2001, the 20th Annual Joint Conference of the IEEE Computer and Communications Societies,* vol. 3, pp. 1548-1557, Apr. 2001.

[60] http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion/.

[61] http://www.isi.edu/nsnam/ns/.

[62] X. Zeng, R. Bagrodia, and M. Gerla, "GloMoSim: a Library for Parallel Simulation of Large-Scale Wireless Networks," In *Proceedings of the 12th Workshop on Parallel and Distributed Simulation,* pp. 154-161, May. 1998.

[63] R. Bagrodia, R. Meyerr, "PARSEC: A Parallel Simulation Environment for Complex System," *UCLA technical report,* 1997. http://pcl.cs.ucla.edu/projects/parsec/.

Name: Qiao Yunhai
Degree: Master of Engineering
Department: Electrical and Computer Engineering
Thesis Title: Location-Dependent Data Caching with Handover and Replacement for Mobile Ad Hoc Networks

Abstract

Over the last few years, the field of wireless ad hoc networks has attracted tremendous interest from the research community. The ultimate goal of setting up a network is to exchange information. However, it is a challenging task to maintain data accessibility due to physical constraints of ad hoc networks.

In this thesis, we propose some location-dependent data caching algorithms to increase data accessibility in mobile ad hoc networks. Location-dependent data handover and replacement schemes are proposed to maintain and further improve the performance of proposed caching schemes. Most of previous work used only Random Waypoint mobility model. In this thesis, Random Direction, Gauss Markov, Manhattan Grid and RPGM are also included. The impact of memory size, the request generating time, the maximum moving speed of mobile nodes on data accessibility is verified. The energy consumption is also examined.

Keywords:
Ad Hoc Networks, Location-dependent Caching, Cache Handover, Cache Replacement, Data Accessibility, Mobility Models.