Environment Aware Location Service for Mobile Ad Hoc Networks

Sabbir Ahmed

A thesis submitted for the degree of
Doctor of Philosophy at
Monash University

April 2010

Typeset in Computer Modern by TEX and LATEX 2ε.

Declaration

This thesis is my own work and has not been submitted in any form for another degree or diploma at any university or other institute of tertiary education. Information derived from the published and unpublished work of others has been acknowledged in the text and a list of references is given.

Sabbir Ahmed

April 2010

Dedicated to my parents for all their love and inspiration.

Acknowledgments

At first I would like to express my gratitude to almighty Allah for giving me the opportunity to undertake this research.

I would like to express my profound indebtedness to my supervisors Gour C. Karmakar and Joarder Kamruzzaman for their constant guidance, insightful advice, helpful criticism, valuable suggestions, commendable support, and endless patience towards the completion of this thesis. I feel very proud to have worked with them. Without their inspiring enthusiasm and encouragement, this work could not have been completed. I also thank Associate Professor Manzur Murshed, Head of School, for his encouragement and support. For providing an excellent environment for research and financial support in the form of scholarships I also thank Monash University.

My heartfelt thanks go to my parents for their inspiration, to my wonderful wife Qumrun Nahar Sumi for her love, care, patience, and understanding during this work, and without their encouragement and moral support this dissertation would have been impossible. I express my special thanks to my younger brothers Sayeed Ahmed and Mohammad Salim Ahmed for their encouragement and valuable suggestion in every aspect of simulation and modeling conducted during my research. I would also like to thank my lovely daughter Sameeha Ahmed and son Sakib Ahmed for constantly keeping me busy.

I would also like to extend my thanks to Patrick Leegel, Ahsanul Haque, Mortuza Ali, Nahdia Tabassum, Atiur Rahman Siddique, Kamrul Islam, and Mahfuzul Haque for their kind assistance in proofreading this thesis. Finally, I thank all the staffs, graduate students, and friends at Gippsland School of Information Technology (GSIT), Monash University, for their support and encouragement during the last few eventful years in my life.

Abstract

For a successful and efficient communication in mobile ad hoc network, routing protocols play a vital role. Among those proposed in literature, location based routing protocols show their superiority over others by exhibiting low communication overhead, higher scalability and better adaptability in scenarios where node mobility is high; especially high scalability makes them attractive for large-scale deployment of ad hoc network. As the name indicates, location based routing protocols greatly rely on the precise knowledge of the mobile devices' location, which is provided by a *location service*. Huge research efforts have been dedicated in developing location services where the researchers have emphasized on making the services scalable but little consideration was given on their applicability in real-world environment as most services have been designed assuming an ideal deployment environment that fails to represent the impact of the real-world on node mobility and signal propagation.

A location service most suitable in a realistic environment must consider the impact of environmental contexts (obstacles, pathways, etc.), making the mobility model a vital factor through which the environmental impact is reflected in simulation. This study develops a novel mobility model which emulates realistic node mobility more accurately by considering various environmental contexts. Non-uniform node distribution caused by environmental contexts impacts the performance of the existing location services drastically. This demands a pressing need for a location service that can adopt itself while ensuring seamless service. In this study non-uniform node distribution arising from natural phenomena is considered while designing a novel location service protocol that adapts itself to the presence and nature of the environmental context. As the location service utilizes the underlying location based routing protocol for transferring service related messages, signal attenuation experienced due to environmental contexts is also taken into account in the routing protocol for finding routable paths, which in turn further enhances the performance of the proposed location service significantly. The proposed location service shows a superior ability to provide location

information while maintaining scalability. A detailed mathematical model of the scalability is also presented. Exhaustive simulation has been conducted using a real-world map, which demonstrates the significant superiority of the proposed scheme over other competitive location service protocols (e.g., HLS (Hierarchical Location Service) and GLS (Grid Location Service)) for both uniformly and non-uniformly distributed nodes, and establishes its greater applicability to real-world environments.

Acronyms

ADLS	Adaptive Demand-driven Location Service
AFR	Adaptive Face Routing
ALM	Adaptive Location Management
AMM	Anchor-based Mobility Model
AODV	Ad-hoc On-Demand Distance Vector routing
CBRP	Cluster Based Routing Protocol
CGSR	Cluster-head Gateway Switch Routing
DLM	Distributed Location Management
DLS	DREAM Location Service
DREAM	Distance Routing Effect Algorithm for Mobility
DSDV	Destination-Sequenced Distance-Vector routing
DSR	Dynamic Source Routing
EDV	External Door Vertex
EGV	External General Vertex
FSR	Fisheye State Routing
GG	Gabriel Graph
GHLS	Geographic Hashing Location Service
GLS	Grid Location Service
GPS	Global Positioning System
GPSR	Greedy Perimeter Stateless Routing
GPSR-EA	Environment Aware Greedy Perimeter Stateless Routing
GPSR-MA	Mobility Aware Greedy Perimeter Stateless Routing
GRSS	Geographical Region Summary Service
GUI	Graphical User Interface
HALS	Hierarchical Adaptive Location Service
HALS-EA	Environment Aware Hierarchical Adaptive Location Service

HALS-MA	Mobility Aware Hierarchical Adaptive Location Service
HGRID	Hierarchical GRID Location Management
HIGH-GRADE	HIerarchical Geographical Hashing with multi-GRained Address DEelegation
HLS	Hierarchical Location Service
IDV	Internal Door Vertex
IGV	Internal General Vertex
LAN	Local Area Network
LAR	Location-Aided Routing
LLS	Locality aware Location Service
MAC	Media Access Control
MAN	Metropolitan Area Network
MANET	Mobile Ad hoc Network
OLSR	Optimized Link State Routing
PDA	Personal Digital Assistant
PSR	Proactive Source Routing
PV	Path Terminal Vertex
RLS	Reactive Location Service
RNG	Relative Neighborhood Graph
RPGM	Reference Point Group Mobility
SLALoM	Scalable Ad-hoc Location Management
SLAW	Self-similar Least Action Walk
SLS	Simple Location Service
SLURP	Scalable Location Update-Based Routing Protocol
STRAW	STreet RAndom Waypoint
SVG	Scalable Vector Graphic
TORA	Temporally Ordered Routing Algorithm
WLAN	Wireless Local Area Network
XYLS	Column Row Location Service
ZHLS	Zone-Based Hierarchical Link State protocol
ZRP	Zone Routing Protocol

Nomenclature

ν	Node movement speed
ν_{max}	Maximum node movement speed
ν_{min}	Minimum node movement speed
v	Vertex
u	Node
s_p	Starting point
d_p	End point
t_{pause}	Pause time
s_i^{area}	i-th sub-area
k_i^{nodes}	Number of nodes in s_i^{area}
ϖ	Clustering exponent
$p_i^{cluster}$	Probability of selecting s_i^{area}
J	A packet's lifetime represented by hops
$t_{backoff}$	Backoff time
$delay_{max}$	Maximum delay
ϵ	Scaling factor of the size of home
M_i^s	i-th location server node
r_t	Signal transmission range
b	An obstacle
$\|V\|$	Total number of vertices
E	Set of edges
$\|E\|$	Total number of edges
G_{ex}	Set of External General Vertices(EGV)
Ω_{ex}	Set of External Door Vertices(EDV)
V_p	Set of Path Terminal Vertices(PV)
Ω_{inb}	Set of Internal Door Vertices(IDV) of obstacle b

G_{inb}	Set of Internal General Vertices(IGV) of obstacle b
K	Set of obstacles
Ω_b	Set of doors of obstacle b
ϱ	Door of obstacle
$v_{in}(\varrho_b)$	IDV of door ϱ of obstacle b
$v_{ex}(\varrho_b)$	EDV of door ϱ of obstacle b
v_{ip}	Vertex derived from a node's initial position
v_{ep}	Vertex derived from a node's end position
$\aleph$	Set of nodes
u	Node
k_{obs}	Number of obstacles
I	Number of IGV
F	Number of EGV
I_b	Number of IGV of obstacle b
η_b	Number of doors of obstacle b
η	Number of doors
μ	Number of PV
w	Number of walls, and General vertices in the scenario
w_b	Number of walls of obstacle b
μ_r	Number of user-defined roads
N	Number of nodes
T	Simulation duration
Υ	Average number of destination selection by a node within simulation time T
I_i^{roi}	i-th region of interest
P^t	Signal transmission strength
P^r	Signal received strength
λ	Wave length
φ	System loss in signal propagation
G^t	Transmitter gain
G^r	Receiver gain
α	Signal decaying factor

h^t	Height of transmitter
h^r	Height of receiver
w^i	i-th wall
γ_i	Attenuation factor of i-th wall
P^b_{ri}	Received signal strength before i-th wall
P^a_{ri}	Received signal strength after i-th wall
$D_{t,r}$	Distance between transmitter and receiver
R_x	Minimum threshold for receiving signal successfully
L_i	Level i
R_i	Rank i
ψ	Highest rank, i.e., playground is an $L_{\psi+1}$ cell
DB	Database
H^A_i	L_i home of node A
κ	Granularity of composing L_1 cell (e.g., $\kappa \times \kappa$)
a	Side length of an L_0 cell
ρ^c_i	Update packet generation rate for R_i home in crossing cell triggered scheme
ρ^t_i	Update packet generation rate for R_i home in time triggered scheme
ρ^d_i	Update packet generation rate for R_i home in distance triggered scheme
t^u_i	Time interval for R_i home update in time triggered scheme
d^u_i	Distance interval for R_i home update in distance triggered scheme
σ_i	Maximum number of hops required from a node to its R_i home
σ^q_i	Maximum number of hops required to complete a query (including reply) bounded by an L_{i+1} cell
δ_i	Maximum distance from a node to its R_i home
z	Average progress by each hop
τ	Average node density
W	Playground (network deployment) width
Δ	Network deployment area
ξ_i	The number of nodes whose location information needs to be stored by a node in an R_i home

p_i	Probability that L_{i+1} cell is the smallest common cell of both querying and queried nodes
C_u^c	Update overhead for cell triggered update method
C_u^t	Update overhead for time triggered update method
C_u^d	Update overhead for distance triggered update method
C_q	Query overhead
C_s^i	Storage overhead for nodes within R_i home
Γ	Current time
u_i	i-th neighbor of node u
$t_{beacon}(u_i)$	Received time of beacon originated by node u_i

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

Wireless network is a data communication system among various types of communication devices (e.g., computer, laptop, PDA, cellular phone) that uses wireless media (e.g., radio waves, infra-red signals) to transmit and receive data with or without the existence of a wired connection. The devices participating in the communication are commonly referred to as *nodes*. The last several decades have witnessed significant advancement in the field of wireless communication through the development of breakthrough technologies like cellular, bluetooth, wireless local area network (WLAN), metropolitan area network (MAN), satellite network and so on. These technologies have become an integral part of our everyday life, and have drawn particular interest of the research community for a wide range of reasons. One of the key factors behind this advancement is the support for mobility where users are able to access information in real-time while roaming around the network without getting disconnected. This feature opens a gateway for various applications and services, which is not possible with wired networks. Moreover, the network setup cost and time are considerably lower in a wireless network, which provides an opportunity to extend the communication domain, even to places which are unreachable by wired connection. Furthermore, wireless networks offer more flexibility in maintenance, and are easily adaptable to changes in the network configuration.

Based on the infrastructure requirements, wireless networks can be classified into two broad categories. The first category includes networks that require predefined infrastructure. Cellular network and wireless LANs belong to this category, where the base stations and wireless access points create the infrastructure, respectively. This type of network generally performs through *one-hop* communication, i.e., the nodes communicate to the infrastructure devices directly, and therefore, the existence of in-

frastructure is essential. The second category is termed as *ad hoc network*, where an established infrastructure is not required. In most cases the nodes creating an ad hoc network are mobile and such network is often referred to as Mobile Ad hoc Network (MANET). MANET is a self-configuring network of mobile devices such as laptops, PDAs or smart phones, all connected by wireless connections. Each device is free to move independently and therefore, its connectivity to other mobile devices changes frequently. This type of network does not have any centralized infrastructure in contrast to other networks, e.g., wired and cellular networks. Because of the absence of any predefined infrastructure, all the mobile devices in such a network participate in communication cooperatively in a distributive manner, and determine the communication path dynamically where each device works as a transmitter, receiver and router, performing through *multi-hop* communications. Since this type of network can be established on an ad hoc basis, it is possible to deploy them rapidly with a low establishment cost. This makes MANET attractive when traditional communication system is either non-existent or dysfunctional, and hence such a network has a wide range of applications including (but not limited to) incident management, planning, and recovery in disaster areas (e.g. bushfire, earthquake, and cyclone), battle fields, inter-vehicular communication, because of its lower cost, quick setup time, and reduced complexity than establishing and maintaining an infrastructure-based network.

For the successful deployment of MANET, the existence of a framework that facilitates communication between any two nodes is essential. Thus the importance of a *routing protocol* is crucial and the network performance and reliability is highly dependent on it. It is the responsibility of the routing protocol to establish a path for successful end-to-end communication. Hence, designing and developing routing protocols has become a very important research area in the MANET research community. Devising routing protocols for MANET is a challenging task as the topology of the network is always changing due to the node mobility. The nodes can enter/join or exit the network at any time, and the absence of an infrastructure makes the situation even more complicated. When two communicating nodes move independently, there may not be a direct link between them. Thus a method is needed that can ensure communication between two nodes in a distributed manner with the help of the other nodes while adapting itself to the continuously changing network topology. Furthermore, the presence of various environmental objects (e.g., buildings, trees, hills) results in the sig-

nal fading and attenuation which subsequently affects the performance of the routing protocol adversely. Also, bandwidth scarcity with variable link capacity and energy constrained devices with limited transmission range make ad hoc network deployment and operation much more challenging.

In this regard, several routing protocols have been proposed for communication in mobile ad hoc networks. Some of these protocols can be categorized as *minimum cost routing protocols* which includes Destination-Sequenced Distance-Vector Routing (DSDV) [1], Fisheye State Routing (FSR) [2] and Optimized Link State Routing (OLSR) [3]. These protocols proactively maintain *routing tables*[1] via a periodic exchange of routing information. On the other hand, Dynamic Source Routing (DSR) [4], Ad-hoc On-Demand Distance Vector routing (AODV) [5] and the Temporally Ordered Routing Algorithm (TORA) [6] can be categorized as *on-demand protocols*. These protocols find routes only when they are needed, and thus reduce the routing overhead compared to the minimum cost routing protocols. In the literature, some *hybrid protocols* such as Zone Routing Protocol (ZRP) [7] and Zone-Based Hierarchical Link State Protocol (ZHLS) [8] have been proposed that maintain routes to nearby nodes in a proactive manner and far away nodes in on-demand manner. Another group of protocols relies on the construction of a *hierarchy* where nodes are grouped using clustering algorithms at the lowest level in the hierarchy, and two nodes in two different clusters communicates via their cluster heads. Cluster-head Gateway Switch Routing (CGSR) [9], Proactive Source Routing (PSR) [10] and Cluster Based Routing Protocol (CBRP) [11] belong to this category. Another category of protocols, called *location based routing protocols*, is based on the *geographic locations* of the mobile devices, which includes the Distance Routing Effect Algorithm for Mobility (DREAM) [12], Location-Aided Routing Protocol (LAR) [13], Adaptive Face Routing (AFR) [14] and Greedy Perimeter Stateless Routing (GPSR) [15]. These protocols use the geographic location of the destination nodes to forward data.

While numerous routing protocols have been developed to date, the *scalability* has been the focus of interest for a long time. For this reason, over the last decade, a significant research effort has been dedicated to a challenging research issue – to assure scalable routing – which is essential for deploying an ad hoc network effectively. Routing

[1] A table that contains a list of next hops for sending packet to all other nodes

protocols other than location based ones, maintain routes either proactively, on-demand or in a hybrid manner. However, if the network size increases, these approaches fail to scale well due to their flooding nature of route discovery requests. Moreover, increased mobility results in higher resource consumption while computing routes. In this respect, the location based routing protocols in MANET exhibit better scalability than the other protocols, and appear more promising to the research community.

Location based routing protocols (also known as position based routing protocol and geographic routing protocol) use the nodes' location in order to make routing decisions. With the easy availability and decreasing cost of Global Positioning Systems (GPS) and other relevant techniques, a mobile device can easily find its own location. The nodes can find their neighbors'[2] location through periodic beacons, known as *HELLO* messages. This enables a source node to choose a neighbor which is the nearest to the destination among all of its neighbors stored in the *neighbor list*. However, this forwarding approach fails if there is no such neighbor in the direction of the destination. Several geographic routing protocols e.g., GPSR [15], recover from this situation by forwarding the packet using a different route, until it reach a node closer to the destination.

Location based routing protocols offer several key features that make them more attractive than the routing protocols belonging to other categories. Unlike other protocols, they do not need to maintain any routing table. Node mobility causes the routing tables to become outdated within a short time. Thus maintaining routing table becomes a complicated and resource consuming task which sometimes involves transferring the routing tables even over the whole network. This situation becomes much worse when node mobility increases, resulting in the routing tables becoming obsolete more frequently. Since location based routing protocols are designed without routing tables, they experience lower communication overhead. Moreover, unlike many other routing protocols, the location based protocols do not rely on any flooding mechanism. Thus routing protocols belonging to this category are very promising for networks of any size, from small to large, even in a high node mobility scenario. This capability of scaling well with network size and coping with high node mobility has motivated many researchers to contribute to the domain of location based routing

[2]Nodes that are present within their transmission range

protocols.

However, two basic prerequisites must be fulfilled for effective execution of a location based routing protocol in MANETs. Firstly, each node must be able to identify its location. This requirement can be fulfilled quite easily by incorporating GPS with the mobile devices. Now-a-days mobile devices such as PDAs, smart phones are mostly equipped with a built in GPS. On the other hand low power devices e.g., sensor nodes, can adopt any of the available methods in [16, 17, 18] to acquire their own locations.

However, knowing a node's own location is not enough for location based routing protocols, and a node must be able to acquire the location of any other node in the network in order to communicate. Since knowing the approximate location of a destination node is needed in geographic routing protocols, the presence of a *location service* is mandatory. A location service is defined as a service which is exploited by every node in a network to retrieve the location of any other node, and the performance of the location based protocols (e.g., packet delivery success rate) depends on the precise knowledge of the destination's location. In this regard, several location service schemes have been proposed in the literature. Most of the proposed location services' working principles rely on hash function based location server assignment (e.g., Scalable Location Update-Based Routing Protocol (SLURP) [19], Scalable Ad-hoc Location Management (SLALoM) [20], HIerarchical Geographical Hashing with multi-GRained Address DEelegation (HIGH-GRADE) [21], Hierarchical Location Service (HLS) [22], etc.). On the other hand, Column Row Location Service (XYLS) [23] and SEEKER [24] are based on assigning servers by building *quorums* in specific directions (e.g., north-south, east-west). Some location services utilize both quorum building and hash function mechanisms, i.e. are hybrid in nature; Adaptive Location Management (ALM) [25] and Locality Aware Location Service (LLS) [26] belong to this category.

Irrespective of the location service used, the more accurate the location of a node can be achieved through such service, the better the geographic routing protocol can perform. However, due to node mobility, providing accurate location of the nodes is a complicated task which becomes more acute if the node mobility increases. While designing a location service, it must be kept in mind that the key strengths of a geographic routing protocol are its scalability and low communication overhead, so the location service must not be a burden to the routing protocol itself. For this reason,

designing a location service that can provide the location of nodes accurately while maintaining scalability is challenging, and at the same time vitally important, as the performance of geographic routing protocol depends upon it.

It is important to note that the main purpose of a location service is to aid the geographic routing protocols designed to be applicable in a real-world environment. Thus, while designing a location service protocol, the impact of the environmental contexts (e.g., pathways, buildings with doorways, inaccessible obstacles in the form of hills, lakes, vegetations etc.) is an important issue to consider as well, which has been given little importance in the development of the existing location services. Since simulation has become very popular in ad hoc network research and most of the location services have been evaluated through simulation, it is very important to reflect the environmental impact in the simulation while evaluating such protocols. The mobility model is the key factor through which the environmental impact is reflected. Hence, it is necessary to use such a mobility model that mimics the realistic movement we observe in the real-world. The more realistically a mobility model can represent node movement, the better the environmental impact is experienced and consequently, the evaluation of any system including location service protocol becomes much more meaningful and accurate.

In this regard, there have been a number of mobility models proposed in the literature. However, most of them either assume free space (empty space) as the deployment environment [27, 28] or, the movement pattern does not emulate real-world situation properly in the presence of obstacles because of the generation of restricted paths [29]. Moreover, the existing pathways in the terrain, doorways to enter and exit buildings, inaccessible obstacles (e.g., hills, lakes) are very prominent and common environmental attributes that impact our movement and hence, need to be taken into account while directing node mobility. The lack of the reflection of environmental impact on node mobility in the literature highlights another research challenge for developing a realistic mobility model whose importance is paramount in ad hoc network research. By considering the environmental impact through a realistic mobility model while designing the location service, and subsequently using the mobility model for evaluation purposes will provide a better assessment and definitely improve the applicability and suitability of the location service in a real-world environment. This advancement of

location service will contribute to improve end-to-end communication in an ad hoc network using geographic routing protocols.

Designing a location service suitable for real-world environment poses a number of research challenges that need to be tackled. These challenges are the main motivating factors for this research, which are explained in detail in the following section.

1.1 Problem Statement and Motivation

The absence of infrastructure in ad hoc networks leads to a vital problem from the location service point of view as the establishment of dedicated servers is out of scope. The nodes creating such a network are mostly low powered devices. As a result they need to share the traffic load and perform the duty of a location server in a distributed manner. In this regard, a number of algorithms to select the location servers have been proposed in the literature. The simplest location service can be developed using broadcasting. In this method every node broadcasts its location over the network at fixed intervals [12] so that every node becomes aware of the locations of all other nodes. Another way is to broadcast a request for a destination node's location throughout the network [30]. When the request reaches the destination, it returns a unicast reply passing its location to the source node. The requirement of broadcasting makes these methods unattractive as it increases the network traffic drastically and the scalability is highly affected. Thus a common practice is to select a set of nodes that will act as the location servers for each node. When any node moves by a predefined threshold distance, it updates its absolute or relative location information to the servers. Among the various location server selection algorithms, hash function based schemes [19, 22] appear to be attractive due to their nature of uniformly selecting nodes which causes no excessive load on any node. These hash functions choose a geographic segment based on a node's ID, and the nodes in that segment become the location servers of that node. For instance, a node D selects a subset of nodes as its location servers using a uniform hash function on D's ID that maps to a specific geographic region, called the *home region* of D. The nodes residing in the home region become the location servers for D. D periodically updates its location to its location servers. When any node S wants to communicate with D, it first needs to know the location of D. Thus in a similar way using the same hash function with D's ID, S identifies D's home region

and sends a query packet. The query packet also contains the location of S. When the packet reaches any node in the home region, it sends D's location information back to S using geographic routing. Any geographic routing protocol can be used to update the location to the location servers, to query for destination's location and to send data. Different types of home region selection and maintenance schemes are available which are described in detail in Section 2.2.

Although many location service schemes have been proposed for MANET, the hash function based *hierarchically* organized location services [22, 21] have better scalability in terms of the location update and query overhead. One of the inherent characteristics of hash function based server assignment is that the whole network area is divided into square grids and each square is assigned uniformly as home region for nodes based on their IDs. In this way every node in a specific region becomes responsible for acting as the location server for some nodes. Since every square is the home region of a subset of nodes, due to the node mobility it may occur that there is no node in a particular cell, triggering the so called *empty home* problem. Due to the non-existence of nodes, this empty home region causes both location update and query failure for the nodes it has been assigned as home. Several schemes [22, 19] try to recover from this situation by storing the update information and forwarding the query around that region. Though this recovery process may handle the *empty home* problem for a short time, over a long time or a large area containing multiple empty regions, this recovery process may not be suitable and increases the overhead for the nodes creating a perimeter around the empty region. As a result, though the existing approaches can perform reasonably when the nodes are uniformly distributed, with non-uniform distribution of nodes (due to the nature of their mobility) and the creation of empty regions is expected to degrade performance considerably [31].

Handling non-uniform node distributions is crucial while designing location services. Because, realistic movement patterns are likely to create non-uniformly distributed nodes due to many factors, e.g., environmental influence through the existence of various environmental contexts, different movement habits of people, selection of destinations based on interest, etc. Thus it is unrealistic to assume a uniform node distribution throughout the network lifetime. Hence the design philosophy of the existing location services severely lacks in their ability to handle real-world scenarios properly, as most

of them have been designed based on the assumption that the nodes are uniformly distributed throughout the deployment area for the whole network lifetime and a smooth execution of the existing location services depends on this assumption.

Any network protocol, including the location services, must be designed keeping the environmental impact in mind, and more importantly, where the protocol is expected to be used, so that it becomes suitable for the real-world deployment. Such theoretical designs can only be verified through test beds, real-world deployment or appropriate simulation tools that emulate the practical scenarios closely. Simulation is a widely used technique for evaluating the performance of MANET through various metrics, since it does not require the cost and time for real-world study through test beds or practical deployment. Furthermore, simulation allows to generate a wide variety of repeatable scenarios which may not be possible in real-world deployment. This repeatable scenario generation aids in analyzing different perspectives of the network performance through various performance metrics, and the readjustment of protocols under consideration is possible [29]. Due to the capability of rapid development of a wide range of scenarios within a short time with the ability to implement refinements, protocol designers use simulation tools for investigating the performance of protocols. However, for accurate evaluation of MANET models, the simulation environment must be based on realistic assumptions, without which simulation becomes ineffectual.

One of the key factors to reflect and assess a real-world environment is to model the node mobility in a MANET as realistically as possible in a simulation environment. After the deployment of nodes, it is the mobility model that is responsible for directing when and where the nodes will move. A variety of mobility models proposed in literature for studying mobile network are described in Section 2.1. Among them Random Waypoint model, Random Walk model [27] and Random Direction model [28] are widely used where the nodes are assumed to be deployed in a rectangular shaped unobstructed simulation area. In these models the nodes can move anywhere in the simulation area (playground) according to the specification of the respective mobility model. However, these mobility models have been designed without any consideration for the environmental context (e.g., pathways, obstacles such as buildings, lakes, hills, etc.) and thus they lack in representing real-world properly. These models may be used to analyze and verify the theoretical working principle of the protocols, but they

are not sufficient to demonstrate the appropriateness of the protocol for the real-world deployment. In this regard, it is worthy to note that most of the existing location service protocols have been evaluated using these simple and unrealistic mobility models, and designed assuming an unrealistic nature of node movement that causes a more or less uniform node distribution.

The environmental influence is not limited to node mobility alone, but also impacts radio signal propagation which affects the availability of the next hop nodes and end-to-end communication. In every aspects of message transfer, the location services use the underlying routing protocol. Since most of the routing protocols are designed considering an ideal environment without considering signal distortion caused by environmental objects, their presence impacts on routing performance adversely, and consequently the workability of location service is affected. This highlights the importance of considering environmental influence in routing algorithms as well. Most of the current location services have paid little attention to address these important phenomena that occur in the real-world, making these protocols less effective in realistic environments. It is extremely important to consider the real-world perspective in every step of the development cycle of a location service, which remains the main focus of this thesis.

To summarize, while location service protocols have been an active research topic to aid the smooth performance of geographic routing protocols, most of the research works overlooked their suitability for real-world deployment by ignoring the impact of the environment. This section identifies the potential problems that need to be addressed while providing a realistic location service to a geographic routing protocol. This thesis presents novel strategies to address these matters and thereby, improves the applicability of geographic routing protocols by fulfilling the requirement of being aware of other nodes' location through a location service suitable for use in the real-world.

1.2 Research Objectives

From the various highlighted research opportunities discussed in the previous section, the following key research objectives are formulated in order to improve the applicability and performance of geographic routing protocols through location service in a real-world environment:

1. Assessment of the existing location services in a real-world scenario in the presence of environmental objects and identification of the key limitations and issues that need to be addressed.

2. Development of a mobility model that can represent real-world more realistically by considering the environmental contexts and better emulate node movement patterns in their presence.

3. Development of a location service that can adapt itself to the impact and influence of real-world entities as well as attain an overall improved performance.

4. Performance enhancement of geographic routing protocol by devising an improved packet forwarding method through consideration of the environmental context's impact on radio signal.

1.3 Overview of Contribution

To fulfill the above major research objectives, this dissertation presents a number of original contributions. In this respect Fig. 1.1 shows a schematic diagram of the flow of the research conducted to address the objectives presented in the previous section. In this figure, Block 1 represents the performance evaluation phase of a representative existing location service to address Objective 1. Through this study the necessity to address the mobility model shown as Block 2 (Objective 2) and the location service shown as Block 3 (Objective 3) is highlighted. The location service in Block 3 is designed using the knowledge gained in Block 2. To improve the performance of GPSR (taken as a representative geographic routing protocol in Block 4) in the presence of signal attenuation, environment aware GPSR as shown in Block 5 is proposed (Objective 4). Finally Block 6 proposes a new environment aware location service - the combination of the environment aware routing protocol and location service in a real-world environment model. The overall contributions in this dissertation are:

1. Evaluation of Scalable Location Update-Based Routing Protocol (SLURP) [19], taken as a representative of the existing location service protocols, is conducted in the presence of obstacles with the widely used Random Direction mobility model. This study actually highlights the limitations of the existing location services and

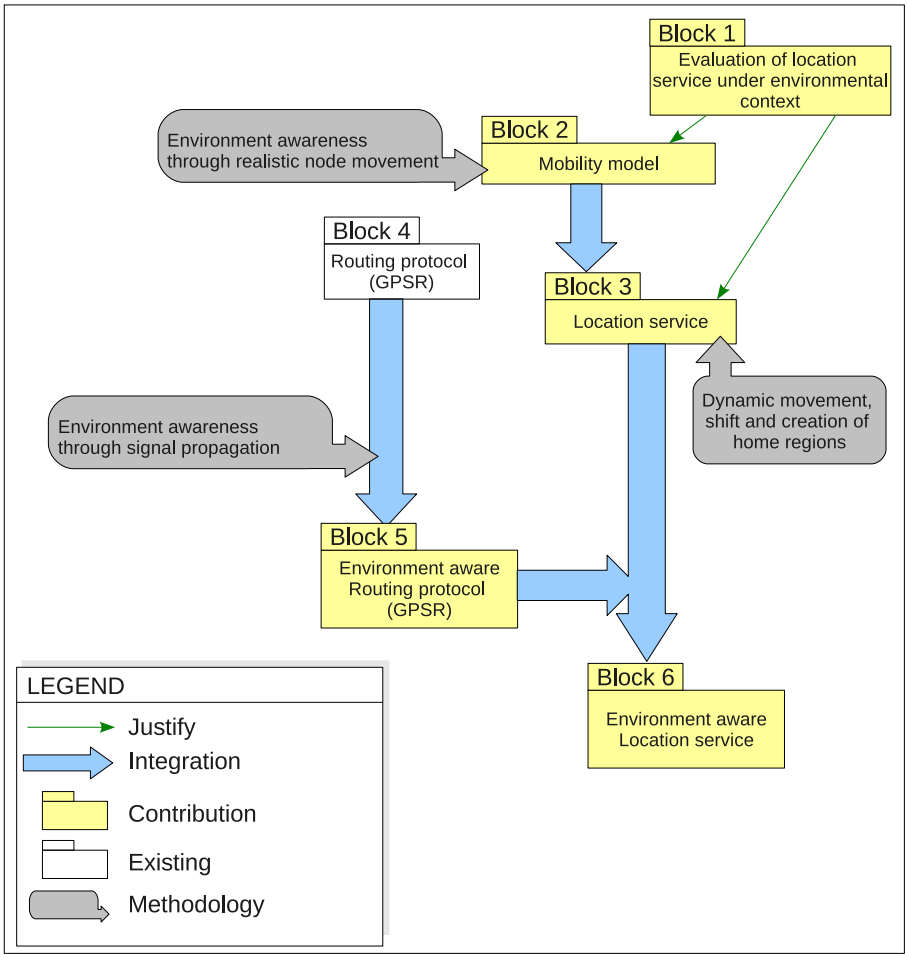


Figure 1.1: A schematic diagram illustrating this dissertation's contributions.

identifies the inadequacy of the existing mobility models in representing a realistic environment and consequently creates the logical foundation of this thesis. This evaluation is published by Ahmed *et al.* in [32].

2. As the mobility model plays the key role in representing the deployment environment, a novel mobility model is developed that considers obstacles (both accessible through doorways and inaccessible ones), pathways and regions of interest, and simulates the node movement in a realistic manner in their presence. A detailed computational complexity analysis is also presented for this model. Through extensive simulation using a real-world map containing all the environmental elements, the impact of environmental attributes on node mobility are explored in detail. Additionally, a more appropriate radio signal propagation

model is developed that considers signal fading due to obstacle more realistically. The initial version of this mobility model was published by Ahmed *et al.* in [33], and a detailed and extended version has been accepted in [34] (currently in press).

3. In the process of developing the above mobility model, a mobility trace generation tool is developed which allows to incorporate a real-world map by placing prominent entities such as obstacles of any shape, that can be inaccessible or accessible (buildings) through doorways, as well as pathways. By choosing various options a wide variety of mobility traces can be generated. These mobility traces can be plugged into any discrete event simulator e.g., ns2. The mobility generator software is publicly available at http://arrow.monash.edu.au/hdl/1959.1/109933.

4. A new location service scheme is developed that can cope significantly well with the impact observed in a real-world environment. For this scheme a novel methodology is introduced making the location service hierarchically organized without utilizing hash function in the assignment of location servers. A new strategy is adopted that allows the shifting, creating and destroying home regions whenever necessary. These features, unique to this location service protocol, provide the required flexibility needed in varying node distribution scenarios, and are crucial to attain the desired level of accuracy in maintaining up-to-date location information. More importantly, the service is scalable and is analyzed through a detailed analytical model. Extensive simulation results confirm the scheme's superiority over other contemporary location services. This location service protocol is published by Ahmed *et al.* in [31].

5. A novel methodology is introduced that takes the signal fading encountered in the physical layer into account, and therefore, allows a node to create a more accurate reachable neighbor list towards routing decision making in the network layer. This methodology improves the routing protocol considerably by ensuring a higher packet delivery rate.

6. Finally, the enhanced routing protocol is combined with the proposed location service and evaluated on the platform of a real-world environment model mentioned in item no. 3. Various aspects of the location service are explored and the results demonstrate its better applicability than the existing schemes to the

real-world environment.

1.4 Organization of Thesis

The thesis is organized as follows:

Chapter 2 presents an overview of relevant research in mobility models and location services. This dissertation aims to improve the applicability of geographic routing protocols through the location service for the real-world environment model. As one of the key criteria to reflect the simulation environment is the mobility model, it is necessary to study the existing models first. In this process the advantages and limitations of the existing models are identified. Then the strength and weakness of the existing location services are also discussed from a realistic environment point of view.

Chapter 3 presents a performance evaluation study regarding the applicability and performance of a representative location service protocol in the presence of obstacles. This study provides the philosophical foundation of this thesis by identifying some key research challenges. These research challenges are addressed in the subsequent chapters of the thesis.

Chapter 4 introduces a novel mobility model which considers real-world environment features in defining node mobility. The proposed mobility model captures node movement in the real-world more effectively by simulating features of the environment not considered in the current models in the literature. In this regard a Graphical User Interface (GUI) based mobility trace generation software is developed where a real-world map with all its environmental settings can easily be incorporated. The computational complexity of the proposed model is also presented in detail. Extensive simulation study confirms its effectiveness in representing real-world mobility.

Chapter 5 proposes a new location service protocol that is adaptive in nature. A novel home region assignment and shifting mechanism is developed that can cope with non-uniformly distributed nodes. Furthermore, a strategy for on demand home region creation and destruction is also integrated, which introduces a completely

new and unique concept in location service protocols. A mathematical model is developed that analyzes the scalability of the proposed location service. The effectiveness of the proposed location service is analyzed in detail through extensive simulation and the results show a superior outcome over the existing schemes.

Chapter 6 introduces a new methodology to take the environmental impact on signal propagation into consideration while making routing decisions in a location based routing protocol. A mathematical model is developed in this respect. Extensive simulation studies confirm the effectiveness of the improved geographic routing in a real-world environment scenario. As the location service uses the underlying routing protocol for transmitting control messages, the integration of the enhanced routing protocol which is aware of the environment further improves the performance of the proposed location service, as demonstrated by extensive simulations.

Chapter 7 presents the conclusions and explores some new research directions based on the original findings presented in the thesis.

Chapter 2

Mobility Models and Location Services - An Overview

As highlighted in Chapter 1, devising a location service suitable for ad hoc networks is the primary research goal in this dissertation. Thus two interwoven research areas of mobility models and location services need to be investigated. In order to emulate the real-world environment in simulation and assess network protocols, the mobility model is the key aspect as the environment is reflected through it. If the mobility of the nodes does not properly emulate the true movement pattern in the real-world scenario with the consideration of environmental context, any assessment of a network protocol does not become fruitful. This motivates the current work to first concentrate on the development of a mobility model which realistically reflects the environment, and secondly, on devising innovative strategies to develop a location service considering the environmental impact. Before presenting these strategies in subsequent chapters, the fundamentals and key features of mobility models and location services are explored and presented in detail in this chapter. Section 2.1 explores the characteristics of various mobility models, and more importantly, their accuracy in representing the real-world, and their respective limitations. Similarly, Section 2.2 presents the state of the art of location service schemes and highlights their capabilities and limitations in providing accurate location information.

2.1 Mobility Models

Mobility models are designed to describe the movement of mobile nodes along with their location, velocity and acceleration – that change over time. Different mobility models impact the network performance differently; as a result, it is necessary to de-

sign a flexible mobility framework that caters for generating a wide range of scenarios and identifies their impacts. Since mobility models play a significant role in network performance evaluation, it is of paramount importance that the model reflects the mobility pattern of the targeted real life application. If the selected mobility model does not emulate the application environment properly, the obtained results and conclusion may be misleading. For instance, it is not appropriate to evaluate the network performance with the simple Random Waypoint model [35] when the projected application is focused on a scenario containing nodes that move in groups. For this reason the selection of the mobility model should be made very carefully.

A variety of mobility models have been proposed by describing node movement in a detailed and realistic manner [36]. This approach has become an attractive platform for simulation and is the preferred research method. Mobility models can be classified into two major categories: 1) *Trace* and 2) *Synthetic* models. Traces are based on movement patterns observed in real life, and involve a large number of participants observed over a long time to obtain a more accurate information. However, it is not always possible to model ad hoc network by traces [27]. On the other hand, synthetic models attempt to realistically represent node movement behavior, but in many cases, does not allow derivation of analytical expression [37]. Figure 2.1 presents a detailed classification of the mobility models used for ad hoc networks.

These mobility models can be further classified as either *macro* or *micro* mobility models based on the level of detail generated. Macro mobility models specify a set of *spots* a node visits and spends time. These models define when a node should move from one spot to another, but do not concentrate on guiding the node through a more specific pathway or route. On the other hand, micro mobility models concentrate on defining the actual route a node takes [38].

The remainder of this section is organized as follows: Section 2.1.1 presents various mobility models based on real-world traces. In Section 2.1.2 a detailed review of Synthetic models is presented. Section 2.1.3 presents a few projects related to defining and analyzing mobility models currently being carried out by various researchers. Finally, the summary of this section is provided in Section 2.1.4.

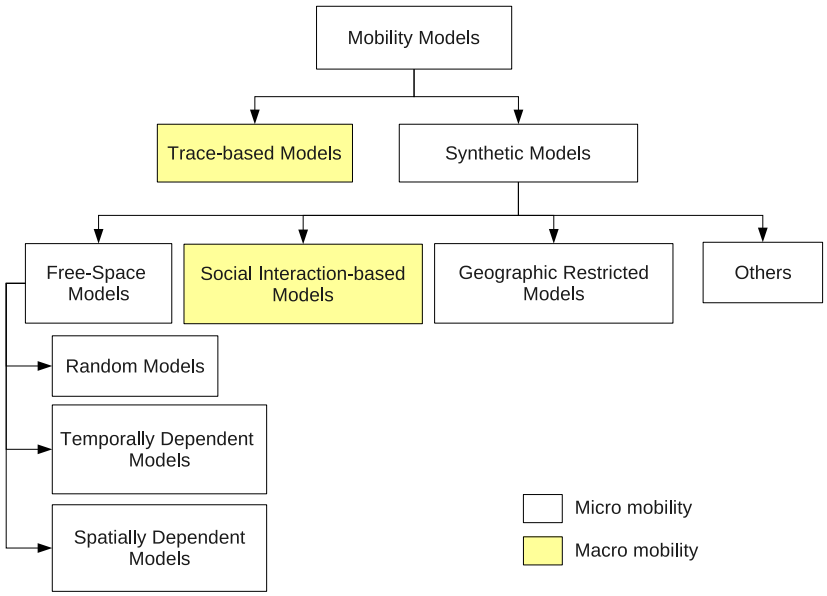


Figure 2.1: Classification of mobility models.

2.1.1 Trace-based Models

Traces are mobility patterns generated from the real-world systems. They are usually developed by analyzing stored information in an existing network system with a large number of participants over a long observation period. Most of the trace based models are macro mobility models; i.e., they do not define the actual path a node takes, rather specify where it goes at a higher level of abstraction.

Hsu *et al.* [39] proposed the Weighted Waypoint model which is based on the observation that pedestrians do not randomly choose their destinations, rather they select popular locations and go there more often. For instance, in a university campus, students mainly focus on moving towards the library, cafeteria and classrooms. To find the regions of attraction, 268 students were asked to maintain diary of their daily movements for a period of one month. Their movements mainly led to classrooms, library and cafeteria. With this survey data, the authors calculated the distribution of visiting those locations and their respective pause time. Rather than selecting the destination randomly, the Weighted Waypoint model identifies popular locations (hotspots) in the target environment and assigns different weights to them according to the probability of choosing a particular spot based on the achieved survey data. The weight of choosing the next destination spot depends on the current time and place, and a time-variant

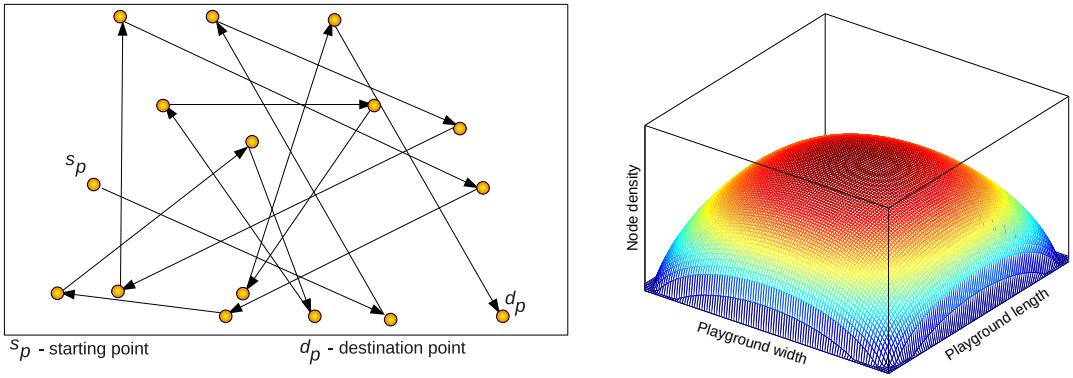
Markov model is used to capture this phenomenon. This model also adopts a pause time distribution based on the nature of the spots. For instance, students stay in the library for longer average duration than in the cafeteria. Similarly, Patterson *et al.* [40] focused on studying common destinations in a user's daily life with GPS data with the intention of providing transportation assistance to people with mild cognitive disabilities. In this work a hierarchical activity model was designed to represent a person's outdoor activity, where the top level estimates the current goal, the middle level represents segments of a trip and the lowest level estimates the person's location on the street.

A more recent study has been done by Kim *et al.* [41] making use of the data collected over several years by a Wi-Fi network comprising of approximately 560 access points and around 10000 users at Dartmouth College. While the users are on the move, they register with the nearest access points, and with this stored information, the authors extracted a synthetic mobility model and identified hot spot regions with a pause time distribution. However, as mentioned by Kim *et al.* [41], the extracted mobility model describes user movement from one hot spot to another but does not validate the real path taken between hotspots.

Through traces a variety of important features of mobility can be identified, but as stated by Camp *et al.* [27], it is difficult to simulate a new network environment if traces have not yet been created using an existing setup established for a long period of time. Moreover, since MANETs have not yet been deployed on a wide scale, obtaining real mobility traces becomes a major challenge [42]. For this reason *Synthetic models* are developed. Here, the researchers have proposed different types of mobility models where mobile users and their movement paths are created probabilistically, in an attempt to capture various characteristics of realistic mobility.

2.1.2 Synthetic Mobility Models

In this section we present some of the contemporary synthetic mobility models widely used in ad hoc network research. Synthetic mobility models attempt to represent behavior and mobility of nodes without the use of traces. This approach has become popular among researchers because it does not require any existing setup or stored information that has been collected over a long duration. Initially synthetic models were



(a) (b)

Figure 2.2: Random Waypoint model, (a) node movement strategy, and (b) Non-uniform spatial distribution of nodes in simulation.

developed by considering simple environments (e.g., free space). However, currently researchers are concentrating in developing more complex models that can represent deployment environments more realistically. Section 2.1.2.1 describes various free space models including temporally and spatially dependent models. Some researchers have developed mobility models based on social interaction, which mainly focus on defining the movement of nodes from one interaction point to another at a higher level of abstraction, as described in Section 2.1.2.2. Section 2.1.2.3 presents some mobility models that consider the presence of various environmental contexts (e.g., buildings, roads) and define node movement in a high level of detail. Finally, in Section 2.1.2.4, a few specific application focused mobility models are presented.

2.1.2.1 Free Space Models

In the initial stages of ad hoc network research, most researchers used free space models in analyzing the network performance. As the name indicates, this type of model considers the mobility playground free of obstructions, where any node can move from one position to another. Even today, researchers are interested in investigating these models analytically to gain a better insight of their characteristics. The most popular and widely used free space models are presented in the following:

Random Models: The Random Waypoint model is one of the most widely used mobility models. In this model a node randomly selects one location in the

simulation space as its destination. It then travels towards the destination with a constant velocity chosen uniformly and randomly within the range of $[0, \nu_{max}]$, where ν_{max} represents the maximum allowable velocity for every mobile node. This velocity is selected independent of the velocity of other nodes. When the node reaches its destination, it pauses for some time, t_{pause} which is termed *pause time*. Then the node selects another destination and the whole process is repeated again and again until the simulation ends. Figure 2.2(a) shows how a node moves from an initial position s_p to a destination d_p by selecting several points as intermediate destinations. If $t_{pause} = 0$ then this model represents a continuous mobility pattern. Though the Random Waypoint model has been used more frequently than any other random model due to its ease of implementation, it has some special characteristics that have a notable influence on the performance measurements. The spatial node distribution of the Random Waypoint model transforms from uniform to non-uniform after the simulation starts, and over the course of time the imbalance in spatial node distribution increases until it reaches a steady state [43, 44]. In this steady state, the center of the simulation area has the maximum node density whereas it is almost zero around the boundary of the simulation area. This phenomenon is termed *non-uniform spatial distribution* [42]. Here, nodes are more likely to move towards the middle of a simulation area or to an area where it is required to move through the middle area. Thus nodes tend to cluster near the middle of the area and the node density decreases around the boundary areas as shown in Fig. 2.2(b). Royer *et al.* [28] observed another phenomenon called *density wave*, where the average number of neighbors of a node periodically fluctuates as the time passes.

The Random Direction model [28] was introduced to overcome the density waves in the average number of neighbors and the non-uniform spatial distribution produced in the Random Waypoint model. In the Random Direction model, a node chooses a random direction and speed uniformly distributed between $[0, 2\pi]$ and $[0, \nu_{max}]$, respectively. If while moving, a node reaches the boundary, it pauses for t_{pause}, then again randomly selects another direction and speed, and this process continues until the simulation ends. Figure 2.3 illustrates a node's movement behavior in the Random Direction model. Since nodes are allowed to move towards the boundary of the simulation area, nodes at the boundary are expected to have

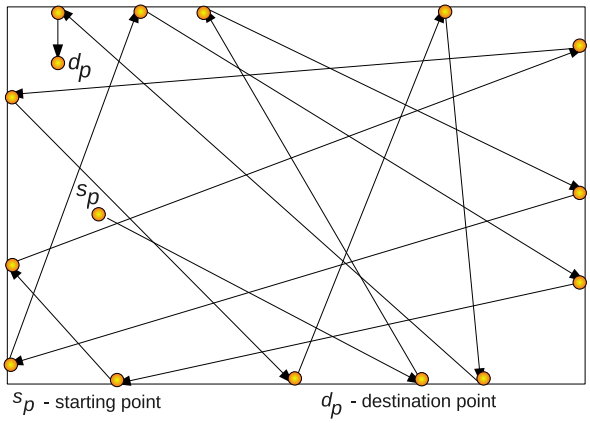
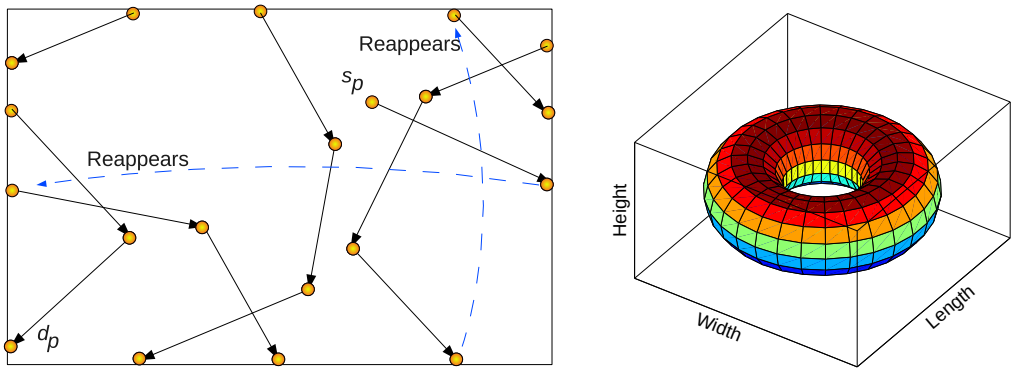


Figure 2.3: Node movement strategy in the Random Direction model.

a lower density function compared to the ones in the middle. Moreover, a portion of the boundary nodes' transmission area that extends beyond the boundary becomes useless, resulting in a non-uniform number of neighbors and link duration that consequently affect effective connectivity measurement. This phenomenon is termed the *boundary effect* [45]. Moreover, network partitions are more likely to happen in the Random Direction model compared to the Random Waypoint model.

A slight modification to this model has also been proposed by the same authors where the nodes continue to choose random directions [28], but they are no longer forced to travel to the simulation boundary. Before stopping at the boundary to change direction, a node chooses a random direction and selects a destination anywhere along that direction to travel. The node then pauses at this destination before choosing a new direction. In this model, the level of boundary effect and network partitioning is reduced but the density wave increases slightly over the Random Direction model.

The Boundless Simulation Area model was introduced to alleviate the boundary effect observed in the Random Direction model. In this model [7], the nodes that reach one side of the simulation area continue traveling and reappear on the opposite side of the simulation area as shown in Fig. 2.4(a). This technique creates a torus-shaped simulation area (Fig. 2.4(b)) allowing the nodes to travel unobstructed. Though the Boundless Simulation Area model effectively reduces the *boundary effect*, the node movement is unrealistic because of their reappearing



(a) (b)

Figure 2.4: Boundless Simulation Area model, (a) node movement strategy, and (b) Torus shaped mapping resulting from the movement.

at the opposite side of the simulation area.

Another frequently used mobility model is the Random Walk model which was originally formulated to emulate the unpredictable movement of particles in Physics [27], also referred to as *Brownian motion*. It has similarities with the Random Waypoint model, and can be viewed as a Random Waypoint model with a zero pause time. In this model, a node changes its speed and direction at each time interval. For every new interval each node randomly and uniformly chooses its new direction θ_i, within the range $[0, 2\pi]$ and a new speed ν_i which follows a uniform or Gaussian distribution within $[\nu_{min}, \nu_{max}]$ and then moves in that direction at that speed for either a specified number of steps or a time period. At the end of the period, the nodes repeat this process. During time interval i, the node moves with the velocity vector $(\nu_i \cos\theta_i, \nu_i \sin\theta_i)$. While moving in this way if a node reaches the boundary of the simulation field, it bounces back into the simulation field according to the law of reflection of light with an angle of θ_i or $\pi - \theta_i$. This phenomenon is called the *border effect* [43]. One interesting observation made by Polya [46] about the Random Walk model is that, a random walk returns to its starting point with a probability of 1.0.

The mobility models described so far are all memory-less mobility models where any future movement decision is fully independent of the previous movement parameters (e.g., speed and direction). Since the future speed and direction of a node is independent of its previous speed or direction, these mobility models

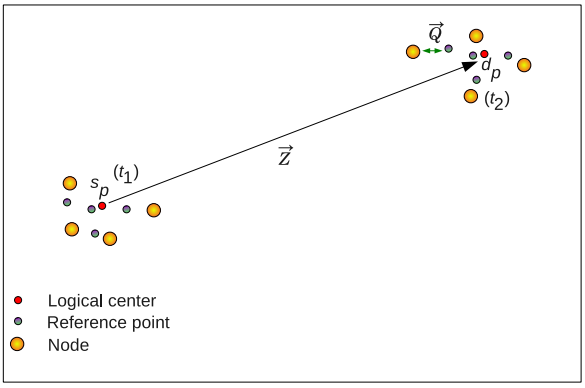


Figure 2.5: Node movement strategy in Reference Point Group Mobility model.

can generate unrealistic behavior (e.g., sharp turns and sudden stops). Thus *Temporally dependent models* were introduced to overcome these issues.

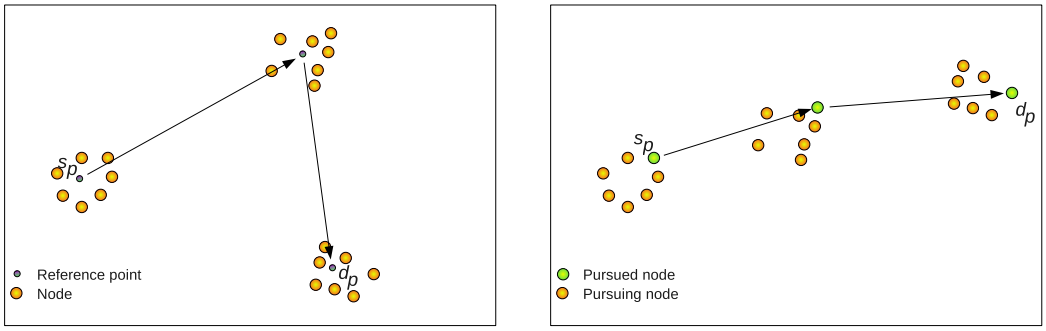
Temporally Dependent Models: The Gauss-Markov Mobility model was designed to introduce temporal dependency in a mobility model. Initially, each node is assigned a speed and a direction. Afterwards the speed and direction at the i-th instance is calculated on the basis of their values at the $(i-1)$-th instance [47] using the following formula

$$\nu_i = \zeta\nu_{i-1} + (1-\zeta)\bar{\nu} + \sqrt{(1-\zeta^2) \times \nu_{i-1}}, \tag{2.1}$$

$$\theta_i = \zeta\theta_{i-1} + (1-\zeta)\bar{\theta} + \sqrt{(1-\zeta^2) \times \theta_{i-1}}, \tag{2.2}$$

where, ν_i and θ_i are the speed and direction of the i-th time interval, respectively; ζ $(0 \leq \zeta \leq 1)$ is the tuning parameter for degree of randomness; $\bar{\nu}$ and $\bar{\theta}$ are the constant mean values of speed and direction, respectively, and ν_{i-1} and θ_{i-1} are the speed and direction of $(i-1)$-th instance, respectively.

Spatially Dependent Models: Spatially dependent mobility models introduce a dependency in a node's movement pattern on the movement of other nodes. For this reason the Group mobility models have been proposed to model the situations where the nodes or a subset of the nodes move as a group, maintaining spatial dependency.



(a) (b)

Figure 2.6: Node movement strategy in (a) Nomadic mobility, and (b) Pursue mobility model.

Among the group mobility models available in ad hoc network research, the Reference Point Group Mobility model (RPGM) [48] is the most frequently used. This scheme generates a movement pattern by defining the random motion of a group of nodes. At the same time it defines how individual nodes will roam within their group. For this reason, a logical center is defined for a group. This logical center is used to calculate the group motion through a motion vector $\overrightarrow{Z}$ (Fig. 2.5). The motion of the group center characterizes the movement of the nodes in the group including their direction and speed. A reference point for each node is also defined, and a node randomly moves around its own predefined reference point. As shown in Fig. 2.5, the reference points of all nodes are updated along with $\overrightarrow{Z}$, as time advances from t_1 to t_2. After updating each reference point, it is combined with the respective node's motion vector $\overrightarrow{Q}$. The length and direction of $\overrightarrow{Q}$ are chosen uniformly distributed with specified radius centered at its reference point and $[0, 2\pi]$, respectively. Thus a node can roam around its own reference point randomly. In this model, all nodes pause for the same duration when the group reference point reaches its destination. RPGM was designed for situations such as avalanche rescue operations, where responding teams consist of human and canine members. Here, each human guide defines the general path for his dogs and the dogs create their own random path.

A variation of the RPGM is the Nomadic Community Mobility model [49]. In this model, a group maintains a reference point and the nodes in a group can move around the reference point with a given degree of freedom. Each node uses

any of the random mobility models (e.g., Random Waypoint, Random Direction) and its freedom of movement from the reference point is defined by its mobility model parameters. This type of mobility model can be observed when a group of children visits a museum as the members of the group follow a guide but roam around with a small degree of freedom. After a time interval the guide (reference point) moves and all the children (nodes) in the group also move along with it. As shown in Fig. 2.6(a), the reference point of a group moves from the starting point s_p to the destination point d_p, and nodes belonging to the group move along with it while keeping a predefined flexible distance from the reference point.

Another group mobility model is the Pursue Mobility model [49] that attempts to represent a group of mobile nodes tracking a particular target node, just like a group of police chasing a criminal. Figure 2.6(b) shows how the pursued node moves from s_p to d_p and the pursuing nodes are following it. To capture a different scenario, the Column Mobility model [49] is proposed, which is another spatially dependent model specially useful for scanning or searching purposes where the nodes are allowed to move around a given column (line). This type of mobility is observed when a row of soldiers march together.

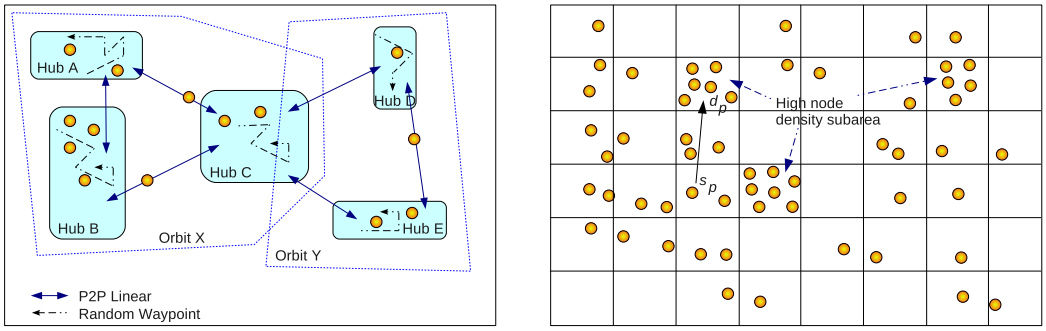
Temporally and spatially dependent models tend to replicate some important aspects i.e., temporal and spatial dependency in node movement, which is not addressed in random models. However, in all of these models the destination is selected randomly, nonetheless to mention in a free space environment. Thus they fail to realistically represent the environment, because in the real-world people tend to move to some specific places (e.g., office, home, gym) rather moving randomly. In the following section a group of mobility models is presented, that do not select their destination fully randomly, rather by considering the influence of social interactions.

2.1.2.2 Social Interaction-based Models

Similar to the trace-based models, social interaction-based models can also be classified as macro mobility model as they do not define the exact movement within a hotspot or in between hotspots. Most of the popular and contemporary social interaction based models are discussed in this section.

Orbital mobility model The Orbital mobility model [38] has similar characteristics to the Weighted Waypoint model. Here, the authors tried to emulate node movement based on sociological movement patterns. A user's mobility exhibits both time and space based hierarchy. For instance, on a typical weekday, a student may leave home in the morning, go to school during the day, visit the gymnasium in the evening and come back home at night. According to this model a number of *hub*s (e.g., gymnasium, school, home) are modeled and a node follows these hubs periodically to represent social movement. Initially a number of rectangular shaped hubs are generated randomly in the simulation terrain. Each node is assigned to a subset of randomly chosen hubs creating a random orbit. Each hub is assigned a *hub stay time* that defines how long a node will spend there. The Orbital mobility model defines two mobility patterns: (i) movement inside hubs is termed *intra-hub movement* (IHM) that follows the Random Waypoint model with a defined *intra-hub speed* and *intra-hub pause time*; (ii) movement in between the hubs is termed *inter-hub movement* where a node randomly selects a point in the destination hub and starts to move there with predefined *inter-hub speed*, and this type of mobility is termed as *P2P Linear* model. Figure 2.7(a) shows how nodes follow Random Waypoint model and P2P Linear model, while moving inside a hub and from one hub to another, respectively. Since any known practical mobility model can be used for intra-hub and inter-hub movement in place of the Random Waypoint and P2P Linear models [38], this model is flexible in catering for many sociological movement patterns. However, since the choices of the next destination hub is completely random, it lacks the regular patterns of human movement [50].

Clustered Mobility model The Clustered mobility model [51] follows a different approach in defining hot spots, and adopts social, natural and biological network's scale-free power-law connectivity distribution [52] which explains the existence of hot spot regions whose connectivity (clustering of nodes) is much larger than that of an average node and applies it in wireless network. It also facilitates the forming of hubs, similar to hot spots or regions of interest, where higher node density takes place. In this model, the nodes tend to move towards higher concentration of nodes resulting in the nodes clustering around hubs. The authors



(a) (b)

Figure 2.7: Node movement strategy in (a) Orbital mobility, and (b) Clustered mobility model.

introduce two phases, called the *growth* and *rewiring* for this mobility model. In the growth phase, the entire simulation area is divided into an n number of sub-areas, s_i^{area}, $1 \leq i \leq n$. Initially all the sub-areas are empty, so, the probability of a sub-area being assigned to the next node is equal. As a new node is assigned to a sub-area, the probability of selecting a sub-area increases or decreases depending on the current number of nodes in the specific sub-area and is calculated as follows: having k_i^{nodes} nodes in the sub-area s_i^{area}, the probability of the sub-area being selected is

$$p_i^{cluster} = \frac{(k_i^{nodes} + 1)^{\varpi}}{\sum_{j=1}^{n}(k_j^{nodes} + 1)^{\varpi}} \tag{2.3}$$

where ϖ is the clustering exponent. Following this process some sub-areas gain a higher probability of being selected than others and hence become hubs. The growth phase ends when the assignment of all nodes is complete. The rewiring phase defines the mobility of each node from one sub-area to another. At first a sub-area s_i^{area} is chosen randomly based on the value $p_i^{cluster}$ and then a random point within it is selected by each node. The node then selects a uniformly distributed random speed within $[\nu_{min}, \nu_{max}]$ and starts to move. In order to achieve a steady state node speed, ν_{min} is chosen non-zero. Figure 2.7(b) shows that a node selects a sub-area with a higher node density and moves to d_p from s_p in a simulation area divided by grids where the nodes are non-uniformly distributed.

Self-similar Least Action Walk (SLAW) In a very recent work, Lee *et al.* proposed Self-similar Least Action Walk (SLAW) [50] in order to emulate human

walking behavior, and an improvement over Levi-walk[1] nature of human movement patterns [54]. In SLAW, the authors identified some special characteristics of human walking behavior such as: people tend to travel longer distance less frequently; they mostly move only within their own confined areas while different people may have widely different mobility areas. Moreover, people usually follow a routine of visiting the same places everyday, such as going to office, but at the same time make irregular trips as well. People do not always randomly select places to visit, or follow a random order if they have multiple destinations.

This model was designed based on the collected GPS data of 226 daily traces of 101 volunteers in five different outdoor sites. Analyzing the real trace data, the authors tried to develop a synthetic trace, observing that the waypoints of human could be modeled by *fractal points*. At first SLAW generates fractal waypoints using a technique similar to Fractional Gaussian noise [55] or Brownian motion [56]. To enforce heterogeneously bounded walking areas among the different nodes, SLAW proposes to model an individual walker by selecting a subset of fractal waypoint clusters. In order to impose randomness, it also allows nodes to occasionally visit other clusters with a controlled probability. When visiting a set of destinations, nearby destinations are visited first before visiting farther destinations.

The mobility models presented in this section can emulate human mobility at a higher level of abstraction more realistically than the free space models described in Section 2.1.2.1. However, social interaction based models exhibit a common property: they do not consider movement within small time intervals or distances, and thus do not employ the actual route traveled by a node. For this reason the impact and influence of environmental aspects (e.g., obstacles, pathways) is not properly reflected in node mobility. In the following section some prominent mobility models are presented that can handle environmental aspects more appropriately.

[1]Introduced by Shlesinger *et al.* [53] to explain atypical particle diffusion not governed by Brownian motion.

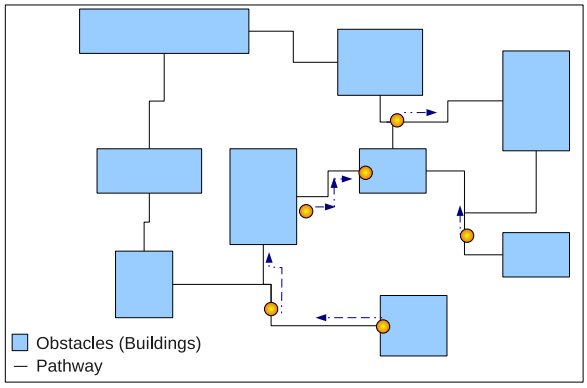


Figure 2.8: Node movement strategy in Pathway mobility model.

2.1.2.3 Geographic Restricted Models

Each of the mobility models described above assumes an open and unobstructed area where the nodes can move freely, but in the real-world situations the existence of this type of unobstructed scenario is rare, and thereby, they fail to emulate true node movements for a real-world environment. In real life, we observe that our movement is restricted by environmental contexts. For instance, in an urban area the motion of vehicles is restricted to roads and streets. Again, in a pedestrian friendly area (e.g., a school campus), the position of buildings and other obstacles influences our movement pattern. To address this issue, the node movement becomes pseudo-random fashioned on predefined simulation sub-areas or pathways. Thus various researchers have attempted to introduce environmental attributes in mobility models, and these models are classified as *Geographic Restricted models*. In the following, we describe a few mobility models that belong to this category.

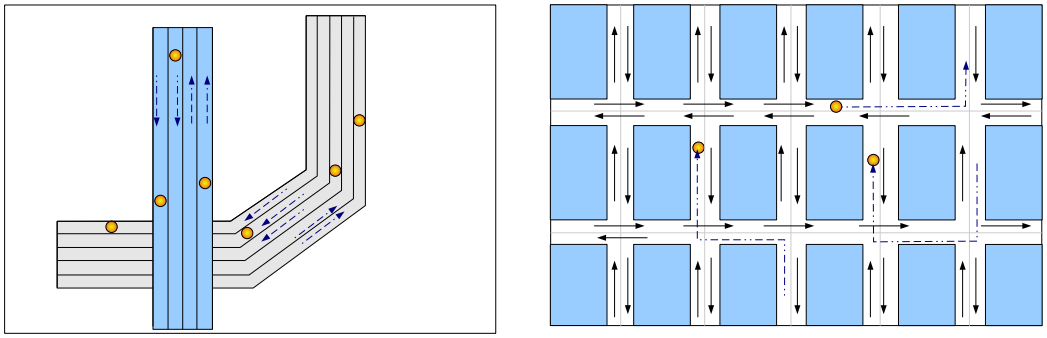
Pathway Mobility model One of the very first mobility models that takes geographic restrictions into account is the Pathway Mobility model [57, 58], which uses a predefined graph in the simulated field. The graph is generated either randomly or based on the map of a real city [59], where the vertices represent buildings and the edges model the streets and freeways linking those buildings. The nodes are only allowed to move along the predefined edges. At the beginning of the simulation, the nodes are randomly deployed on the edges of the graph. A node then randomly selects a destination and moves there with a randomly

selected speed within $[0, \nu_{max}]$, using the shortest path along the edges of the pathway graph. Upon arriving at its destination, the node pauses for some time and then selects another destination, and this process continues until the simulation ends. Figure 2.8 shows the pathway graph used in a Pathway model representing a campus, where the nodes move along the defined pathway edges. Though the Pathway Mobility model describes the environment more realistically than the free space models, it lacks randomness as the nodes are only permitted to follow the existing pathways while moving from one building to another. Even in such pedestrian friendly area, human movement can not be always expected to follow the existing pathways only.

City Section Mobility model The City Section Mobility model is actually a variation of the Pathway model where the scenario is mostly focused on a detailed implementation of node mobility within a city. In this model, the simulation area consists of the streets and blocks of a section of a city. The streets are based on the type of city being simulated and lanes and speed limits are also incorporated accordingly. The nodes are deployed randomly on arbitrary streets. After deployment each node randomly selects a point on a street and calculates the path requiring the shortest time to reach its destination taking into account the speed limit of the streets. Moreover, realistic movement behavior such as the minimum distance allowed between two nodes also exists. Upon arriving at its destination, a node pauses for some time and then randomly selects another destination in a similar manner.

In the real-world, people do not have the ability to roam freely without facing obstacles and traffic regulations. Thus the City Section mobility model provides realistic movement of nodes within a city since it significantly restricts node movement behavior by adopting real-world conditions such as traffic rules.

By enhancing the City Section mobility model Bai *et al.* [60, 61] proposed the *Freeway* and *Manhattan* mobility models. The Freeway model exploits a generated map that includes many freeways, each side of which comprises multiple lanes as shown in Fig. 2.9(a). As movement in freeways is simulated here, no intersections are considered. Initially the nodes are randomly placed on the lanes, and move using a history based speed. Two subsequent nodes in a lane main-



(a) (b)

Figure 2.9: Node movement strategy in (a) Freeway Mobility, and (b) Manhattan Mobility model.

tain a minimum safe distance. Acceleration and deceleration are also introduced into node movement and if the distance between two subsequent nodes in a lane becomes less than the safe distance, the rear node decelerates and lets the front node move away. Lane changing is not permitted in this model, which is unlike how people drive on freeways and hence is a limiting factor. The nodes continue to move until they arrive at the border of the simulation area, then they are placed randomly in another position on the freeway.

Another generated map based model focusing on urban areas is the Manhattan Mobility model. Here, the map contains vertical and horizontal roads as shown in Fig. 2.9(b). Each of the roads consists of two lanes, on which the nodes move in opposite directions. Initially the nodes are randomly placed on the roads. Upon arriving at a crossroad, a node randomly chooses a direction at either forward, left or right with a respective probability of 0.50, 0.25 and 0.25. A safe distance between nodes is also incorporated as in the Freeway model, but unlike the Freeway model, the nodes can change lanes at the intersections. However, this model does not introduce any pause time or stopping at the intersections.

The Stop Sign model [62] is the first model where traffic control mechanisms have been integrated. At every crossroad, a stop signal is placed that causes the nodes to slow down and pause at the intersection. Every node adjusts its speed to keep a safe distance from the node in front. If many nodes arrive at the intersection they form a queue where each node waits until the node in front

moves away giving it space to move. The Stop Sign model was improved by the Traffic Sign model [62] where the stop signals are replaced by traffic lights. In this model, unlike the Stop Signal model, a node stops at the intersection only if it faces red light. The light is turned red and green with the probability of p and $(1-p)$, respectively. A further enhancement was made by Gorgorin *et al.* [63] by introducing an overtaking mechanism in the mobility model.

The models described in this section are focused on simulating the movement of vehicles and are more appropriate for evaluating vehicular and cellular networks rather MANETs. A detailed survey on vehicular mobility models can be found in [64, 65]. Geographic restricted mobility models more appropriate for evaluating MANET are the obstacle mobility models discussed in the following section.

Obstacle Mobility models

In the Obstacle Mobility models, obstacles representing buildings of regular shape are distributed randomly or based on a map throughout the simulation area. While moving in the simulation area, whenever a node faces an obstacle on the way, it changes its direction and starts moving again. Johansson *et al.* [66] developed three mobility scenarios to depict the movement of mobile users in real life, which includes:

Conference scenario: This scenario consists of 50 mobile nodes representing people in a conference, where most of them are static while a small number of them are mobile with a low mobility.

Event Coverage scenario: This scenario models a group of highly mobile entities, e.g., people, vehicles or nodes that frequently change their position.

Disaster Relief scenario: This scenario models a disaster area where some nodes move very fast while the others move very slowly.

Another important issue addressed in the Obstacle model is the radio propagation model. When radio signal propagates through obstacles, a significant amount of signal strength is expected to be reduced. In the above scenarios (conference, disaster relief and event coverage), if there is an obstacle in between two nodes, the link between them is considered to be broken until one comes out of the shadowed area of the other, i.e., the signal is fully absorbed by the obstacle present in the line of sight between the two nodes.

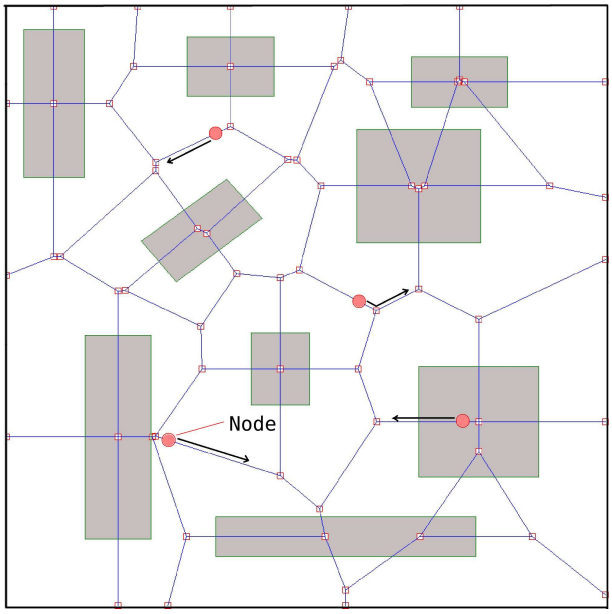


Figure 2.10: Node movement strategy in Voronoi graph based model.

Jardosh *et al.* [67, 29] also investigated the impact of obstacles on mobility modeling in detail. They observed that, in real life people mostly follow predefined pathways between buildings, instead of moving randomly and bouncing back off the buildings. To capture this type of behavior, they proposed a Voronoi graph based mobility model that generates a Voronoi graph, based on the corner positions of obstacles and directs the node movement along the edges. In their simulation field, a number of obstacles were placed to model buildings within a campus environment. Then a Voronoi graph is generated to compute the movement path for the nodes as shown in Fig. 2.10. In the simulation area, the nodes are randomly deployed only on the Voronoi edges. For each node, a random point on the Voronoi edges is selected as its destination and the node starts to move towards it following the shortest path through the generated pathway graph. Here, the nodes are only allowed to move on the edges of the Voronoi cells and enter or exit buildings. According to them, it is a general observation that when a pathway in the real-world passes between two adjacent buildings, it usually keeps an equal distance from each building. The Voronoi graph captures this behavior as it is a planner graph having the edges as line segments equidistant from two corner points of obstacles. Another observation made by Jardosh *et al.* is that, usually, doors are created along the middle of a wall. Since in the Voronoi graph, an edge is generated

keeping an equal distance between two adjacent corner points, there always exists an edge for each wall of the obstacles, that intersects the corresponding wall at the middle position representing a *door* to enter and exit. In this model the nodes are allowed to enter and exit buildings only through these doors.

Since in this model obstacles are considered, the authors have proposed a radio propagation model applicable for such scenarios. In [67], they propose a model where the link between two nodes is assumed to exist if there is a straight line-of-sight between them, as long as both of them are within each others transmission range. A more realistic propagation model is proposed by Jardosh *et al.* [29] where the signal attenuation due to the obstacles is considered based on the obstacle material. Here the signal strength is reduced according to the number of walls present between the transmitter and receiver and if the resultant signal strength is more than a receiving threshold value, it is assumed that the link exists in between the transmitter and receiver. Thus this model designs a partially penetrating radio signal propagation model that is not necessarily based on straight line-of-sight transmission, and is considered to be a more realistic propagation model.

As observed by Konishi *et al.* [68], the Voronoi based mobility model considers microscopic movement but lacks the ability to represent microscopic behavior, e.g., how each node changes its behavior dynamically depending on its surrounding information obtained from network system and time. In this regard, they proposed a rule based mobility model to describe the movement behavior of mobile nodes. Here, the nodes are classified into multiple groups depending on their behavior patterns. Each group of nodes is assigned a rule-based description where dynamic and realistic behaviors of nodes (e.g., visiting favorite shops, and avoiding congested routes) are specified. In this model, the node movement is mainly focused on existing pathways designed through the *MobiREAL* simulator, and thus can be considered as an improved version of the Pathway mobility model. This model also incorporates the presence of obstacles and for this reason a radio propagation model similar to [67] is used.

Lu *et al.* [69, 45] have also proposed a mobility model belonging to the geographic restricted mobility models category. Based on an environment where MANETs can be deployed, several environmental objects such as routes, junctions, hotspots, lanes, paths, normal accessible areas, and inaccessible areas are considered. The definition of

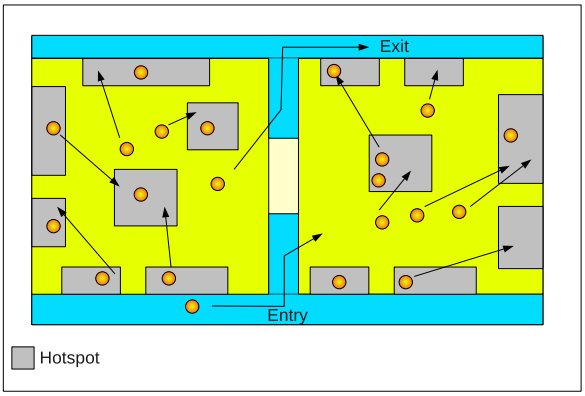
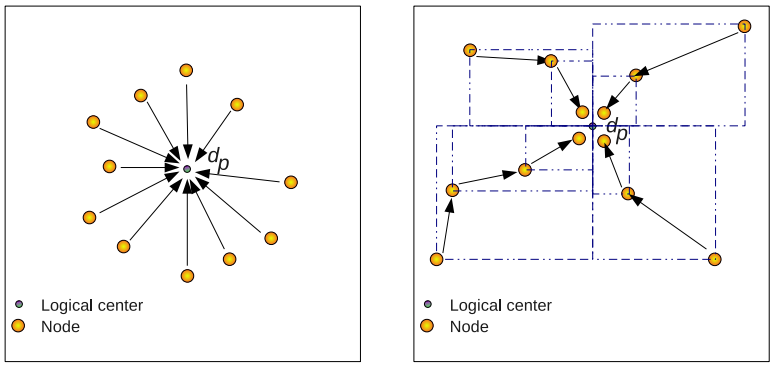


Figure 2.11: Node movement strategy in the Hotspot model applied to the Art gallery scenario.

these objects is made through Scalable Vector Graphic (SVG) which uses Extensible Markup Language (XML) to describe two-dimensional graphics. SVG contains a set of objects of basic shapes, e.g., rectangle, line, circle etc. It also has the ability to change the vector graphics over time for visual enhancement. The movement trajectory of a mobile node is correlated with the sub-area within which it is located and is allowed to be changed during the simulation. In this regard, they proposed have two different mobility models: the *Route model* and the *Hotspot model*.

Route model: This model focuses on relatively complex environments, such as a city area or urban road network similar to the Pathway model or Manhattan model. Here, the nodes move on the predefined routes and when they reach the end of a route, reappear at the entry of a new route.

Hotspot model: In this model, a number of hotspots are specified and the nodes become interested in visiting these spots. Two different scenarios for this model are evaluated, namely, the *Art Gallery* and *Festival Event*. In the Art gallery scenario, the simulation area is divided into two *rooms* (Fig. 2.11) and visitors can enter the rooms through the *entry*. After entering into a room, a visitor moves randomly towards one of the initially defined hotspots. When he finds his hotspot too crowded, he selects another spot. Unlike the Art gallery, the Festival event scenario models some public places, such as a park, where the nodes move around freely. This can be considered as a variation of the Random Waypoint model with hotspots, where the nodes randomly select a position in a randomly chosen hotspot.



(a) (b)

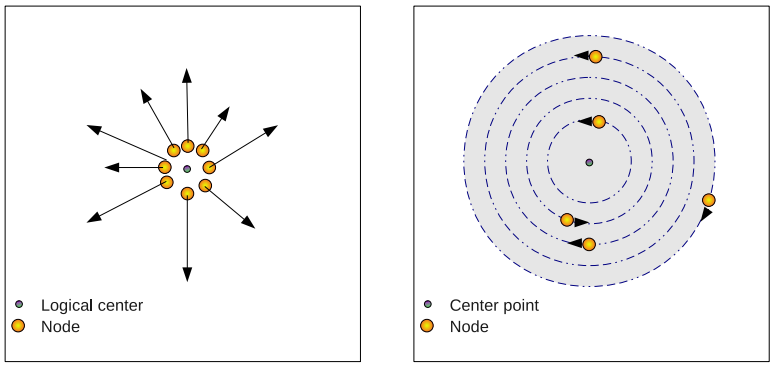
Figure 2.12: Node movement strategy in (a) Contraction, and (b) Modified Contraction model.

2.1.2.4 Other mobility models

There has been a number of mobility models proposed to represent specific events and are only applicable for such events. In this section, some of these recent models are presented.

Lu *et al.* [70] divided the mobility models into two groups called *Recurring* and *Terminal* mobility models. They classified the existing free space models described in Section 2.1.2.1 as recurring models, where the same synthetic model is repeated throughout the simulation without termination or changing of the mobility parameters. While in the Terminal mobility models, the node movement changes over time (e.g., some nodes may stop), which has significant impact on the time related characteristics of performance measurements. In this regard, the *Contraction*, *Expansion* and *Circling* mobility models have been proposed.

The Contraction model emulates the movement of mobile nodes from all directions towards a logical center. In the beginning, all the nodes are uniformly distributed in the area. Each node moves towards the center in a straight line and at every time interval (e.g., 10 sec), it selects a random speed within $[1, \nu_{max} - 1]$ and may pause for a random time depending on the pause probability and maximum pause time. Upon arriving at the center, the nodes stop (Fig. 2.12(a)). Since in reality people do not always walk along a strict straight line, a modified contraction model is proposed where for each time interval, a node selects a uniformly distributed random destination point within



(a) (b)

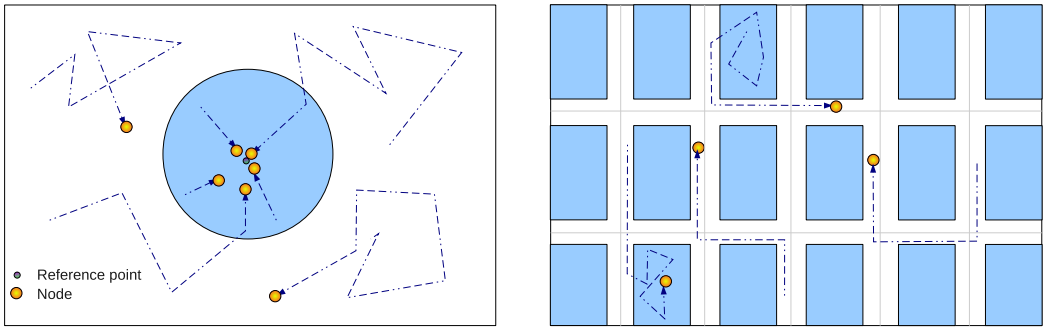
Figure 2.13: Node movement strategy in (a) Expansion, and (b) Circling model.

the rectangular area defined by its current position and the center reference point with a randomly chosen speed within $[1, \nu_{max} - 1]$. In this way, the nodes eventually reach the center by moving closer to the center in each interval, as shown in Fig. 2.12(b). This type of mobility is observed in scenarios such as students gathering for a class.

The opposite of the movement pattern in the contraction model is defined in the expansion model. Here, nodes move away from the center reference point towards the border of the simulation area, as shown in Fig. 2.13(a). Here, the nodes are initially distributed uniformly throughout the area, or in some cases, the initial node deployment is constrained to a small area around the center. When a node arrives at the boundary of the area, it stops and stays there until all other nodes have arrived at the border, completing the simulation. Applications of this model include evacuation processes, where people are evacuated from a specific place.

In the Circling model (Fig. 2.13(b)), the mobile nodes move in a circular manner around a center point. Each node is assigned a unique radius value and it moves in a specific direction. For each interval, a node picks the next destination position at the same radius and moves with a uniformly distributed random speed $[1, \nu_{max} - 1]$. After arriving at its destination it pauses for some time, and the process continues until the simulation ends. A similar pattern of movement is observed when a car circles around to park in a car park.

In reality, a simple mobility model with only one movement pattern is not adequate in capturing all the characteristics of realistic movements. Thus, *hybrid* models



(a) (b)

Figure 2.14: Node movement strategy in the (a) Hybrid Contraction and Random Waypoint model, and (b) Hybrid Manhattan and Random Waypoint model.

are proposed, where the movement of a node may switch from one mobility model to another based on its location or simulation time. The *Hybrid Contraction and Random Waypoint model* (Fig. 2.14(a)) divides the simulation area into two regions: a circular region and a region external to it. Initially the nodes are uniformly distributed in the simulation area. The nodes in the circular area move according to the Contraction mobility model and the other nodes use the Random Waypoint model. When a node moving according to the Random Waypoint model enters the circular area, its mobility changes to the Contraction model and it starts moving towards the center. Applications of this model include student mobility in a campus where students move around different places reflected by the Random Waypoint model. The mobility of a student follows the Contraction model when he enters a building to reach a classroom. Another model is the *Hybrid Manhattan and Random Waypoint model*, where the area is a map with horizontal and vertical streets and blocks. When the nodes move on the streets, they move according to the Manhattan Model as described in Section 2.1.2.3. On the other hand, the nodes move according to the Random Waypoint model while moving inside the blocks. The nodes can move from street to block and vice versa, and their mobility changes according to the mobility model of their specific place. This type of movement is observed in the city where people move on the streets, and at times enter and exit buildings.

Though the Geographic restricted mobility models presented in Section 2.1.2.3 give a flavor of the realistic movement of mobile nodes suitable for MANETs, some essential

elements of the real-world mobility are still lacking. These models generate very rigid node movement mainly based on pathway graphs, thereby lacking randomness, but people are not always expected to move in such a way. Thus these models are incapable of generating a wide range of mobility scenarios and need further refinement.

2.1.3 Mobility Model Projects

As highlighted in Chapter 1, mobility modeling is an important issue in analyzing ad hoc networks. Such modeling has resulted in a number of projects where various researchers tried to emulate realistic movement. In this section a number of such projects are presented.

Realistic Mobility Model [71] The objective of this project is to model the mobility of mobile nodes in a better way and is mainly focused on vehicular ad hoc networks. This project uses TIGER (Topology Integrated Geographic Encoding and Referencing) database from the United States Census Bureau where a detailed street maps with automobile traffic is available.

Obstacle Mobility Model [72] This project considers various obstacles in the simulation environment and at the same time incorporates a radio signal propagation model that can create a more realistic environment. Through the corner points of the obstacles designed in the simulation playground, a Voronoi graph is generated and the nodes are directed to move on the Voronoi edges. Here, nodes are randomly placed on the pathways (Voronoi edges) and use the shortest path to reach their randomly chosen destination on the pathways or within an obstacle.

Self-similar Least Action Walk (SLAW) [73] This project tries to emulate human walking behavior. Here, real-world traces have been used to identify few human walking characteristics and then synthetic mobility traces are generated using fractals. Further details were presented in Section 2.1.2.2.

Mobility Models for Mobile Ad Hoc Network Simulations [74] This project focuses on developing mobility models from the collected movement data of wireless users. It also analyses different types of node distributions caused by different mobility models.

IMPORTANT Mobility Framework [75] In this project various protocol independent metrics are proposed to capture the mobility characteristics of mobile nodes. The characteristics of several mobility models including the Random Waypoint, Random Walk, Group Mobility, Freeway, Manhattan and City Section models are evaluated through this framework.

MobiREAL [76] MobiREAL is a network simulator built in C++ for simulating MANET. It can simulate the realistic mobility behavior of humans and automobiles, and enables to change their behavior depending on a given application context. Here, a probabilistic rule-based model to describe the behavior of mobile nodes is adopted. The proposed project allows to describe how mobile nodes change their destinations, routes, speeds and directions based on their positions and surrounding environment. It also incorporates a visualization tool to observe the nodes' mobility.

STreet RAndom Waypoint (STRAW) [77] This project is focused on vehicular network where the mobile nodes move on the streets based on the real maps of United States cities. Here, the nodes emulate movement on the streets with a simplified traffic control system. This project also considers the propagation of signal affected by the presence of obstacles (e.g., buildings, trees etc.) and thus models signal propagation more realistically.

DAVIS Smart Mobility Modeling Project [78] The goal of this project was to optimize individual mobility options at the University of California in Davis through an improved network connectivity and transportation system. It was originated to analyze the impact of transportation during campus expansion.

2.1.4 Summary

Mobility models in wireless ad hoc networks define an individual node's motion behavior. A number of mobility models concentrate on node movement in free space. Some variants define the speed and direction of node mobility based on the node's previous speed and direction. Some others emulate group mobility. A different class of mobility models focuses on defining node mobility at a higher level of abstraction, considering existing real life trace data or by analyzing human behavior in social interactions. Some

Table 2.1: Comparison of different mobility models.

Model	Type	Detail	Radio model	Movement	Hotspot	Group	Applicable	Effect	Realistic
Weighted Waypoint	trace	macro	no	random	yes	no	campus	-	medium
Dartmouth	trace	macro	no	random	yes	no	urban	-	high
Random Waypoint	synthetic-free space	micro	no	random	no	no	-	density wave	no
Random Direction	synthetic-free space	micro	no	random	no	no	-	boundary effect	no
Boundless Simulation Area	synthetic-free space	micro	no	random	no	no	-	border effect	no
Random Walk	synthetic-free space	micro	no	random	no	no	-	-	no
Gauss-Markov	synthetic-free space	micro	no	random	no	no	-	temporal	low
Reference Point Group	synthetic-free space	micro	no	random	no	yes	-	-	medium
Nomadic Community	synthetic-free space	micro	no	random	no	yes	-	-	medium
Pursue	synthetic-free space	micro	no	random	no	yes	-	-	medium
Column	synthetic-free space	micro	no	random	no	yes	-	-	low
Orbital	synthetic-social	macro	no	pseudo-random	yes	no	urban/pedestrian	-	medium
Clustered	synthetic-social	macro	no	random	yes	no	urban/pedestrian	-	medium
SLAW	synthetic-social	macro	no	random	yes	no	pedestrian	-	high
Pathway	synthetic-geographic	micro	no	graph based	no	no	campus	-	medium
City Section	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	medium
Freeway	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	medium
Manhattan	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	medium
Stop Sign	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	medium
Stop Signal	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	high
Obstacle	synthetic-geographic	micro	yes	graph based	no	no	campus-pedestrian	-	high
Route	synthetic-geographic	micro	no	graph based	no	no	vehicular	-	medium
Hotspot	synthetic-geographic	micro	yes	graph based	yes	no	pedestrian	-	high
Contraction	synthetic-geographic	micro	no	random	yes	no	pedistrian	-	medium
Expansion	synthetic-geographic	micro	no	random	no	no	pedistrian	-	medium
Circling	synthetic-geographic	micro	no	circular	no	no	vehicular	-	low
Hybrid	synthetic	micro	no	random	mixed	no	mixed	-	high

mobility models aim to incorporate environmental influence on the mobility model in a much more detailed way, such as by considering radio signal propagation and the environment's impact on radio signal. Table 2.1 summarizes the major mobility models available in the literature.

By exploring various state of the art mobility models in Section 2.1, it is crucial to understand that each mobility model is designed for specific scenario or environment. Therefore, there is no single model which is capable of representing all situations. However, mobility models belonging to the category of geographic restricted models provide some key attributes in representing the environment in more detail, making them more attractive over other models. But as they create very rigid node movement over the generated graph edges, they lack the ability to handle obstacles in a realistic manner. These issues need to be addressed to represent the environment more accurately. Since mobility models are designed to aid wireless network simulation, it is very important to devise them to be as much realistic as possible so that they can reflect the environment where the potential network is expected to be deployed.

2.2 Location Services

While the movement of nodes governed by the chosen mobility model has an enormous impact on the performance of the network protocol, estimating the exact location of the nodes through a location service protocol during movement as accurately as possible is also extremely important for successful and reliable delivery of data.

In a mobile ad hoc network where most of the nodes are mobile, the approximate location of the destination node is needed in geographic routing protocols, thus making the presence of a location service mandatory. It is utmost important to consider that a location service must not be a burden on the routing protocol itself. The simplest way to *query* the location of a destination is to broadcast the query throughout the network. When the queried node receives the message, it returns a uni-cast *reply* to the source node. The requirement of broadcasting makes this procedure less attractive as it lacks scalability. To overcome the overhead, each node selects a subset of nodes as its *location servers* either forming *quorums* in a specific direction (e.g., north-south, east-west, spiral) or by using uniform hash functions on the respective node's ID that maps it to a specific geographic region. From time to time each node updates its location servers

with its current position information. In a similar way, when a node needs to retrieve the location of a destination node, it sends a query towards the destination's location servers either in a quorum structure or by using the hash function. Upon receiving the query packet, the responsible location server sends back the location information. It should be noted that the location servers are selected among the usual nodes; there is no requirement of special hardware for a node to work as a location server. Different types of location server selection and maintenance schemes are available in the literature.

Based on different types of location information dissemination strategies and location server selection, location services can be categorized as *Flood-based*, *Quorum-based* and *Hash-based* location services. In Flood-based services, the location information of a node is flooded to the neighboring nodes or to the whole network. On the other hand, Quorum-based services maintain location information by forming quorums in selected directions, as mentioned before. The common characteristics of Hash-based location services is that they divide the whole network area into *grids* and assign *cells* as the home region for nodes in a *Flat*, *Two-level* or *Multi-level* structure. Figure 2.15 shows a detailed classification of location services.

Whatever the location service technique may be, there are some common tasks that every service must accomplish, which are described as follows:

- *Location update:* In this phase, each node sends a packet called the *location packet* or *update packet* with its current location information to its location servers which then store this information in a table or database.
- *Location maintenance:* Since most of the nodes in the network are mobile, they must organize and share location database with other nodes in such a way that any node can access the location information of others through the location servers, and this is ensured in the location maintenance phase.
- *Location query/request:* In order to determine the location of a destination node, a source node sends a packet containing the destination node's ID towards the destination node's location server. This packet is called a *query packet* or *location request packet*.
- *Location reply:* If a node upon receiving a query packet finds that it has destination node's location information, it sends a packet containing the destination

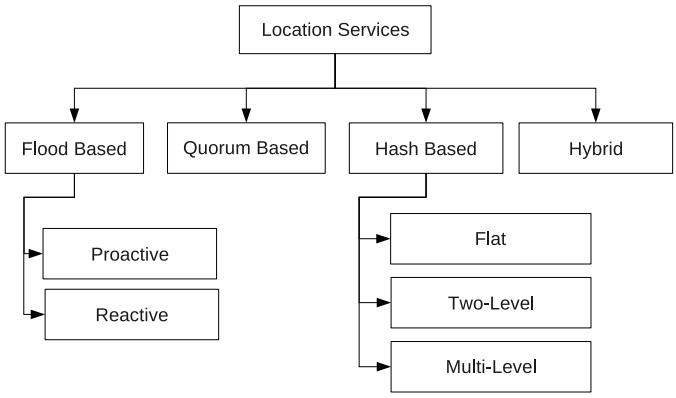


Figure 2.15: Classification of Location Services.

node's location to the query originator node. This packet is known as a *reply packet*.

The remainder of this section is organized as follows: Section 2.2.1 presents the flood-based location services. The Quorum-based services are described in Section 2.2.2. Section 2.2.3 and Section 2.2.4 explore a number of hash-based and hybrid approaches, respectively. Finally, we summarize this section.

2.2.1 Flood-based Location Services

Flood-based location services maintain the location information by flooding which is made restricted within the neighbors of a node or based on the distance from the node. These location services can be divided into two categories based on their location update frequency, namely, *Proactive* and *Reactive* services.

2.2.1.1 Proactive Services

Proactive location services maintain the location information through a periodical exchange of location database. Here, each node periodically broadcasts its location to notify other nodes. In this section some prominent proactive methods are discussed.

DREAM Location Service (DLS) Distance Routing Effect Algorithm for Mobility (DREAM) [12] is one of the prominent geographic routing protocols. DREAM does not assume that the knowledge of the destination is available without any

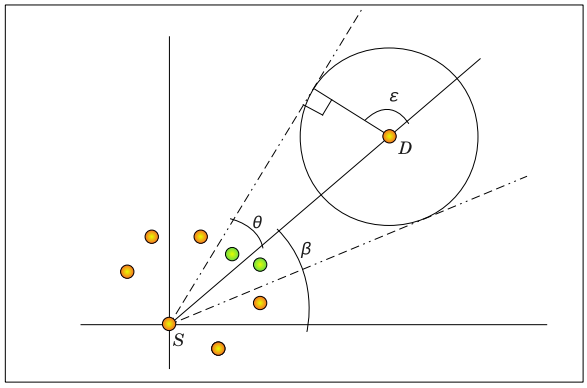


Figure 2.16: Data forwarding in DLS.

cost. Thus DREAM incurs higher routing overhead for incorporating a location service in the routing protocol. In DLS (the location service of DREAM) [12], each node periodically transmits a location packet that contains its coordinates. The source node's speed and transmitting time is also included in the location packet. The rate of transmitting a location packet is higher for nearby nodes and lower for more distant nodes, and adapts according to the distance traveled by the source node from its last update location. This is done by including a maximum distance $distance_{max}$ that is based on the geographic distance the packet traverses from sender. The majority of the packets have short life spans and are sent more frequently. When a node receives a location packet, the receiving node determines how far the packet has traveled by calculating the distance between the source node and itself. If the distance is greater than the $distance_{max}$ associated with the packet, the packet is no longer forwarded. By differentiating near and far nodes and updating far nodes less frequently, the overhead of the location update is kept limited. Location packets for distant nodes update all nodes in the network including the nearby nodes. In this way each node maintains a location table that contains the location and speed information at a specific time of a subset of nodes in the network.

When a node S needs to send a data packet to D, S looks into its location table for the location and speed information of D. Upon retrieving this information, S selects some nodes among its neighbors and forwards the data packet to them. Each of the nodes do the same until the packet is received by D. The subset of

neighbors within the wedge defined by $[\beta-\theta, \beta+\theta]$ where β denotes the direction towards D centered at S, are selected for forwarding the packet. The angle θ defines the area of the wedge which is chosen in such a way that the probability of finding D in the wedge sector is at least a predefined value p. As shown in Fig. 2.16, the displacement of D with speed ν within the time interval t_1 to t_2 is limited by the circle centered at the original position with radius $(t_2 - t_1)\nu$, with the assumption that D can move in any direction ε uniformly chosen between $[0, 2\pi]$.

Simple Location Service (SLS) SLS [79] follows a similar algorithm to DLS for disseminating location information. In SLS, each node also transmits location packets at a predefined rate, but unlike DLS, SLS transmits a table containing multiple nodes' locations to neighbors. Each location packet in SLS contains up to E_{SLS} entries from the node's location table, where the E_{SLS} entries are chosen from the table in a round robin fashion. Since multiple location packets are transmitted with a subset of the location table, it is most likely that all of the location information a node has will be shared with its neighbors. Upon receiving a location packet, a node updates its location table with the *latest* information and hence maintains the most recent location information. When a node sends a data packet, it follows similar strategy to that used in DLS.

2.2.1.2 Reactive Services

Proactive location services maintain location information by sending location packets periodically at a predefined rate. On the contrary, reactive location services update the location information on demand basis. This process reduces the bandwidth usage but increases the time a node needs to retrieve another node's location compared to that for the proactive services.

Reactive Location Service (RLS)[79] As mentioned above, in the Reactive Location Service (RLS)[79], the nodes do not disseminate their location periodically. Location information dissemination takes place only when a node looks for the location of another node. If node S needs to know the location of node D, at first S looks for it in its own location table. If the information is not present, S queries its one hop neighbors. If none of the neighbors respond with a location reply, S

floods a location request to the entire network. Upon receiving the request, any node that does not have the location information of D, propagates the flooding. However, if a node's location table contains D's location, it returns a location reply packet via the *reverse source route*[2] obtained in the location request packet. This reverse source route building mechanism increases the size of the location packet with each hop it traverses. Moreover, this methodology results in multiple reply packets increasing the query overhead.

Each node using RLS updates its location table upon overhearing any new location packet. However, this promiscuous mode operation is power consuming [66]. The entries in the location table includes *age* - a time duration used to identify a location entry's lifetime which is dependent on the speed of the node associated with it. An entry carries a lower lifetime if the associated node moves quickly, and vice-versa.

Reactive Location Service (RLS)[30] Another Reactive Location Service has been proposed in [30], which is referred to here as RLS[30]. Though the working principle of RLS[30] is very similar to RLS, there are quite a few differences in maintaining location information and forwarding location requests. In RLS[30], a location request is forwarded until it reaches the node whose location is queried. On the contrary in RLS, any intermediate node that possesses the requested information replies. RLS[30] does not incorporate any location caching mechanism due to the understanding that it may cause multiple replies resulting in a higher congestion in the communication channel.

Moreover, when replying, RLS[30] does not use reverse source route, rather it utilizes the underlying routing protocol. This reduces the overhead in RLS[30] because the route information is not recorded in a location request packet as it traverses. This functionality helps RLS[30] to use less bandwidth in the location finding process. Furthermore, RLS[30] introduces three types of flooding mechanisms: Linear flooding, Exponential flooding and Binary flooding.

Linear flooding: This variant floods a small neighborhood region first by limiting the packet TTL (Time to Live) value, J, to a small number of hops. If the

[2]The location request packet carries a list of nodes that it traversed on the way to reach a node having the location of the queried node. This list is used in reverse order to send the reply.

nodes within J hops do not have the required location information, the value of J is increased by J_{step} and the location request is restarted, and this process continues until the value of J reaches the maximum allowable value J_{max}. After failing to get a location reply within J_{max}, the destination is labeled unreachable. This flooding method may cause nodes to remain *hidden*[3]. In this scheme, if the queried node D moves away fast enough (at least J_{step} hops in the timeout period) from the querying node S, it can remain hidden resulting in query failure.

Exponential flooding: This variant's working principle is very similar to that of Linear flooding. However, instead of increasing the value of J by an additive constant J_{step} , it is multiplied by a factor and thus reduces the chance of any nodes staying hidden.

Binary flooding: This scheme actually divides the flooding scheme into *near* and *far* communication. Here the source node first floods a close range neighborhood to check if the request could be answered. If there is no reply, the traffic is classified as *far* and J is set to J_{max} rather than increasing it gradually.

In the flooding process, many nodes try to transmit packets simultaneously, resulting in higher rate of collisions. In RLS[30], the authors included a *radial* component in the flooding process that increases the expansion speed of flooding while still providing the congestion alleviation of a random back-off. Each node, upon receiving a request, computes its distance to the last hop node and calculates a back-off time with respect to the distance as follows:

$$t_{backoff} = delay_{max} \times \left(1 - \left(\frac{d_{last}}{r_t}\right)^2\right) \tag{2.4}$$

where $delay_{max}$ is the maximum delay a packet may be backed off, d_{last} is the distance to the last hop node and r_t is the radio range. This way it is assured that the farther away a node is, the sooner it will rebroadcast the query.

Proactive and reactive location services have high update and query overhead, respectively; thus they do not scale well in larger networks. On the other hand, the

[3]A node is considered to be hidden if while being a part of a network, its location can not be acquired by a requesting node.

failure of individual nodes does not affect the location service operation and performance, making these schemes robust.

2.2.2 Quorum-based Location Services

A different strategy for developing location services found in the literature is Quorum-based services. In a Quorum-based system, a node selects a set of nodes that produce a quorum for holding that node's location information. Different quorum building strategies are followed while maintaining and querying location information. In this section, the major location services belonging to this group are presented.

Column Row Location Service (XYLS) In Column Row Location Service (XYLS) [23], a node transmits its location to the north-south direction and the selected nodes become the quorum to hold the node's location information. Any node willing to track a node sends a query in the east-west direction, and whenever the query packet reaches the intersection at the column quorum, the location information of the destination is acquired. As shown in Fig. 2.17, node D is updating its location in the north-south direction and a quorum is created. Node S transmits a query in the east-west direction. The node residing at the intersection of the query and the update quorum is expected to know the location of D and replies to S. XYLS exhibits $O(\sqrt{N})$ overhead for the update, query and storage, where N is the number of nodes. In [23], the authors introduced three variants of the location update and query. The simplest is to update the location in the north-south direction while the query is forwarded in the east-west direction. To increase the probability of finding an intersection between the update and query quorums, the second variant proposes to forward the update in both the east-west and the north-south directions, while forwarding the query in the similar manner. This results in two potential intersections. The third variant creates more quorums by updating toward the east-west, north-south, northwest-southeast and northeast-southwest directions. Each of the later variants increases the probability of finding the queried location, however, with a considerable location update and query overhead.

There is a drawback in the XYLS procedure, called the *local maxima problem*. If while traveling, the location update packet reaches a node which is the northern-

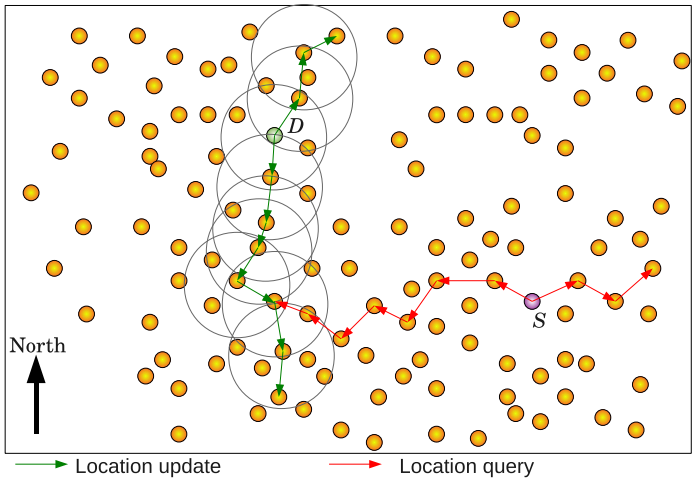


Figure 2.17: Location update and query in XYLS.

most in the column but not the northernmost in the simulation area, the built quorum is incomplete. As a result, there is always a chance that a query traveling in the east-west direction may not intersect the column quorum. Various healing up procedures have been proposed, among them there is a quorum development process in [80]. When an update packet reaches the northernmost node of the quorum, the update process transforms into a *face routing* [15] and then the packet travels to all the nodes that reside at the boundary of the whole network. In this modification, all boundary nodes belong to every quorum which increases query success, but makes the storage overhead among the nodes in the network unequal.

SEEKER Another quorum based location service found in the literature is SEEKER [24]. In this scheme, the whole geographic region is divided into grids of equal size. A quorum is developed in each row of grid squares. Through periodic beacons, every node determines whether it is the terminal node of the row of grids in which it resides. Whenever a node finds that there is no more node towards its east or west directions, it assigns itself as an *initial* node. Each of the two initial nodes in each row selects a virtual destination point (x, y), where y is the same as the initial node's y coordinate and the x coordinate is the extreme in the east or west direction farthest from itself. Then the initial nodes start updating procedure towards their virtual destinations. Each of the intermediate nodes the update

packet traverses includes its location information in the update packet. In this way when the packet reaches at the nearest node at the virtual destination point, the node receives all the nodes' location information, belonging to a specific row. Since this process is initiated from both extreme positions of a row, all nodes become aware of the location information of the nodes that reside in that specific row. This is called an *aggregate update* scheme. This location update procedure is shown in Fig. 2.18.

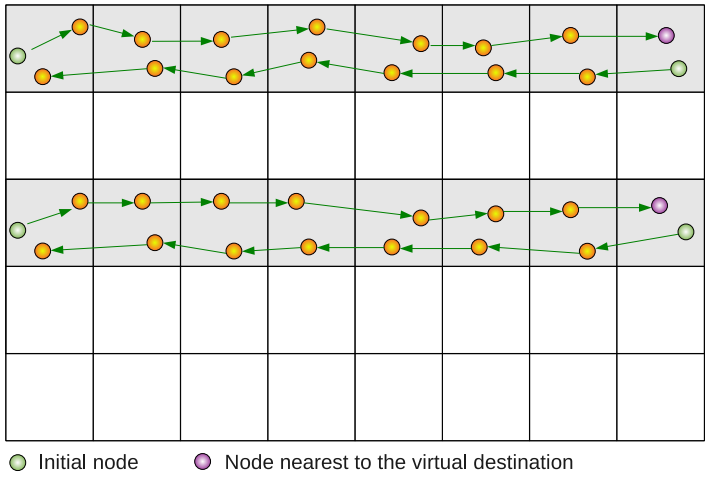


Figure 2.18: Location update procedure in SEEKER.

The query is done in a similar way to that for XYLS, but the direction is north-south rather that east west. When the query packet reaches any node belonging to the row of the queried node, a reply is initiated in a greedy mode to the query initiator. SEEKER has better performance than XYLS because it produces fewer update packets. However, since the size of the location update packet is increased in each hop, it lacks scalability. If the network size is large, the last hop of every update packet will contain an enormous amount of data, and a single packet may not be enough to transfer all the information, resulting in multiple update packets. Also, if the node mobility is considerably high, it may show relatively lower performance. If a node moves into a new row, there is no way for the initial nodes to be aware of the newly arrived node; so periodic update makes the update cost higher. Another major issue is that, if the update packet is dropped for any reason, the location service may perform poorly, since only the initial nodes initiate the update mechanism.

Quorum based services are mainly attractive in cases where all nodes move towards more or less the same direction keeping their relative position the same to avoid frequent location updates. But, as observed in [23], quorum based schemes may also suffer from the *distance effect*, since a source node may need to search and update the whole network even when the source and destination are relatively close. To fulfill guaranteed delivery, unsuccessful searches for the destination node need to be converted to network wise flooding. If this happens frequently, quorum based strategies become less attractive. High mobility of nodes also affects the performance of the quorum based location services.

2.2.3 Hash-based Location Services

In order to elevate the issues observed in quorum based systems, the hash function based location services were introduced. In this approach, uniform hash function is used to map a node's ID to a geographic position in the deployment area and every node in the network knows this hash function. Based on this position a subset of nodes is assigned as location servers. Since every node in the network is mobile, a node does not have any prior knowledge of the location of the other nodes, other than their IDs. Thus, through the hash function a node can identify any destination node's location servers and consequently retrieve the location of the destination node. This type of location services, based on location server maintenance, can further be classified as *flat*, *two-level* and *hierarchical* structure, and discussed in this section.

2.2.3.1 Flat-structured Location Services

The basic type of hash based location service is the flat structure. In flat structured services, a single hash function is applied to map each nodes ID to a geographic location. Some well known location services belonging to this group are the Scalable Location Update-based Routing Protocol and Geographic Hashing Location Service.

Scalable Location Update-Based Routing Protocol (SLURP) [19] One of the earliest location services found in the literature is SLURP. In this method, the network area is divided into a flat grid of squares. For each arbitrary node D, a uniform hash function is applied to its ID to obtain the (x, y) coordinates in the simulation area. The square H containing the hashed coordinate is defined as D's

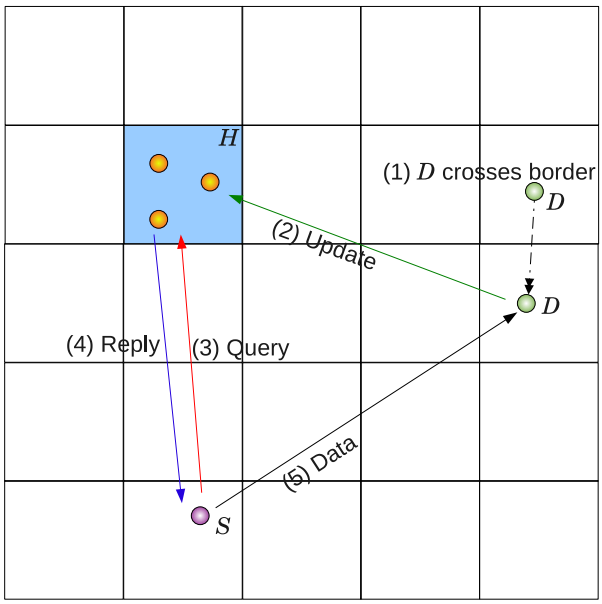


Figure 2.19: Location update and query procedure in SLURP.

home region (the shaded region shown in Fig. 2.19) and D's location information is stored in all the nodes in H. When D crosses the border of any square, it sends an update packet containing its new absolute location to its home region. Upon receiving the update packet, any node residing in H broadcasts D's updated location to all the nodes within H. Since this update process is made limited within the area of the smallest square, it does not incur much overhead. When any other node A enters into H, A collects the location information from the other nodes in H and becomes a location server of the nodes for those H is the home region. Whenever another node S wants to query D's location, it applies the same hash function with D's ID to obtain D's home region. S then sends a query packet to H to retrieve D's location. When any node in H receives the query, it sends a reply packet attaching D's location. After receiving D's location information, S can forward the data packets to D directly without any involvement of the home region. SLURP experiences $O(\nu\sqrt{N})$ and $O(\sqrt{N})$ for location update and query overhead, respectively, where ν is the node speed and N is number of nodes.

Since a uniform hash function is used to map a node's ID to its respective home region, each of the smallest squares is designated as the home region for some nodes. Thus the location information storage overhead is uniformly distributed

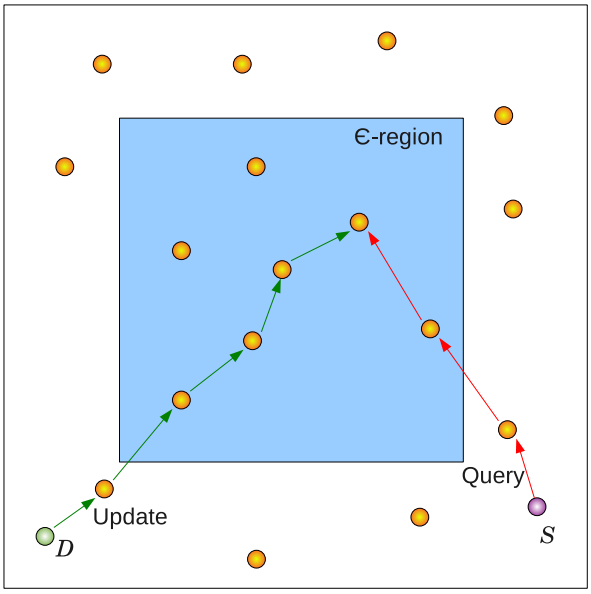


Figure 2.20: Location update and query procedure in GHLS.

among the nodes in the network. However, due to the home regions being uniformly distributed, the *distance effect* problem arises. In this case, if nodes S and D are geographically close to each other but D's home region is far away, S has to find D's location by making a query that has to travel a long distance, thus increasing query cost.

Geographic Hashing Location Service (GHLS) Geographic Hashing Location Service (GHLS) [81] is another hash-based location service similar to SLURP. Unlike SLURP this scheme does not generate any grid or rely on the home region, rather selects a node to store its location. The server node is selected using a hash function on a node's ID. The hash function chooses a point (x, y) in the simulation area and the node which is the nearest to that point becomes the location server. Since each beacon from a node contains its location, the neighboring nodes can easily find out who is the nearest to the hashed point. Each time a node moves further than a threshold distance, it uses the hash function with its ID to find the server point, and the nearest node to that point is updated with its location information. A *handoff* mechanism is initiated when the server node finds another node nearer to the hashed point, and transfers all the location information to it.

In order to reduce the distance effect, GHLS uses a hash function that generates locations within a small sub-region near the center of the entire region. The ratio of the length of the scaled location region to that of the whole region is termed the *scaling factor* ϵ, and this region is called the ϵ-*region* (Fig. 2.20). In this way only a subset of nodes residing in the ϵ-region are assigned to be location servers, which leads to load imbalance among the server nodes in the network. Using a single node as the location server also makes this method vulnerable to a single point of failure in case of high node mobility. However, due to using a single node as the server, GHLS reduces the control packet overhead as no broadcasting is required.

2.2.3.2 Two-level Location Services

To elevate the distance effect problem in the flat structured location services described above, two-level structure was introduced in Scalable Ad-hoc Location Management (SLALoM) [20] and Adaptive Demand-driven Location Service (ADLS) [82]. In this section, we provide an overview of these two location services.

Scalable Ad-hoc Location Management (SLALoM) SLALoM combines the concept of two-level hierarchical grid and SLURP to overcome the drawbacks of SLURP. In this method, the whole region is divided into equal sized grids like SLURP, which are called order-1 squares. Multiple order-1 squares ($m \times n$) are combined to form an order-2 square. A node uses a hash function on its ID to select an order-1 square in every order-2 square as its home region. SLALoM incorporates a two-level hierarchy by introducing *near* and *far* home regions. The home region in the order-2 square containing the node and the home regions within the surrounding eight order-2 regions are called the near home region. The nodes in these regions store the actual location of that node. The other home regions are called far regions and nodes in the far home regions know in which order-2 region the node resides in. When a node crosses the boundary of an order-1 region, it only updates its near home regions, while if it crosses an order-2 region, it has to update all the home regions as well.

As shown in Fig. 2.21, one order-1 square in each order-2 square is assigned as the home region of node D based on a hash function. The home regions in the

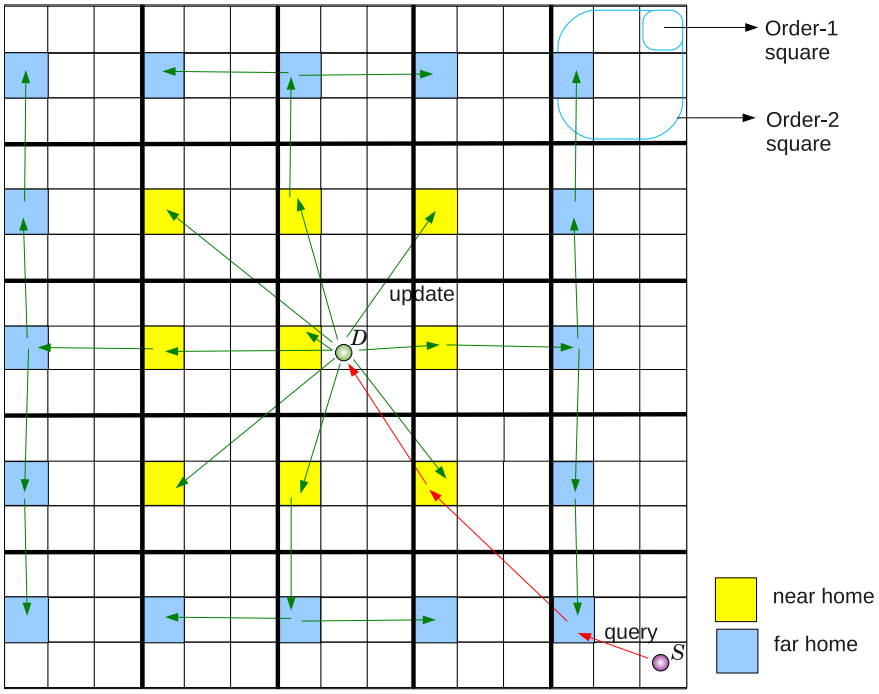


Figure 2.21: Location query procedure in SLALoM.

order-2 square containing D and the surrounding 8 order-2 squares are marked as the near home regions (marked in yellow). The rest of the home regions (marked in blue) are D's far home regions. When D crosses an order-1 square within the same order-2 square, it only updates its near homes with its absolute location. But if it crosses the boundary of an order-2 square, it needs to update the far home regions as well. The far home regions are only updated with the information that contains the order-2 cell's ID which D resides in. For this reason, far home regions do not need to be updated as long as D stays within an order-2 square. In Fig. 2.21, each order-2 square consists of 3×3 ($m = 3, n = 3$) order-1 squares. When node S looks for D's location, it uses the hash function with D's ID within its own order-2 square, and sends the query to the home region of D. Since the home region is a far home of D, the query is forwarded to the nearest *near* home and thus the location of D is acquired. This method reduces the query cost which is higher in SLURP but at the expense of a very high location update cost because of maintaining one home region in every order-2 region.

Adaptive Demand-driven Location Service (ADLS) ADLS tries to combine the

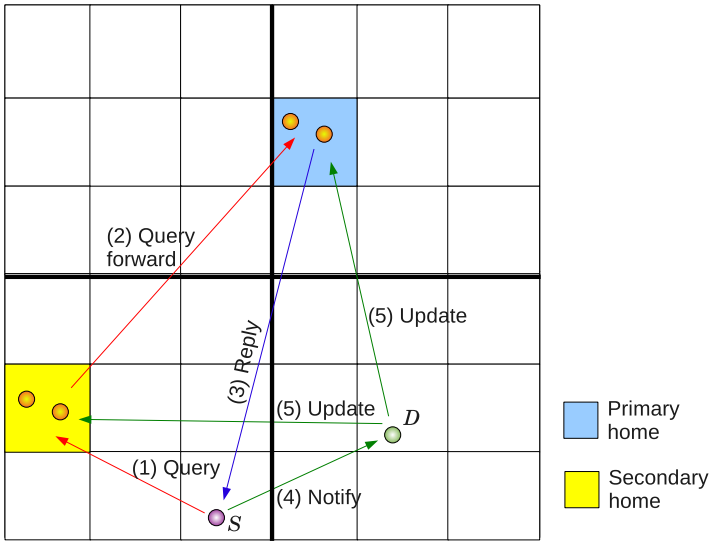


Figure 2.22: Location update and query procedure in ADLS.

advantages of SLURP and SLALoM. Similar to SLURP, each node maintains a home region called the *primary* home region where every node in the network uses the same hash function to query for a destination node. Similar to SLALoM, ADLS also uses a two-level grid-hierarchy, and each order-2 square has an order-1 square that is selected by the hash function. Unlike SLALoM, ADLS does not need to setup at least one home region in every order-2 square, so ADLS has a lower update cost than that of SLALoM specially in the presence of localized traffic. Moreover, ADLS avoids using the order-1 squares at the border of the area in assigning the home regions. The query node will setup a *secondary* home region in its current order-2 square on demand. This reduces the update cost to some extent, but if the query is generated from all around the network, it will eventually be transformed into SLALoM and the update cost will be increased as in SLALoM. Moreover, since the secondary home region is created on demand basis, the query delay is higher compared to that for SLALoM.

As illustrated in Fig. 2.22, S wishes to send packets to D but S does not have the location information of D. So, S sends a query packet to the potential secondary home region of D within S's own order-2 square, using the predefined hash function. If any node residing in the mapped region knows D's location, it simply replies to S. Otherwise, the query is forwarded to D's primary home

region. When S receives the reply, it starts transmitting data to D and at the same time notifies D that D does not have a secondary home region in S's order-2 square. D then proceeds to create a secondary home region by sending an update packet. Any subsequent location queries by nodes residing in S's order-2 square will now be replied through the secondary home region without the necessity of forwarding the query to the primary home region.

2.2.3.3 Multi-level Location Services

Multi-level location services organize home regions in a higher level hierarchy in order to balance the location update and query overhead. A few well-known location services are GLS, DLM, HIGH-GRADE, HLS, GRSS, and HGRID which are described in the following:

Grid Location Service (GLS) The first location service protocol developed to use a hierarchical structure is Grid Location Service (GLS) [83]. GLS divides the network area into a square grid where the smallest squares are termed the order-1 square. Four order-1 squares make up an order-2 square and four order-2 squares make up an order-3 square, and this process continues until the whole area is covered. Furthermore, the squares of the same order do not overlap and every node is aware of this hierarchy. In this way a node's location can be identified by exactly one square at each level. Figure 2.23 shows how the lower order squares are combined into upper order squares.

The way the location servers are selected in GLS is different from the other services. As shown in Fig. 2.23, the order-1 square containing node D has three *sibling* order-1 squares, and these four order-1 squares compose an order-2 square. For location servers, D recruits *one* node belonging to each order-1 square (except the order-1 square containing D) that make up an order-2 square. The node having the lowest ID among the nodes in a square but still higher than that of D is recruited as the server node. Similarly, one location server is selected in each of the order-2 squares (except the order-2 square containing D) that constitute an order-3 square. Therefore, the density of location servers for a node is high in areas close to the node and becomes exponentially less dense as the distance from the node increases.

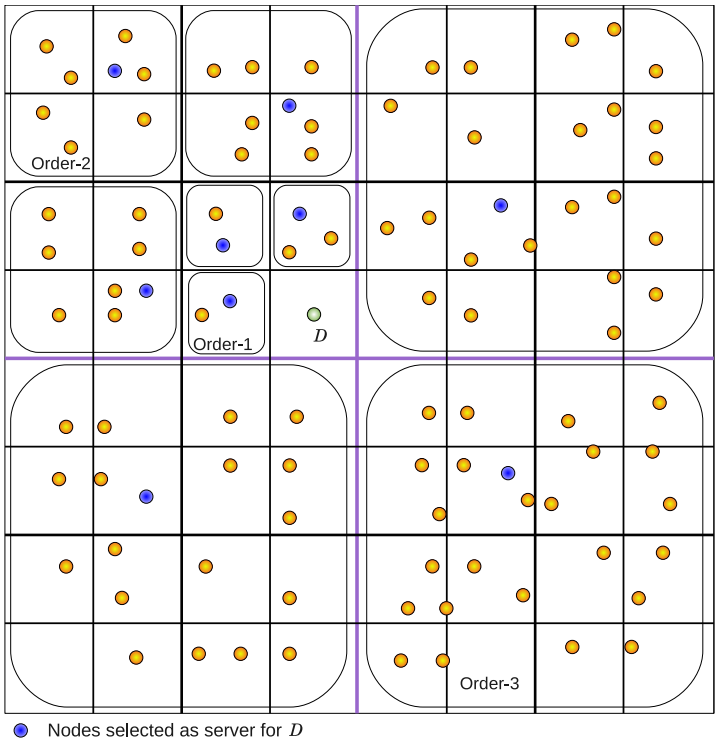


Figure 2.23: Hierarchy building and location server selection in GLS.

When a node moves, it needs to update its location server with its new location. A node updates its order-2 location servers every time it moves away a threshold distance d_{GLS}. Similarly it updates its order-i servers when it moves away $2^{i-1} \times d_{GLS}$ distance. So, the generation of update packets is proportional to the movement speed of nodes. As a result the updating of higher order servers takes place less frequently than that of lower order servers.

To find the position of a node D, any arbitrary node S sends a request to the node with the smallest ID but greater than that of D, for whom S is working as a server. The query is forwarded to nodes with decreasing IDs, each step for each level in the hierarchy, until it reaches a location server of D. From this location server, the query is forwarded to D and upon receiving the query, D generates a reply to S.

The update and query mechanisms of GLS require the packet to traverse a series of nodes based on the nodes' ID. It may work fine in a stationary scenario but as investigated in [84], traversing a chain of mobile nodes may increase the update

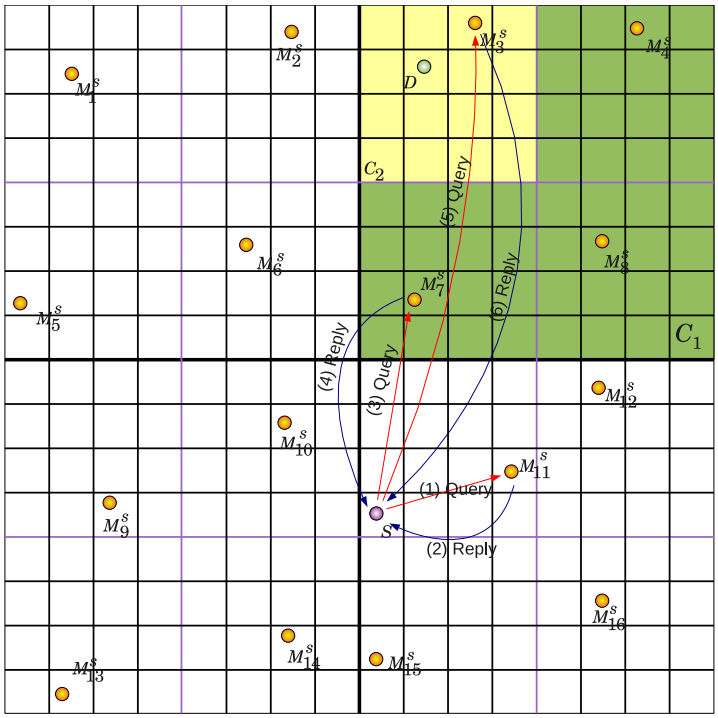


Figure 2.24: Location query procedure in DLM.

and query failure significantly if the node mobility is high; since, if one of the nodes in the chain cannot be reached, the update or query process fails. GLS requires $O(\nu\sqrt{N})$, $O(\sqrt{N})$ and $O(\log N)$ overhead for the location update, query and storage, respectively.

Distributed Location Management (DLM) In DLM [85], the deployed region is termed the 0-order region which is partitioned into $a_{DLM} \times b_{DLM}$ square regions. Each of these squares is called 1-order region. In a similar way, each of these 1-order squares are partitioned into a number of 2-order squares, and this process continues to the minimum size squares that can be fully covered by a node's transmission range. As shown in Fig. 2.24, the 0-order (whole area) region is divided into 2×2 1-order squares. Similarly, each of the 1-order squares is divided into 2×2 2-order squares and each of the 2-order squares is further divided into 4×4 3-order squares. In this example, a 2-order square is selected as granularity (denoted by m-order) and one node residing in each of the 16 2-order squares is assigned as the location server for a particular node.

DLM facilitates two different types of addressing schemes. In the *full address*

scheme each of the location servers stores the absolute location of each node for which it has been assigned as the server. Whenever a node crosses the boundary of the smallest square, it updates all of its servers. This scheme facilitates a better query response since any node can retrieve the location information from a server located near to it. However, the update overhead becomes higher because a node needs to update all the servers with its absolute location. To reduce the update overhead, the *partial address scheme* has been proposed where the location information of a node appears more detailed to a nearer server. In this scheme if the smallest common square within which an arbitrary node A and any of its location servers M_i^s reside in is an m-order square, the full address (absolute location) of A is stored in M_i^s. Otherwise, if the smallest common region of A and a location server M_i^s is an n-order ($n < m$) region, then M_i^s stores A's $(n+1)$-order square's address.

In Fig. 2.24 the query procedure in partial address scheme is explained. Here, $m = 2$ and D's location servers are $M_1^s \cdots M_{16}^s$ where, D's 1-order square address is stored in M_1^s, M_2^s, M_5^s, M_6^s, M_9^s,$\cdots$, M_{16}^s. Again, D's 2-order square address is stored in M_4^s, M_7^s, M_8^s. Only M_3^s stores D's absolute location. When any node S looks for D's location, it first queries M_{11}^s as it resides within S's 2-order square. M_{11}^s replies to S with D's 1-order square ID C_1 (marked in green). Upon receiving the reply, S again initiates a query and continues to locate a location server within C_1. As shown in the figure, server M_7^s is found which subsequently returns a reply with an 2-order square ID C_2 (marked in yellow). After receiving the reply S again sends the query to the location server M_3^s within C_2. Upon receiving the query, M_3^s replies to S with D's absolute location.

This partial address scheme reduces the update overhead at the cost of an increased query overhead, as for each order square, S needs to send a query. Similarly, for each order square a reply with a more specific location of D is sent to S. The even distribution of the location servers limits DLM's scalability, since whenever a node crosses the borders of an m-order region, all location servers residing in that particular region need to be updated. Since the location servers are evenly distributed in the network, the resulting location maintenance overhead becomes similar to that of network-wide flooding [22]. Moreover, relying on

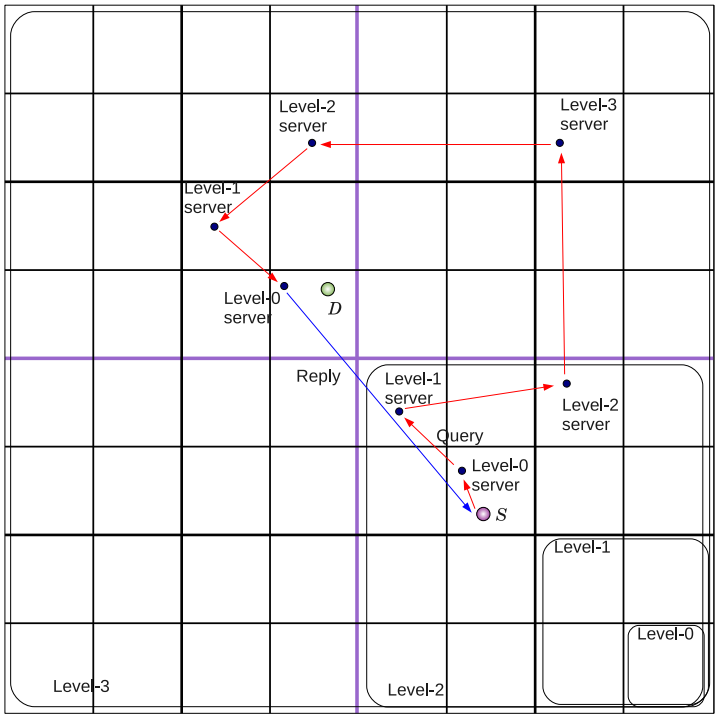


Figure 2.25: Location query procedure in HIGH-GRADE.

a single node in a region as the location server makes DLM vulnerable to location update and query failure.

HIGH-GRADE Another hash-based location service is HIerarchical Geographical Hashing with multi-GRained Address DEelegation (HIGH-GRADE) [21]. In HIGH-GRADE, the entire area is called the level-U square, where U is a system parameter representing the number of levels in the hierarchy. The level-U square is divided into four quadrants, called the level-$(U-1)$ squares, each of which is further divided into four quadrants until the entire region is divided into 4^U level-0 squares. For each level, a different hash function is used to map a node's ID to a specific geographic point in a specific level. The nodes that create a *perimeter* around that point are assigned as the location servers. Relative location information is stored in the location server nodes at different levels, for instance, an arbitrary node D's level-i servers store information in which level $(i-1)$ square D resides in. Only the level-0 servers store D's exact location. When a node crosses a level-i region, it updates up to level $(i+1)$ home regions, thus reducing the update cost. When a node S needs to find D's location, it uses

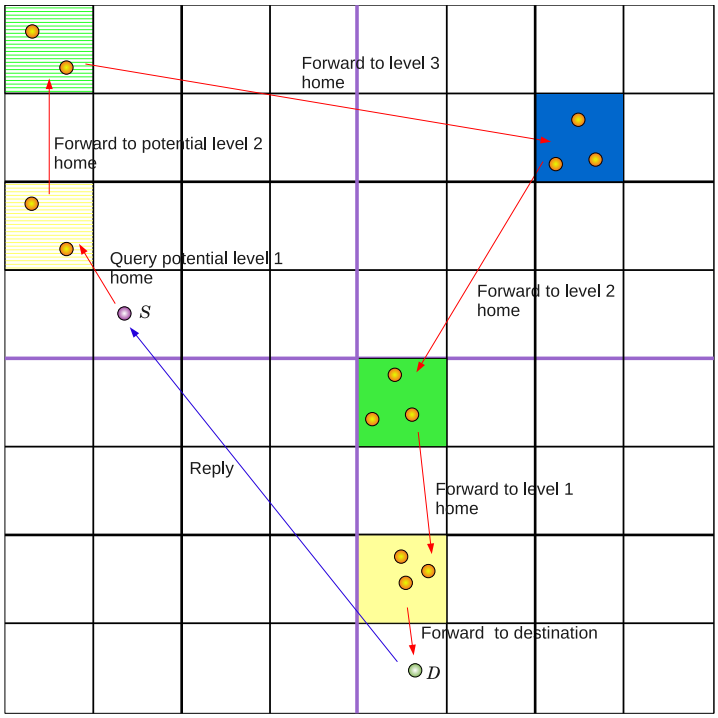


Figure 2.26: Location query procedure in HLS.

hash function on D's ID to find its home region and if D's location is found, a reply is sent. Otherwise, the query is forwarded to the next $(i+1)$-th level home region and this process is continued until the j-th $(j \leq U)$ level is reached, which is the common region where both S and D reside in (Fig. 2.25). The update cost is reduced at the price of extra query cost since there may be a number of unsuccessful queries until level j is reached, but performs better for localized traffic.

Hierarchical Location Service (HLS) Similar to HIGH-GRADE, HLS [22] is another hash-based hierarchical location service. It also divides the whole deployment area into grids and combines a number of grids into higher level grids forming a hierarchy. The location update and query process in HLS is very similar to those of HIGH-GRADE. However, unlike HIGH-GRADE, HLS does not select the location server nodes in a perimeter around the mapped point based on the hash, rather it selects the level-0 cell as the home region, and all nodes residing in the home region become the location servers for a particular node (Fig. 2.26). Moreover, if there is no node present in a specific home region re-

sulting in the empty home problem, HLS handles by building a perimeter around the home region, and the nodes creating the perimeter act as temporary location servers. These temporary location servers always look for any nodes coming into the empty home region and forward the temporary responsibility to them. Thus, HLS is more fault tolerant and balances the location update and query costs. HLS has $O(\sqrt{N})$, $O(\sqrt{N})$ and $O(\log N)$ overhead for the location update, query and storage, respectively.

Geographical Region Summary Service (GRSS) Hsiao developed another hierarchical location service called Geographical Region Summary Service (GRSS) [86]. It constructs a hierarchical structure like HIGH-GRADE or HLS. The nodes in each order-0 square use a local routing protocol in such a way that a node knows the location of every other node in the same order-0 square. This scheme uses a summary development mechanism by introducing a special purpose for the nodes who reside at the boundary of each square of every order. These nodes are called the *boundary nodes* and are responsible for developing a summary of the nodes in their levels. For instance, every boundary node in an order-0 square generates a summary of the nodes in the square and sends it as a summary update message to other sibling order-0 squares, where sibling squares are defined as those squares that make an upper order square. Some of the boundary nodes in the order-1 square then generate summaries of the order-1 square and send it to their sibling order-1 squares. This procedure continues recursively until the whole simulation area is covered. For query purpose, a node searches its database whether the destination node's location is available. If it finds the location, directly sends data to the destination node. Otherwise, it forwards the query to the center of the smallest order square whose summary it has. However, for the node mobility the overhead in maintaining summary of boundary nodes becomes higher and the number of hops for a query may increase since it does will follow the optimum path [86].

Hierarchical GRID Location Management (HGRID) HGRID [87] is another multi-level structured location service designed to improve on the performance of SLALoM. It creates a hierarchy similar to HLS and HIGH-GRADE but the hierarchy building mechanism is slightly different. Instead of using the hash functions, it assigns

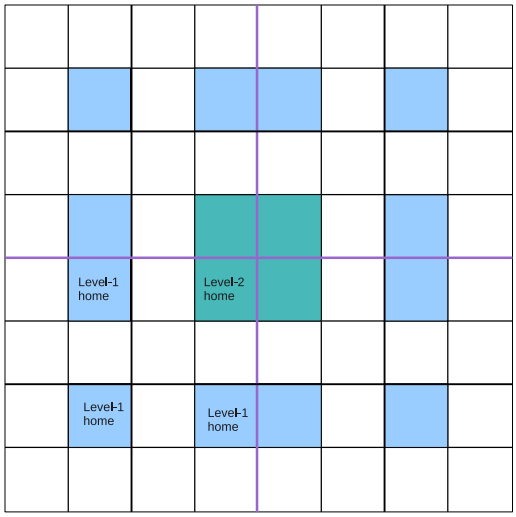


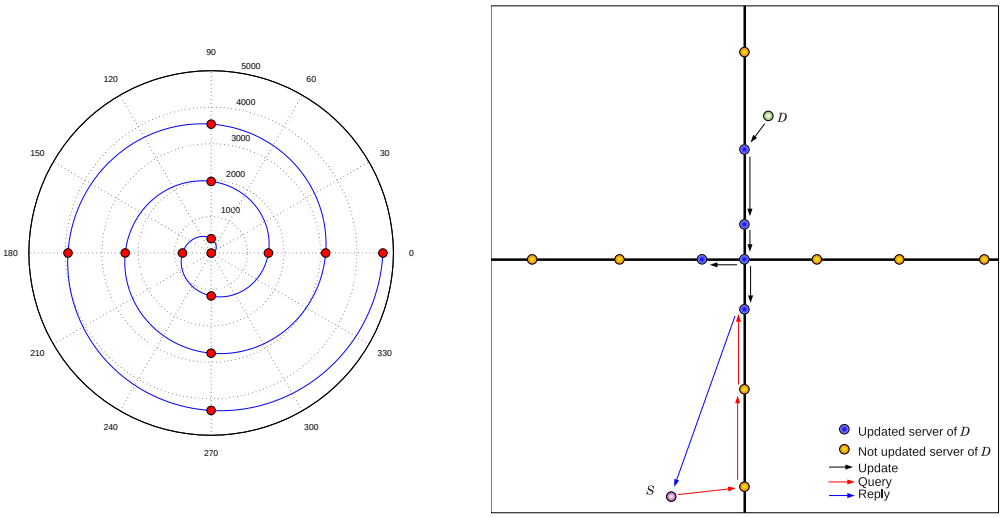
Figure 2.27: Home region assignment procedure in HGRID.

the home region in each level of the hierarchy in such a way that the higher level home regions are kept in the center of the simulation area as shown in Fig. 2.27. In this way the distance traveled for location updates and query packets becomes less compared to HIGH-GRADE and HLS. However, the empty home region problem has not been addressed in this scheme.

2.2.4 Hybrid Location Services

While the location services described in the previous section are based on hierarchical structure, there are also a number of location services that are hybrid in nature. These services are mainly developed in order to utilize the advantages of building a hierarchy, using hash functions, and/or using quorums. In this section, a number of these location services are discussed.

Adaptive Location Management (ALM) A different quorum generation technique is proposed in ALM [25]. Here, a node publishes its location information to a set of virtual points that form a virtual spiral increasing exponentially with distance. The center of the spiral is calculated using a hash function with the ID of a node. When another node initiates a lookup operation, it also uses the same hash function on the ID of the queried node. As the spiral generation function is known to all, the location of the servers can be also calculated. As shown in



(a) (b)

Figure 2.28: ALM's (a) distribution of servers based on spiral generation function $R_{ALM} = k\theta$, where, $k = 250$ and $0 \leq \theta \leq 6\pi$, and (b) location update and query mechanism.

Fig. 2.28(a) an arbitrary node D calculates a spiral with $R_{ALM} = k\theta$, where k is spiral tuner, $\theta+ = \pi/2$, $0 \leq \theta \leq 6\pi$, and the origin is selected based on the hash function. At $\pi/2$ intervals on the spiral, a node is selected as a location server of D.

ALM follows a special fuzzy algorithm to find out how frequently the location update has to be initiated and how many servers need to be updated based on the node mobility and distance from the origin. The location update and query procedure is explained in Fig. 2.28(b). In this example, node D initiates a location update mechanism from its closest server. The message is then forwarded to the next closest server. Here, using the fuzzy algorithm, D decides to update 5 servers among all of its servers. When another node S looks for D's location it uses the same hash function and spiral building function and becomes aware of the position of D's location servers. S sends a query towards the server of D that is the nearest to S. As the server node in the figure does not have D's updated location, it continues to forward the query towards the next servers until the location of D is retrieved. The server having D's location sends a reply to S with D's location. Similar to the update scheme, during the lookup phase, the fuzzy algorithm is used to determine how many servers need to be visited in the query

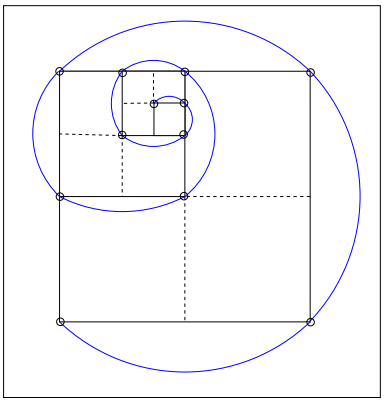


Figure 2.29: Location update and query procedure in LLS.

process and how frequently the query needs to be retransmitted upon failure. For an undisrupted service, ALM needs the nodes to be uniformly distributed throughout the area [25] as an uneven node distribution causes a higher failure for both update and query process due to its home assignment mechanism.

Locality Aware Location Service (LLS) Similar to ALM, the Locality Aware Location Service (LLS) [26] also exploits the concept of spiral shaped location update and query. However, the spiral building procedure in LLS is different as it divides the network area into a hierarchical structure while propagates the update and query packets in spiral form.

Initially LLS divides the network area into grids and defines a virtual hierarchy, whose origin is dependent on the node IDs. For a given node D, a double index hash function is used to map D's ID to a specific coordinate at each level of the hierarchy. The four corners of the grid at different levels are used for potential location servers. As shown in Fig. 2.29, a spiral is generated that traverses through these points. Like Geographic Hash Table (GHT) [88], a virtual coordinate system is used here. At the mapped point, let M_p, there may be no node to be used as a location server. So the node nearest to M_p actually becomes responsible for this task. When it moves away from M_p, the node transfers its responsibility to another node which is currently nearest to M_p. In this way LLS maintains the location information through the nodes present in a spiral structure based on the corner points of squares of each level in hierarchy, as shown in Fig. 2.29.

Moreover, to address the worst case scenario, a modified *spiral flood* scheme is

introduced. Here, at the lowest level of hierarchy, in addition to the four corner points, the surrounding eight grids' corner points are also used for the location update. When any node S needs the location of D, it uses the same hash function used by D for update process, and the query is forwarded in a spiral structure similar to the location update mechanism.

MLS MLS [89] is one of the recently developed location services, which is a hybrid of HIGH-GRADE and LLS. As in HIGH-GRADE, the whole area is divided into grids and a hierarchy of grids is created. Each node uses a hash table similar to GHT to map its ID to a geographic location at each level of the hierarchy. Unlike HIGH-GRADE, only a single node near to the mapped point is assigned as the location server. Instead of storing the exact position, a location server stores in which of the four sub squares a node resides in.

The query process is initiated similar to LLS. When a node S looks for node D's location, S first searches D's location in the immediate neighborhood of S and then incrementally increases the search area until a location server is found. After that the query is forwarded to the lower level in hierarchy until D is found. Though the authors have provided some important metrics such as the upper bound of the node speed for a successful operation of the scheme, a very dense node distribution and reliable communication is necessary for MLS, which may be unrealistic in real-world scenario.

2.2.5 Summary

Due to being scalable, geographic routing protocols are very much attractive for ad hoc network research. However, they need an appropriate location service protocol. Without a suitable location service, a node can not find any other node's location. In Section 2.2 various location service schemes are explored along with their special features, advantages and disadvantages. Table 2.2 summarizes some of the important aspects of these services.

The location services available in the literature have their own advantages and limitations; some require higher update overhead but are robust in nature, others try to reduce the update overhead by using on-demand location requests at the expense of increased query response delay. A few location services try to balance the location

update and query overhead in different manner. However, a common characteristic of most of the location services is that the nodes need to be distributed more or less uniformly throughout the deployment area.

The selection of the location service for a geographic routing protocol should be application oriented. For instance, a bandwidth critical application should not use any flood based location service, rather the location services that utilize less bandwidth (e.g., hierarchical services). Thus depending on the nature and requirements of the application, the most suitable location service should be selected, and only then a geographic routing protocol can be benefited the most from a location service.

2.3 Conclusion

This chapter presents the fundamental research strategies and concurrent works in relation to mobility models and location services from a real-world perspective. The research objectives for this thesis presented in Section 1.2 are summarized based on the limitations of these works. The contributions outlined in Section 1.3 made in this thesis by developing new strategies that address these issues are presented in the subsequent chapters.

From the above study it is clear that most of the location services are designed based on a simple and unrealistic assumption, i.e., all nodes are evenly distributed throughout the deployment area all the time. But environmental context causes node distribution to be non-uniformly distributed if a realistic mobility model in real-world environment is considered. This underpins the necessity to evaluate the existing location services considering real-world environmental components (e.g., obstacles, buildings, lake etc.), and to examine their applicability in such scenario, which is addressed in the next chapter.

Table 2.2: Comparison of different location services.

Service	Type	Update	Query	Storage	Update type	Query type	Area division
DLS	proactive	$O(N)$	$O(\phi)$	$O(N)$	Flood	-	No
SLS	proactive	$O(N)$	$O(\phi)$	$O(N)$	Flood	-	No
RLS[79]	reactive	0	$O(N)$	0	-	Flood	No
RLS[30]	reactive	0	$O(N)$	0	-	Flood	No
XYLS	quorum	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Unicast	Unicast	No
SEEKER	quorum	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Unicast	Unicast	Yes
ALM	quorum	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Unicast	Unicast	No
SLURP	flat-hash	$O(\nu\sqrt{N})$	$O(\sqrt{N})$	$O(\phi)$	Geocast	Unicast	Yes
GHLS	flat-hash	$O(\nu\sqrt{N})$	$O(\sqrt{N})$	$O(\phi)$	Unicast	Unicast	No
SLALoM	2-level-hash	$O(\nu\sqrt[3]{N})$	$O(\sqrt[3]{N})$	$O(\sqrt[3]{N})$	Multicast	Treewalk	Yes
ADLS	2-level-hash	$O(\nu\sqrt[3]{N})$	$O(\sqrt[3]{N})$	$O(\sqrt[3]{N})$	Multicast	Treewalk	Yes
DLM	2-level-hash	$O(\nu\sqrt[3]{N})$	$O(\sqrt[3]{N})$	$O(\sqrt[3]{N^2})$	Multicast	Treewalk	Yes
HIGH-GRADE	hierarchy-hash	$O(\nu \log N)$	$O(\sqrt{N})$	$O(\log N)$	Unicast	Treewalk	Yes
HLS	hierarchy-hash	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\log N)$	Geocast	Treewalk	Yes
GRSS	hierarchy-hash	$O(N)$	$O(\phi)$	$O(N)$	Flood	Unicast	Yes
HGRID	hierarchy	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\log N)$	Geocast	Treewalk	Yes
GLS	hierarchy-hybrid	$O(\nu\sqrt{N})$	$O(\sqrt{N})$	$O(\log N)$	Treewalk	Treewalk	Yes
LLS	hybrid	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$	Unicast	Unicast	Yes
MLS	hierarchy-hybrid	$O(d \log d)$ time	$O(d)$ time	$O(\log N)$	Unicast	Unicast	Yes

N: number of nodes; ϕ: constant; ν: node speed; d: distance

Chapter 3

Impact of Environmental Context on Location Service Protocol

As depicted in Chapter 1, mobile ad hoc network is an attractive solution in various real-world environments for a wide range of applications including search and rescue operations in disaster areas, war front communication, inter-vehicular communications, and so on. Considering the potential deployment environment, it is found that real-world environment is rather complex due to having various environmental contexts such as buildings, vegetation, hills, lakes and pathways that impact human mobility. Since mobile devices are mostly carried by humans or vehicles, defining their mobility considering environmental context is important. Furthermore, analyses of the existing location services in Chapter 2 identifies that most of the location services were designed assuming only ideal situation of node distribution. Only mobility models, such as Random Waypoint and Random Direction that tend to distribute nodes more or less uniformly throughout the deployment playground for the whole network lifetime, were used. The basic design philosophy of these location services is based on the fact that the nodes have to be uniformly distributed. Thus it can be concluded that most of the location service schemes in the literature focused only on making the services scalable on location maintenance and query, but little importance was given to their suitability to be deployed in practical environments. A real-world scenario is likely to have various obstacles that may play a vital role in the performance of geographic routing protocols as well as location services since those areas occupied by the obstacles may have no or very low node density available. No study has yet been done to evaluate the influence of obstacles on location services' performance [32]. Thus the applicability of these

location services to a realistic environment is questionable, highlighting the necessity that the location services need to be evaluated in a realistic environment. Such an evaluation will lead us to important research questions that need to be addressed to make location services suitable for a real-world environment. With this aim, in this chapter, a representative example of location service is investigated and analyzed through simulation considering one of the prominent environmental contexts - obstacles (e.g., buildings). The findings of this chapter underline the necessity for further research by raising valuable research challenges, and establish the foundation of the subsequent research carried out in the following chapters.

3.1 Real World Environment Model

In the real-world, the presence of obstacles with different shapes and sizes is very common. It is also an important issue to consider the position and placement of obstacles while designing a real-world model. From the previous research [29], it is well accepted that different scenarios have influence on the network performance metrics in different ways. However, for our initial study a simple scenario is considered in this chapter, where various regular shaped obstacles are placed as illustrated in Fig. 3.1 to evaluate the impact of the presence of obstacle on location service protocols. Incorporating irregular shaped obstacles for representing objects like lakes, hills, can obviously make this model even more realistic.

In ad hoc and sensor networks, the device power is an important issue to consider. Power is mostly consumed for transmitting data where the consumption is dependent on the transmission range. Most of the location services discussed in Section 2.2 were evaluated on the basis of a single transmission range, e.g., 250m, ensuring a moderate to high level of node density. However, while evaluating network performance through simulation, the parameters must be tuned based on the application of the system. It is trivial that a higher transmission range causes the mobile devices to consume more power and at the same time causes higher congestion in the MAC layer. On the other hand, a lower transmission range results in increased hop count, and consequently a higher delay in packet delivery. This is why the impact of the transmission range should also be taken into account so that a better insight from various view points can be obtained, which motivated us to perform the simulation with different transmission

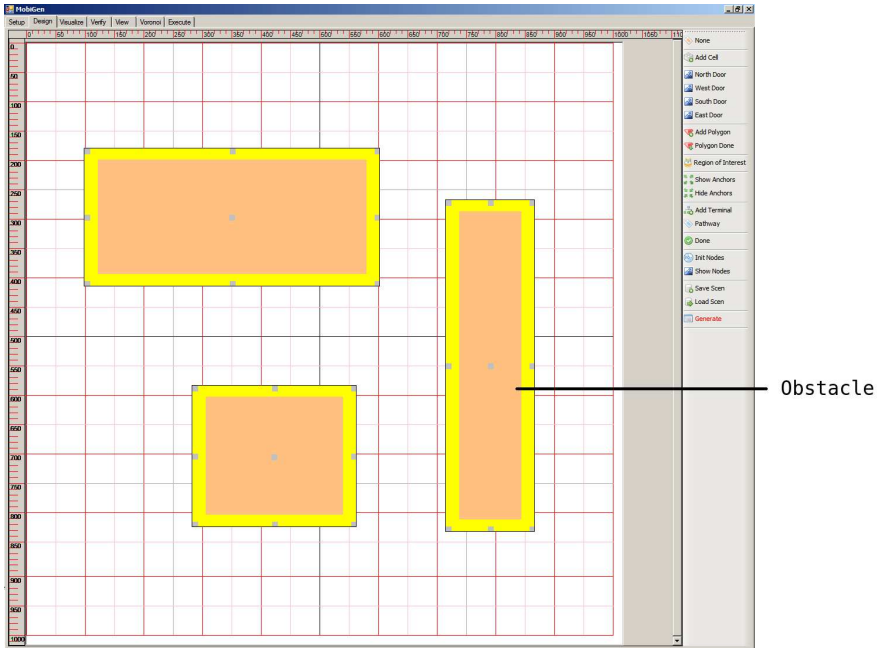


Figure 3.1: A simple real-world environment model with obstacles.

ranges.

In this model the obstacles are assumed to be inaccessible, and nodes can not occupy the area taken by the obstacles. This eventually results in the nodes being distributed non-uniformly and empty home problem arises. Though SLURP [19] and HLS [22] adopt a similar types of mechanisms to recover from this situation, they are only applicable for a short time. In cases where the empty home problem sustains for a long time, the effectiveness of these solutions is yet to be analyzed. Among the location service schemes mentioned in Section 2.2.3, only SLURP can be adopted in the real-world model for considering obstacles with slight modifications. The other schemes need to be modified significantly to cope with the obstacles. But as SLURP is the basic form of most of the location service schemes, experimenting with SLURP will provide an indication of how the other schemes would perform in the presence of obstacles.

3.1.1 SLURP with Obstacles

The evaluation phase considers two scenarios, one without any obstacle and the other with obstacles as shown in Fig. 3.1. It is expected that comparing these two scenarios

will demonstrate the impact of the obstacles. In the free space scenario, we used a simple modulo based hash function for the home region assignment. A slight modification in the hash function is also made so that obstacles can be considered, which is outlined in Algorithm 3.1.

Algorithm 3.1 assignHome

Require: u : node,
$mode$: with/without obstacle,
$Rows$: number of rows,
$Cols$: number of columns.

1: **if** $mode = 0$ **then** {without obstacles}
2: $u.home \leftarrow u.id \bmod (Rows \times Cols)$
3: **else** {with obstacles}
4: $u.home \leftarrow u.id \bmod (Rows \times Cols)$
5: **if** $u.home$ is a cell occupied by an obstacle **then**
6: **repeat**
7: $u.home \leftarrow u.home - 1$
8: **if** $u.home < 0$ **then**
9: $u.home \leftarrow (Rows \times Cols) - 1$
10: **end if**
11: **until** $u.home$ is a cell not occupied by an obstacle
12: **end if**
13: **end if**

As described in Section 2.2.3, at the beginning SLURP divides the whole area into square grids. The size of a square is selected based on the transmission range of nodes in such a way that any two nodes residing in the same square remain within each others transmission range. By dividing into squares, the values of $Rows$ and $Cols$ are determined; these values represent the number of rows and columns of the grid covering the playground, respectively. Each square is given a unique ID starting from 0 to $(Rows \times Cols - 1)$, in the direction of left to right and bottom to top. Every node is assumed to be equipped with the layout of the obstacles, and hence, can determine which squares are occupied by obstacles. Based on the layout of obstacles, each node executes a hash function as described in Algorithm 3.1 and assigns a square as its home region.

In Algorithm 3.1, *mode* identifies whether the simulation is done in the presence of obstacles or not. If there are no obstacles, a simple modulo based hash function is used on the nodes' ID. Otherwise, the hash function is modified (Steps 5-12) in such a way that no node assigns a square fully occupied by obstacles as its home. This is done by decrementing the potential home ID until a square which is not covered by

any obstacle is found. By decrementing, if the assigned home ID becomes negative, the ID is assigned to the highest ID ($Rows \times Cols - 1$) representing the top-rightmost square of the grid and the process is repeated until a valid home is found. In this way, though the nodes are able to successfully assign squares as homes where some nodes are expected to be present, the storage overhead of the nodes becomes uneven, because the nodes residing in some of the squares need to store the location information of more nodes than that of the others.

3.1.2 Mobility Model in Presence of Obstacles

Among the widely used mobility models we have chosen the Random Direction model for this study. The Random Direction model maintains almost uniform node distribution throughout the simulation area while the other models such as the Random Waypoint model creates non-uniform node distribution after passing some time in simulation. Since we are interested in assessing the impact of the environmental contexts on a location service, it seems more appropriate to choose a model that creates uniform node distribution within the area where there is no obstacle. In the Random Direction model a node randomly selects a direction within $[0, 2\pi]$ and speed within $[0, \nu_{max}]$. It then moves forward in this direction until it hits the boundary, then pauses for some time, and again randomly selects another direction and speed. To cope with the real-world model, modifications are made to this model so that a node selects new direction and speed when it hits the boundary of the playground as well as the border of an obstacle. This modification is shown in Fig. 3.2 where a node moves from its initial position s_p to the destination position d_p.

3.2 Simulation Environment

For evaluating SLURP in the real-world environment model the network simulator ns-2.30 was used. The playground size was kept 1000m × 1000m and three obstacles were placed that occupied 23% of the simulation area (Fig. 3.1). Two sets of experiments were conducted, one without the obstacles in the playground and the other with the obstacles. In the first set of experiments, we implemented SLURP according to the specifications described in [19], i.e., without any obstacles and 100 nodes were used in the simulation. In the second set of experiments, the obstacles were introduced in

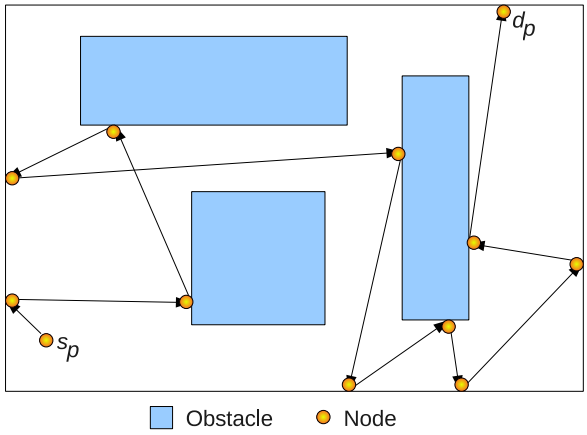


Figure 3.2: Modified Random Direction model in presence of obstacles.

SLURP and the modification in defining the hash function was made as specified in Section 3.1.1. For a proper assessment of the obstacles' impact on the location service, it is appropriate to make the comparison with a scenario having equal node density. Since 23% of the area is occupied by obstacles, 77 nodes were used in the simulation with the obstacles. As done in [19, 83, 20, 21], the experiments were performed with five different maximum node movement speed 10, 15, 20, 25, 30 m/sec. A number of mobility scenes were generated and 10 runs were performed with each scene, and the averaged results are presented here. The duration of the simulation with each scene was 100 seconds. To investigate the impact of the radio transmission range on the obstacle scenario, 250m, 200m and 150m transmission ranges with TwoRayGround propagation model as defined in ns-2.30 were chosen since different transmission ranges have different effects in terms of collision in the MAC layer. GPSR [15] was chosen as the routing protocol because SLURP uses this underlying geographic routing protocol for transmitting various location service related packets. Moreover, GPSR is scalable, robust and incurs a low routing overhead. Table 3.1 summarizes the parameters chosen for the simulation.

The simulation area was divided into grids according to the specified transmission range. For example, for transmission range of 150m, the square grids are of size 150m $\times$ 150m. Whenever a node crosses a region it updates its location to the respective location servers. In each second, four pairs of source and destination nodes are randomly selected, and each source uses the hash function to locate the respective destination

Table 3.1: Parameters used in simulation.

Notation	**Meaning**
Playground size	1000m × 1000m
Simulation duration	100 sec
Number of nodes N	100, 77
Mobility model	Random Direction and Modified Random Direction model
Transmission range	150m, 200m, 250m
Max node speed	10, 15, 20, 25, 30 m/sec
Request per second	4
Number of runs	10
MAC layer	IEEE 802.11
Routing protocol	GPSR

node's home region and looks for the location by sending a query packet to the location servers. If the location of the destination node is found, the location server sends a reply packet attaching destination's location to the source node.

3.3 Simulation Results

Various network parameters including the location update success rate, location update packet overhead, average hops for update and query, routing packet overhead and query success rate at different speed and transmission range have been evaluated. These parameters are widely accepted for evaluating the performance of location services in other studies [19, 21]. Moreover, the number of update and query packets successfully delivered with respect to the distance between transmitter and receiver has also been analyzed. For clarity in representing the results depicted in figures the following notations have been used: $R250$, $R200$ and $R150$ denote transmission ranges of 250m, 200m and 150m, respectively and for the obstacle scenario $R250-obs$, $R200-obs$ and $R150-obs$ denote transmission range of 250m, 200m and 150m, respectively.

Figure 3.3 shows the location update success rate at different speed of node movement and with various transmission ranges. The results show that the success rate decreases as the transmission range is reduced from 250m to 150m, and the presence

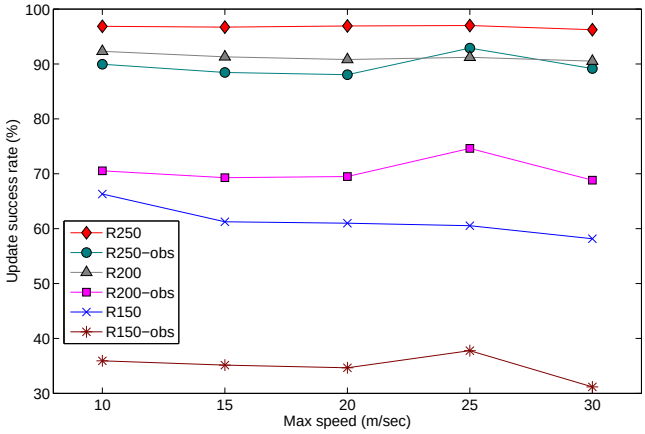


Figure 3.3: Location update success rate at different speed with various transmission range.

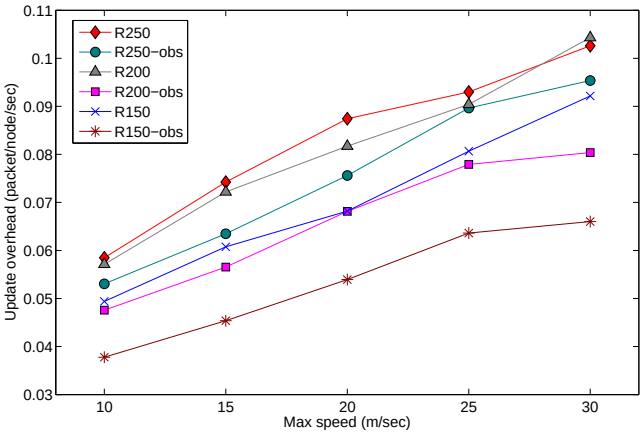


Figure 3.4: Location update overhead at different speed.

of obstacles makes it much worse. For instance, at 20 m/sec speed, the update success rate with transmission range of 200 meter is about 90.8% while with obstacles the rate reduced to approximately 69.5%. With a transmission range of 150 meter, the success rate is reduced drastically in the presence of the obstacles (less than 40%). This result indicates that, when obstacles are present and a lower transmission range is used, in most cases the nodes do not have enough neighbors towards the direction of the destination node, and consequently, forward packets in perimeter mode where it may turn into routing loops and traverse the outer face of the network due to the nature of the GPSR protocol. This results in the nodes' failure to update their locations to the location servers.

Figure 3.4 shows the location update overhead at various speeds. The location

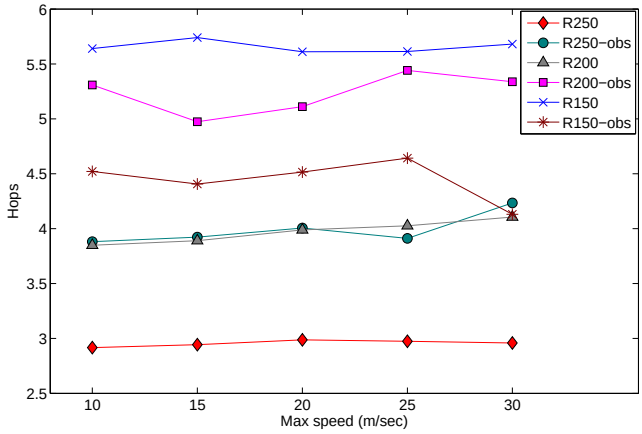


Figure 3.5: Average number of hops required for the update packets to reach their destinations.

update overhead for SLURP consists of two parts: firstly, the generation of an update packet by any node crossing the border of a square and secondly, when the packet is received by any location server residing in the update sender's home region, it is broadcast to all the nodes in that particular region. It is observed from the figure that the packet generation rate per node increases almost linearly with the increase of the speed; this is because at a higher speed, the nodes cross the boundary of squares more frequently. Moreover, it is observed that the update overhead is lower at lower transmission range. Since the update success rate is lower at lower transmission range irrespective of the node movement speed, the number of broadcasts within the home region is less, reducing the overall update overhead. Location update overhead in the presence of obstacles is lower compared to that of free space. Due to the restricted nature of movement, the nodes cross fewer regions resulting in generating less update packets. Moreover, the presence of the obstacles reduces the successful update packet delivery. This contributes to the lower update overhead in the presence of obstacles.

Given similar level of contention in the MAC layer, the packet transfer delay is proportional to the required number of hops. The fewer the average number of hops required to update and query, the better the performance from time delay perspective. The delay in update packet transfer in terms of hops from a source node to its home region is presented in Fig. 3.5. As expected, it is observed that in the free space scenario, a higher transmission range leads to fewer hop traversals for packet delivery compared to that for a lower transmission range. In the case of 250m and 200m ranges,

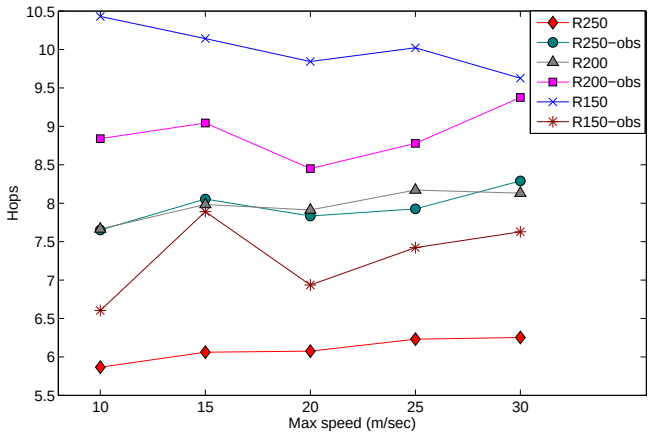


Figure 3.6: Average number of hops required for the query packets to reach their destinations.

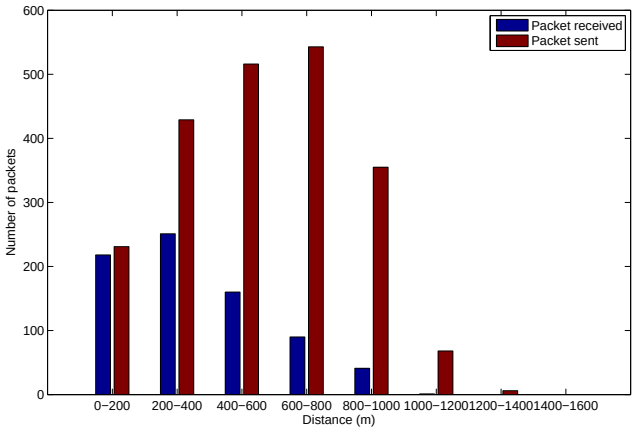


Figure 3.7: Update packets delivery at different distances (at 10 m/sec).

it takes a higher number of hops in presence of the obstacles than in free space. The existence of any obstacle in between the sender and receiver forces the packets to travel around the obstacles in many cases, as no node was present inside the area occupied by the obstacles. This results in increased number of hops in the obstacle scenario. A similar type of result is also experienced in the delay for query packets in terms of hops and is presented in Fig. 3.6.

On the contrary, in the case of 150m transmission range, the required number of hops for both the update and query packets in the presence of obstacles became lower than that of the free space scenario. Thus further analysis was conducted, which revealed an interesting result. Figure 3.7 presents the number of update packets sent and successfully received within a specific range of distance between a node and its

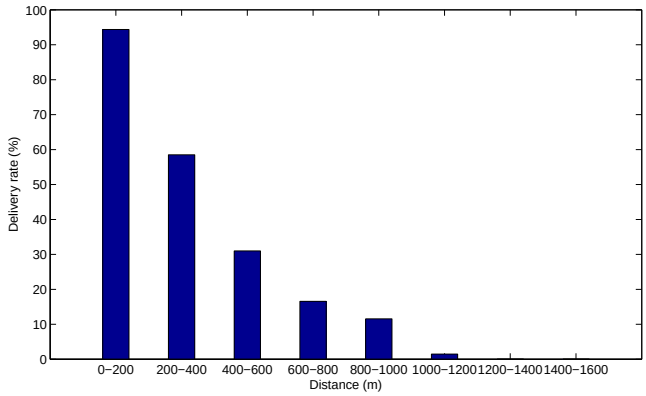


Figure 3.8: Percentage of update packets successfully delivered at different distances (at 10 m/sec).

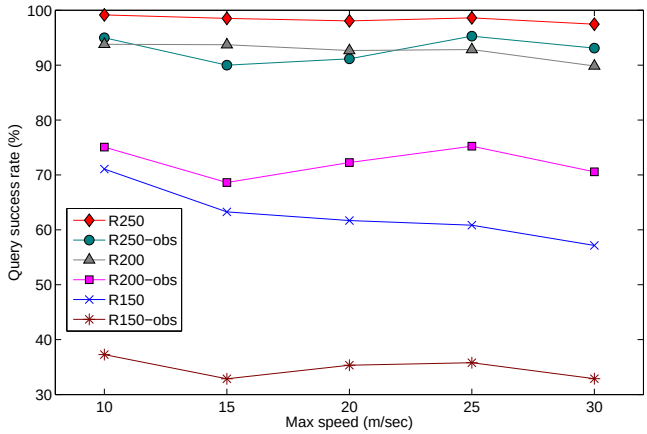


Figure 3.9: Location query success rate at different speed.

respective home region. Similarly, Fig. 3.8 shows the percentage of update packets successfully received at different distances. These results demonstrate that, as the distance increases, the update success rate decreases exponentially. Thus for 150m transmission range, in most cases, the nodes whose location servers were nearer were able to update their locations, resulting in a fewer average number of hops in obstacle scenario than that of the free space scenario (Fig. 3.5). This phenomenon also explains the similar trend observed for location query in Fig. 3.6.

The location query success rate is one of the key metrics in a location service scheme. It is always desirable for a geographic routing protocol to employ a location service that can ensure higher query success rate. Figures 3.9 and 3.10 illustrate the location query success rate at various speeds and transmission ranges, respectively. The results show

that, as the transmission range decreases, the query success rate also decreases. The location query success rate also depends on the update success rate, since a query is not expected to be answered by the location server if it has not been updated with the node's current location. Again, in the presence of obstacles, the query packets need to traverse around the obstacles resulting in a higher number of hops (Fig. 3.6). The same phenomenon is experienced by the reply packets. The requirement of a high number of hop traversals to send a location query and receive its reply in the obstacle scenario also has impact on the query success rate, especially when the transmission range is low in which case query success rate is very low (less than 40%).

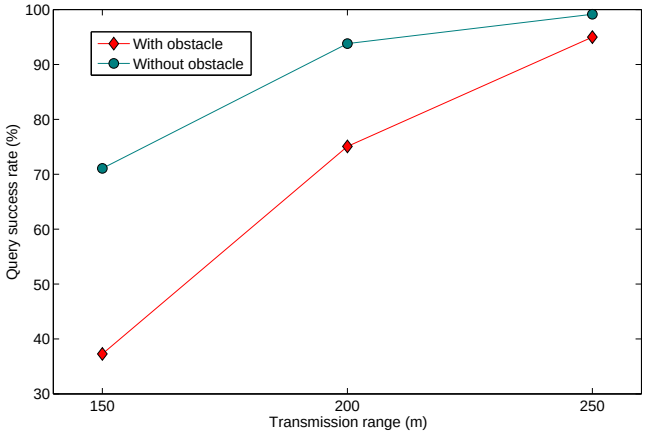


Figure 3.10: Location query success rate at different transmission ranges (at 10 m/sec.)

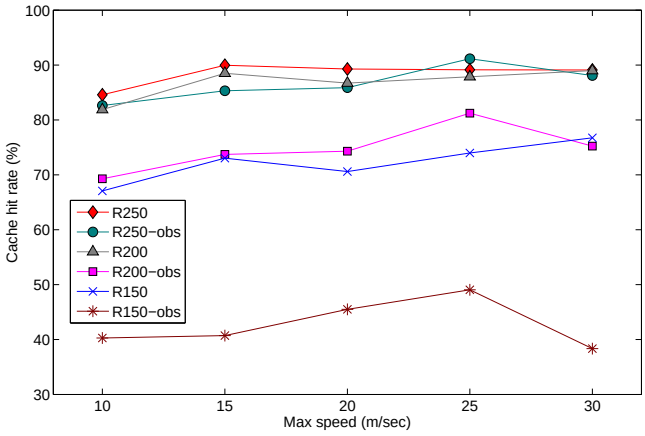


Figure 3.11: Query cache hit rate at different speed.

For query purpose, the packet drop can take place in different phases. Among the query packets sent, some packets are dropped and the remaining packets are successfully

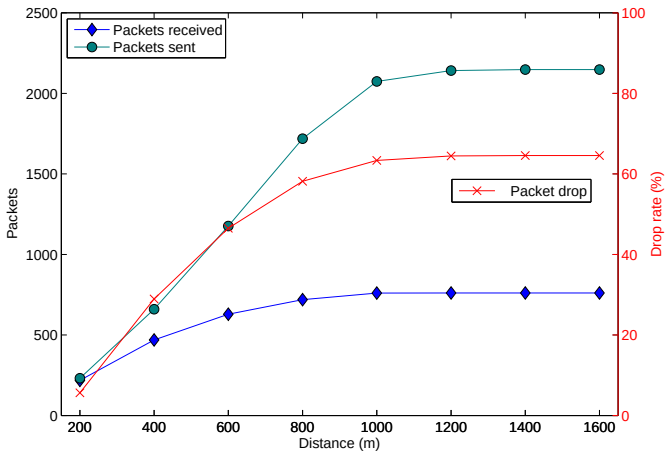


Figure 3.12: Packets delivered at different distance.

received by the location servers. Among the successfully received query packets, the location servers become unable to answer some of the queries (location servers could not find the queried node's location in the database), and so those packets are dropped. The remaining queries are answered by sending reply packets. Some of these reply packets are lost on the way to the query generator node, and the remaining ones are successfully received. The cache hit rate counts the percentage of replies sent by the location servers among the received queries, and is presented in Fig. 3.11. The figure indicates that, for a higher transmission range, the cache hit rate is higher, which is justified by a higher update success rate at higher transmission range, as illustrated in Fig. 3.3.

Figure 3.12 shows the number of packets (both update and query) sent and received successfully in the presence of obstacles with respect to the distance between transmitter and receiver when the transmission range is 150m and the maximum speed of node movement is 10 m/sec. As observed in the figure, the presence of obstacles has a significant impact on packet delivery, and thus SLURP's performance degrades considerably.

Figure 3.13 shows the routing overhead per query packet which increases almost linearly with respect to the node speed. Routing overhead includes all the packets transmitted for GPSR's beaconing, SLURP's location update, query and maintenance. We were interested in investigating this metric as it explains the overall overhead

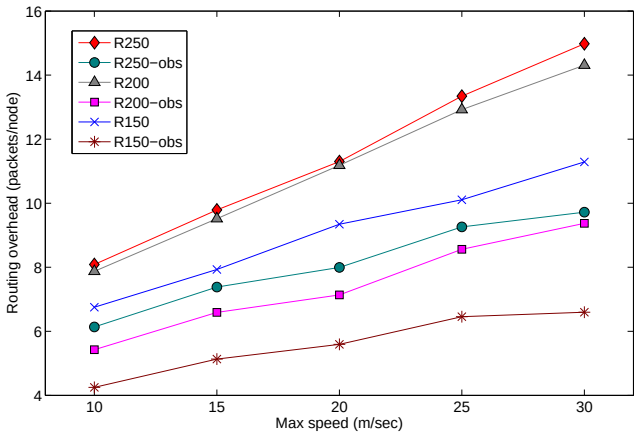


Figure 3.13: Routing overhead at different node speed.

incurred in a network. All speed variations (e.g., 10, 15, 20, 25 and 30 m/sec) considered in a specific scenario (e.g., free space) have an equal beaconing overhead while the other types of overheads vary. The presence of obstacles leads to a lower total overhead due to the restricted node movement but lower success rate of updating location servers.

3.4 Key Findings and Research Challenges

The analyses of the location service conducted in this chapter have identified a number of research challenges. These are summarized as follows:

- The mobility model used in the obstacle scenario is a simple modified Random Direction model. However, this model is not robust enough to cater realistic node movement. For example, if the obstacles have doors to enter and exit, the mobility model should direct nodes in and out of the obstacles. Moreover, the model should determine an appropriate speed inside the obstacles, converge or diverge nodes to and from specific regions of interest, and so on. Incorporating a realistic node mobility is expected to reveal a more realistic picture in evaluating the network performance. In fact, a realistic mobility is a pre-condition for evaluating MANET protocols.

- The evaluation of most of the location services available in the literature did not consider any obstacle in the simulation environment. As a result, the impact of obstacles on signal attenuation has been not considered. Though the study pre-

sented in this chapter considers obstacles, signal fading caused by the obstacles has not been taken into account. As experienced in [29], there is a significant impact of signal fading on performance measurement metrics and hence, in the presence of signal attenuation, further substantial degradation in performance is expected. The consideration of signal propagation more realistically will eventually provide a better assessment of the network performance.

- The presence of obstacles causes location packets to be successfully delivered over small distances, but a significant number of packets are dropped when packets need to travel longer distances. Moreover, the presence of obstacles results in non-uniformly distributed nodes, and consequently the empty home problem arises if the location service algorithm is not adjusted to cope up with the problem. These observations emphasize that obstacles have significant impact on the scalability and effectiveness of location services.

- The study in this chapter suggests that the obstacles cause significant number of location update, query and reply packets to drop resulting in a lower location query success rate.

- The above findings stress the need for an efficient location service that can adapt itself taking environmental impacts into account, and by doing so improvement of location service performance can be achieved.

3.5 Conclusion

The main focus of this chapter is to thoroughly investigate the influence of obstacles on a location service protocol through various network performance metrics. In this regard, a few major issues that are extremely crucial but have not been adequately addressed in mobile ad hoc network research are revealed and it is evident that further research needs to be conducted to improve these services in the presence of obstacles to make them more suitable for a real-world environment. This study also reveals how transmission distance holds a greater impact on the performance in the presence of obstacles. Some key research issues have been identified and highlighted in Section 3.4. These issues establish the foundation of this thesis by underlining the next research directions. In Chapter 4, we developed a realistic mobility model that can reflect

the real environment more accurately by considering the signal attenuation caused by the presence of obstacles and guiding node mobility by considering inaccessible objects, accessible obstacles through entry points (doors) and pathways. In Chapter 5, we proposed a novel location service protocol that is capable of coping with these environmental contexts and performs well in a realistic environment.

Chapter 4

Environment Aware Mobility Model

As identified in Chapter 3, it is necessary to consider a realistic mobility model to design and evaluate a location service protocol for the real-world deployment of mobile ad hoc networks. Among the widely used geographic restricted mobility models, Voronoi graph based model is the contemporary and most popular in ad hoc network simulation where terrains with different obstacles are considered. However, some important loopholes are identified in this chapter regarding the Voronoi graph based model's applicability in real-world environment modeling and consequently, a new mobility model is proposed that can emulate node mobility more realistically.

This chapter is organized as follows. After providing a brief overview of the proposed mobility model in Section 4.1, the limitations of Voronoi graph based model and the demand for a refined mobility model is presented in Section 4.2. Section 4.3 presents the proposed mobility model, while its computational complexity is analyzed in Section 4.4. Section 4.5 and 4.6 present the simulation environment used for network performance evaluation and the corresponding simulation results, respectively. Finally, Section 4.7 concludes the chapter.

4.1 Introduction

One of the key factors for a simulation environment to reflect and assess a real-world scenario accurately is to model the node mobility in MANET as realistically as possible. After deployment of nodes, the mobility model used in the simulation is responsible for directing when and where the nodes should move. A variety of mobility models that have been proposed in the literature for studying mobile network are described

in Section 2.1. Among them the Random Waypoint model, Random Walk model and Random Direction model are widely used where nodes are assumed to be deployed in a rectangular sized unobstructed simulation area. Nodes can move any where in the given simulation playground according to the specification of the respective mobility model.

In most of the existing mobility models the nodes move in a random fashion in free space, but the real-world is complex and vastly diverse that makes the existing models inappropriate for representing the real-world context properly. The essential constituents of a natural environment are pathways and obstacles. Many obstacles such as buildings, lakes, hills, playgrounds and vegetation are often found in possible deployment areas of MANET, that restrict our movement in different ways. Here, by the word "obstacle" both accessible (e.g., buildings) and inaccessible (e.g., lakes, hills, restricted area) objects observed in the environment are referred. Irrespective of the position of pathways, obstacles and their entry points in a real-world setup, existing obstacle based mobility model [29] produces a fixed number of restricted paths and entry points to each side of an obstacle and uses the shortest path for node movement. However, there are some obstacles, e.g., buildings, where people can move inside and others where they are not expected to enter e.g., lakes. People also do not move in a random fashion but rather on a region of interest basis; usually they have specific regions/destinations in mind. There may be defined pathways or no pathways at all for going to a specific region. In many cases people do not use the shortest paths to reach their destinations. To address this and hence a major unresolved problem, "there can be no single model that is the best for all terrains" [29], it is essential to develop a mobility model where the node movement is guided considering the elements in a deployment area, such as obstacles and existing pathways and a framework which could accurately represent their natural structures. In this chapter, we accomplish this by exploiting the flexible positioning of *anchors* in the simulated deployment area and by proposing a novel mobility model called the Anchor-based Mobility Model (AMM).

The proposed mobility model incorporates all of the above mentioned real-world features. The anchors are assumed to be square shaped geographic areas and can be located anywhere in the deployment area depending on the context such as predefined paths, obstacles and their entry points, if exist. Therefore, they are positioned in such

a way to define the mobility framework of the deployment area. For example, anchors can be defined around each convex corner of an obstacle and entry points. Thus the anchors will contribute to generating a graph that will guide the node movement in an appropriate way. A node can reach any position in the simulation area by selecting a suitable anchor as its next step. The size of an anchor defines the freedom and flexibility of the node movement. Since, according to this model a node selects a random point in the next anchor to be visited, it can produce a more flexible node movement scenario which is able to capture a wide variety of movement patterns. The model also reduces the *surrounding effect* [45]. Moreover, a novel signal fading model is introduced that can simulate a more realistic radio propagation in the presence of obstacles. The main contributions of this work are as follows:

- We propose a new mobility model that generates node movement patterns more realistically than the existing mobility models and is thereby more suitable for simulating MANET in a real-world deployment because of the following reasons:
  - its ability to accommodate any arbitrary shaped obstacle. Obstacles within obstacles in a nested fashion can also be supported. Moreover, the internal floor plan of a building can be adopted as well.
  - the placement of doorways at any position in the side of an obstacle (e.g., building) so that the nodes can move inside the obstacle through these doors only.
  - the ability to adopt existing pathways in any portion of the simulation area.
  - the ability to compute node movement based on automatically generated anchors.
  - it allows visualization of the node movement and the generated mobility traces.
- We analyze the computational complexity of the model in detail and assess the impact of the model on the measurement of different network performance evaluation metrics with and without the signal attenuation due to the presence of obstacles.
- We developed a GUI based program "MobiGen"that includes all the above features and generates node movement scenario files which can easily be integrated

with any network simulator, e.g., ns2 [90].

To evaluate the proposed mobility model, a map of the Clayton campus of Monash University in Australia with infrastructures in real positions was used. Greedy Perimeter Stateless Routing (GPSR) [15] which uses the geographic location of a destination to forward data is used as the routing protocol. Our mobility model is compared with the widely used Voronoi graph based obstacle model [29] as it is capable of generating scenarios in the presence of obstacles. The simulation results show that the proposed model has a significant impact on the performance of the geographic routing protocol GPSR; most importantly, it simulates the true movement pattern of a node under a real-world scenario more closely than the other mobility models.

4.2 Need for a Realistic Mobility Model

As alluded in Chapter 1, ad hoc network is attractive for communication in battlefields, disaster areas and inter-vehicular communication where it embraces a huge number of different deployment contexts. In disaster areas after earthquake, flood, fire or war there might be many buildings, streets and roads destroyed. Established road infrastructure may be totally or partially damaged. Rescue operators may need to save people stuck under the demolished or semi-demolished buildings. In the absence or demolition of existing infrastructure, peoples' movement can not be expected to be limited on pathways only and restricting movement may not serve the purpose. In a university campus to enter a building one has to use doorways only. Also in battlefields soldiers tend to move on a point of interest basis considering the obstacles. Thus a realistic mobility model must be flexible enough to cater a variety of situations.

Voronoi graph based mobility model is not suitable for these types of scenarios for the following reasons. This model only uses buildings of regular shapes as obstacles and restricts the node movement to the edges of the Voronoi cells where movement paths are generated as the boundary edges of the Voronoi cells considering the corners of obstacles. This restricted movement lacks the randomness of peoples' movement as they are not always expected to move on pre-defined paths. Moreover, in the Voronoi graph based mobility model, it is not possible to create an inaccessible obstacle which does not permit to enter. Voronoi cell building procedure renders a number of entry

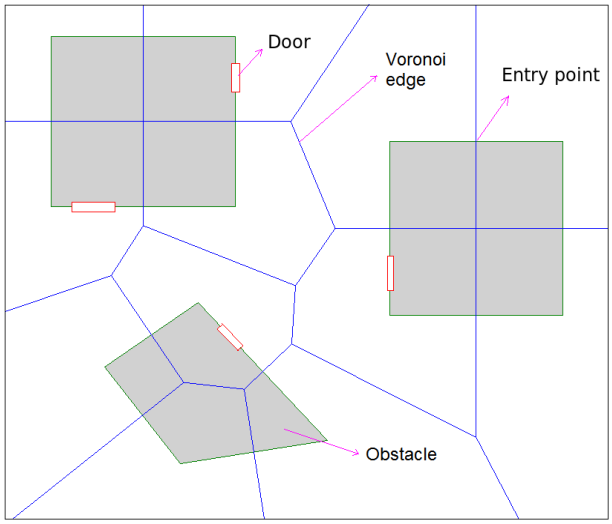


Figure 4.1: An example of Voronoi graph generated by considering corner points of 3 obstacles. Note that entry points to obstacle generated by Voronoi edges may not match with the existing doorways.

points at specific locations only which may not match the existing doorways of an infrastructure, as shown in Fig. 4.1. Furthermore, every side of a building may not have a door and consequently, the model fails to capture the actual environment. As a result, this model is incapable of generating a wide range of node movement variations.

All the above issues underpin the pressing need for a new mobility model that simulates a wide range of terrains and provides a movement pattern which resembles the real-world scenario more closely, and this remains the major focus of this study.

4.3 Anchor-based Mobility Model (AMM)

In the real-world when a person finds an obstacle on his way he tries to move around it - human beings can see obstacles from a distance, they do not necessarily collide with a side and then move along the walls like blind movement [91, 92]. This observation motivates us to exploit the concept of anchors to illustrate the movement paths by considering the obstacles and existing pathways in the mobility model which will consequently lead to a more realistic node movement. Once the anchors are generated, a node can move to a randomly selected destination using the shortest path. Several procedures [93, 94, 95] have been proposed in computational geometry to find the shortest path around polygons. The proposed algorithm follows a similar strategy

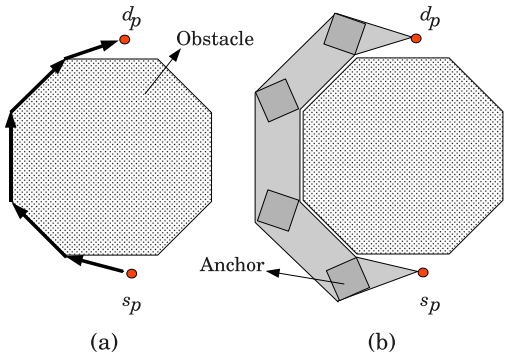


Figure 4.2: (a) The surrounding effect without anchors when a node moves from s_p to d_p (b) The anchors reduce surrounding effect.

but instead of using the single point of a corner, it uses anchors and thus reduces the effect of higher availability of nodes at the borders and corners of obstacles termed the *surrounding effect* [45] as shown in Fig. 4.2. Figures 4.3 and 4.4 show different types of anchors generated in the presence of obstacles with or without pathways and how they guide a node to move from start point s_p to destination point d_p.

It should be noted that our main objective is to develop a realistic mobility model that can adopt doorways in buildings and support pathways rather than finding the shortest path to the destination.

4.3.1 Generation of Anchors

Anchors are specific geographic positions that guide node movement while considering the existing pathways and obstacles including buildings with entry and exit points through doorways. The concept of anchors has been used in Terminode routing protocol [96] where the anchors are generated by selecting a point around which the node density is higher, with the help of monitoring node density throughout the network, in order to guide the packets to destination. However, in this model the generation procedure and usage of anchors are novel. Here, the anchors are generated according to the curvature, e.g., corners of obstacles and the placement of doorways and pathways. Depending upon the context of the scenario different types of anchors, namely, general, doorway and pathway anchors are generated as shown in Fig. 4.3 and Fig. 4.4. A general anchor is created at the side of every convex corner viewed from both the inside and outside of

an obstacle. As explained in Fig. 4.3, o is the center of a square shaped external general anchor with width q, a distance at g away from the nearest corner of an obstacle and its diagonal is the perpendicular to the tangent at that corner. When two obstacles become close enough, the anchor of an obstacle may intersect with the other obstacle. In this situation the corresponding anchor is shifted towards the corner of its obstacle and it can be shifted by at most $g = q/\sqrt{2}$. If the anchor still intersects with the obstacles, the size is reduced further so that the intersection diminishes, i.e., until $q > 0$. If $q = 0$ as a result of two obstacles touching at the corner then the anchor is removed.

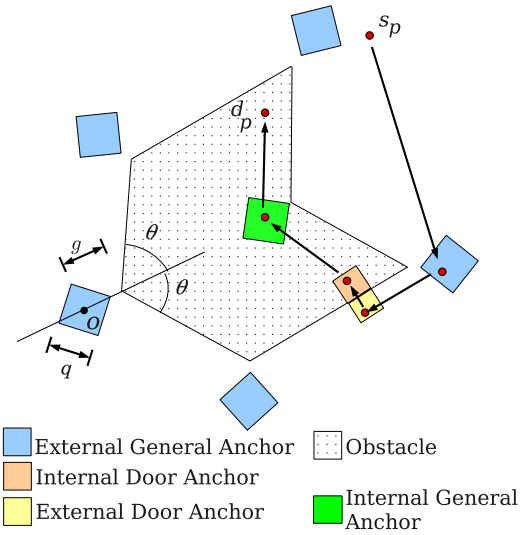


Figure 4.3: Node movement guided from s_p to d_p by anchors in the presence of an obstacle with an entry point when there is no existing pathway.

By choosing suitable values for q and g, it is possible to reduce the surrounding effect, and the nodes will be more spatially distributed along the spaces in between the obstacles. Moreover, the framework of using anchors is so flexible and generalized that it can be used to support any existing pathways and doorways for entering and exiting from any obstacle. Furthermore, in the real-world it is seen that, when a person moves around a corner of an obstacle he does not always keep the same distance from that corner. This phenomenon is simulated by selecting a random point within an anchor. This will make a node keep a random distance from the corner of an obstacle, as it happens in the real-world and leads to a more realistic movement behavior.

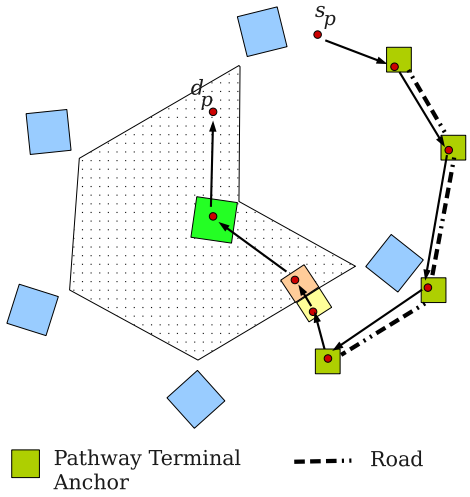


Figure 4.4: Node movement guided from s_p to d_p by anchors in the presence of an obstacle with an entry point when there is an existing pathway.

4.3.2 Graph Generation

After implementing the anchors, the graph creation mechanism starts. Here, a vertex is created for each anchor with the corresponding center point o. As shown in Fig. 4.5(a), when $\pi > 2\theta > \pi/2$, it is possible to reach d_p from s_p by choosing any intermediate point d_{p1} in the anchor, without intersecting the walls OD and OE. For special cases as shown in Fig. 4.5(b), if the convex angle $2\pi - 2\theta$ is reflex, i.e., $2\theta < 90^0$, the corresponding anchor is divided into two triangular segments BOX and OXY as this makes it possible to reach d_p from s_p by choosing any intermediate point d_{p1} and d_{p2} in each of the respective triangular segments without intersecting any edge OD and OE as $\theta \geq 0$.

In illustrating the Scenario generation algorithm, the various notations used are explained in Table 4.1 and the algorithm is presented in Algorithm 4.6. For better clarity, the algorithm is broken into sub algorithms (Algorhtims 4.1-4.5). Here, Algorithm 4.1 generates the internal edges of the graph by adding edges composed of the internal door vertex and external door vertex of a specific door, internal general vertices and internal door vertices for each obstacle and internal door vertices for each obstacles, to a list of edges called the *EdgeList*. Algorithm 4.2 generates the external

edges composed of the external general vertices, pathway vertices and external door vertices and adds them to the EdgeList. Algorithm 4.3 adds the pathway edges to the EdgeList. Algorithm 4.4 generates temporary edges based on a node's starting and ending position for each iteration. While adding edges, it is made sure that none of the edges intersect with any wall. Algorithm 4.5 specifies how a node selects a set of edges from the generated graph specified by the EdgeList as its path in order to reach its destination position. Finally, Algorithm 4.6 integrates Algorithms 4.1-4.5 together and specifies how the node movement trace is generated.

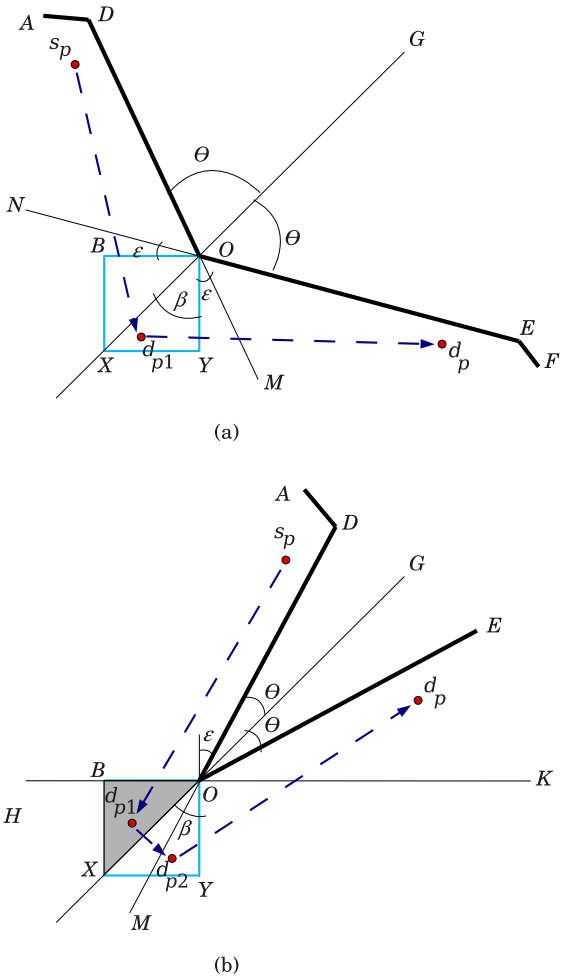


Figure 4.5: (a) Anchor $OBXY$ is generated when $90^0 < 2\theta < 180^0$ (b) Splitting of anchor $OBXY$ into two triangular segments BOX and OXY when $2\theta < 90^0$.

Here the $adjustWeight()$ function is used to specify the weight (length) of each

Table 4.1: Notations used in the algorithms.

Notation	Meaning
G_{ex}	Set of External General Vertices (EGV)
Ω_{ex}	Set of External Door Vertices (EDV)
V_p	Set of Path Terminal Vertices (PV)
Ω_{inb}	Set of Internal Door Vertices (IDV) of obstacle b
G_{inb}	Set of Internal General Vertices (IGV) of obstacle b
K	Set of obstacles
Ω_b	Set of doors of obstacle b
$v_{in}(\varrho_b)$	IDV of door ϱ of obstacle b
$v_{ex}(\varrho_b)$	EDV of door ϱ of obstacle b
v_{ip}	Vertex derived from a node's initial position
v_{ep}	Vertex derived from a node's end position
$\aleph$	Set of nodes

edge. For scenario generation purposes the weight of an edge that represents a road has been intuitively reduced to half of the length of the edge. As a result, roads get a preference in the shortest path computation and in moving to a destination, a node will prefer to travel on a road, if any, as in most cases we do in real life. Upon considering the actual phenomena in people's movement, AMM is flexible enough to select a path based on a particular criteria set by the users or any predefined path or a combination of paths which may include parts of both the shortest and predefined path.

Algorithm 4.1 createInternalEdges

```
1: for each obstacle b ∈ K do
2:    for each ϱ_b ∈ Ω_b do
3:       Add edge (v_in(ϱ_b), v_ex(ϱ_b)) to EdgeList
4:    end for
5:    for every pair of vertices v_i, v_j ∈ Ω_inb ∪ G_inb ∧ v_i ≠ v_j do
6:       if edge (v_i, v_j) does not intersect with any wall of b then
7:          Add edge (v_i, v_j) to EdgeList
8:          adjustWeight(v_i, v_j)
9:       end if
10:   end for
11: end for
```

Algorithm 4.5 returns a set of vertices required by a node to reach its destination

Algorithm 4.2 createExternalEdges

for every pair of vertices $v_i, v_j \in G_{ex} \cup V_p \cup \Omega_{ex}$ **do**
  if $v_i, v_j \notin V_p$ and the line v_i, v_j does not intersect with any wall **then**
    add edge (v_i, v_j) to EdgeList
    adjustWeight(v_i, v_j)
  end if
end for

Algorithm 4.3 createPathwayEdges

for all v_i, v_j, such that $v_i \in V_p \wedge v_j \in V_p \wedge v_i \neq v_j$ **do**
  add edge v_i, v_j (user defined path) to EdgeList
  adjustWeight(v_i, v_j)
end for

Algorithm 4.4 createTemporaryEdges

for each v in v_{ip}, v_{ep} **do**
  if v is inside obstacle b **then**
    for each vertex $v_k \in \Omega_{inb} \cup G_{inb}$ **do**
      if edge (v, v_k) does not intersect with any wall of b **then**
        add edge (v, v_k) to EdgeList with "temp" tag
        adjustWeight(v, v_k)
      end if
    end for
  else if v is outside **then**
    for each vertex $v_k \in \Omega_{ex} \cup G_{ex} \cup V_p$ **do**
      if edge (v, v_k) does not intersect with any wall **then**
        add edge (v, v_k) to EdgeList with "temp" tag
        adjustWeight(v, v_k)
      end if
    end for
  end if
end for

Algorithm 4.5 generateMobility

if to reach destination, a node u wishes to follow user defined path **then**
  use user-defined path
else
  use the path u wishes to take
end if

Algorithm 4.6 scenarioGenerator

Ensure: Algorithm 4.1-4.5
call createInternalEdges /* Algorithm 4.1 */
call createExternalEdges /* Algorithm 4.2 */
call createPathwayEdges /* Algorithm 4.3 */
call createTemporaryEdges /* Algorithm 4.4 */
call generateMobility /* Algorithm 4.5 */
remove edges marked "temp" from EdgeList
Set $v_{ip} \longleftarrow v_{ep}$ and $v_{ep} \longleftarrow$ new random destination
Repeat steps 4 to 7 until simulation duration T expires
Repeat steps 4 to 8 for each node $u \in \aleph$

from its starting point. For every movement, a node selects a random point inside the next vertex (for special anchors that need to be split into two triangular segments, one random point in each of the segments is used as the intermediate destinations as explained in Fig. 4.5(b)) in the list and continues to do so until it reaches its destination without intersecting any wall. This process continues until the simulation duration T expires for each node.

4.3.3 Other Node Movement Strategy

In reality, if the destination is known, people tend to use shortest path to reach there. However, if the destination is unknown or the person does not care for the shortest path a less direct route may be taken. To cater all types of node movement patterns, a different strategy is adopted which does not necessarily follow the shortest path. This movement can be defined as follows. Let $E = \{e_1, e_2, ..., e_k\}$ be the set of edges for a vertex v_{start}, and $dist(v_1, v_2)$ denote the distance between vertices v_1 and v_2. Then the next hop will be one of its neighbor (vertices that have an edge with the current vertex) vertices v_k for the destination v_{end}, if $(v_{start}, v_k) \in E \wedge \forall v^*$ where $(v_{start}, v^*) \in E \wedge dist(v_k, v_{end}) \leq dist(v^*, v_{end})$. When a node reaches a vertex, it selects the vertex among its neighbors which is nearest to the destination, similar to the packet forwarding method of GPSR in greedy mode. Due to the obstacles, the next vertex chosen may not necessarily be present in the shortest path.

In practice when people enter into an obstacle (e.g., building) their movement inside is restricted to walking speed, as they are pedestrians. To accommodate this feature in the proposed mobility model, the following node movement strategy while entering, exiting, or moving inside buildings is incorporated: when a node moves outside of the obstacles (e.g., on the road) a random speed is selected within the corresponding maximum speed (e.g., 10, 15, 20 m/sec) specified by the user. Whenever the node selects to enter or exit buildings or move inside buildings using Door Anchors and/or Internal General Anchors, it selects a random speed within the human walking speed (2 m/sec) as the average human walking speed is 3 miles per hour (1.4 m/sec) [97]. From the list of vertices to be traversed, a node identifies the type of anchor it needs to visit next and accordingly adjusts its speed. Thus the obstacle and anchor impact on user mobility is reflected in AMM.

For simulating an event like emergency relief or special event in a locality, AMM has another feature which permits the user to specify a number of arbitrary shaped regions of interest or *Hot spots* [45, 41]. If these regions of interest exist then a node randomly selects one region and a point within it and moves there. After arriving it again selects another region and this process continues until the simulation time expires. While moving from one region to another, nodes use the anchor based mobility model to deal with the environmental context. An example of the situation where this can be used is a rescue operation in a disaster area, if some people are found stranded in a specific region, the rescuers become interested to rescue them and move in order to reach there.

4.3.4 Mobility Trace Generation

As stated in Section 4.1, an independent GUI based program called MobiGen was developed to design the scenarios. These scenarios include various obstacles with doors, pathways and regions of interest. For each node i, $0 \leq i < N$, where N is the total number of nodes, MobiGen creates a mobility trace file that includes two parts: a) the initial position of the nodes and b) the node movement directives at specific times. The initial position part takes the following pattern:

\$node_ (i) set X_ x_0^i

\$node_ (i) set Y_ y_0^i

\$node_ (i) set Z_ 0

$\vdots$

The second part contains the directives for node movement and tells *when* and *where* to move at *what speed* for the simulation duration T. This part takes the following form:

\$ns_ at t_0 "node_(i) setdest x_1^i y_1^i ν_1 "

\$ns_ at t_1 "node_(i) setdest x_2^i y_2^i ν_2 "

$\vdots$

\$ns_ at t_j "node_(i) setdest x_{j+1}^i y_{j+1}^i ν_{j+1} "

where $j \geq 0$ and $t_j \leq T$. This implies, an arbitrary node i will start to move at time t_0 to destination (x_1^i, y_1^i) with speed ν_1. After arriving there it pauses for pause time t_{pause} if $t_{pause} > 0$. Thus t_{k+1} is calculated recursively as:

$$t_{k+1} = t_k + \frac{\sqrt{(x_{k+1}-x_k)^2+(y_{k+1}-y_k)^2}}{\nu_{k+1}} + t_{pause}. \tag{4.1}$$

where $0 \leq k < j$. After the mobility trace file is generated it is attached to the ns2 simulator as the scenario file. The scheduler in ns2 schedules each of the events and executes them at the appropriate time. Thus the nodes move according to the directives specified in the file starting from the initial positions based on the various scenarios generated by AMM.

4.3.5 Applicability of AMM

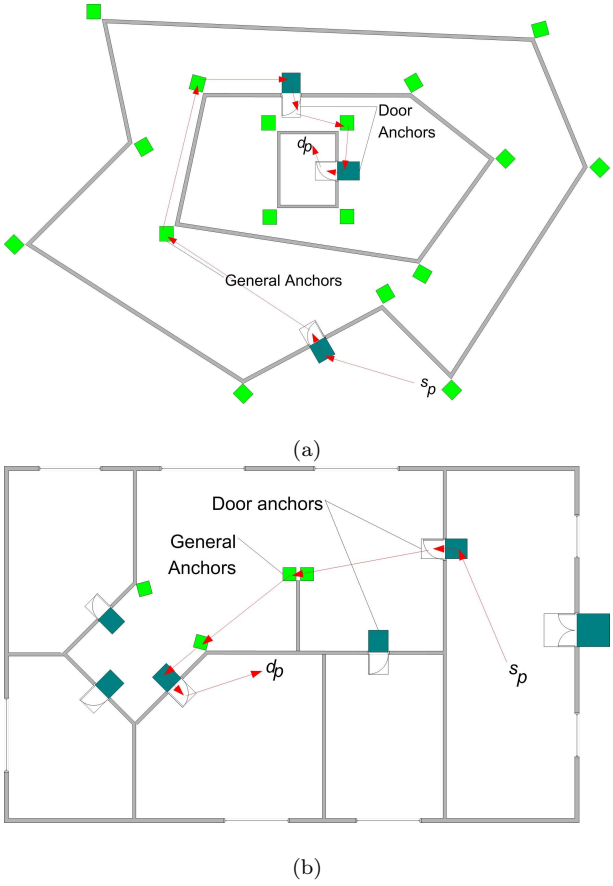


Figure 4.6: A node is guided from s_p to d_p by anchors in (a) nested obstacles (b) an internal floor-plan with doors.

It is obvious that there can be no single mobility model that is the best for every terrain. The proposed mobility model is focused on mimicking the movement of people in a pedestrian friendly area (e.g., university/school campus, techno park) where people can move anywhere except the inaccessible obstacles in the specified area and have the freedom and flexibility of using or not using the existing pathways. By following the

generated graph and considering the anchors, it is possible to move anywhere in the terrain without colliding with the wall of an obstacle. AMM assumes people are aware of the exact positions of the obstacles, doors and pathways. This model can also support nested obstacles (obstacles within obstacle) at any level as well as internal floor-plans of buildings as shown in Fig. 4.6. However, the proposed mobility model may not be suitable for representing City mobility where one mobile entity's mobility is affected by other entities and needs further modification. It can be inferred that AMM is a specialized Random Waypoint model that can adopt various geographic attributes. If there are no obstacles or pathways, no anchor is needed to be generated and the model turns into a simple Random Waypoint model.

4.4 Computational Complexity

This Section presents the computational complexity of the proposed mobility model. Table 4.2 illustrates the notations used for this analysis.

It can be inferred that the total number of corner points of obstacles = the total number of walls = w. So, w general anchors ($w = (I + F)$) are placed at the corners of each obstacle, either inside or outside. An IDV and EDV are generated for each door. As a result, the total number of IDV = total number of EDV = η. Again, finding the intersection of two lines and the $adjustWeight()$ function in Algorithms 4.1-4.4 have constant time complexity. For the anchor computation each general anchor is tested whether it intersects with any obstacles. Finding intersections using a brute force method for two polygons needs $O(mn)$ computation [98] where m and n denote the number of corners of each polygon, respectively. For this model where anchors have a fixed number of (four) corner points, the complexity becomes $O(4m) = O(m)$. Resizing the anchors after finding intersections requires constant time. Then the complexity for computing the general anchors is:

$$\left(\sum_{b=1}^{k_{obs}} O(w_b)\right)\left(\sum_{b=1}^{k_{obs}} w_b\right) = O(w^2) \tag{4.2}$$

since $\sum_{b=1}^{k_{obs}} w_b = w$. As the doorway and pathway anchors are user defined, it can be reasonably assumed that they will not intersect with any obstacle, making the

Table 4.2: Notations used in computational complexity analysis.

Notation	Meaning
k_{obs}	Number of obstacles
I	Number of IGV (Internal General Vertex)
F	Number of EGV (External General Vertex)
I_b	Number of IGV of obstacle b
η_b	Number of doors of obstacle b
η	Number of doors
μ	Number of PV (Path Terminal Vertex)
w	Number of walls, and General vertices in the scenario
w_b	Number of walls of obstacle b
μ_r	Number of user-defined roads
N	Number of nodes
T	Simulation duration
Υ	Average number of destination selection by a node within simulation time T

complexity

$$O(w^2 + 2\eta + \mu) = O(w^2 + \eta + \mu). \tag{4.3}$$

Step 1 in Algorithm 4.6 generates $O(\eta) + O\left(\sum_{b=1}^{k_{obs}} \binom{\eta_b + I_b}{2}\right)$ edges, and therefore, $O\left(\sum_{b=1}^{k_{obs}} \binom{\eta_b + I_b}{2}\right)$ edges need to be checked for intersecting the walls of respective obstacles resulting in

$$\begin{aligned} &O(\eta) + O\left(\sum_{b=1}^{k_{obs}} \binom{\eta_b + I_b}{2} w_b\right) \\ &\leq O(\eta + k_{obs} w_{max} (\eta_{max} + I_{max})^2), \end{aligned} \tag{4.4}$$

where w_{max}, η_{max} and I_{max} are the maximum values of w_b, η_b and I_b, respectively, for all values of b.

In step 2, each edge is checked with every wall for intersection requiring:

$$\begin{aligned}&\left(\binom{F+\eta+\mu}{2}-\binom{\mu}{2}\right)\times\sum_{b=1}^{k_{obs}}w_b\\&\leq O(w(F+\eta+\mu)^2).\end{aligned}\tag{4.5}$$

Step 3 requires

$$O(\mu_r).\tag{4.6}$$

Step 4 requires:

$$\begin{cases}O(2w(F+\eta+\mu)) & \text{both } v_{ip} \text{ and } v_{ep}\\ & \text{outside obstacle;}\\ O(2w_b(I_b+\eta_b)) & \text{both } v_{ip} \text{ and } v_{ep}\\ & \text{inside obstacle;}\\ O\begin{pmatrix}w(F+\eta+\mu)\\+w_b(I_b+\eta_b)\end{pmatrix} & \text{other cases.}\end{cases}$$

Since $F \geq I_b$, we have $\eta \geq \eta_b$ and $w \geq w_b$ in the worst case (when both the start and end points are outside obstacles) it becomes:

$$\begin{aligned}&O(2w(F+\eta+\mu))\\&=O(w(F+\eta+\mu)).\end{aligned}\tag{4.7}$$

In step 5, the shortest path algorithm [99] was used on the graph with vertices and edges to reach the destination which requires:

$$\begin{aligned}&O(|V|^2+|E|)\\&=O\left(\begin{array}{l}(I+F+2\eta+\mu+2)^2+\eta+\mu_r\\+\binom{F+\eta+\mu}{2}-\binom{\mu}{2}+\sum_{b=1}^{k_{obs}}\binom{\eta_b+I_b}{2}\end{array}\right)\end{aligned}\tag{4.8}$$

where $|V| = (I+F+2\eta+\mu+2)$, and $|E| = \eta+\mu_r+\binom{F+\eta+\mu}{2}-\binom{\mu}{2}+\sum_{b=1}^{k_{obs}}\binom{\eta_b+I_b}{2}$.

But as in the generated graph $|E| < \binom{|V|}{2}$, when $|V| \neq 3$, the complexity can be approximated as:

$O(|V|^2)$

$$= O((I + F + 2\eta + \mu + 2)^2)$$
$$= O((I + F + \eta + \mu)^2). \tag{4.9}$$

Removing temporary edges in step 6 it requires:

$$O(2(F + \eta + \mu))$$
$$= O(F + \eta + \mu). \tag{4.10}$$

Anchor generation, step 1, 2 and 3 are executed only once and so require (by summing up (4.3)-(4.6)):

$$O\left(\begin{array}{l} w^2 + k_{obs} w_{max} (\eta_{max} + I_{max})^2 \\ +w(F + \eta + \mu)^2 \end{array}\right) \tag{4.11}$$

since $\mu_r \leq \binom{\mu}{2}$.

If the average number of destinations selected within the simulation duration T for one node is Υ, then the total complexity for scenario generation which is loosely bound becomes:

$$O\left(\begin{array}{l} w^2 + k_{obs} w_{max} (\eta_{max} + I_{max})^2 \\ +w(F + \eta + \mu)^2 \\ +\Upsilon \times N \times (I + F + \eta + \mu)^2 \end{array}\right). \tag{4.12}$$

On the other hand the required complexity for generating the Voronoi graph is $O(w \log w)$ [100] since the number of generated Voronoi vertices is $O(w)$. As a result, the total complexity for the scenario generation in the Voronoi model is:

$$O(w \log w + \Upsilon \times N \times w^2). \tag{4.13}$$

The complexity of the proposed model is higher than the Voronoi graph based model because the Voronoi model does not take into account the position of doorways and pathways, and fails to capture the actual environment as explained in Section 4.2. Whereas the AMM model can produce better traces of the mobility pattern considering the environmental attributes as observed in the real-world at the cost of a higher computational complexity.

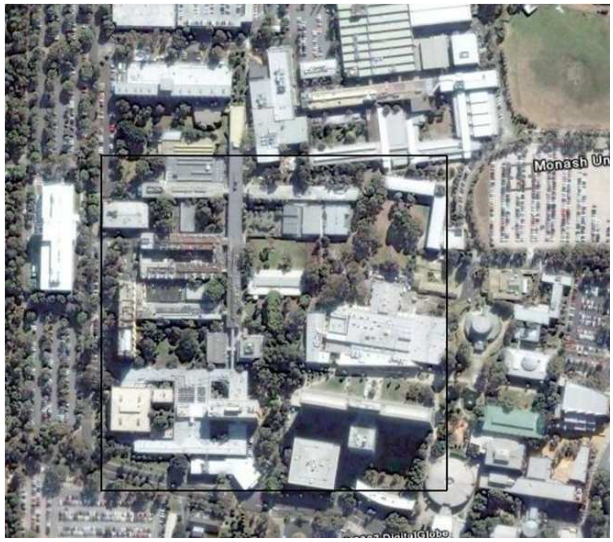


Figure 4.7: Snapshot of a portion (inside the square box) of Clayton campus, Monash University, Australia, selected for performance evaluation (courtesy: Google Earth).

4.5 Simulation Environment

MobiGen (Section 4.3.4) was used to generate the scenarios and these scenarios were based on a real context, the locations of buildings in the Clayton campus of Monash University, Australia as shown in Fig. 4.7. A portion of the map was selected keeping the playground size 1000m × 1000m and within this terrain the existing pathways, buildings and the actual positions of main entrances (doors) to buildings were incorporated in MobiGen as shown in Fig. 4.8.

ns-2.30 was used to simulate the proposed and existing mobility models by plugging the scenario files generated by MobiGen. 150 nodes were used in the simulation where 95% of the nodes follow the AMM algorithm, while the rest use the method which does not necessarily follow the shortest path as described in Section 4.3.3. The sides of the general and doorway anchors were 10 meters and 5 meters, respectively. For the node movement speed five different maximum speed of 2, 5, 10, 15 and 20 m/sec were considered where 2 m/sec represents the human walking speed. Additionally, as outlined in Section 4.3.3, irrespective of the maximum node speed, each node in AMM model travels at a speed randomly selected within human walking speed (2 m/sec) while entering, exiting and moving inside the buildings. 10 runs for each speed variation were performed and the averaged results are presented here. The parameters

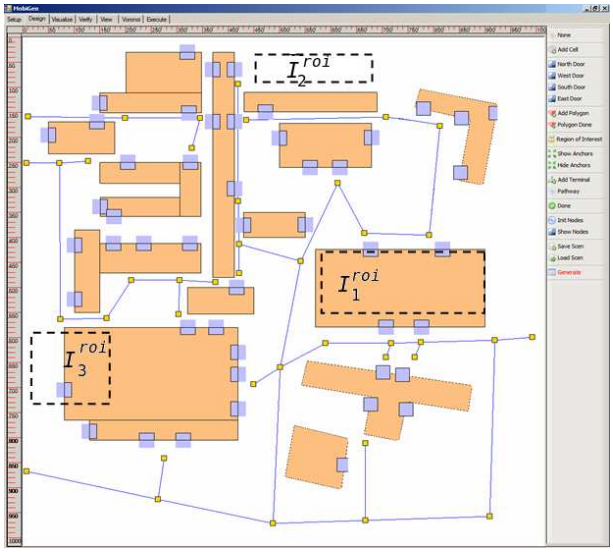


Figure 4.8: A snapshot of MobiGen implementing the portion selected in Fig. 4.7 with buildings, pathways and doors at exact positions. Blocks 'I_1^{roi}', 'I_2^{roi}' and 'I_3^{roi}' show location of regions of interest.

used in simulation are summarized in Table 4.3.

The duration of the simulation was 900 seconds. In each second 15 pairs of nodes were randomly selected, each pair consisted of one sender and one receiver and a 128 byte packet was sent to the receiver. When a node reached the destination, it paused there for random time between 10 and 20 second and then randomly selected another destination. For routing data packets, the GPSR protocol was used while the IEEE 802.11 protocol was used at the MAC layer.

Table 4.3: Parameters used in simulation.

Notation	**Meaning**
Playground size	1000m × 1000m
Simulation duration	900 sec
Number of nodes N	100, 150, 200
Transmission range r_t	150m, 200m
Max node speed	2, 5, 10, 15, 20 m/sec
Number of runs	10
MAC layer	IEEE 802.11
Routing protocol	GPSR

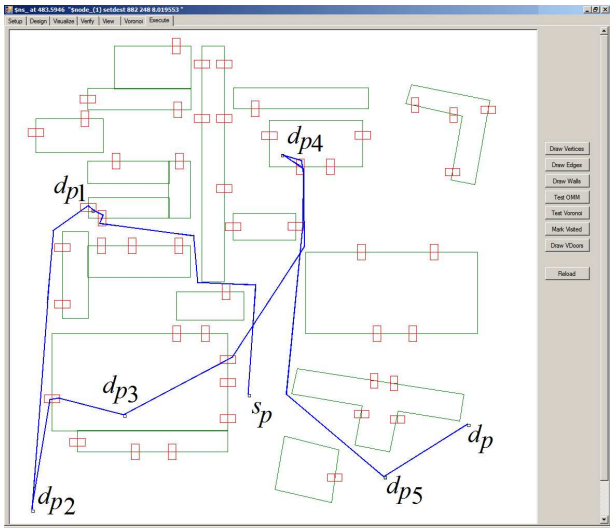


Figure 4.9: A Snapshot of MobiGen tracing a node's movement path for AMM (NoROI-NoPath).

4.5.1 Variation of Scenarios Generated by AMM

In order to simulate situations possibly occurring in the real-world, a wide variety of simulation scenarios were developed by the AMM model and experimented with. However, for the sake of clarity in presentation, only five variations are presented in this study :

1. InaccObs: Obstacles are inaccessible, the nodes are not allowed to enter the buildings and there is no regions of interest or pathways,
2. ROI: Obstacles are accessible through doors and there are regions of interest but no pathways,
3. NoROI-NoPath: Obstacles are accessible through doors and there is no regions of interest or pathways,
4. NoROI-Path100: Obstacles are accessible through doors and 100% of the nodes use the user defined pathways, and
5. NoROI-Path50: Obstacles are accessible through doors and 50% of the nodes use the user defined pathways.

Three rectangular shaped regions of interest were used as highlighted with dotted lines (I_1^{roi}, I_2^{roi}, I_3^{roi} in Fig. 4.8). The visualization of a node movement is presented in Fig. 4.9. A node starts from s_p at t_0 and progressively selects d_{p1}, d_{p2}, d_{p3}, d_{p4}, d_{p5} at t_1, t_2, t_3, t_4, t_5, respectively, and finally ends at d_p at t_6 when the simulation expires. For comparison with the proposed model the widely used Voronoi graph based obstacle mobility model [29] was selected. Since the assumption made in [29] that "there exists a door in the wall where the Voronoi edge intersects" does not always match with the obstacles placed in real-world setting, the edges residing in the buildings that do not coincide with the doors were trimmed to find out how the Voronoi based model would perform in real-world door placement.

4.5.2 Signal Propagation Model

The performance of wireless networks is significantly affected by signal attenuation and interference caused by the presence of different objects (obstacles). Because of radio signal's reflection, diffraction and scattering, it is a challenging task to design a realistic signal propagation model.

The most commonly used propagation models are listed as follows:

1. *Free space model*: It assumes ideal propagation condition and considers only one clear line-of-sight path between the transmitter and receiver and calculates the received power P_r at distance d according to:

$$P_r(d) = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^\alpha \varphi} \tag{4.14}$$

   where, P_t is the transmitted signal power, φ is the system loss ($\varphi \geq 1$), λ is the wavelength, G_t and G_r are the transmitter and receiver antenna gains, respectively, and α is the signal strength decaying factor over distance.

2. *Two-ray ground model*: This model includes a ground reflection path along with the direct line-of-sight path and gives a more accurate prediction at long distances than the Free Space model [101]. Here, the received power is predicted by

$$P_r(d) = \frac{P_t G_t G_r {h_t}^2 {h_r}^2}{d^\alpha \varphi} \tag{4.15}$$

where h_t and h_r are the heights of the transmit and receive antennas, respectively. The typical value of α lies in the range of $2 \sim 4$. In the Free space and Two-ray ground model $\alpha = 2$ and $\alpha = 4$, respectively.

3. *Shadowing model*: In this model a Gaussian random variable is added to the path loss to account for environmental influences.

However, all of these radio propagation models assume a flat ground surface and the simulation playground does not contain any obstacle [102, 103] and hence, are not directly usable in simulations where obstacles are considered. For this reason various researchers have tried to incorporate the influence of obstacles on radio propagation in different ways. Marinoni *et al.* [103] used Two-ray ground model and assumed the radio signal is fully obstructed by obstacles. As a result, if there exists a wall between the transmitter and the receiver, the receiver can not hear the transmission. A similar type of model has been introduced in [102] suitable for IEEE 802.11p to include *Wireless Access in the Vehicular Environment*. IEEE 802.11p works in a high frequency band (5.9 GHz) which makes it highly directional and experiences a very low depth of penetration. So any obstacle between the transmitter and receiver fully blocks the radio wave. Mahajan *et al.* [104] followed a different approach. Their work is based on IEEE 802.11b and the signal strength is attenuated or reduced when the signal passes through obstacle wall rather than being fully blocked. In their model logarithmic linear regression was applied and a simplified received signal strength was introduced as follows:

$$P_r = P_t + f_c - f_d \log(d) \tag{4.16}$$

where, f_c and f_d are tunable parameters represented as the *constant factor* and *distance factor*, respectively. However, selecting appropriate values for f_c and f_d is challenging as they can be different for various urban environments, and even across different regions within a single urban environment [104].

The propagation model must simplify the calculations in order to allow a reasonably accurate simulation within acceptable amount of time [103]. Here, a simple radio propagation model is introduced that can perform in the presence of obstacles considering the attenuation factor of the object materials. There have been many studies to find the attenuation factors of various types of materials [101]. The proposed radio

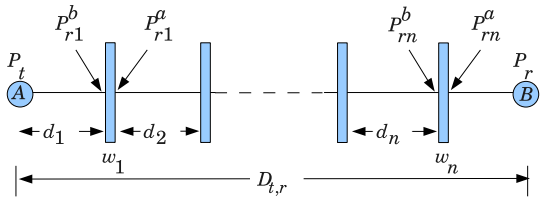


Figure 4.10: Signal propagation model.

propagation model is similar to [29] but also takes into account the distance and material of different obstacles while calculating the reduced signal strength. This model is based on the Two-ray ground propagation model and considers n walls (obstacles) $(w_1, w_2, \cdots, w_n)$ between the transmitter A and receiver B as shown in Fig. 4.10. Let w_1 be d_1 distance away from the transmitter, then the received power P^b_{r1} just before w_1 can be calculated as:

$$P^b_{r1} = \frac{P_t G_t G_r h_t^2 h_r^2}{(d_1)^\alpha \varphi}. \tag{4.17}$$

P^b_{r1} can be assumed as the power that is attenuated through the obstacle wall and after w_1 the received power P^a_{r1} is:

$$P^a_{r1} = 10^{(20\log(P^b_{r1}) - \gamma_1)/20} \tag{4.18}$$

where, γ_1 represents the attenuation factor (in dB) of w_1's material.

Similarly, the received power before and after the wall w_n with attenuation factor γ_n becomes:

$$P^b_{rn} = \frac{P^a_{r(n-1)} G_t G_r h_t^2 h_r^2}{(d_n)^\alpha \varphi} \tag{4.19}$$

and

$$P^a_{rn} = 10^{(20\log(P^b_{rn}) - \gamma_n)/20}. \tag{4.20}$$

$D_{t,r}$ distance away from A, the receiving power P_r at B is given by

$$P_r = \frac{P^a_{rn} G_t G_r h_t^2 h_r^2}{\left(D_{t,r} - \sum_{i=1}^{n} d_i\right)^\alpha \varphi} \tag{4.21}$$

Table 4.4: Transmission power attenuation factors.

	Home	**Office**
Single wall	6∼20 dB	6∼20 dB
Double wall	40∼50 dB	50∼60 dB

Similar to [29], the value of the attenuation factor γ_i $(1 \leq i \leq n)$, in Table 4.4 are used which considers whether the signal penetrates through a single or double wall of a home or office. Since a university campus is being modeled, only the attenuation factors for *office* are considered. Furthermore, only the boundary walls of obstacles as shown in Fig. 4.8 are considered here.

The formula to calculate the received signal strength is given in equation (4.22). In particular, the equation is exactly the same as the Two-ray ground model when no obstacle is present between the transmitter and receiver, and (4.21) is used elsewhere that eventually reduces the signal strength according to the distance and attenuation factor of each wall that exists between transmitter and receiver.

$$P_r = \begin{cases} \dfrac{P_t G_t G_r {h_t}^2 {h_r}^2}{(D_{t,r})^{\alpha} \varphi} & \text{no obstruction;} \\ \dfrac{P_{rn}^a G_t G_r h_t^2 h_r^2}{\left(D_{t,r} - \sum_{i=1}^{n} d_i \right)^{\alpha} \varphi} & \text{elsewhere.} \end{cases} \tag{4.22}$$

To detect signal, P_r must fulfill $P_r \geq R_x$, where R_x represents the minimum signal threshold required to receive the signal successfully. In this study $\alpha = 4$ was selected. This is how the proposed propagation model introduces partially pass radio signal through obstacles.

4.6 Simulation Results

Various network performance metrics are calculated which includes: a) number of neighbors at different time span and speed, b) link duration, c) path length between packet originator and receiver in terms of hops, d) packet delivery success rate at different node speed and density, e) end-to-end delay and f) routing overhead. These measures are also investigated in the simulation of similar studies [29, 27]. The propa-

gation model as defined in ns-2.30 with signal attenuation as described in Section 4.5.2 was used. Moreover, the same experiment was also repeated without considering the signal attenuation in order to investigate the impact of signal attenuation on the network performance. Unless otherwise stated, signal attenuation is considered in the simulations presented below. The distribution of nodes and partition formation in mobility models and their characteristics in large areas are also investigated.

Figure 4.11 shows the average number of neighbors at different time stamp throughout the simulation. From this figure it can be concluded that the mobility models actually fall into three groups:

1. Voronoi graph based model that distributes nodes sparsely throughout the simulation area (bottom in figure);
2. ROI model that tends to concentrate the nodes in particular geographic regions (top in figure); and
3. The other models (NoROI-NoPath, NoROI-Path50, NoROI-Path100 and InaccObs) that fall in between.

Due to different levels of node concentration caused by the different models, the average number of neighbors (Fig. 4.12) and the durability of the link with respect to node movement speed (Fig. 4.14) are affected. Since in ROI the nodes are concentrated in specific regions, the average number of neighbors is the highest. While average number of neighbors remains less affected with node movement speed, the link duration decreases with the increasing node speed in mobility models in general. The number of neighbors and the link duration have a direct impact on the average number of hops for packets from sender to receiver (Fig. 4.16). Here it is seen that the models belonging to Group 1 result in higher path length than the other groups and Group 2 results in the lowest due to different level of node concentration. Finally, the path length has a direct impact on the packet delivery success rate (Fig. 4.17 & 4.18) as the models having lower path length show a higher success rate. To investigate and assess the impact of signal attenuation on different network performance metrics, the same experiments were repeated without considering signal attenuation and compared to the results of signal attenuation described above. Figures 4.13 and 4.15 show the

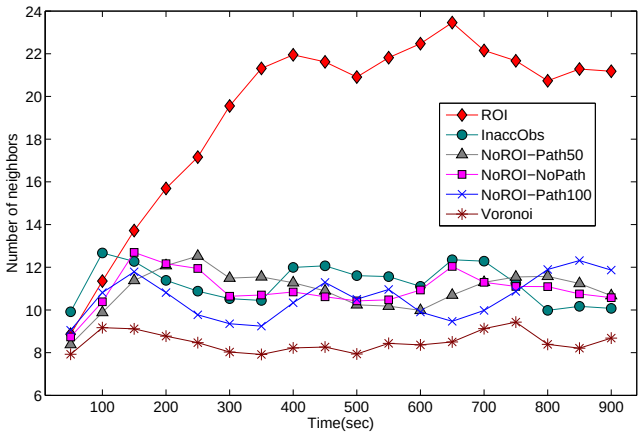


Figure 4.11: Variation of average number of neighbors over time (at 10 m/sec).

average number of neighbors and link duration, respectively, without considering the signal attenuation and consequently their impact on the packet delivery success rate is shown in Fig. 4.20.

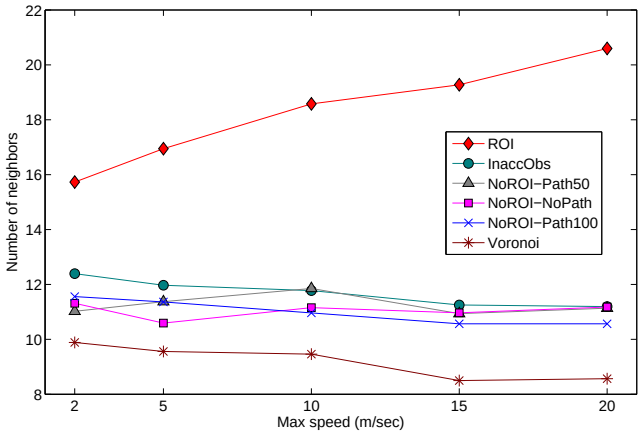


Figure 4.12: Average number of neighbors.

In Fig. 4.11, the Voronoi model shows less fluctuation in the number of neighbors over the simulation duration compared to the other models. This fact is justified as in the Voronoi model, for the whole simulation duration, the nodes follow predefined paths resulting in almost similar number of neighbors over time. On the other hand, in all variations of scenarios of AMM, there is no predefined route that would be followed for the whole duration. This causes the nodes to become more clustered at different time stamp and fluctuation in the number of neighbors is experienced. Moreover, at the

initial stage in AMM, the nodes are randomly placed in the simulation area, but in the course of time the movement pattern becomes more restricted yielding an increasing trend in the number of neighbors during the first part of simulation.

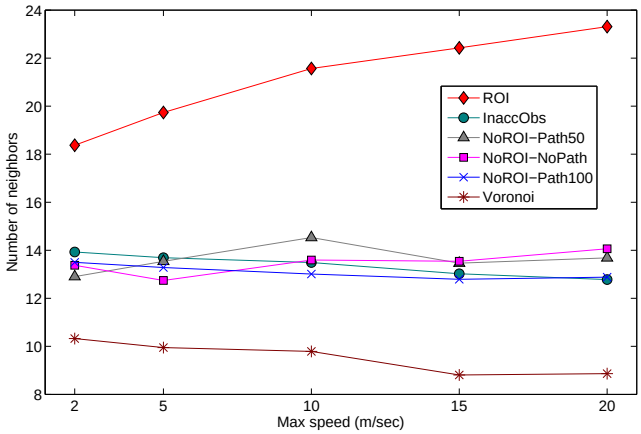


Figure 4.13: Average number of neighbors without considering signal attenuation.

Figure 4.12 shows the average number of neighbors at different speed. When the regions of interest are present, the nodes travel from one region to another and if possible, use doorways to go through the buildings. This makes node movement much more concentrated in the area between the regions and increases the number of neighbors. Figure 4.13 shows the average number of neighbors if no signal attenuation is considered. It shows a similar trend as observed in Fig. 4.12 but the number of neighbors is relatively higher as the signal experiences no attenuation.

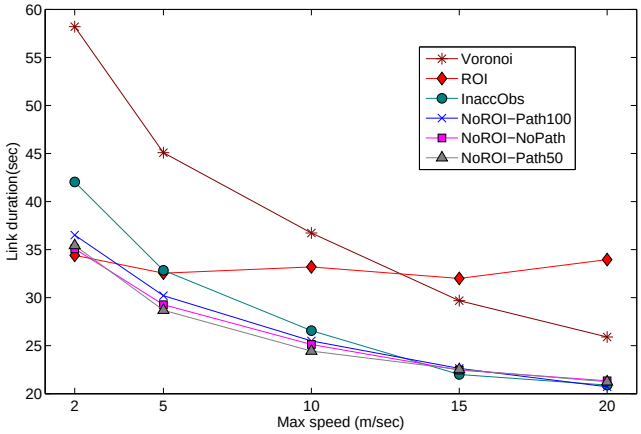


Figure 4.14: Average link duration.

As shown in Fig. 4.14, the Voronoi model experiences the highest link duration in

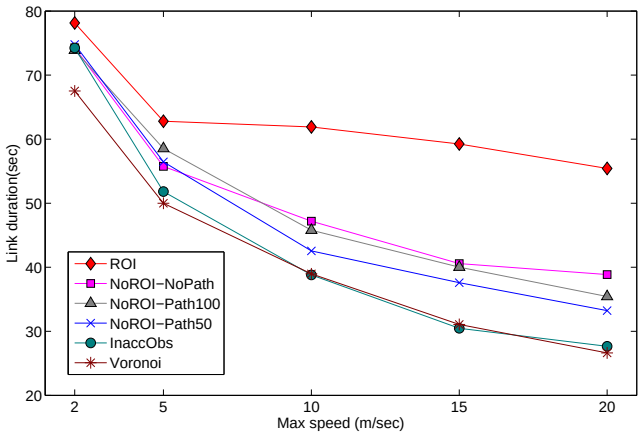


Figure 4.15: Average link duration without considering signal attenuation.

the presence of signal attenuation when the nodes move relatively slowly. On the other hand ROI exhibits the highest link duration (Fig. 4.15) when no signal attenuation is considered. In the Voronoi model, when a node moves along a predefined path, the other nodes on the same edge are moving either in the same direction or in the opposite direction. This situation continues until it reaches the nearest vertex where the nodes may select different edges. Thus by limiting node movement to the Voronoi edges, there is less effective area where the nodes are available, resulting in a higher clustering of nodes. Moreover, because of the inherent property of Voronoi graph, the edges are constructed in the middle of two obstacles and signal attenuation has less impact. On the contrary, region I_1^{roi} and greater portion of I_3^{roi} (regions of interest in Fig. 4.8) is inside buildings, allowing the nodes to enter and exit buildings. In this case the walls play a significant role in attenuating signal which has a higher impact on the link duration causing more frequent link breaks. As a result ROI exhibits a lower link duration than Voronoi although ROI experiences the highest number of neighbors (Fig. 4.11 & 4.12). It is interesting to observe that the link duration of ROI is less sensitive to different speed. In AMM when a node enters or exits a building, it continues to move at walking speed. The regions of interest actually cause the nodes to move at walking speed most of the time (entering, staying inside building and exiting) irrespective of the maximum allowable speed and consequently ROI encounters less change in link duration. However, the link duration as well as the packet transfer success rate, path length and average neighbors of ROI are subject to be affected by the number, placement and size of the regions of interest.

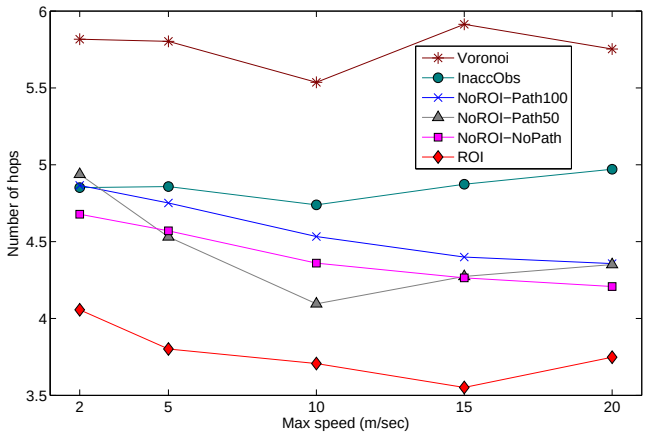


Figure 4.16: Average path length in hops from packet source to destination.

Figure 4.16 shows the average path length from the sender to the receiver in terms of hops. Here the Voronoi model exhibits the higher path length in spite of having the highest link duration. This is because, unlike the link duration, the required number of hops for packet traversing is a global characteristic. For GPSR having sufficient neighbors alone is not enough, rather the existence of nodes in the direction of the receiver is more important for forwarding packets in greedy mode. As the nodes in the Voronoi model move on the Voronoi edges only, the nodes do not find enough nodes in the direction of the receiver because of the unavailability of nodes in the Voronoi cells. Consequently they fail to find shorter routes (for switching to perimeter mode more frequently) and often fall into routing loops due to the nature of the GPSR protocol. On the other hand, in all variations of AMM except InaccObs the nodes are permitted to enter the buildings through doors only. As a result a few nodes are expected to be present there which can act as some sort of gateway between two regions separated by an obstacle. This results in a lower path length and ROI yields the lowest as expected.

Figures 4.17 and 4.18 show the packet delivery success rate at different node speed and density, respectively. In the Voronoi model, the nodes are only permitted to travel on the graph edges resulting in the lowest success rate as packets have to travel a longer path as observed in Fig. 4.16. Moreover, the trimming of edges that do not coincide with the existing doors of obstacles results in larger empty spaces. This forces the packets to select a longer path and consequently the packet loss increases. In InaccObs where the nodes can not enter buildings, the success rate is less than the

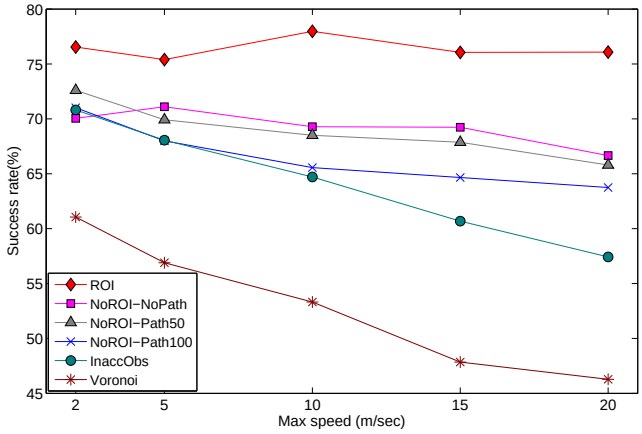


Figure 4.17: Packet delivery success rate.

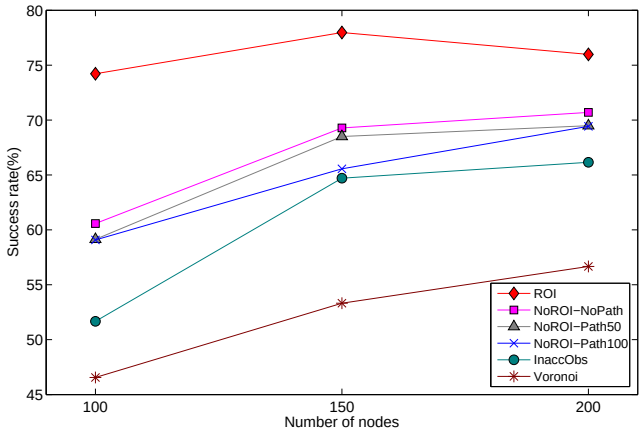


Figure 4.18: Packet delivery success rate at different node density (at 10 m/sec).

other variations of scenarios of AMM. This is because the absence of nodes inside the buildings causes the packets to travel around the buildings and the longer path results in more packet delivery failure. For NoROI-Path100 nodes are concentrated on the predefined pathways and the number of defined roads is less than that in the Voronoi graph. In this case most packets are sent to nearby nodes resulting in a higher success rate than the Voronoi model. The success rate of NoROI-Path50, which is actually a combination of NoROI-NoPath and NoROI-Path100, stays in between as expected, since 50% of the nodes chose to use the roads. The presence of nodes inside the obstacles reduces the chance of perimeter mode forwarding in the NoROI-NoPath model resulting in a higher success rate. In ROI where the nodes move on a region of interest basis, the success rate is highest. This is due to nodes concentrating in the

regions of interest and their interconnecting areas and the average physical distance between the transmitters and receivers becomes smaller. This causes the packets to take a fewer number of hops to reach the receiver as observed in Fig. 4.16 and consequently the packet delivery success rate increases.

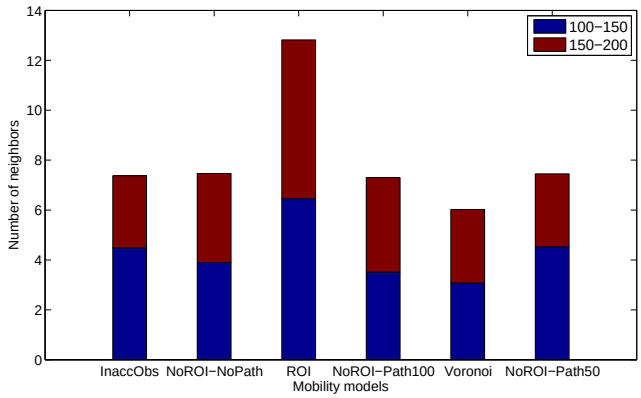


Figure 4.19: Increase in number of neighbors when number of node increases (at 10 m/sec).

In Fig. 4.18 it is observed that the packet delivery success rate increases in all the mobility models when the number of nodes increases from 100 to 150. The Voronoi model shows a similar increasing trend when the number of nodes increases from 150 to 200 but the success remains unchanged or drops slightly in AMM. This phenomenon can be attributed to a significant increase in channel congestion when a higher number of nodes become clustered in a small region resulting in relatively lower packet delivery. Figure 4.19 supports this fact as it is observed that the Voronoi model has the lowest increase in the number of neighbors as the number of nodes increases.

The packet delivery success rate was measured without considering the signal attenuation which is shown in Fig. 4.20. It can be inferred that, in the presence of signal attenuation, a significant number of packets are dropped resulting in a reduced packet delivery success rate irrespective of the mobility models as a comparison between Figs. 4.17 and 4.20 reveals. It can be concluded that while using the models belonging to Group 2, the routing protocol has the highest success rate, whereas with Group 1 it performs the lowest, and for Group 3 the success rate stays in between.

To test the significance of difference in success rate among the models (Fig. 4.17), InaccObs and Voronoi were selected as representatives of the proposed AMM models and existing models, respectively. The t-test at speed 2, 5, 10, 15 and 20 m/sec yielded

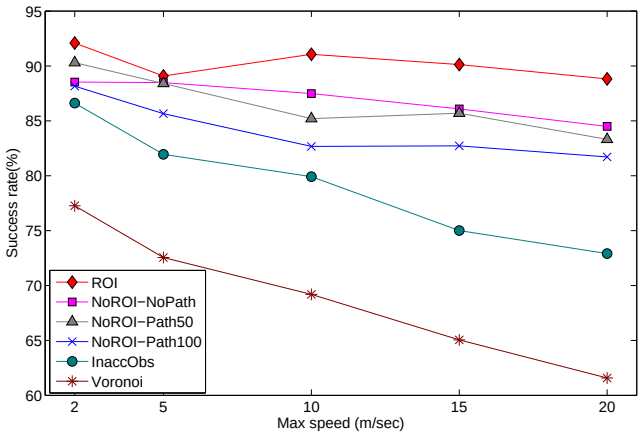


Figure 4.20: Packet delivery success rate without considering signal attenuation.

p-values of $p < 2.11 \times 10^{-8}$, 4.79×10^{-7}, 4.10×10^{-7}, 1.30×10^{-6}, 9.24×10^{-6} at 99% confidence level, validating their performance difference being statistically significant.

4.6.1 Distribution of Nodes

In order to examine how the node distribution takes place in different mobility models similar to [104] the 1000m × 1000m topology was divided into 50m × 50m regions, and the number of nodes was counted in each region starting from 300 sec with 10 sec interval and averaged the result.

Figure 4.21 shows the node distributions and contour plots found in the ROI, NoROI-Path100 and Voronoi models. Figures 4.21(a, b) shows how the node distribution takes place in the ROI model. It is observed that nodes are concentrated at the regions of interest defined in the map (snapshot of MobiGen shown in Fig. 4.8). Also, since the nodes move from one region of interest to another, there are nodes available in the interconnecting areas between the regions of interest. Similarly, Figs. 4.21(c, d) shows the node distribution resulting from NoROI-Path100. In this model the nodes can move any where in the simulation playground and use the existing roads. Comparing Fig. 4.21(d) to the layout of the roads in Fig. 4.8 reveals that the nodes are more concentrated along the existing roadways. In the Voronoi model's node distribution (Figs. 4.21(e, f)), it is observed that nodes are distributed along the Voronoi edges and higher number of nodes are clustered in the middle of the terrain.

In the presence of regions of interest, AMM variation of ROI creates those regions

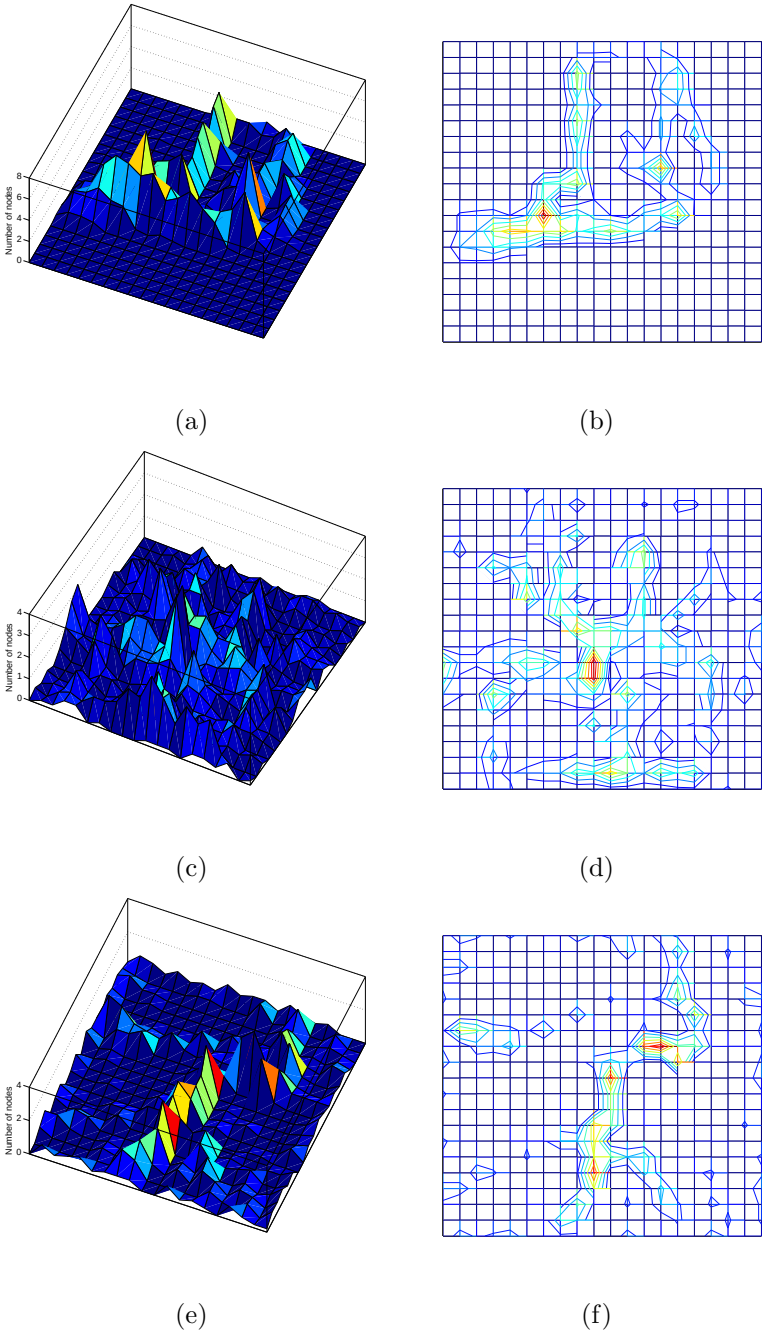


Figure 4.21: Node distribution in the playground and its contour plot respectively with ROI (a & b), NoROI-Path100 (c & d) and Voronoi (e & f). The number of nodes in the simulation is 200.

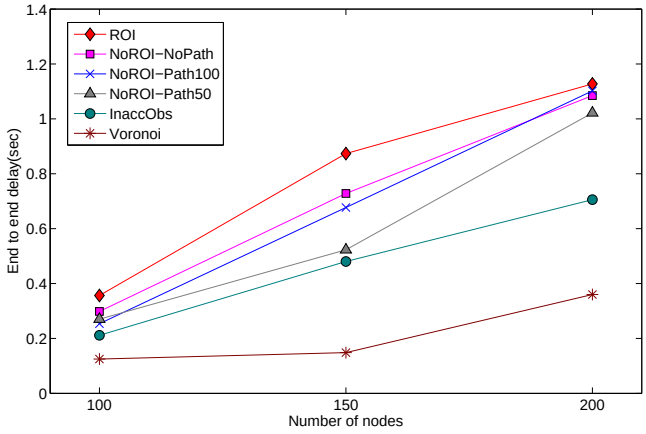


Figure 4.22: End to end delay for packet delivery.

centric node distribution which reflects what happens in the real-world, as it is observed that when there are regions of interest, people go there and node density becomes higher in regions of interest. In the presence of existing roads, people tend to use roads resulting in a greater density along the roadways and this real-world experience is representable through NoROI-Path100 variation. Thus it can be concluded that AMM is flexible enough to cater various situations by considering environmental contexts more realistically while directing node mobility resulting in node distribution similar to that of a real-world scenario.

4.6.2 Protocol Overhead and Packet Delivery Delay

Figures 4.22 and 4.23 represent the end to end delay and routing overhead, respectively. An increased MAC layer congestion causes ROI to encounter higher end to end delay as shown in Fig. 4.22 even though ROI requires a lower path length (Fig. 4.16). Since in the Voronoi model the nodes only move on the Voronoi edges, lower channel congestion is observed and consequently the end to end delay becomes shorter. Having a concentration level of nodes more than that of the Voronoi model and less than that of ROI, the end to end delay of the other variations of AMM is in between.

Figure 4.23 shows the routing overhead encountered by different mobility models. The routing overhead of GPSR has two parts: the beacon overhead and packet forwarding overhead. The beacon interval plays a significant role in the routing performance of GPSR. A larger beacon interval increases the location error while decreasing MAC

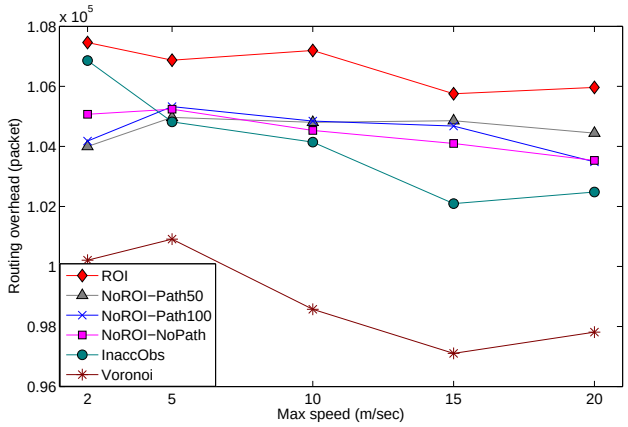


Figure 4.23: Routing overhead.

layer congestion. In the simulation, a beacon interval of 1.0 sec was used for all nodes and in all mobility models. On the other hand the packet forwarding overhead reflects the total number of hops required for routing packets. As observed in Fig. 4.23, ROI and Voronoi exhibits the highest and the lowest routing overhead, respectively. Though the Voronoi model requires a higher average path length (Fig. 4.16) than that of ROI, the success rate of ROI is significantly higher than Voronoi (Fig. 4.17). As a result the total number of forwards becomes higher in ROI than the Voronoi model, and thus ROI shows a higher routing overhead.

4.6.3 Large Area Simulation

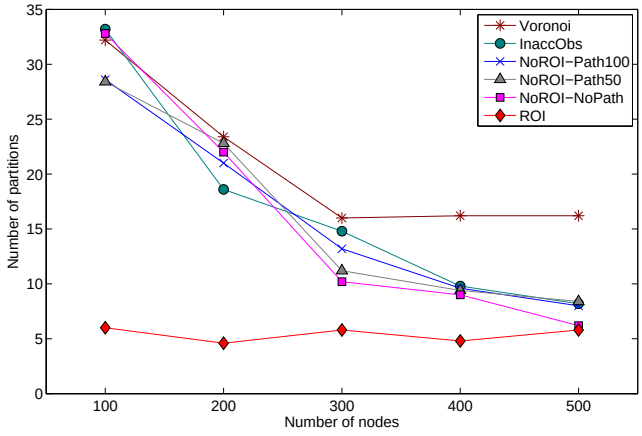


Figure 4.24: Partitions caused by different mobility model in a large area (at 10 m/sec).

A large area simulation was also performed with the same Clayton campus, Monash University. The large area scenario consisted of playground size 2000m × 2000m and the simulation was performed with 100, 200, 300, 400, 500 nodes. Two regions of interest were placed at the upper left and lower right quadrants and the signal attenuation was considered. The other simulation parameters were kept same as those of 1000m × 1000m topology. With a larger area it was possible to have further insight into the partitioning of nodes [105, 104] and how the different mobility models impact on the routing performance in a low node density. Here a few large area simulation results are presented on the number and size of partitions, and packet delivery success rates in Figs. 4.24, 4.25 and 4.26, respectively.

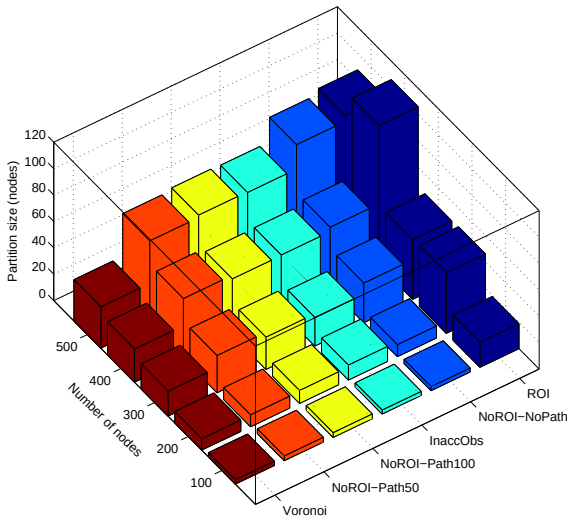


Figure 4.25: Average partition size at different node density (at 10 m/sec).

Since different mobility models result in different node distributions (Fig. 4.21), it is logical that the nodes may form many partitions. In order to estimate the number of partitions, the connectivity of the nodes was measured at 10 second intervals starting from 300 second and averaged results are presented in Figs. 4.24 and 4.25 that represent the number of partitions and average partition size, respectively. Increasing the number of nodes results in different levels of partitions in different mobility models. ROI shows low level of partitioning compared to other mobility models. The number of partitions in the other models drop drastically when the number of nodes in the play-

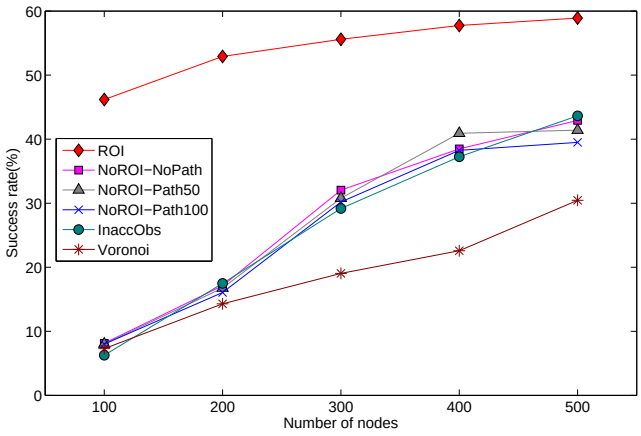


Figure 4.26: Packet delivery success rate in a large area (at 10 m/sec).

ground is 300 or less, beyond which the partition number drops slowly. With fixed and distributed Voronoi edges, the increase in the number of nodes only causes more nodes in the predefined edges. This causes the effective area where the nodes are available to become smaller and a higher clustering of nodes is observed that consequently keeps the partitioning nearly unchanged after 300 nodes, though size of partitions increase (Fig. 4.25). Since ROI exhibits fewer partitions the average partition size is the highest while in the Voronoi model it is the lowest.

Figure 4.26 represents the success rate in packet delivery at different node density. Except for ROI, all the other models exhibit very low success rates when the number of nodes is 100 and 200 as the node density is low [106, 22]. As illustrated in Fig. 4.24 a low node density leads to higher number of network partitions where the nodes in different partitions become unable to communicate with one another, which is the primary reason for low packet delivery success rate reflected in Fig. 4.26. Moreover, the presence of obstacles makes the partitioning more prominent. ROI shows a relatively higher success rate as the nodes become concentrated in the regions of interest and due to experiencing less partitioning effect. When the number of nodes is increased, the success rate increases in NoROI-Path100, NoROI-Path50, NoROI-NoPath and InaccObs more rapidly than that in Voronoi and ROI. With increase of nodes from 400 to 500, NoROI-Path100 and NoROI-Path50 exhibit no change in the success rate as most of the nodes are using roads making the network saturated along the roadways. In Voronoi, the number of edges is much higher than the number of existing roads

causing a lower success rate irrespective of the number of nodes. Because of the higher node concentration in ROI, an increase in the number of nodes results in very small increase in the success rate.

4.6.4 Summary

Though it is well accepted that the mobility model has a significant impact on the performance of routing protocols, here the *level of influence* of different mobility models on the network performance is investigated in detail. It is found that the existence of regions of interest and obstacles has a significant impact on the node mobility which is consequently reflected on performance evaluation metrics. Again, the existence of accessible obstacles makes a medium level difference compared to inaccessible obstacles. However, the path preference details make a relatively small difference. The presence of regions of interest, obstacles and pathways has an impact on the network partitioning which affects the packet delivery success rate. Moreover, signal attenuation due to the presence of obstacles has a significant impact on the network performance.

The primary objective in this chapter is to closely emulate the movement patterns of real-world scenario in directing node mobility in network simulation. It should be noted that the simulation results only demonstrate the impact of the proposed mobility model on the routing protocol rather than evaluating the protocol itself. The results show that there is a significant impact of the choice of mobility model on the routing protocol performance. Ignoring or approximating the environmental features and conditions as in the existing mobility models will lead to a less accurate assessment of the protocols and services.

4.7 Conclusion

The success and efficacy of the simulation in evaluating MANET's performance depends highly on how closely the mobility of nodes resembles the real-world scenario, which the existing models fall short of. In this chapter, a new mobility model along with a scenario generation tool is proposed for MANET, which enables us to create a wide variety of simulation scenarios closely resembling the real-world environment. The computational complexity and the characteristics and impact of the proposed mobility model in the presence of obstacles and pathways at different movement patterns are

analyzed in details. A comparison of the characteristics of the proposed model with the existing ones reveals the significant impact of the choice of mobility model on performance evaluation. The node distribution and partition formation behavior of the models were also analyzed in the presence of different environmental contexts. The AMM model is flexible enough to cater various realistic movement patterns similar to what we observe in the real-world. Since the generated scenario can be easily used in discrete event network simulators, the MobiGen scenario generator will be helpful to other researchers in MANET and wireless communication in general.

As alluded in Chapter 1, the main objective of this thesis is to develop a location service protocol suitable for use in a real-world environment. In this chapter, we have developed a mobility model that considers various environmental factors, through which the real-world can be modeled more accurately while reflecting the environmental impacts more realistically. The analysis shows that the impact the environment has on mobility, leads the nodes to become non-uniformly distributed. Such a distribution that changes in course of time is required to be taken into account while designing and evaluating a location service protocol. Having defined a mobility model capable of simulating real-world movement closely, in the next chapter we develop a scalable location service protocol that can cope with real-world environmental context by considering the realistic node mobility designed in this chapter.

Chapter 5

Hierarchical Adaptive Location Service

Most of the existing location services in mobile ad hoc network use the mobility models where nodes move in a random fashion in free space and mainly focus on the scalability of the services. However, as stressed in earlier chapters the real-world is complex and vastly diverse that makes the existing mobility models inadequate for representing the actual environment properly. Thus in order to design and evaluate a location service applicable to a realistic environment, the initial step is to represent the environment that can reflect the real-world more accurately. Through a novel node mobility model the realistic aspects of an environment has been introduced in the last chapter where it is observed that the environmental characteristics greatly influence the node mobility and produce a non-uniform distribution of nodes in the course of time. Since most of the contemporary location services tend to organize home regions uniformly throughout the simulation area, they are not attractive for a real-world deployment and their performance degrades considerably due to the non-uniformly distributed nodes as observed in Chapter 3. This observation motivates us to investigate and propose a new location service that adapts itself for time varying non-uniformly distributed nodes resulting from realistic node mobility in the presence of various environmental contexts, while still maintaining scalability at the same time. This chapter presents the design philosophy of such a location service, and addresses the major aim of this thesis i.e., to develop a location service which is capable of coping with varying node distribution which takes place in a realistic environment.

This chapter is organized as follows: Section 5.1 gives a brief introduction of the proposed location service, while the detailed design of the service is presented in Sec-

tion 5.2. A scalability analysis is given in Section 5.3. Sections 5.4 and 5.5 provide the simulation environment used for evaluating the location services, and corresponding results, respectively. Finally, Section 5.6 concludes the chapter.

5.1 Introduction

The location services are developed considering four phases: firstly, the location update phase where each node updates its location to the location servers, secondly, the location query phase where a node initiates a query in order to retrieve the location of another node. In the third phase, the location servers maintain the location information stored in their databases in such a way that the most up-to-date information is available and the servers are ready to respond when any query arrives and finally, the location reply phase where the location servers reply to the query initiator with the location of the queried node. However, due to node mobility, managing the location information is a challenging task. There exists a design tradeoff in reducing location update and query costs [21], but as discussed in the literature review in Chapter 2, among the existing location services multi-level structured location services have the right balance in the update and query costs while maintaining better scalability over the other schemes. This makes a multi-level structured approach most appropriate for location services and forms the basis of our proposal in this thesis.

The proposed location service comprises novel techniques in order to maintain scalability while coping with a non-uniform node distribution. In our approach, any type of hash function based home region assignment method is not used; rather a home information dissemination procedure is developed. Moreover, home region shifting, creating and destroying methodologies have been incorporated that can take place on-demand basis. Through these methodologies the proposed location service can avoid the empty home problem caused by non-uniformly distributed nodes and consequently the success in the location update and query increases. Furthermore, the home region selection is hierarchically organized and the need for message broadcasts is eliminated. The analytical model presented in Section 5.3 validates the scalability and the simulation results in Section 5.5 demonstrates a superior performance over other existing location services in different environments and node distributions.

5.2 Hierarchical Adaptive Location Service Scheme

In this section the basic structure and the theoretical underpinnings of our proposed location service protocol called the Hierarchical Adaptive Location Service (HALS) are presented. The formulation of HALS begins with dividing the whole deployment area into a grid of squares. This dividing procedure is outlined in Section 5.2.1. The home region selection and home information dissemination mechanisms are presented in Sections 5.2.2 and 5.2.3, respectively. As HALS does not use hash functions, the nodes need to communicate with the home regions in a special way which is presented in Section 5.2.4. In order to address the empty home problem and adapt with non-uniform node distributions, a novel home region *shifting* mechanism is incorporated in HALS, which ensures non-empty home regions at all times. Section 5.2.5 provides a detailed methodology for the shifting home regions. Due to node mobility when a sub-area becomes empty, it is necessary to introduce a home *destroy* methodology. In addition, whenever the nodes enter an empty sub-area where there is no existing home, the creation of a home is mandatory to ensure uninterrupted service. In this way, homes can be destroyed and created on demand basis and detailed mechanisms for these processes are given in Sections 5.2.6 and 5.2.7, respectively. Finally, Section 5.2.8 provides HALS's location update, and Section 5.2.9 presents the location query methodology in details.

5.2.1 Generation of Grids

Like other contemporary and efficient hierarchical location services (e.g., HIGH-GRADE [21] and HLS [22]), HALS also divides the whole region into grids. The smallest square regions are termed $Level - 0$ (L_0) cells. The relationship between the transmission range and the size of an L_0 cell should be such that any two points in an L_0 cell must lie within each others communication range as shown in Fig. 5.1. As a result, for the transmission range r_t the width of an L_0 cell a is given by

$$\begin{aligned} a &= r_t \cos 45^\circ \\ &= \frac{r_t}{\sqrt{2}} \end{aligned} \tag{5.1}$$

After generating the L_0 cells, multiple cells are grouped hierarchically into higher level cells. Since empty home regions reduce the update and query success rate, in this

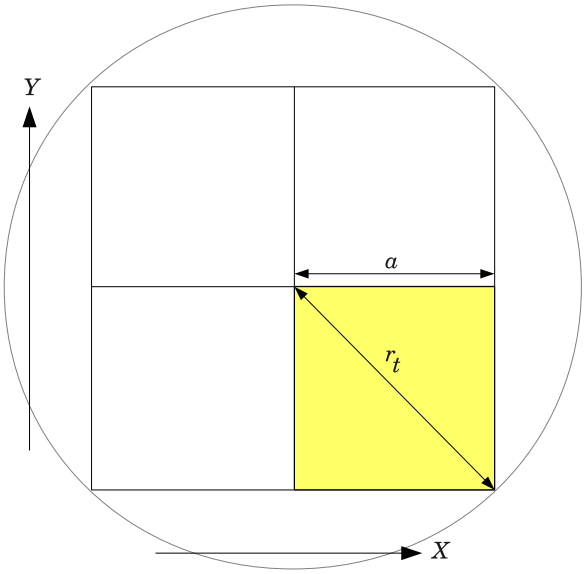


Figure 5.1: Determining the size of L_0 (smallest) cell.

model we intend to shift the home regions on demand, so that the home regions do not become empty if such a situation arises. An L_1 cell is composed of a higher number of L_0 cells, so that the shifting mechanism does not cause overhead at higher levels of the hierarchy as HALS will require shifting of low level home regions more frequently than higher level homes. Unlike other location services, the hierarchy generation includes two different grouping mechanisms, i.e., $\kappa \times \kappa$ L_0 cells create an L_1 cell, and at subsequent levels 2×2 L_i $(i > 0)$ cells create an L_{i+1} cell. In this hierarchy development procedure, a low value of κ will result in increased overhead for shifting, creating and destroying homes due to increased levels in the hierarchy. On the other hand a higher value of κ will cause fewer levels in the hierarchy and thus an increasing distance effect. Through repetitive experiments we concluded that $\kappa = 5$ (Fig. 5.2) is the best suitable value for κ, to obtain a balance between home shifting, creating and destroying overheads and the distance effect. It should be noted that all cells of a particular level are mutually exclusive, i.e., each L_i cell must be the member of exactly one L_{i+1} cell. The four L_{i-1} $(i > 1)$ cells that compose an L_i cell are called the *sibling* cells within L_i.

The lower left point of the grid is assigned to the origin of the co-ordinate system with X and Y values increase towards the right and top, respectively. An ID number is assigned to each cell in each level and this ID is incremented in a left to right, bottom

to top fashion. Since a node knows its location, it can easily identify in which L_i cell it resides. Let the width of the playground be W and node A's location is (x_A, y_A).

The maximum number of L_0 cells in a row of L_0 cells is

$$max_0 = \left\lceil \frac{W}{a} \right\rceil \tag{5.2}$$

The maximum number of L_1 cells in a row of L_1 cells is

$$max_1 = \left\lceil \frac{W}{a \times \kappa} \right\rceil \tag{5.3}$$

The maximum number of L_2 cells in a row of L_2 cells is

$$max_2 = \left\lceil \frac{W}{a \times \kappa \times 2} \right\rceil \tag{5.4}$$

Similarly, the maximum number of L_i $(i \geq 1)$ cells in a row of L_i cells is

$$max_i = \left\lceil \frac{W}{a \times \kappa \times 2^{i-1}} \right\rceil \tag{5.5}$$

A's L_0 cell ID is calculated as:

$$CellId_0(A) = max_0 \times \left\lfloor \frac{y_A}{a} \right\rfloor + \left\lfloor \frac{x_A}{a} \right\rfloor \tag{5.6}$$

Similarly, A's L_1 and L_2 cell IDs are calculated as:

$$CellId_1(A) = max_1 \times \left\lfloor \frac{y_A}{a \times \kappa} \right\rfloor + \left\lfloor \frac{x_A}{a \times \kappa} \right\rfloor \tag{5.7}$$

$$CellId_2(A) = max_2 \times \left\lfloor \frac{y_A}{2 \times a \times \kappa} \right\rfloor + \left\lfloor \frac{x_A}{2 \times a \times \kappa} \right\rfloor \tag{5.8}$$

By, generalizing for $i \geq 1$

$$CellId_i(A) = max_i \times \left\lfloor \frac{y_A}{2^{i-1} \times a \times \kappa} \right\rfloor + \left\lfloor \frac{x_A}{2^{i-1} \times a \times \kappa} \right\rfloor \tag{5.9}$$

Thus, this ID scheme and hierarchy generation mechanism ensures a specific cell at L_i to be a member of only a single cell at L_j where $j > i$.

5.2.2 Selection of Home Region

Most of the grid based location services use uniform hash functions known to all nodes. These schemes perform well in uniform node distribution but due to the mobility of

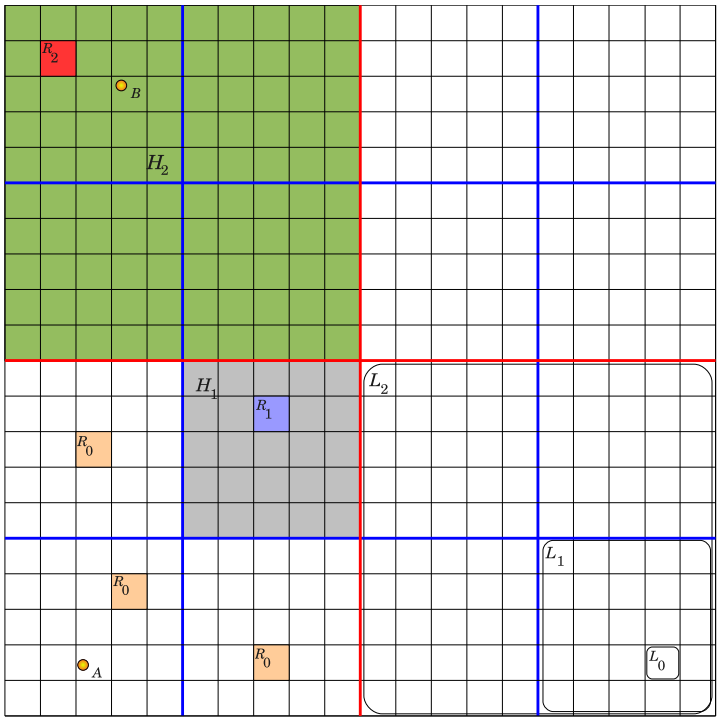


Figure 5.2: Generation of grids where A, B are nodes, R_0, R_1 and R_2 are ranked homes at different levels.

nodes in a realistic way where empty cells are frequent and the node distribution is not uniform for a long duration, a significant amount of location update and query failure is expected. This motivates us not to use hash functions to map the home regions for the whole lifetime of HALS. The term "home" is used to denote the server region for a subset of nodes. To explain the working principle of HALS two different terms are introduced: 1) $Rank - i$ home (R_i home), and 2) $Level - i$ home (L_i home).

$Rank - i$ **home (R_i home):** For each L_1 cell one of the $\kappa \times \kappa$ L_0 cells is assigned as $Rank - 0$ home (R_0 home); for each L_2 cell one of the four R_0 homes is assigned as an R_1 home and this process continues for each level. Formally, For each L_{i+1} cell one of the four R_{i-1} homes is assigned as an R_i home. A $Rank - i$ home (R_i home) is defined as an L_0 cell, nodes within which act as server for nodes within its L_{i+1} cell. An R_i home will act as R_0 home for the nodes within the same L_1 cell, act as R_1 home for nodes within the same L_2 cell and so on up to R_i home for nodes within the same L_{i+1} cell. For instance, as shown in Fig. 5.2, R_2 home in the top left corner acts as an R_0 home for L_1 cell within which it resides, as

R_1 home for L_2 cell (marked in green) and as R_2 home for L_3 cell (the whole playground in this case). Thus, nodes in R_i know "in which cell nodes within the L_{i+1} cell boundary resides" at different levels of accuracy.

Let a node S working as a server in a cell with home R_i and another node A resides in its same L_j cell ($1 \leq j \leq i+1$), S stores A's location in the following format:

$$Data(S, A) = \begin{cases} L_{j-1} \text{ for } j > 1; \\ x_A,\, y_A \text{ of } A \text{ for } j = 1. \end{cases} \tag{5.10}$$

This implies, an R_i home works as a server for all nodes residing within its same L_{i+1} cell at different levels of precision according to (5.10). This scheme keeps a more accurate location of nearby nodes and a less accurate location for farther nodes. Since the accurate locations become outdated sooner, the nearby nodes need to update their locations more frequently than the farther nodes.

From the implementation point of view, each server node which resides in an R_r home, where $0 \leq r \leq \psi$ (ψ represents the highest rank), maintains a database (a two dimensional array of adjustable size) having the following format:

$Table[r][]$: here Table[i][], where $0 \leq i \leq r$, contains the location of nodes as

$$Table[i] = \begin{cases} \text{a list of tuples} < \text{node\_id , } L_i \text{ cell ID the node resides in} > \\ \quad \textit{(for } i > 0 \textit{ and for those nodes present within } R_r\textit{'s } L_{i+1} \textit{ cell)} \\ \text{a list of tuples} < \text{node\_id , absolute location of node} > \\ \quad \textit{(for } i = 0 \textit{ and for those nodes present within } R_r\textit{'s } L_1 \textit{ cell)} \end{cases} \tag{5.11}$$

According to (5.11), a server node stores the absolute location of nodes residing in the same L_1 cell in the database's L_0 *ID space* and stores the L_1 cell IDs of nodes residing in the same L_2 cell in the database's L_1 *ID space*, up to the rank of the home where the server resides in.

***Level* $-$ i home (L_i home):** An L_i home for a node is defined as an L_i cell in which the node's R_i home resides in. Similar to the location storing scheme followed by *ranked* nodes, each node also stores its home information in terms of L_i homes.

$$Home_i(A) = L_i \text{ Cell ID of } L_i \text{ home;} \tag{5.12}$$

This home information keeping scheme ensures that a node does not need to be notified of its L_i home as long as its R_i home stays inside the boundary of the corresponding L_i cell.

As shown in Fig. 5.2, for any arbitrary node A, R_0 is an L_0 cell that acts as its L_0 home. An L_1 cell called H_1 (marked as grey) is the L_1 home of A. Within H_1, R_1 is actually working as server at $rank-1$ for A. An L_2 cell called H_2 (marked as green) is A's L_2 home. Within H_2, R_2 is the $rank-2$ home for A. It should be noted that A does not have precise knowledge about the location of R_2, it only knows that R_2 is within the boundary of H_2. Any non-uniform node distribution will cause the ranked cells to shift, but as long as R_2 is within H_2, A does not need to update its L_2 home information.

Again, for another node B, R_2 acts as B's $rank-0$, $rank-1$ and $rank-2$ homes. Since, R_2 resides within B's L_1 cell, R_2 acts as B's L_0 home, similarly the L_1 cell within which R_2 resides is B's L_1 home and H_2 is B's L_2 home. Thus, the nodes in R_2 know "in which cell nodes within its L_3 cell boundary resides in" at different level of accuracy. Figure 5.3 shows an example of how different cells are designated as ranked cells at different levels within a Level-3 hierarchy.

Algorithm 5.1 initialize

Require: u as node,
$seed_j^i$ as seed for j-th cell in i-th level where $1 \leq i \leq \psi + 1$
and $0 \leq j \leq max$(cell ID in i-th level),
DB as database.

1: **for** each j-th cell in each i-th level l_j^i **do**
2: $h_j^i \leftarrow$ randomly select the ID of an L_{i-1} cell within l_j^i with $seed_j^i$
3: **end for**
4: **for** each level i $(1 \leq i \leq \psi + 1)$ **do**
5: **if** u resides in l_j^i **then**
6: DB.home_id$[i-1] \leftarrow h_j^i$
7: **end if**
8: **end for**

Initially the nodes are assumed to be distributed throughout the area, and every node is equipped with an initialization function that maps each node with its home regions based on the *initial position* of the nodes during deployment. It should be

noted that unlike the hash function used by other methods, our initialization function (Algorithm 5.1) uses nodes initial deployment position rather than their ID, and after initialization there is no use for this function any more.

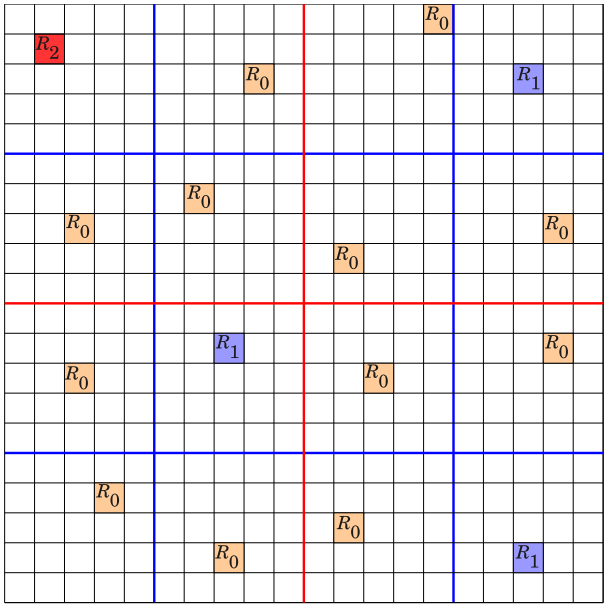


Figure 5.3: An example of assigned ranked homes.

5.2.3 Home Information Dissemination

Since HALS adopts a home region shifting mechanism to handle empty homes caused by the non-uniform node distribution, it is not possible to predict where the home regions will be with respect to time, as they will change their position monitoring node density to their neighboring cells. For this reason no static hash function is applicable throughout the lifetime after deployment. Thus to facilitate the shifting of home region, we employ a home region information disseminate scheme that will make the nodes aware of the home regions of different levels in the course of time.

HALS disseminates home information through the routing protocol's *beacons* (also termed as *HELLO* packet). Since each node in a geographic routing protocol sends beacons to its neighbors, HALS includes the following additional information in the beacons:

- *home_id*[]: an array of $\psi+1$ elements (indexed from 0 to ψ) where ψ is the highest

rank (the whole simulation area is a cell of level $\psi + 1$), $home\_id[i]$ contains the ID of L_i home.

- $freshness$: a time stamp that indicates how recent the $home\_id[]$ information is.
- no_of_nodes: the number of nodes in the beacon generator's L_0 cell.

Each node also stores these values in its database depending upon the information freshness and disseminates the most updated values it has. On receiving a beacon from a neighbor, a node updates its database (DB) if

$$freshness_{beacon} > freshness_{DB} \tag{5.13}$$

Only the nodes residing in R_i home assign $freshness_{beacon}$ with the current time stamp. This implies the latest home information is actually generated from a Ranked cell, and the other nodes only update their database and propagate it. This propagation is kept limited within the area of an L_1 cell. So when an R_0 home is shifted within an L_1 cell, the beacon packets notify others of the newly assigned home cell. If a node receives a beacon from any node that does not reside in its same L_1 cell, it does not consider updating its home information database.

As described in Section 5.2.2, the $home\_id$ values also adopt a relative cell ID scheme similar to how the home servers store node locations at different precisions. For an L_i home, a node stores the L_i cell ID in which its R_i home resides. Thus even if the R_i home is shifted, as long as it remains within its L_i cell boundary that node does not need to be notified.

Since nodes propagate the $home\_id$ through beacons, when a node crosses an L_i boundary from cell c_1 to c_2 it receives beacons from any node in c_2 and updates its home information (when a node crosses L_i cell boundary it needs to update its database for home value from level 0 to L_{i-1}) and starts propagating this updated data. Procedures for attaching home information to beacons and updating the database upon receiving beacons, are outlined in Algorithms 5.2 and 5.3, respectively.

5.2.4 Forwarding Packet to R_i home:

In order to facilitate home region shifting, no static hash function can be used in HALS. Hence it is not possible to predict the position of the R_i homes directly. But

Algorithm 5.2 sendBeacon

Require: u as node
1: **if** u resides in ranked cell R_i **then**
2: Beacon.freshness ← NOW
3: Beacon.no_of_nodes ← count only those neighbors that reside in R_i
4: Beacon.home_id[] ← DB.home_id[]
5: **else**
6: Beacon.no_of_nodes ← count only those neighbors that reside in its L_0 cell
7: **end if**
8: Transmit Beacon

Algorithm 5.3 receiveBeacon

Require: u as node
1: c_{orig} ← Beacon originator's L_1 cell id
2: **if** u's L_1 cell id = c_{orig} **then**
3: **if** u does not resides in ranked cell R_i **then**
4: **if** Beacon.freshness > DB.freshness **then**
5: DB.home_id[] = Beacon.home_id[]
6: **end if**
7: **else**
8: update node density map
9: **end if**
10: **else**
11: discard this Beacon.
12: **end if**

as mentioned earlier, within an L_i cell the highest ranked home works at least at R_{i-1} which keeps track of all the nodes within its L_i cell at different levels of precision. As a result, for a location update, query and other message transfer for creating and destroying homes, a node needs to send packets to the highest ranked home within an L_i cell. Then the highest ranked home can respond by replying or redirecting the packet to a higher or lower level in hierarchy based on the stored information in database. For a node A to send such a packet, it only uses the L_i cell ID, but does not need to know the precise information of the highest ranked cell in terms of location or ID. Thus A sends a packet with the following information:

$<$ Level $= i$, ID of the L_i cell $>$

When the packet reaches any node B inside destination L_i cell, the highest ranked home within L_i is actually residing within the L_{i-1} home of B. So, B modifies the tuple as:

$<$ Level $= i - 1$, ID of L_{i-1} home of $B >$

and forwards it to the center of the L_{i-1} home of B. In this way, when the packet

arrives inside the boundary of an L_1 cell where the ranked home is working as R_0, the packet is forwarded towards the ranked home as every node within an L_1 cell knows the location of its R_0 home. After that any node within the corresponding R_0 home can receive that packet. In this way a packet can be sent to an R_{i-1} home, the highest ranked cell within an L_i cell, with only the cell ID information of the respective L_i cell. It should be noted that forwarding in this way will select a non-optimal path from the sender to destination. However, through this way by using the higher level cell ID, it is possible to route a packet to the ranked home which changes its position based on node density. Algorithm 5.4 presents this packet forwarding procedure.

Algorithm 5.4 forwardPacket

Require: u as node, C as target cell ID of level i
1: **if** a packet arrives at a ranked home **AND** $i = 0$ **then**
2: receive the packet
3: **end if**
4: set packet direction to the center of C and send to the appropriate neighbor u
5: **if** u resides in C **then**
6: $C \leftarrow$ cell ID of u's L_{i-1} home
7: $i \leftarrow i - 1$
8: **end if**
9: call forwardPacket with $< i, C >$

Figure 5.4 explains this forwarding method with the aid of an example. Here an arbitrary node S needs to send a packet to the ranked home of an L_2 cell C (marked in yellow). There are one R_2 and three R_0 homes within C, where R_2 is the highest ranked home and responsible for the whole cell. This R_2 home is also working as an R_1 home for nodes within C, and S actually needs to send the packet to R_2 while only knowing that it resides within C. Thus S sends the packet towards the direction of the center of C. While being forwarded, the packet is received by a node A within C. Since the smallest common cell of R_2 and A is smaller than that of R_2 and S, A has more precise location information about R_2 as R_2 is actually working as an R_1 home for A. Thus A modifies the packet and sets the direction towards the center of its L_1 home H_1^A (marked in green), and forwards. In a similar manner the packet is received by any node B within the L_1 cell H_1^A. The R_2 home is working as an R_0 home of B, and it knows R_0's actual location (through the beacons). Thus B modifies the packet once more and forwards it to its R_0 home (R_2 home for S). Any node within this R_0 home can receive this packet. In this way through cooperative forwarding and by using

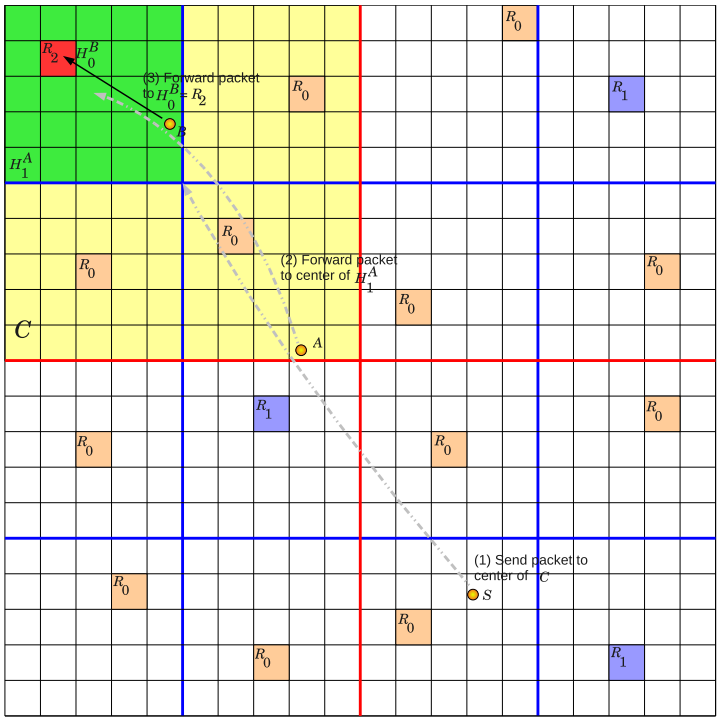


Figure 5.4: Sending packet to R_2 home with L_2 cell's ID.

an L_i cell ID a node can forward a packet to the highest ranked cell (Within an L_i cell, the highest ranked home works at least at R_i).

It should be noted that by mentioning "a node sends a packet to its L_i home", we mean the packet is actually being received by the node's R_i home (which resides in its L_i home). Similarly, "a node sends a packet to its R_i home" means that the node is using its L_i home ID to send the packet to its R_i home.

5.2.5 Shifting Home Region

As mentioned earlier, a non-uniform distribution of nodes creates the empty home problem when the location servers move away from their designated area. As a result the nodes fail to locate the location servers and consequently the performance of the location service degrades considerably. To support non-uniform node distribution a home region shifting mechanism has been adopted. In this method a home shifts to a region which is more populated with nodes. This methodology ensures the homes to be non-empty all the time and addresses the empty home problem faced by the other location services.

For each R_i home, a leader node is selected. Here, a node is selected as leader if it has the smallest ID within an R_i home. This leader is colored as *Red*. When a *Red* node moves away from its designated L_0 cell, it marks itself as a normal node and sends a message to nodes within its old L_0 cell so that a new *Red* node can be elected depending upon the ID space.

The *Red* node is actually responsible for making the decision about shifting the home region. As every node sends with beacons the number of nodes present in the L_0 cell in which it resides, the *Red* node stores this information and gets an assessment of how the node distribution within the L_1 cell is changing. When it finds the number of nodes in the current L_0 home has reduced to one, it decides to shift the home to any of the neighboring L_0 cells within the same L_1 cell that has the highest number of nodes. This type of shifting is termed *Soft shifting*. The *Red* node then handovers all of its server responsibilities to the nodes of the newly selected L_0 home cell, sends a message to elect a *Red* node and finally makes itself a normal node. Subsequently, the newly selected L_0 home starts attaching latest information about the new home with beacons and within a short time all nodes in the L_1 cell update their database with the updated data. It should be noted that propagating information through beacons is not as fast as broadcasting. But the nodes do not need this updated information immediately as the shifting commences when the old home is not completely empty and is still capable of responding to queries during this transition period. Algorithm 5.5 outlines this soft shifting mechanism.

Algorithm 5.5 softShift

Require: u as node
1: **if** u is *Red* **then**
2: **if** u's L_0 cell c_{old}'s number of nodes = 1 **then**
3: **for all** cells $c \in c_{old}$'s neighboring L_0 cells within its L_1 cell **do**
4: $c_{new} \leftarrow c$ with maximum number of nodes.
5: **end for**
6: u sends *elect_red* message to c_{new}
7: u transfers server responsibility to c_{new}
8: u transforms itself to normal
9: **end if**
10: **end if**

Figures 5.5(a) and (b) explain how the soft shifting takes place through an example. In Fig. 5.5(a), the *Red* node within a ranked cell R_i^{old} finds the cell has too few nodes and decides to shift the home. In this case, it concludes by analyzing its neighbor table

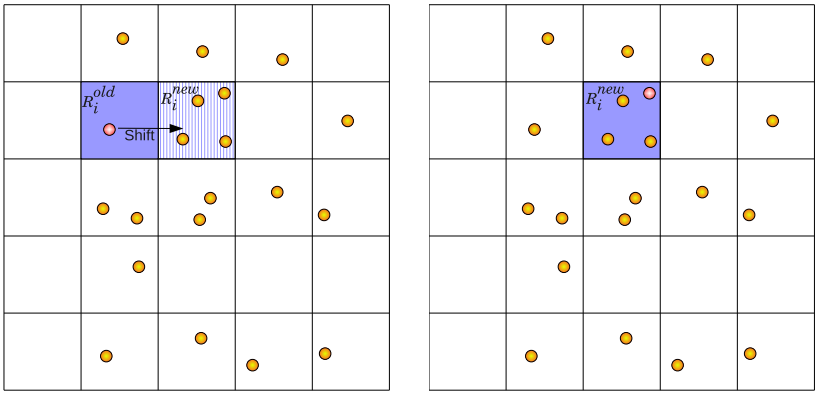


Figure 5.5: Soft shifting of a ranked home, (a) before shifting, and (b) after shifting.

that, the R_i^{new} cell contains the highest number of neighbors among the 8 surrounding L_0 cells of R_i^{old}. Thus the *Red* node sends a message to elect a new *Red* node within R_i^{new}, transfers its database there, and makes itself as a normal node. After the shifting is complete, the organization of the home region becomes as shown in Fig. 5.5(b).

5.2.6 Destroying Home Region

Soft shifting of the R_i home cell is kept restricted to within the boundary of an L_1 cell. Shifting the home region will keep the home region non-empty. However, if all the nodes start to move out of an L_1 cell, the R_i home will be the last to move as the home region shifts to the area where the node density is higher. If this happens, the R_i home cell, notifies its R_{i+1} home that the L_1 cell in which R_i was residing in is going to become empty. When there is no remaining nodes in the corresponding L_1 cell, the R_i home is simply destroyed.

A simple home destroying mechanism is presented through an example shown in Fig. 5.6. The last node previously working as a *Red* node at R_1, crosses the boundary of an L_2 cell denoted by C (marked in yellow), notifies its R_2 home that cell C is about to become empty, and destroys the R_1 home. Since the R_2 home is the highest ranked home in the whole L_3 cell, it keeps track that the L_2 cell C is empty. Storing this information is important and will be utilized when creating a new home within C if it becomes populated again. The current situation is simple, because within the L_3 cell the highest possible rank is R_2, and this destruction takes place for a home working

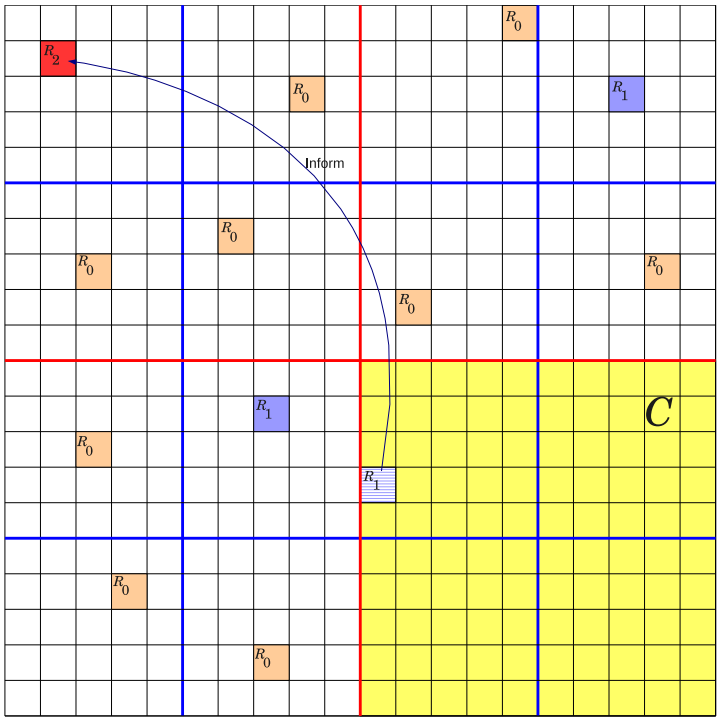


Figure 5.6: Destroying home region.

at rank R_1 where the L_2 cell C is about to become empty. However, if the R_2 home needs to be destroyed, the home destruction mechanism is more complicated and is explained in Figs. 5.7(a) and (b).

Figure 5.7(a) presents a situation when the highest ranked home R_2 finds that its L_1 cell denoted as C (marked in yellow) is about to become empty. Within R_2's L_2 cell the other homes are working at rank 0. As a result R_2 can not simply destroy itself, and stop from doing its tasks as an R_2 home, rather it needs to transfer its responsibilities to another home. In this case, R_2 selects any of the homes in the 3 siblings of C (e.g., C' in Fig. 5.7(a), marked in grey), within the current L_2 cell as a potential candidate to delegate its duty. R_2 transfers its database to the selected R_0 home and notifies the other two R_0 homes of this transfer, and destroys itself. Then the selected R_0 home upgrades its rank from R_0 to R_2 and keeps track that C is empty. Afterwards, if the new R_2 finds that C' is going to become empty, it will transfer its responsibility to one of the remaining R_0 homes within its L_2 cell. This type of responsibility transfer ensures that as long as the entire L_2 cell is not empty, transferring R_2 will be constrained within the current L_2 cell.

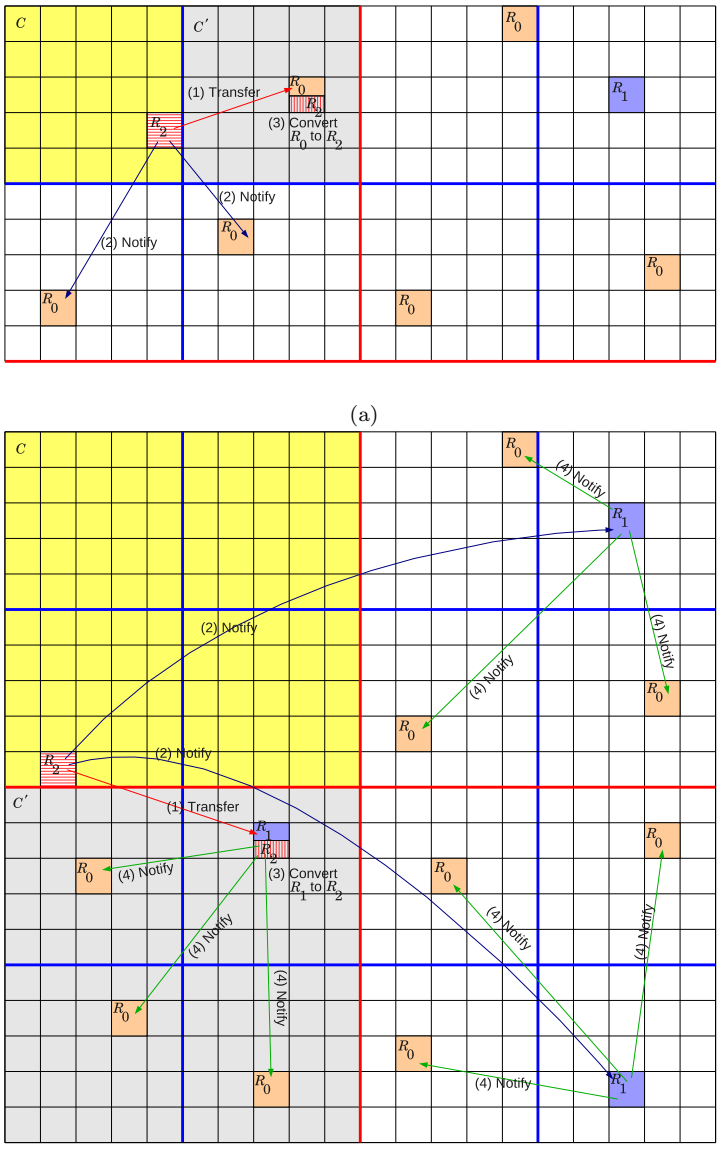


Figure 5.7: Hard shifting of homes when R_2's (a) L_1 cell becomes empty, and (b) L_2 cell becomes empty.

Figure 5.7(b) shows a different situation where the whole L_2 cell C (marked in yellow) is going to be empty. Since there is no other home in C, this time R_2 selects one of the siblings of C within the L_3 cell as a potential destination to transfer its responsibilities (R_2 knows that all of the siblings of its current L_1 cell are empty because R_2 is the highest ranked home within C and keeps track of empty L_1 cells when the destruction of homes or hard shifting takes place). It selects a sibling L_2 cell C' as the destination, and transfers its database to the R_1 home (highest ranked home) within it. At the same time R_2 notifies the other two R_1 homes residing within the same L_3 cell. Each of the three R_1 homes recursively notifies the R_0 home within their own L_2 cells. The selected R_1 home to whom the database is transferred upgrades its rank from R_1 to R_2 and becomes the new R_2 home for the whole L_3 cell. The new R_2 also keeps track that C has become empty. This change in home information will be subsequently propagated to all the nodes through periodic beacons as described in Section 5.2.3.

Algorithm 5.6 destroyHome

Require: u as node
1: **if** u is *Red* at Rank r, where $0 \leq r < \psi$ **then**
2: **if** u finds current L_{r+1} cell empty **then**
3: inform EMPTY to R_{r+1}
4: **else if** u finds current largest empty cell at L_i where $1 \leq i \leq r$ **then**
5: transfer database to any one of R_{i-1} within L_{i+1}
6: notify selected R_{i-1}
7: upgrade Rank of the selected R_{i-1} to R_r
8: notify remaining R_{i-1} within L_{i+1}
9: inform EMPTY to R_r
10: **end if**
11: **end if**

Algorithm 5.7 notify

Require: u as node
1: **if** u is *Red* at Rank i, where $i > 0$, **AND** receives notification **then**
2: notify non-empty siblings within u's L_{i+1} cell
3: **end if**

The home destroying mechanism is summarized in Algorithm 5.6. An arbitrary home R_r ($0 \leq r < \psi$), needs to be destroyed if either of the following two situations arises:

1. R_r finds that the cell L_{r+1} within which R_r resides is going to be empty: According to the home assignment scheme, in a L_{r+1} cell, the highest ranked cell is at

least of rank r. This implies, the highest ranked home within a cell needs to be destroyed. In this case the *Red* node in R_r sends a message to the R_{r+1} home, informing that its current L_{r+1} cell is empty and then simply destroys itself from working as a home. This situation is shown in Fig. 5.6.

2. R_r finds that, its largest empty cell in which it resides in is a L_i cell, where $1 \leq i \leq r$: This means that the L_{r+1} cell within which R_r resides is not fully empty. Rather a sub-region within the L_{r+1} cell is empty and there are other R_k homes, where $k < r$, still active. As a result, R_r can not simply destroy itself from working as home, rather it transfers its duty to a lower ranked home. Thus R_r selects any one of the non-empty siblings of its L_i cell to continue its responsibility. It transfers its database to the selected sibling's R_{i-1} home which upgrades its rank from $(i-1)$ to r and notifies this rank upgrade to the other siblings where the R_{i-1} home are present. Each of these R_{i-1} homes notifies the lower ranked homes within the same L_{i-1} cell. This process continues towards the lowest level in the hierarchy until R_0 home receives the information as outlined in Algorithm 5.7. It should be noted that, as long as the R_r home is within its L_i cell, where $1 \leq i \leq r$, this recursive notification towards the lowest level in the hierarchy is bounded within the corresponding L_i cell. Moreover, the new R_r keeps track that the L_i cell where old R_r was, is now empty. Subsequently, updated home information is propagated through periodic beacons and every node becomes aware of the new arrangement of homes.

   For keeping track of empty sibling, each R_r home $(0 < r \leq \psi)$, maintains a table (two dimensional array) and keeps this information in the following format:

$$EmptySibling[i][k] = k\text{-th sibling within } L_{i+1} \text{ cell, for } 0 < i \leq r \text{ and } 0 < k \leq 3. \tag{5.14}$$

   as an L_i cell has 3 siblings within its L_{i+1} cell.

   This shifting process ensures that an R_r home $(r > 0)$, will always try to keep itself within the smallest cell as long as it is not fully empty. For instance, an R_5 home will try to keep itself within an L_2 cell. When this L_2 cell becomes empty, R_5 will keep the shifting limited within L_3 cell as long as the L_3 cell is not empty

and so on. In this way the overhead in shifting is kept to a minimum. This type of home destroying and shifting is termed as *Hard shifting*.

Through these examples and the algorithms presented in this section it is clear that the hard shifting of a home tries to constraint the transfer within the lower level before moving to a higher level. At the same time it is ensured that a higher level home keeps track of whether its siblings are empty or not. This information is needed for creating home regions and is explained in the following section.

5.2.7 Creating Home Region

When a node crosses the boundary of an L_i cell ($i \geq 1$), it expects to get the appropriate home information for residing in the new cell through the beacon packets originating from the nodes already present there. It should be noted that when a node u crosses the boundary of an L_i cell from c_{old} to c_{new}, it also crosses all the L_j cell boundaries where $j < i$. In this case the following situations may arise:

1. There exist nodes in c_{new} whose beacons u listens to and updates its database and there is no need to create a home.

2. u did not hear any beacon within c_{new}, because either

   - there is no node in c_{new}. So u does not have any idea about homes from L_0 to L_{i-1}. Since its L_i home information is still up to date, it queries about c_{new} whether c_{new} is empty (while destroying an R_i home the R_{i+1} home is notified). Upon receiving the reply that c_{new} is empty, u simply makes itself (the L_0 cell) R_{i-1} home and starts acting as a server. Or,
   - nodes are already there but are not near enough for receiving the beacons. In this case u queries its L_i home about the status of the c_{new}. When it receives the reply that c_{new} is not empty, it starts a unicast query at fixed intervals to the center of cells from L_1 to L_{i-1} asking about homes at L_0 to L_{i-1} within c_{new}. Since there exist nodes in c_{new}, eventually the request will be received by a node and that node replies with the home information. Depending upon the content of the reply, u can conclude about the missing home information, converts itself to a server and upgrades itself to the corresponding rank.

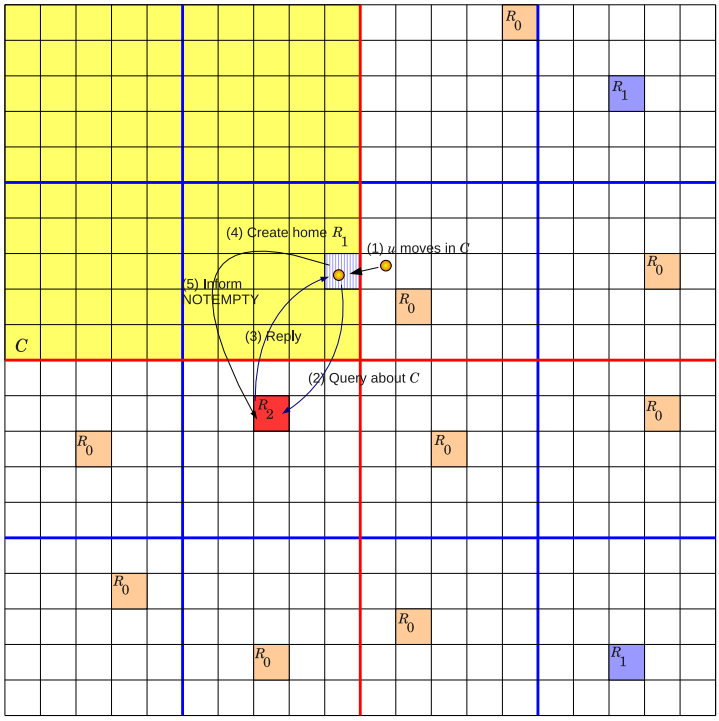
The newly created home at R_i then notifies its R_{i+1} home that the current L_{i+1} cell is NOTEMPTY. Whenever a home is created in this way at rank i, where $i > 1$, it also keeps track that its other three L_i sibling cells within the same L_{i+1} cell are empty. This procedure ensures the highest ranked home to keep track the status (whether empty or not) of its siblings within a cell.

Algorithm 5.8 createHome

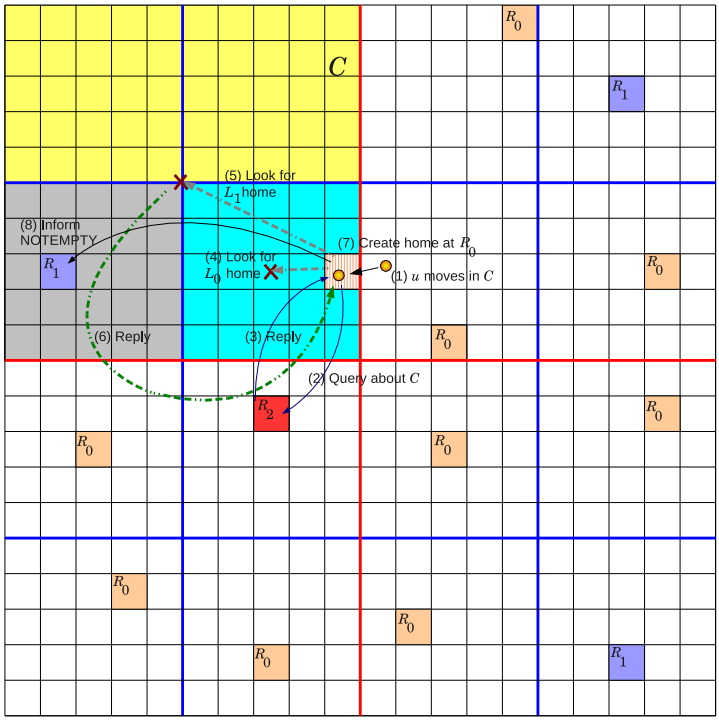
Require: u as node
1: **if** u crosses largest cell C_l at level l, where $1 \leq l \leq \psi$ **AND** does not receive beacon **then**
2: u queries to R_l whether C_l is empty
3: **if** reply contains C_l is empty **then**
4: convert u to *Red*
5: make current L_0 cell home at R_{l-1}
6: inform "C_l NOTEMPTY" to R_l
7: **if** $l > 1$ **then**
8: mark the 3 sibling cells within u's C_l as EMPTY.
9: **end if**
10: **else**
11: **repeat**
12: u queries about L_i, where $1 \leq i \leq l$
13: **until** reply received /* if received at L_1 there is no need to create home as R_0 still exists within L_1 but out of communication range */
14: **if** reply is received at level k where $2 \leq k \leq l$ **then**
15: convert u to *Red*
16: make current L_0 cell home at R_{k-2}
17: inform "current L_{k-1} NOTEMPTY" to R_{k-1}
18: **if** $k > 2$ **then**
19: mark the 3 sibling cells within u's L_{k-1} cell as EMPTY
20: **end if**
21: **end if**
22: **end if**
23: **end if**

Algorithm 5.8 outlines the home creation mechanism and the elaboration is presented through examples in Figs. 5.8(a) and (b). In Fig. 5.8(a), a node u crosses the boundary belonging to an L_2 cell at the highest level, denoted by C. Since there is no node in C, u does not receive any beacon, and thus becomes unaware of the home regions of the corresponding L_2 cell C. Thus u needs to retrieve information for the L_0 and L_1 homes because u's L_2 home information is still valid.

When the cell C became empty, the R_2 home was notified and R_2 knows the status of C. Therefore, u queries its L_2 home. The query is eventually forwarded to the corresponding R_2 home following the method described in Section 5.2.4. After receiving a reply from R_2, u becomes aware that C was empty, and converts the L_0 cell



(a)



(b)

Figure 5.8: Creation of a home when a node enters an empty (a) L_2 cell, and (b) L_1 cell.

within which it resides in to an R_1 home - the highest rank in an L_2 cell, and notifies R_2 that C is no longer empty. At the same time, u marks the other three L_1 sibling cells within C as *empty*, so that those sibling cells can be assisted to create the home next time.

Figure 5.8(b) presents another scenario where node u crosses the boundary of an L_2 cell C which is not entirely empty. The L_1 cell within C (marked in light blue), where u enters does not contain any node, but there are nodes in another L_1 cell (marked in grey) within C. As u is not receiving any beacon from within its current L_1 cell, it needs to retrieve its L_0 and L_1 home as the L_2 home is still active. Thus after entering C, u sends a query to its L_2 home about the status of C. Upon receiving the reply, u finds that C is not empty and concludes that

- Since C is an L_2 cell and not empty, it already contains an existing L_1 home.
- There might be nodes within u's current L_1 cell, but outside u's transmission range.
- There may be no nodes within u's current L_1 cell.

In order to obtain this information, u sends a query to the direction of the center of the current L_1 cell. Since there is node there, the query is not responded to and after a timeout period u again sends another query to the direction of the center of C. As there exist nodes in C, the packet is eventually received by any node A already there. A is aware of its L_0 and L_1 homes and sends a reply containing this information to u. When the reply is received, u finds the ID of the L_1 home, because both A and u have the same L_1 home as they both resides in the same L_2 cell. However, the L_0 home ID of A received by u through the reply packet tells u that the corresponding L_0 home ID actually belongs to a different L_1 cell than the one u is in. Thus u concludes that the current L_1 cell is empty and makes itself a *Red* node, then marks its L_0 cell as the R_0 home and notifies the L_1 home based on the recently achieved information that u's current L_1 cell is not empty. In this case a home at R_0 is created and thus there is no need to store information about whether its sibling cells are empty or not.

5.2.8 Location Update

Like other hierarchical location services HALS also updates its corresponding servers (homes) at different levels of the hierarchy with different levels of accuracy. The location update can take place in one of the following ways: *Time triggered*, *Distance triggered* and *Crossing cell triggered*.

Time triggered update In a time triggered scheme, each node updates its location servers in a fixed interval time triggered fashion. The interval for updating home at different levels increases from lower level to upper level as the lower level homes need to know a more accurate location. As outlined in Algorithm 5.9, in a time triggered scheme a node updates its location to its R_0 home with a predefined time interval t_0^u and updates its L_i home twice as frequently as its L_{i+1} home. A smaller value of t_0^u causes more accurate location information to be available in the location servers though the network traffic increases. On the other hand, a higher value of t_0^u results in less accurate location information. Thus the value of t_0^u must be selected properly to balance this trade off.

Distance triggered update Location update in a distance triggered scheme takes place when a node traverses a predefined distance. Similar to time triggered update, in this scheme a node updates its location to the R_0 home after traveling a predefined distance d_0^u and updates its L_{i+1} home after traveling twice more distance than that of its L_i home (Algorithm 5.9). Thus the frequency of location update in this method depends on the speed of node movement.

Crossing cell triggered update Crossing cell triggered scheme updates a node's L_i home when it crosses an L_i cell boundary. It has the least overhead, but if an update packet for a higher level is lost for any reason, the query failure may be substantially high.

When a node wants to update its L_0 home (R_0 home), it sends its absolute location. However, for higher level L_i homes, the node sends the L_i cell ID in which it resides. It knows its L_i home ID which is available through beacons. Therefore, the node sends its location in terms of its L_i cell ID to its L_i home. The packet forwarding mechanism

to an R_i home with L_i cell ID is described in Section 5.2.4. In this way by means of cooperative forwarding, the packets can reach the desired home regions.

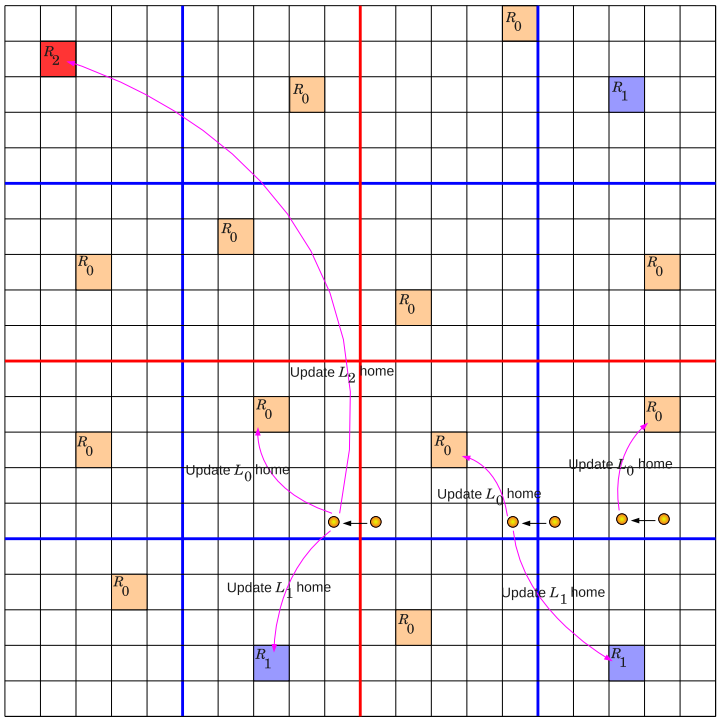
Algorithm 5.9 locUpdate

Require: u as node
1: **if** update method is Time-triggered **then**
2: **for all** i, where $0 \leq i \leq \psi$ **do**
3: **if** $i = 0$ **then**
4: u updates R_i home at t_0^u interval with absolute location
5: **else**
6: u updates R_i home at $t_0^u \times 2^i$ interval with L_i cell ID
7: **end if**
8: **end for**
9: **else if** update method is Distance-triggered **then**
10: **for all** i, where $0 \leq i \leq \psi$ **do**
11: **if** $i = 0$ **then**
12: u updates R_i home after moving d_0^u distance, with absolute location
13: **else**
14: u updates R_i home after moving $d_0^u \times 2^i$ distance, with L_i cell ID
15: **end if**
16: **end for**
17: **else if** update method is Crossing-cell-triggered **then**
18: **if** cell is crossed at highest level l, where $0 \leq l < l_{max}$ **then**
19: **for all** i, where $0 \leq i \leq l$ **do**
20: **if** $i = 0$ **then**
21: u updates R_i home with absolute location
22: **else**
23: u updates R_i home with L_i cell Id
24: **end if**
25: **end for**
26: **end if**
27: **end if**

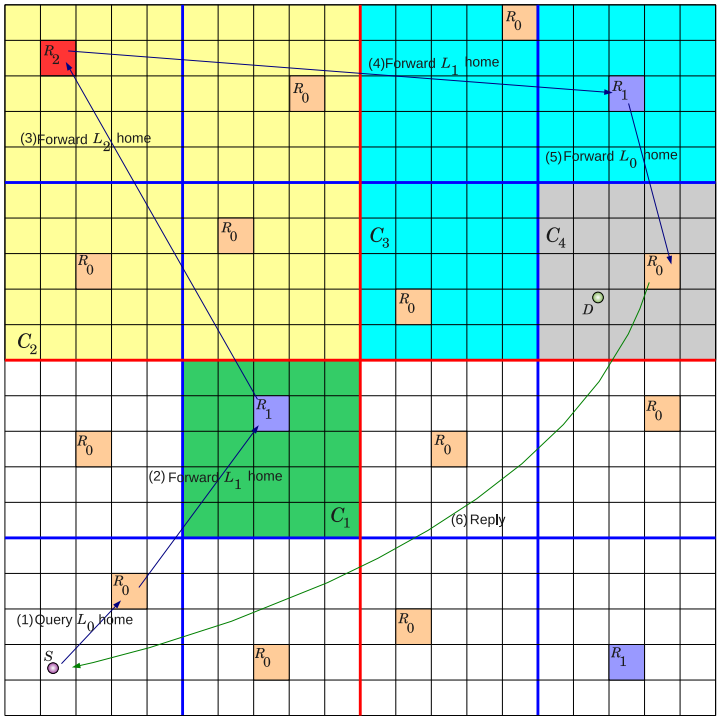
Figure 5.9(a) shows the crossing cell triggered location update scheme. In this figure, it is seen that updating the home regions at different levels depend on the level of the cell boundary crossed by a node. When a node crosses only an L_0 cell border, it updates its R_0 home with its absolute location. If the node crosses an L_1 cell boundary, it needs to update both R_0 and R_1 homes, since crossing the boundary of an L_1 cell also causes a node to cross an L_0 cell's boundary. Finally, when the node crosses the boundary of an L_2 cell, it has to update all of its recent R_0, R_1 and R_2 homes.

5.2.9 Location Query

If a node S wants to query the location of a target node D, the query packet needs to be routed to D's R_0 home which contains the absolute position of D. The query



(a)



(b)

Figure 5.9: Example of (a) location update, and (b) location query.

is bounded within the boundary of the smallest cell which contains both S and D. At first S sends the query to its own R_0 home. The R_0 home then searches for D's location in its database. If D's location is found, a reply is sent to S. Otherwise, the query is forwarded towards S's R_1 home. The R_1 home looks for D's location in its database to see whether D's absolute location can be found. If D's absolute location is found, a reply is sent. Otherwise, R_1 looks for D's location within the L_1 ID space in its database. If D's location in the form of an L_1 cell ID, say C, is found, the query is forwarded to the R_0 home within C. This R_0 home knows D's absolute location and replies to S. In case S's R_1 home can not find D's location in its database, it forwards the query to the R_2 home of S. This forwarding, both up and down the hierarchy, continues until D's location is found.

The algorithm for the query procedure is presented in Algorithm 5.10. Formally, when a query reaches an R_r home ($0 \leq r \leq \psi$) it looks for D's location in every ID space i in the database where $0 \leq i \leq r$. If D's location is found in the form of an L_i cell, the query is forwarded to the found L_i cell that has a more accurate location information of D and thereby reduces the search area (absolute location if $i = 0$, or ID of an L_i cell where D resides). In a similar way, the R_{i-1} home within the found L_i cell searches for D's location in its database and forwards the query to a lower ranked home until the absolute location of D is found. However, if the R_r home of S can not locate D's location in any ID space, it simply forwards the query towards the R_{r+1} home, and this process continues until D's location is located in any ID space. It should be noted that since the R_ψ home has all the nodes' location information at different levels of precision, a query never needs to be forwarded beyond that.

Algorithm 5.10 forwardQuery

```
1: if query packet received at rank r, 0 ≤ r ≤ ψ, then
2:    for all i, 0 ≤ i ≤ r, do
3:       search in database at level i
4:    end for
5:    if found at level 0 ID space then
6:       send reply with location
7:    end if
8:    if found at level k with the cell ID C_k, 1 ≤ k ≤ r, then
9:       forwardQuery to R_{k-1} within C_k
10:   else
11:      forwardQuery to R_{r+1}
12:   end if
13: end if
```

The location query procedure is explained through Fig. 5.9(b). Here, node S initiates a query to retrieve the location of another node D. At first S sends the query to its R_0 home within its L_1 cell. As the R_0 home can not find D's location in the database (D is not within its L_1 cell), it forwards the query to S's L_1 home C_1 (marked in green). Eventually the query reaches S's R_1 home by following the method explained in Section 5.2.4. The R_1 home looks for D's location and can find neither the absolute location (L_0 ID space) nor in the L_1 ID space (C_1's siblings within its L_2 cell). Thus R_1 forwards the query to S's L_2 home C_2 (marked in yellow) and R_2 within it receives the query. R_2 starts looking for D's location and fails to find in its L_0 and L_1 ID spaces in the database. However, D's location is located in the L_2 ID space which means R_2 knows D's location C_3 (marked in light blue) - an L_2 cell ID (D resides in C_3). Thus again the query is forwarded to the highest ranked home (R_1 home) of C_3. This time R_1 within C_3 finds D's location C_4 (marked in grey)- an L_1 cell ID. The R_0 home within C_4 actually knows D's absolute location in the database's L_0 ID space since D resides in C_4. Upon receiving the forwarded query, R_0 within C_4 attaches D's location to the reply message and sends to S.

5.3 Scalability Analysis

The scalability of a location service is characterized by the message overhead required to make the location service functional and how this overhead changes with the size of the network. In this section a theoretical model is introduced considering uniform node distribution similar to the other location service schemes [19, 21], that describes HALS's scalability. Similar to the other location services, the scalability of HALS is measured based on three metrics:

1. Update overhead
2. Query overhead
3. Storage overhead

The theoretical analysis for scalability of the above three metrics is presented in the following Sections 5.3.1, 5.3.2 and 5.3.3, respectively. The notations used in this analysis are listed in Table 5.1.

Table 5.1: Notations used in scalability analysis of HALS.

Notation	Meaning
ν	Node movement speed
a	Side length of an L_0 cell
ρ_i^c	Update packet generation rate for R_i home in crossing cell triggered scheme
ρ_i^t	Update packet generation rate for R_i home in time triggered scheme
ρ_i^d	Update packet generation rate for R_i home in distance triggered scheme
t_i^u	Time interval for R_i home update in time triggered scheme
d_i^u	Distance interval for R_i home update in distance triggered scheme
σ_i	Maximum number of hops required from a node to its R_i home
σ_i^q	Maximum number of hops required to complete a query (including reply) bounded by an L_{i+1} cell
δ_i	Maximum distance from a node to its R_i home
z	Average progress by each hop
r_t	Transmission range
τ	Average node density
N	Number of nodes
ψ	Highest rank, i.e., playground is an $L_{\psi+1}$ cell
Δ	Network deployment area
ξ_i	The number of nodes whose location information needs to be stored by a node in an R_i home
p_i	Probability that L_{i+1} cell is the smallest common cell of both querying and queried nodes
C_u^c	Update overhead for cell triggered update method
C_u^t	Update overhead for time triggered update method
C_u^d	Update overhead for distance triggered update method
C_q	Query overhead
C_s^i	Storage overhead for nodes within R_i home

5.3.1 Update Overhead

Any of the three different types of location update mechanisms presented in Section 5.2.8 can be used in HALS and each of the methods has its own scalability domain. In order to analyze the cell crossing triggered update method, Lemma 5.1 and 5.2 give the frequency of the update packet generation and the expected number of hops that

each update packet needs to traverse to reach the destination, respectively.

Lemma 5.1 *An arbitrary node A's L_i cell crossing rate (update packet generation rate) is:*

$$\rho_i^c = \begin{cases} \dfrac{\pi\nu}{2a} & \textit{for } i\textit{=0;} \\ \dfrac{\pi\nu}{\kappa a 2^i} & \textit{for } 1 \le i \le \psi. \end{cases} \tag{5.15}$$

Proof: Let the L_0 cell boundary crossing rate be ρ_0^c. In [19] the authors showed that

$$\rho_0^c \approx \frac{\pi\nu}{2a} \tag{5.16}$$

by approximating ρ_0^c with the crossing rate of a node in a circle with diameter a (Since a node can cross the boundary of an L_0 cell in any direction, the area of an L_0 cell is approximated as a circular region). It should be noted that an L_i boundary crossing always causes crossing boundaries of all L_j cells, where $j < i$. In HALS, L_1 is composed of $\kappa \times \kappa$ L_0 cells and L_i, $1 < i \le \psi + 1$, is composed of 2×2 L_{i-1} cells. Here κ is a user defined constant value. Thus

$$\begin{aligned} \rho_1^c &= \frac{\pi\nu}{2a \times \kappa} \\ &= \frac{\pi\nu}{\kappa a 2^1} \end{aligned} \tag{5.17}$$

and, for any j, $2 \le j \le \psi$

$$\rho_j^c = \frac{1}{2}\rho_{j-1}^c. \tag{5.18}$$

By generalizing,

$$\rho_i^c = \frac{\pi\nu}{\kappa a 2^i}. \tag{5.19}$$

■

Lemma 5.2 *The maximum number of hops traversed by the location update packet from node A to A's R_i home is*

$$\sigma_i = \frac{\kappa \times \sqrt{2} a 2^i}{z}, \tag{5.20}$$

where z is the average progress by each hop for uniformly distributed nodes.

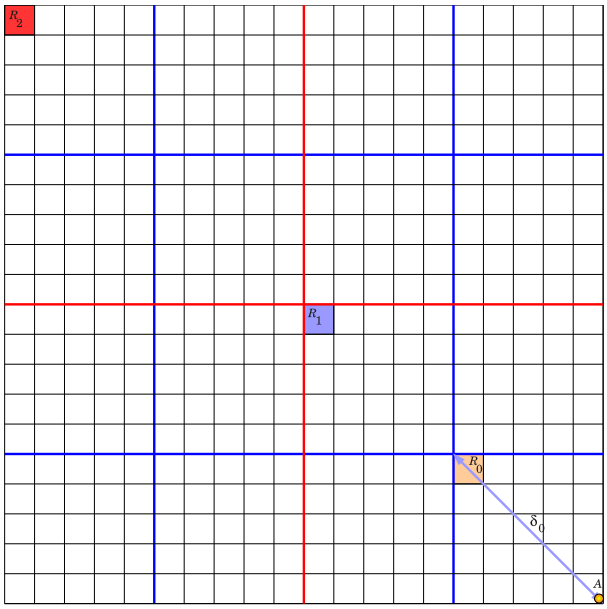


Figure 5.10: Positions of ranked homes are at the farthest from a node.

Proof: In HALS the R_i homes can shift from one cell to another. For this reason, let us consider an R_0 home located at the farthest possible distance from a node A within an L_1 cell, as shown in Fig. 5.10. The diagonal distance is the maximum distance and hence this will give the worst case estimation of the distance from a node to its R_0 home.

Thus the maximum distance from a node to its R_0 home is

$$\begin{aligned} \delta_0 &= \sqrt{(\kappa a)^2 + (\kappa a)^2} \\ &= \kappa\sqrt{2}a \times 2^0. \end{aligned} \tag{5.21}$$

Similarly, as shown in Fig. 5.10, if all other R_i homes are the farthest possible distance from A, then the distance from A to R_i is

$$\delta_i = \kappa\sqrt{2}a \times 2^i, 0 \leq i \leq \psi. \tag{5.22}$$

Now, the expected number of hops in forwarding the update packet to an R_i home is the distance divided by z defined as [19]

$$z = f(r_t, \tau) \tag{5.23}$$

where, r_t is the radio transmission range and τ is the node density. Since r_t and τ are constants, z is also constant.

Thus the maximum number of hops from a node to its R_i home is given by

$$\begin{aligned} \sigma_i &= \frac{\delta_i}{z} \\ &= \frac{\kappa\sqrt{a} \times 2^i}{z}. \end{aligned} \tag{5.24}$$

■

Theorem 5.1 *The crossing cell triggered location update cost is*

$$C_u^c = O(\nu \log_2 N), \tag{5.25}$$

Proof: The location update cost C_u^c is the cost of updating all the $\psi + 1$ homes, therefore,

$$\begin{aligned} C_u^c &= \sum_{i=0}^{\psi} \rho_i^c \times \sigma_i \\ &= \rho_0^c \times \sigma_0 + \sum_{i=1}^{\psi} \rho_i^c \times \sigma_i \\ &= \frac{\pi\nu\delta_0}{2az} + \sum_{i=1}^{\psi} \frac{\pi\nu}{\kappa a 2^i} \frac{\delta_i}{z} \\ &= \frac{\kappa\sqrt{2}\pi\nu}{2z} + \sum_{i=1}^{\psi} \frac{\pi\nu}{\kappa a 2^i} \frac{\kappa\sqrt{2}a2^i}{z} \\ &= \frac{\kappa\sqrt{2}\pi\nu}{2z} + \frac{\sqrt{2}\pi\nu}{z} \times \psi \\ &\approx O(c_1\nu + c_2\nu\psi) \end{aligned} \tag{5.26}$$

where, $c_1 = \frac{\kappa\sqrt{2}\pi}{2z}$ and $c_2 = \frac{\sqrt{2}\pi}{z}$. Since z is constant, c_1 and c_2 are also constant. Therefore,

$$C_u^c = O(\nu + \nu\psi) \tag{5.27}$$

Any arbitrary node has one ranked home in its every L_j cell, $1 \leq j \leq \psi + 1$ (i.e., R_0 in an L_1 cell, R_1 in an L_2,$\cdots$, R_ψ in an $L_{\psi+1}$ cell as shown in Fig. 5.10), and the highest rank value ψ is proportional to $\log_2 \sqrt{\Delta}$, because, $\sqrt{\Delta} = \kappa a 2^\psi$ where Δ represents the deployment area. If the same node density is maintained while increasing the deployment area, i.e., $\Delta \propto N$; this leads to

$$\psi \propto \log_2 \sqrt{N}. \tag{5.28}$$

Thus the update cost becomes

$$\begin{aligned} C_u^c &= O(\nu + \nu \log_2 \sqrt{N}) \\ &= O\left(\frac{\nu}{2} \log_2 N\right) \\ &= O(\nu \log_2 N). \end{aligned} \tag{5.29}$$

■

Theorem 5.2 *The time triggered location update cost*

$$C_u^t = O(\log_2 N) \tag{5.30}$$

Proof: According to the time triggered location update procedure a node updates its R_0 home periodically at t_0^u time intervals which is a predefined constant value. Similarly, its R_i home is updated at t_i^u time intervals where

$$t_i^u = 2^i \times t_0^u. \tag{5.31}$$

Therefore, the update packet generation rate for an R_i home is

$$\begin{aligned} \rho_i^t &= \frac{1}{t_i^u} \\ &= \frac{1}{2^i \times t_0^u}. \end{aligned} \tag{5.32}$$

When the maximum ranked home R_ψ is updated all the other R_i homes, $0 \leq i \leq \psi - 1$, are also updated. Thus the update cost can be calculated with the update packet generation rate and can be defined as:

$$\begin{aligned} C_u^t &= \sum_{i=0}^{\psi} \sigma_i \times \rho_i^t \\ &= \sum_{i=0}^{\psi} \frac{\delta_i}{z} \times \frac{1}{t_i^u} \\ &= \sum_{i=0}^{\psi} \frac{\kappa\sqrt{2}a2^i}{z} \times \frac{1}{t_0^u \times 2^i} \\ &= \frac{\kappa\sqrt{2}a}{z} \sum_{i=0}^{\psi} \frac{1}{t_0^u} \\ &= \frac{\kappa\sqrt{2}a}{z \times t_0^u}(\psi + 1) \\ &\approx O\left(c_3 \times \log_2 \sqrt{N}\right) \end{aligned} \tag{5.33}$$

where $c_3 = \frac{\kappa\sqrt{2}a}{zt_0^u}$ is constant. Therefore, the update cost becomes

$$C_u^t = O(\log_2 N). \tag{5.34}$$

■

Theorem 5.3 *The distance triggered location update cost C_u^d can be defined as:*

$$C_u^d = O(\nu \log_2 N) \tag{5.35}$$

Proof: According to the distance triggered location update procedure a node updates its R_0 home periodically after moving a d_0^u distance away from the last update position where update took place. Here, d_0^u is a predefined constant value. Similarly its R_i home is periodically updated when it moves d_i^u distance from its last update location. Here,

$$d_i^u = 2^i \times d_0^u. \tag{5.36}$$

Therefore, the update packet generation rate for an R_i home is

$$\begin{aligned} \rho_i^d &= \frac{\nu}{d_i^u} \\ &= \frac{\nu}{2^i \times d_0^u}. \end{aligned} \tag{5.37}$$

When the maximum ranked home R_ψ is updated, all the other R_i homes, $0 \leq i \leq \psi - 1$, are updated as well. Thus using the update packet generation rate, the update cost can be calculated as:

$$\begin{aligned} C_u^d &= \sum_{i=0}^{\psi} \sigma_i \times \rho_i^d \\ &= \sum_{i=0}^{\psi} \frac{\delta_i}{z} \times \frac{\nu}{d_i^u} \\ &= \sum_{i=0}^{\psi} \frac{\kappa\sqrt{2}a2^i}{z} \times \frac{\nu}{d_0^u \times 2^i} \\ &= \frac{\kappa\sqrt{2}a}{z} \sum_{i=0}^{\psi} \frac{\nu}{d_0^u} \\ &= \frac{\kappa\sqrt{2}a \times \nu}{z \times d_0^u}(\psi + 1) \\ &\approx O\left(c_4 \times \nu \log_2 \sqrt{N}\right) \end{aligned} \tag{5.38}$$

where $c_4 = \frac{\kappa\sqrt{2}a}{zd_0^u}$ is constant. Therefore, the update cost becomes

$$C_u^d = O(\nu \log_2 N). \tag{5.39}$$

■

5.3.2 Query Overhead

In order to calculate the query overhead, the total number of hops a query packet traverses needs to be estimated. Lemma 5.3 finds the required number of hops for a query in the worst case scenario.

Lemma 5.3 *If a node A queries another node B's location while both A and B reside in the smallest common L_{i+1} cell, $0 \leq i \leq \psi$, the maximum required number of hops is*

$$\sigma_i^q = \frac{\kappa\sqrt{2}a}{z}\left(5 \times 2^i - 3\right). \tag{5.40}$$

Proof: As described in Section 5.2.9, in the worst case scenario node A sends the query towards its R_0 home, then the query is subsequently forwarded upward in the hierarchy to R_1, R_2,$\cdots$, R_i and then downward through R_{i-1}, R_{i-2},$\cdots$, to the R_0 home, when the smallest cell which contains both A and B is an L_{i+1} cell. This scenario will require the maximum number of hops to be traversed. After that from R_0 the reply is sent to A. In the worst case, each of these steps can be viewed as the diagonal distance of the cells in each level.

Therefore, the distance of each step is the same as was derived for δ_i in Lemma 5.2. Thus the maximum possible number of hops σ_i^q requires when both A and B are in

L_{i+1} cell is

$$\begin{aligned}
\sigma_i^q &= steps_up_to(R_i) + steps_down_to(R_0) + step_to_reply \\
&= \{hops(R_0) + hops(R_1) + \cdots hops(R_i)\} + \\
&\quad \{hops(R_i) + \cdots + hops(R_1)\} + hops(R_i) \\
&= \frac{\delta_0}{z} + \frac{\delta_1}{z} + \cdots + \frac{\delta_i}{z} + \frac{\delta_i}{z} + \cdots + \frac{\delta_1}{z} + \frac{\delta_i}{z} \\
&= 2\sum_{j=0}^{i} \frac{\delta_j}{z} - \frac{\delta_0}{z} + \frac{\delta_i}{z} \\
&= 2\sum_{j=0}^{i} \frac{\kappa\sqrt{2}a2^j}{z} - \frac{\kappa\sqrt{2}a}{z} + \frac{\kappa\sqrt{2}a2^i}{z} \\
&= \frac{2\kappa\sqrt{2}a}{z}\sum_{j=0}^{i} 2^j + \frac{\kappa\sqrt{2}a2^i}{z} - \frac{\kappa\sqrt{2}a}{z} \\
&= \frac{\kappa\sqrt{2}a}{z}\left(2\sum_{j=0}^{i} 2^j + 2^i - 1\right) \\
&= \frac{\kappa\sqrt{2}a}{z}\left(2(2^{i+1} - 1) + 2^i - 1\right) \\
&= \frac{\kappa\sqrt{2}a}{z}\left(5 \times 2^i - 3\right). \qquad (5.41)
\end{aligned}$$

■

Theorem 5.4 *The upper bound of the expected location query overhead is*

$$C_q = O(\sqrt{N}). \qquad (5.42)$$

Proof: Lemma 5.3 gives an estimation of the maximum number of hops required to complete a query process when the querying node A and queried node B both reside within the smallest boundary of an L_{i+1} cell. To obtain the expected query overhead the probability p_i, $0 \leq i \leq \psi$, of both A and B residing in the same L_{i+1} cell needs to be calculated. If the traffic pattern is uniform across the network, for a given position of A', if the probability of B's position is also uniform across the network, then

$$p_i = \frac{1}{4^{\psi - i}}, 0 \leq i \leq \psi. \qquad (5.43)$$

Now, the maximum query overhead within an L_{i+1} cell is

$$\begin{aligned}
C_q &= \sum_{i=0}^{\psi} \sigma_i^q p_i \\
&= \sum_{i=0}^{\psi} \frac{\kappa\sqrt{2}a}{z} \left(5 \times 2^i - 3\right) \frac{1}{4^{\psi-i}} \\
&= \frac{\kappa\sqrt{2}a}{z} \sum_{i=0}^{\psi} \left(\frac{5 \times 2^i}{4^{\psi-i}} - \frac{3}{4^{\psi-i}} \right) \\
&= \frac{\kappa\sqrt{2}a}{z} \sum_{i=0}^{\psi} \left(\frac{5 \times 2^i \times 4^i}{4^{\psi}} - \frac{3 \times 4^i}{4^{\psi}} \right) \\
&= \frac{\kappa\sqrt{2}a}{z4^{\psi}} \sum_{i=0}^{\psi} (5 \times 2^{3i} - 3 \times 2^{2i}) \\
&= \frac{\kappa\sqrt{2}a}{z2^{2\psi}} \left(5\frac{8^{\psi+1}-1}{7} - 3\frac{4^{\psi+1}-1}{3} \right) \\
&= \frac{\kappa\sqrt{2}a}{z} \left\{ \frac{5}{7} \left(\frac{8 \times 2^{3\psi}}{2^{2\psi}} - \frac{1}{2^{2\psi}} \right) - \frac{4 \times 2^{2\psi}}{2^{2\psi}} + \frac{1}{2^{2\psi}} \right\} \\
&= \frac{\kappa\sqrt{2}a}{z} \left\{ \frac{5}{7} \left(8 \times 2^{\psi} - \frac{1}{2^{2\psi}} \right) - 4 + \frac{1}{2^{2\psi}} \right\} \\
&\approx O\left(2^{\log_2 \sqrt{N}} - \frac{1}{2^{2\log_2 \sqrt{N}}} + \frac{1}{2^{2\log_2 \sqrt{N}}} \right) \\
&= O(2^{\log_2 \sqrt{N}}) \\
&= O(\sqrt{N}). \qquad (5.44)
\end{aligned}$$

■

5.3.3 Storage Overhead

Theorem 5.5 *The storage overhead of HALS is*

$$C_s^i = \begin{cases} 4^i & \textit{for nodes within an } R_i \textit{ home ;} \\ 0 & \textit{elsewhere.} \end{cases} \qquad (5.45)$$

Proof: The location server nodes in HALS work in more like centralized fashion, as the nodes in an R_i home store the information of all nodes residing in its same L_{i+1} cell. Given the highest rank in the hierarchy ψ, the nodes of an R_i home need to store information of ξ_i nodes where

$$\xi_i = \frac{N}{4^{\psi-i}}. \qquad (5.46)$$

Therefore, the storage overhead for the nodes in an R_i home is given by

$$\begin{aligned} C_s^i &= \xi_i \\ &= \frac{4^i N}{2^{2\psi}} \\ &\approx \frac{4^i N}{2^{2\log_2 \sqrt{N}}} \\ &= O\left(\frac{4^i N}{2^{\log_2 N}}\right) \\ &= O\left(\frac{4^i N}{N}\right) \\ &= O(4^i) \end{aligned} \tag{5.47}$$

and the other nodes not in any ranked home do not have any overhead at all.

■

5.3.4 Summary

Analyzing the theoretical model for the uniform node distribution, it is observed that the scalability of HALS can be represented according to Table 5.2. This table also presents the overheads of HLS and GLS for comparison.

Table 5.2: Scalability of HALS

Overhead	HALS	HLS	GLS
Update (cell crossing triggered)	$O(\nu \log N)$	-	-
Update (time triggered)	$O(\log N)$	$O(\sqrt{N})$	$O(\nu\sqrt{N})$
Update (distance crossing triggered)	$O(\nu \log N)$	-	-
Query	$O(\sqrt{N})$	$O(\sqrt{N})$	$O(\sqrt{N})$
Storage	$O(4^i)$ for nodes in R_i home 0 for others	$O(\log N)$	$O(\log N)$

It should be noted that, with a uniform node distribution the home regions in HALS do not need to be destroyed or created, since no cell is expected to become empty. On the contrary, if the node distribution is non-uniform, the performance of the existing hash function based location services, e.g., HIGH-GRADE [21], HLS [22], SLURP [19], degrades substantially while HALS overcomes this problem with the shift, create and destroy mechanisms for home regions at the expense of some

overhead. As analyzed in Sections 5.2.5, 5.2.7 and 5.2.6, the shifting, creating and destroying mechanisms complete their tasks with a finite number of unicast message transfers which is restricted within a lower level in the hierarchy before moving to an upper level, making them scalable. By selecting the higher ranked homes in an intelligent way, e.g., hosting them near hot spot regions, the number of shifting can be reduced and the overall performance can be improved.

5.4 Simulation Environment

For implementing the proposed location service HALS Network Simulator ns-2.29 was used, and its performance was compared to GLS [83] and HLS [22]. The exact implementation of GLS and HLS was used which is available at *http://www.cn.uni-duesseldorf.de/staff/kiess/software/*. Both GLS and HLS are hierarchically organized location services and HLS shares a design similarity to HIGH-GRADE [21] and balances the update and query costs while being highly scalable. Table 5.3 presents the values of parameters used for comparing the three location services, and most of these values are also used in other studies like [22].

Table 5.3: Parameters used in simulation.

Notation	**Meaning**
Playground size	2000m × 2000m
Simulation duration	300 sec
Number of nodes N	200, 300, 400
Mobility model	Random Waypoint model with and without region of interest
	AMM model with and without region of interest proposed in Chapter 4
Transmission range r_t	200m, 250m
Max node speed	5, 10, 15, 20 m/sec
Request per node	4
Number of runs	10
MAC layer	IEEE 802.11
Routing protocol	GPSR

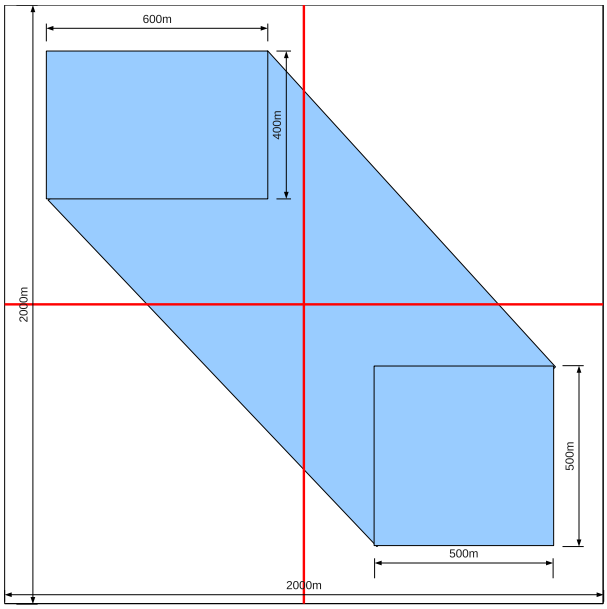


Figure 5.11: Scenario of free space with region of interest.

These location services have been evaluated in both the free space and obstacle scenario. The free space scenario is used in order to validate the working principles of the location services. Moreover, since most of the location services have been evaluated in free space, it is of great interest to see how HALS performs in this situation. The variations in the free space scenario used are as follows:

- **Free space scenario:** Here the nodes move following the Random Waypoint model in the playground without the existence of any obstacle. This scenario aids in analyzing the location services' performance in uniform node distribution.
- **Free space scenario with region of interest:** The formation of region of interest results in a higher number of nodes concentrated in specific regions enabling us to simulate how proposed and existing schemes behave in such a situation. In addition to the Random Waypoint model, two rectangular shaped regions of interest of 600m × 400m and 500m × 500m size were placed in the upper left and lower right portion of the simulation environment. After deployment the nodes tend to move towards the regions of interest, which makes most of the nodes concentrated in the highlighted area (marked in blue) as shown in Fig. 5.11. This scenario helps to analyze the location services' performance with non-uniformly distributed nodes.

Moreover, to evaluate the location services' performance in a real-world environment in the presence of obstacles MobiGen (Chapter 4) was used to generate scenarios where node movement follows the AMM mobility model [34] proposed in Chapter 4, and these scenarios were based on the Clayton campus of Monash University as shown in Fig. 5.12. A portion of the map keeping the playground size 2000m × 2000m was selected and within this terrain the existing buildings and positions of entrances were incorporated in MobiGen as shown in Fig. 5.13. For evaluating the location services, the following variations are considered.

- **Obstacle scenario:** In this scenario the obstacles are accessible through doors and there are pathways. While moving to its destination, a node selects a path in such a way that it avoids inaccessible obstacles, uses the doorways to enter and exit accessible obstacles and uses the pathways, if any exists in between the starting and ending positions.
- **Obstacle scenario with region of interest:** On top of the Obstacle scenario, two rectangular shaped regions of interest I_1^{roi} and I_2^{roi} were placed in the simulation environment as shown in Fig. 5.13. Region I_1^{roi} consists of the Mathematics and Information Technology services building, Engineering Building, Hargrave-Andrew Library and Cafeteria, facilities and Conference room. Region I_2^{roi} includes the Campus center, Humanities building, Administration building, Sir Louis Matheson Library and the Religious center. Since these buildings are frequently used by people (mostly students), they have been considered as regions of interest.

5.5 Simulation Results

Various parameters related to the performance evaluation of location service protocols are calculated, including the location update and query success rate, average number of hops required for update and query, delay in the update and query, and the cache hit rate for query at a wide range of node movement speed and node density. Metrics related to the location update and location query are presented and analyzed in the following Sections 5.5.1 and 5.5.2, respectively. Here, the results for a transmission

Figure 5.12: Snapshot of a portion (inside the square box) of Clayton campus, Monash University, Australia, selected for performance evaluation (courtesy: Google Earth).

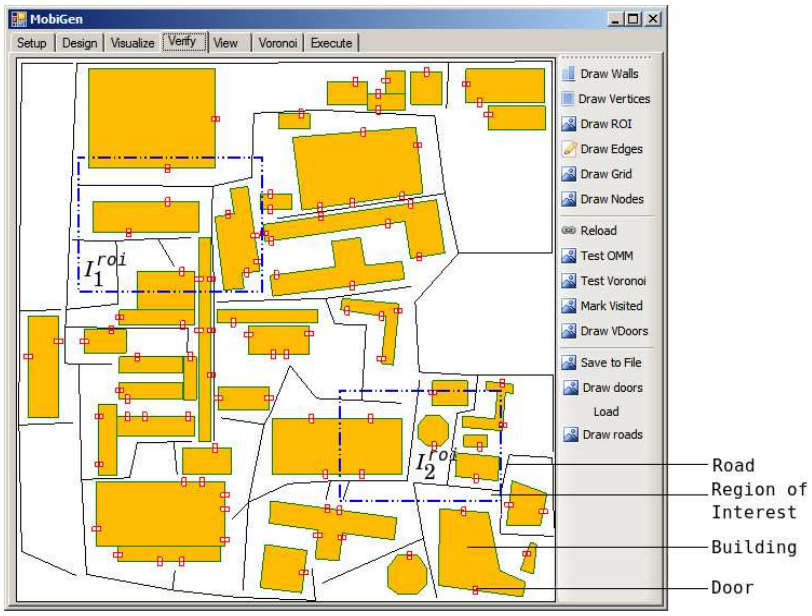


Figure 5.13: A snapshot of MobiGen implementing the selected portion in Fig. 5.12 with buildings, pathways and doors at exact positions. Blocks 'I_1^{roi}' and 'I_2^{roi}' show the location of the regions of interest.

range of 200m are presented and a similar trend in results was also observed for a 250m transmission range.

5.5.1 Location Update

Figure 5.14(a) shows the location update success rate of GLS, HLS and HALS at different node movement speed with 400 nodes, where there is no obstacle, and the node movement follows the Random Waypoint model. Here it is observed that HALS exhibits the highest success rate among the location services. GLS shows a higher success rate than HLS when the node movement speed is higher. Since the home region shifting mechanism in HALS never lets the home region to become empty even for a short duration of time, HALS shows significantly higher success rate compared to that for the other schemes. Moreover, HALS sends update packets in both greedy and perimeter mode, whereas HLS uses greedy mode only. GLS also does not rely only on greedy mode and thus its success rate becomes slightly higher than HLS when the node mobility increases.

Though the Random Waypoint model tends to scatter the nodes throughout the simulation area, some of the cells become empty for short duration if node density is not very high, and during this period the update packets can not be received successfully. To overcome this HLS adopts a mechanism to store and forward pending location information to the designated server cell while it is empty due to node mobility, but this scheme does not guarantee successful update as observed in Fig. 5.14(a).

Figure 5.14(b) shows the update success rate at different speed when regions of interest are present. In the beginning the nodes tend to travel towards the regions of interest and then from one region to another. This makes the node movement more concentrated inside the regions of interest and in the area interconnecting the regions, leaving very few nodes available in the rest of the simulation area. This causes a large number of cells to be empty which in turn decreases the location update success rate of HLS. Moreover, at higher node movement speed, the cells become empty at a higher rate and subsequently result in lower success rate when node mobility is high. Since GLS does not rely on groups of nodes in the home region, rather on a single server node, it shows a higher success rate in location update, and maintains almost steady location update success rate. On the other hand, HALS exhibits much superior performance to

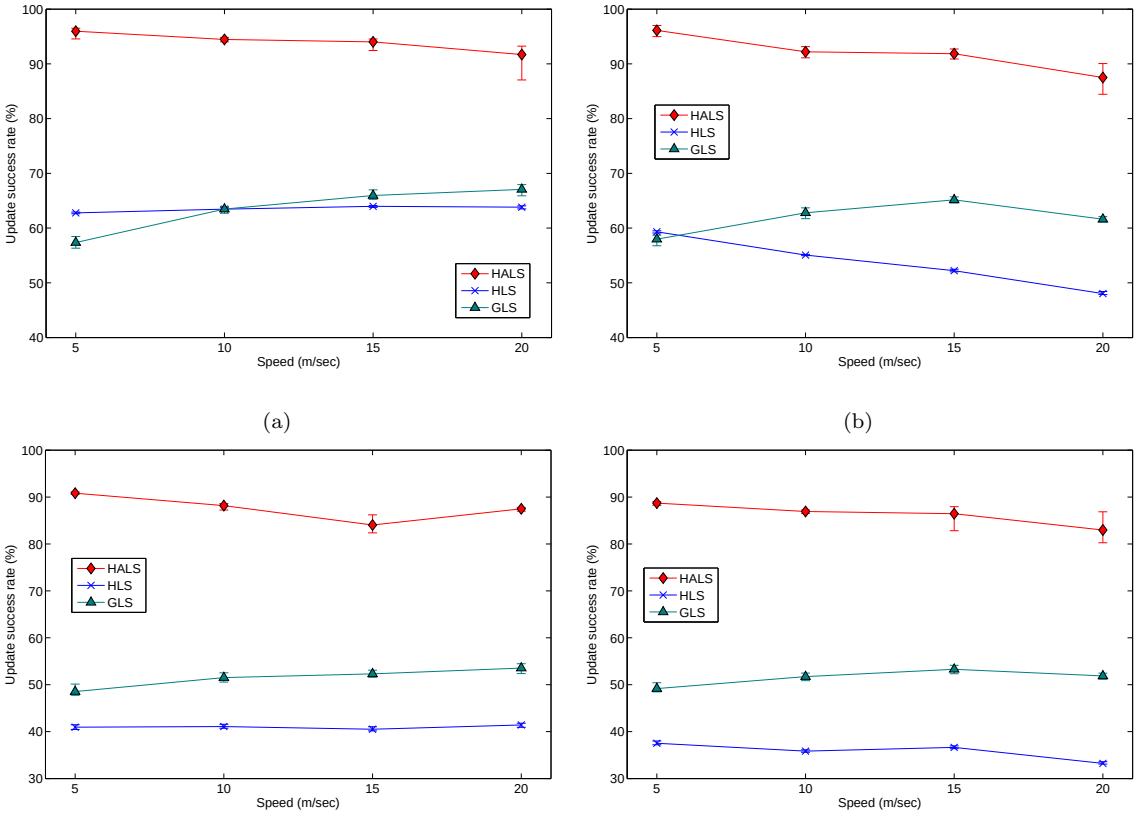


Figure 5.14: Location update success rate at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

HLS and GLS, an almost equivalent results similar to what is observed in Fig. 5.14(a). Figure 5.14(c) and (d) show the location update success rate considering the obstacle scenario without and with region of interest, respectively. The obstacles cause signal attenuation and thus the update success rate is lower compared to that in the free space scenario. However, the obstacles cause higher drop in the success rate of HLS than that of HALS and GLS because unlike others, HLS only uses greedy mode forwarding during the update process. Moreover, GLS and HALS are less sensitive to the node distribution than HLS.

The location update success rate of different location services with various node density is presented in Fig. 5.15. In the free space scenarios (Fig. 5.15(a) & (b)), it is observed that though HLS shows the lowest success rate, it has the highest increase in

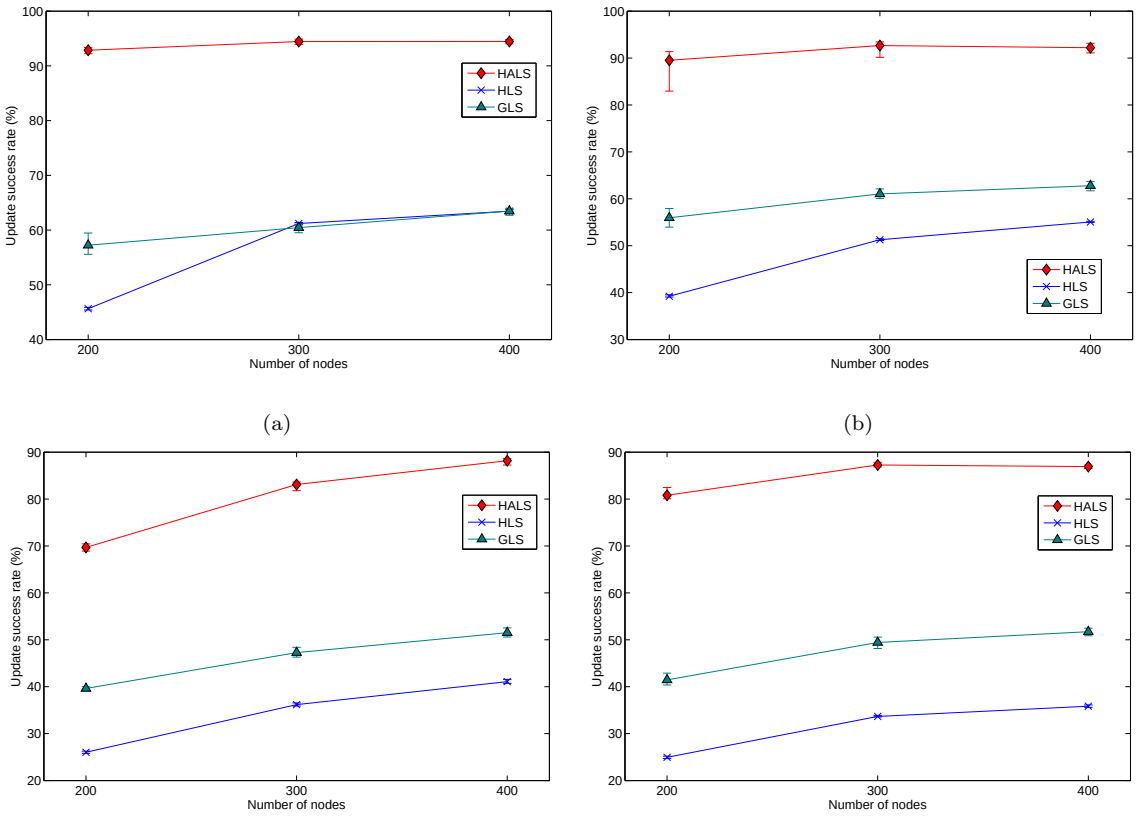


Figure 5.15: Location update success rate at different node density (at 10 m/sec) in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

the success rate when the number of nodes increases from 200-400. HALS maintains almost a similar update success rate at varying node density by ensuring non-empty homes irrespective of the node density. On the other hand, GLS shows a slightly increasing trend when the number of nodes is increased.

In the obstacle scenario (Fig. 5.15(c)), the update success rate of HLS remains the lowest while HALS shows the highest success rate and GLS lies in between. A similar trend in the update success rate is also observed in presence of region of interest (Fig. 5.15(d)). As the number of nodes increases, the update success rate also increases almost steadily, though the rate of increase is slightly lower in every location service in the presence of regions of interest (Fig. 5.15(c) vs. (d)).

Figure 5.16(a) shows the average number of hops required for the location update

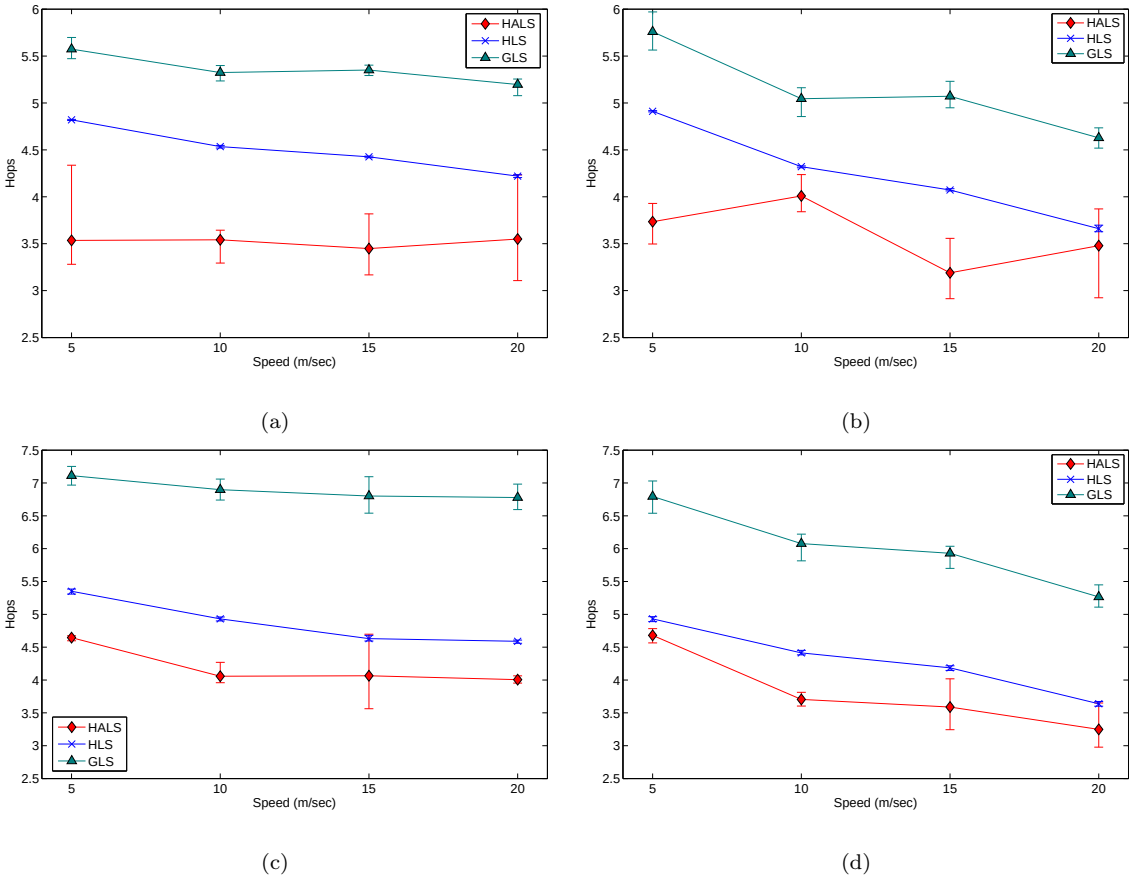


Figure 5.16: Average number of hops for location update at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

at different node movement speed in the free space scenario. Here, HALS requires the lowest number of hops for updating the location information compared to HLS and GLS. In HLS, because of empty home regions, in many cases nodes can not find their respective home regions and location is stored in the nodes along the perimeter of the empty home cells, which results in a higher number of hops than that of HALS. In GLS, the update packets need to travel a series of nodes in order to reach the location servers. Traversing a series of single nodes causes vulnerability in the location update scheme and thus GLS requires the highest number of hops.

With regions of interest (Fig. 5.16(b)), due to the presence of a large number of empty cells, the update packets roam around more before a successful update takes place. For this reason HLS needs a higher hop traversal for the update packets to

reach their destination compared to the without region of interest scenario (Fig. 5.16(a)) when the node speed is low. However, when the node mobility increases, the packet drop increases as well resulting in a lower number of hops at higher node movement speed as observed in Fig. 5.16(b). In this case nearer home regions are updated more successfully than farther homes, resulting in a lower number of hops. A similar effect is also observed with GLS.

In the presence of obstacles, the required number of hops is shown in Fig. 5.16(c) which exhibits a similar trend in result as observed in Fig. 5.16(a). However, the presence of obstacles causes the packets to travel around the obstacles to reach their home regions. Thus all the location services require a higher number of hops to update. The presence of regions of interest in the obstacle scenario (Fig. 5.16(d)) has similar impact as observed in Fig. 5.16(b) on the required number of hops, and the nearer location servers are updated more successfully than the farther servers at higher node movement speed.

Figure 5.17 shows location update delay in various simulation environments. In Fig. 5.17(a), when the nodes move following the Random Waypoint model, GLS and HLS exhibit the highest and the lowest update delay, respectively, irrespective of the node movement speed. In this case HALS shows a slightly higher delay than HLS, though it requires a smaller number of hops (Fig. 5.16(a)). Since the nodes in an R_i home in HALS work as the location servers for all nodes within its L_{i+1} cell, a higher channel congestion causes a higher delay than that of HLS.

In the presence of regions of interest (Fig. 5.17(b)), the nodes become concentrated within the regions of interest resulting in a higher network congestion. Thus the update packet delivery delay increases compared to the without region of interest scenario (Fig. 5.17(a)). At higher node movement speed, the nodes become concentrated more rapidly, and for this reason when node speed is high, all the location services exhibit a higher delay compared to that of without region of interest.

However, in the obstacle scenario (Fig. 5.17(c)), it is observed that HALS experiences almost same delay as without obstacle scenario, but the delay in HLS becomes more than that of HALS. Thus the non-uniform node distribution caused by the environmental contexts (e.g., obstacles, pathways) has higher impact on the delay metric in HLS because of the empty home cells. It should be noted that in the case of an empty

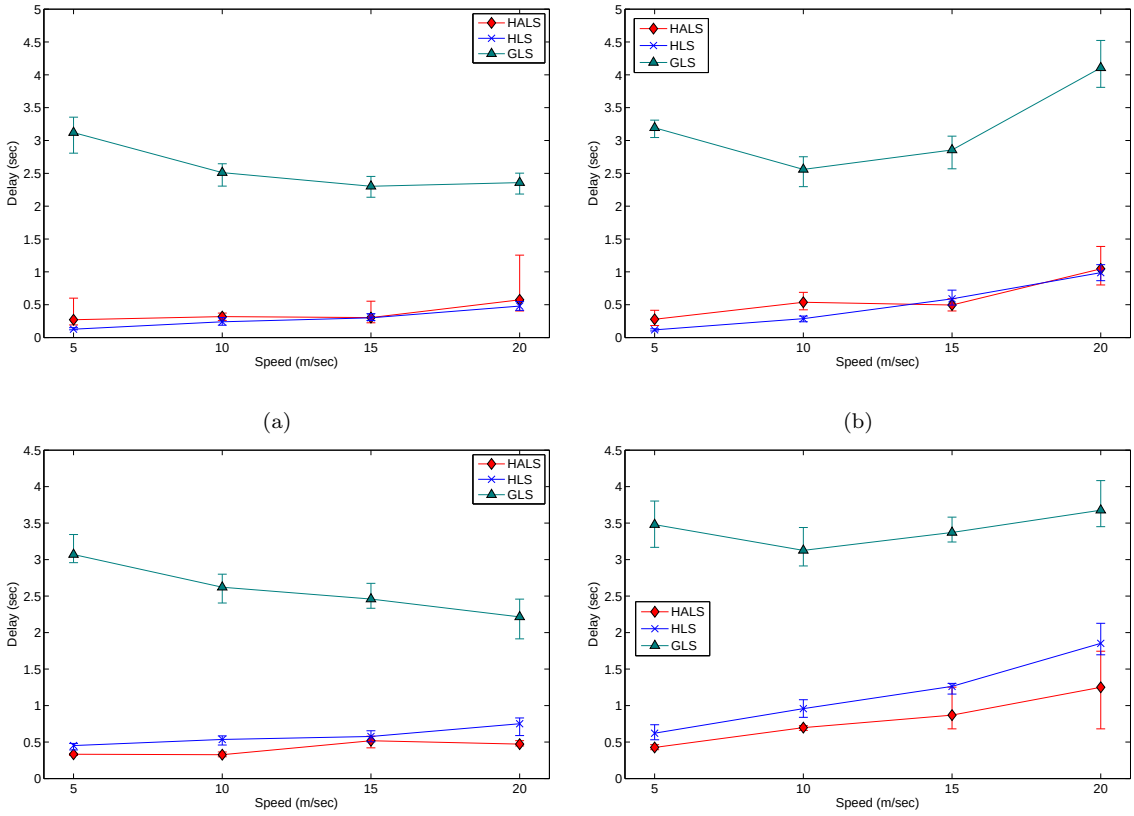


Figure 5.17: Average delay of location update at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

cell, the nodes creating the perimeter around that cell wait to forward the pending update packet until any node enters that cell. Thus HLS experiences a higher delay. Furthermore, the presence of regions of interest in the obstacle scenario (Fig. 5.17(d)) causes a higher delay in the location update when the node speed is higher, because of building network congestion more rapidly.

The delay for location update at various node density is presented in Fig. 5.18 where the maximum node speed is 10 m/sec. In the free space scenario (Figs. 5.18(a) and (b)), the relative performance of the location services based on delay metric remains the same as Figs. 5.17(a) and (b), respectively. HALS requires a slightly higher delay than HLS to update the locations in free space (Fig. 5.18(a)). The rate of increase in the delay with the node density is higher than that of HLS in the presence of regions

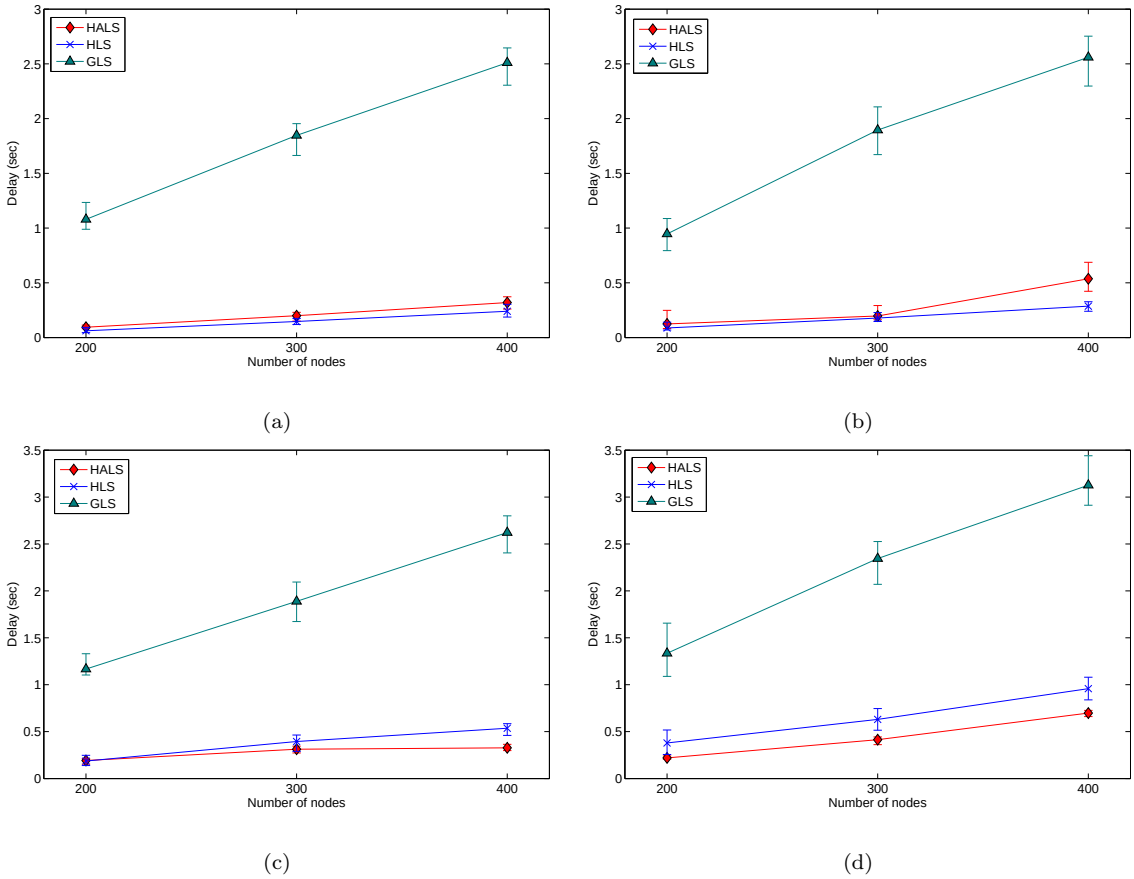


Figure 5.18: Average delay in location update at different node density (at 10 m/sec) in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

of interest (Fig. 5.18(b)).

However, in the presence of the obstacles (Fig. 5.18(c)), HLS suffers a longer delay than that of HALS for update as the node density increases. In the presence of regions of interest (Fig. 5.18(d)), due to the increased network congestion all services experience a higher delay than that of without region of interest scenario (Fig. 5.18(a)). However, it is evident from the figures that the delay in GLS increases more rapidly than the other schemes with increased node density. In this case GLS faces difficulty in finding an appropriate node in the forwarding chain of nodes. On top of this, the added congestion caused by the increased number of nodes results in higher delay in the location update.

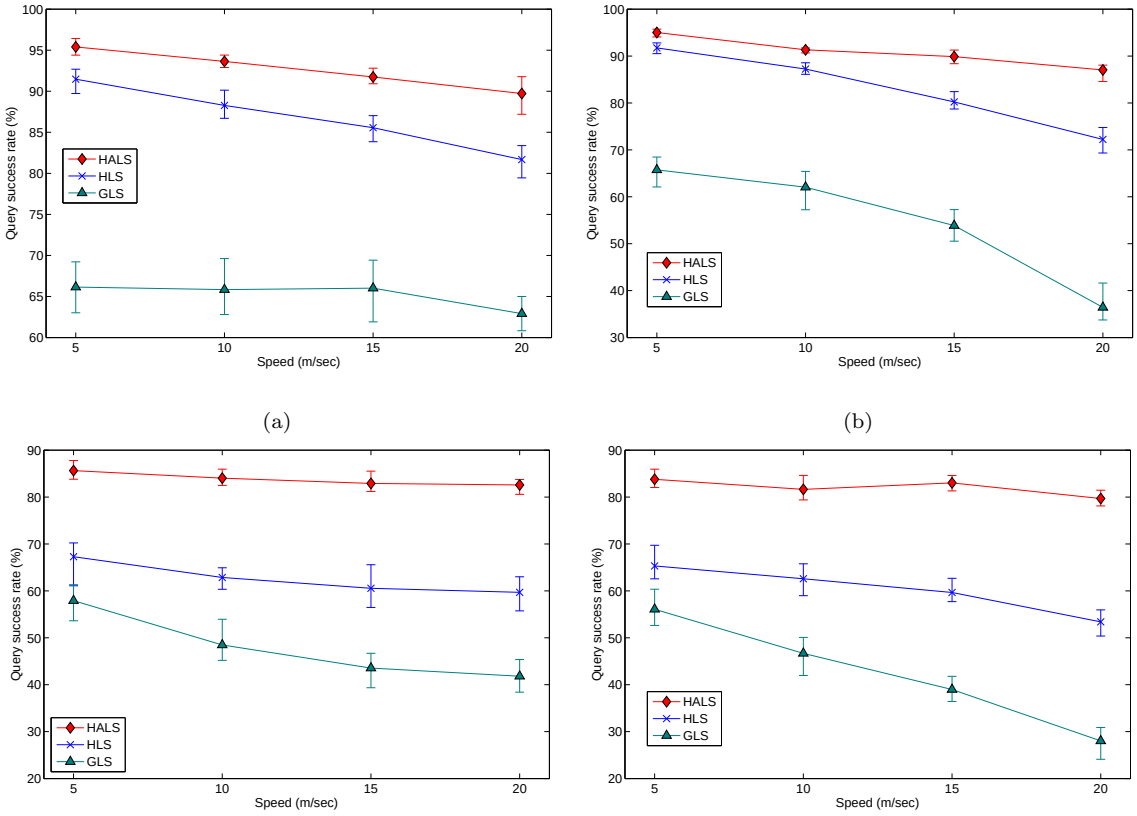


Figure 5.19: Location query success rate at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

5.5.2 Location Query

Figure 5.19(a) shows the query success rate of GLS, HLS and HALS at different speed while the nodes move following the Random Waypoint model in free space. The result shows that GLS has the lowest success rate irrespective of the node movement speed. On the other hand, HALS performs the best among the location services. Similar to the update method, GLS needs to forward the query packet through a chain of nodes in order to retrieve the appropriate location server. This results in frequent query failure in GLS.

As the node movement speed increases the performance of HLS degrades more rapidly than HALS. Every node in HLS is involved in the handover of location infor-

mation when they cross the smallest cell, as a result at increased speed the rate of handover increases as well. Whereas in HALS, the number of home cells is less than that of HLS and the handover process is limited to ranked cells only. So, increased node mobility has less affect on HALS. Moreover, using a higher number of cells for the composition of an L_1 cell (5×5 vs. 2×2) results in fewer hierarchy levels in HALS. Furthermore, a lower update success rate for HLS causes relatively lower query success rate because of the failure to update the servers properly. When the location is not updated properly in the servers, they can not provide the up-to-date node location.

The effect of lower update success rate has a direct impact on the query success rate of HLS which drops drastically in the presence of regions of interest at higher node speed shown in Fig. 5.19(b). As observed in this figure, at 10 m/sec speed HALS has 92% success rate compared to 85.8% of HLS, but when the speed increases to 20 m/sec, HALS shows 87.5% compared to 71.5% of HLS. This is because in presence of regions of interest, the empty cells are generated at a faster rate which causes failure in finding the home regions to response queries in many cases. Again, since GLS relies on a series of single nodes to reach the appropriate location server, it shows the lowest success rate. Moreover, when the speed increases the neighbors' locations stored in neighbor list become outdated more frequently. On top of this, regions of interest result in increased network congestion and the packet delivery delay increases. This makes locating the chain of nodes more complicated, and the success rate drops rapidly as the node movement speed increases.

In the obstacle scenario (Fig. 5.19(c)), the presence of obstacles lowers the query success rate of every location service scheme irrespective of the node movement speed. However, HLS and GLS suffer more compared to HALS. Additionally, HALS is more resilient to node mobility compared to GLS and HLS. In the presence of regions of interest in the obstacle scenario (Fig. 5.19(d)), HALS maintains almost the same query success rate at different node movement speed since HALS ensures the presence of home regions all the time. However, for HLS, a query needs to be forwarded to an appropriate lowest level home of the queried node. The non-uniform node distribution caused by the environmental context results in many home regions to become empty. Thus the probability of query drop is higher, because the query is forwarded through many home regions one after another at different levels. It should be noted that the

query needs all the home regions at each level to be responsive, otherwise failure occurs. A similar phenomenon also affects GLS, as the packets need to travel following a chain of nodes to retrieve the location of the queried node. Moreover, higher node mobility in the presence of region of interest makes the situation worse as the generation of empty cells becomes faster. For this reason the query success rate degrades drastically when the node movement speed is higher in the presence of regions of interest.

The interesting part is, though the location update success rate in HLS is significantly lower (Fig. 5.14) than GLS, the query success rate is relatively higher. In the implementation of HLS, it is found that a location caching mechanism has been implemented which improves the query success rate. In HLS, when a node forwards an update packet towards the destination, all the intermediate nodes engaged in forwarding the packet, as well as any other nodes that overhear the packet forwarding in the MAC layer store the location information. Thus it can be concluded that a high number of location queries are not answered from the designated home regions, rather from other nodes that have cached data and therefore, the query success rate is higher compared to the update success rate. On the other hand, since the success rate of the location update in HALS is higher and the home regions never become empty, the query success rate is also significantly higher than HLS.

Figure 5.20 presents the location query success rate at various node density for each of the concerned location services. In Figs. 5.20(a) and (b) where there is no obstacles, HALS maintains almost same query success rate irrespective of the node density. However, in GLS and HLS the success rate increases as the number of nodes increases. Thus HALS's query success rate is less sensitive to the node density, making HALS attractive for use in various node density from low to high. In the presence of obstacles (Fig. 5.20(c)), it is observed that all the location services' query response capability increases at higher node density. A similar trend is also observed in HLS and GLS in the presence of region of interest (Fig. 5.20(d)). On the other hand, HALS maintains a steady query success rate at different node density. The capability of HALS's steady query response performance once again highlights the effectiveness of ensuring the existence of the home regions all the time.

Figure 5.21 presents the cache hit rate of different location services with respect to the speed. Comparing to query success rate it is observed that cache hit rate follows

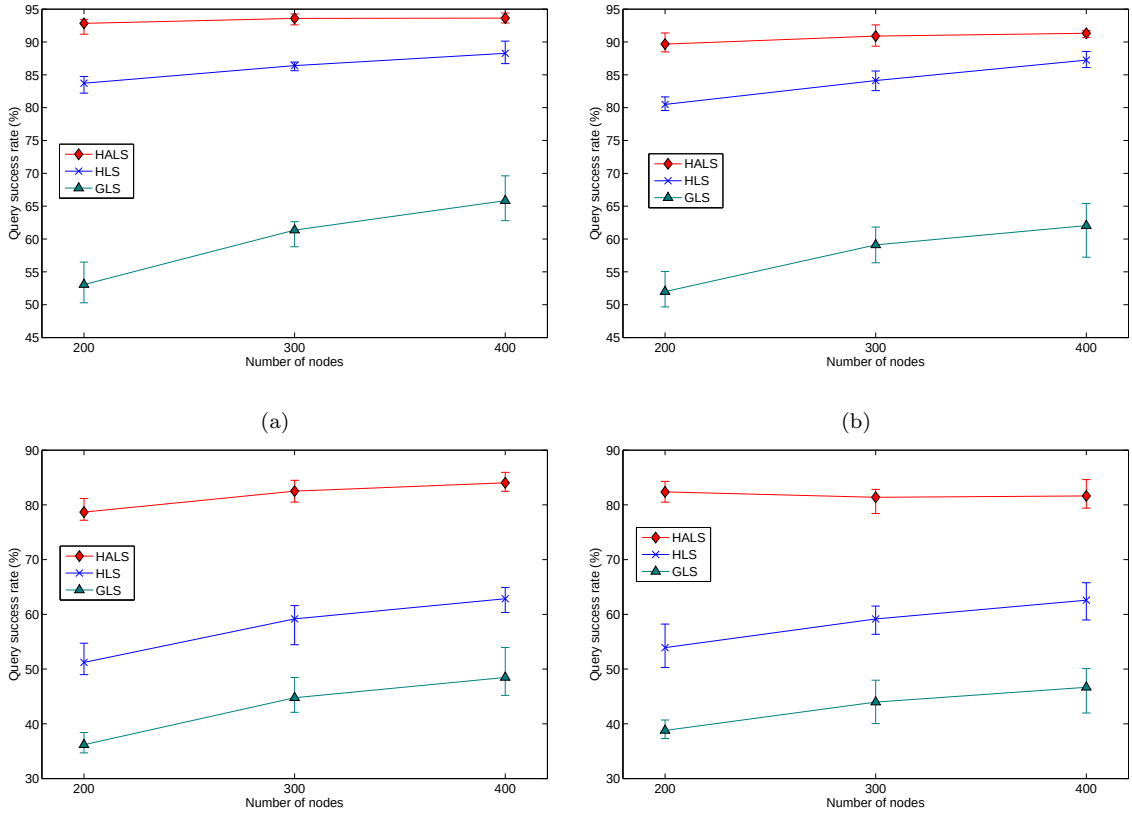


Figure 5.20: Location query success rate at different node density (at 10 m/sec) in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

similar trend as observed in the query success rate shown in Fig. 5.19. The cache hit rate calculates the percentage of queries that can locate the location of the queried node successfully. Since this metric includes the reply packets that were dropped on the way to the querying node, the relative value is slightly higher than the query success rate metric (Fig. 5.19).

The average number of hops required for the query response is plotted in Fig. 5.22. The results show that in free space (fig. 5.22(a)) HALS requires the lowest average number of hops among all the location services while GLS requires the highest number of hops. In HALS, higher number of cells for composing an L_1 cell results in fewer levels in the hierarchy compared to that of GLS and HLS. Thus GLS and HLS require traveling through a higher number of server regions/nodes in order to find the absolute location of the queried node. Moreover, in HLS, when a query packet can not find a

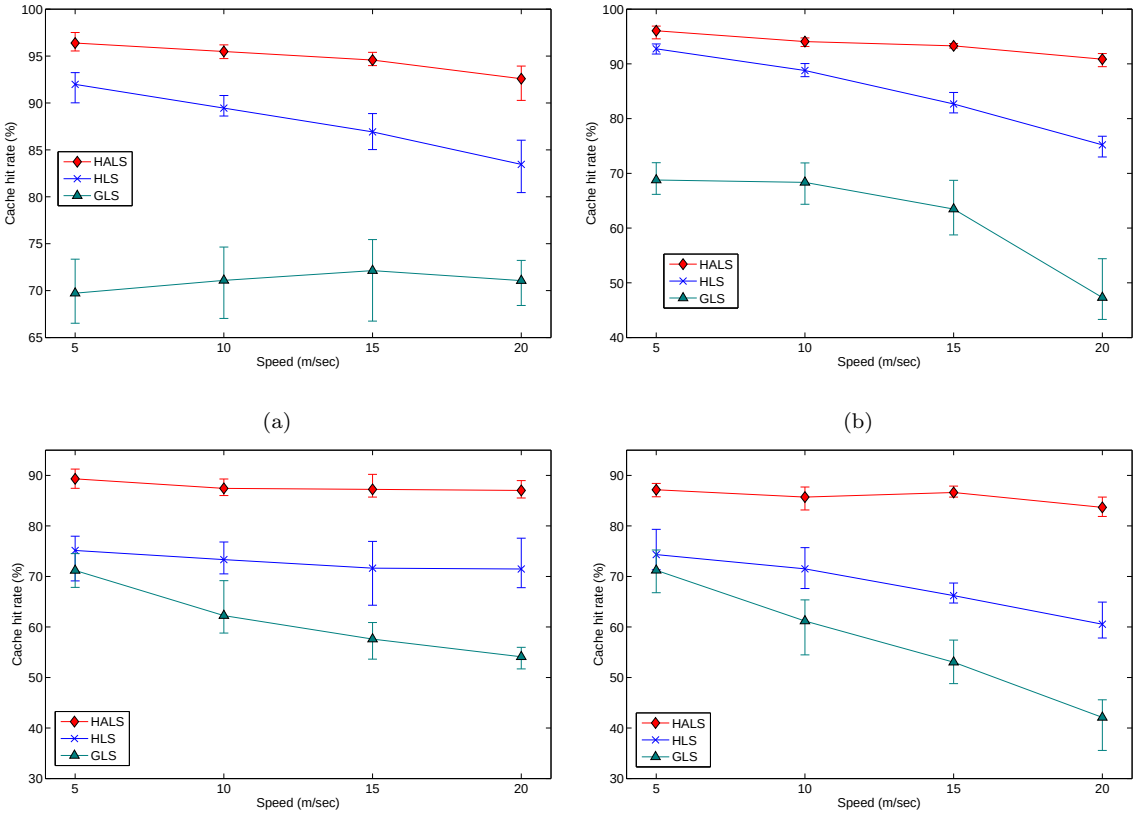


Figure 5.21: Query cache hit rate with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

home region (due to the home being empty), it is forwarded around the empty home region expecting that there might be nodes that may answer the query. Thus the required number of hops becomes higher than HALS where the existence of the home regions is always ensured.

In should be noted that for HLS a lot of queries are responded to by the intermediate nodes using cached data. Thus in many cases the query packets do not need to travel to higher levels in the hierarchy. For this reason HLS requires a fewer number of hops on average as compared to that of GLS. When the nodes move at higher speed in presence of regions of interest (Fig. 5.22(b)) GLS requires fewer hops. Referring to the query success rate in a similar scenario (Fig. 5.19(b)), it is observed that GLS's query response is very low in high node mobility scenario. This phenomenon again establishes that in high node mobility with regions of interest, the queries for the further nodes

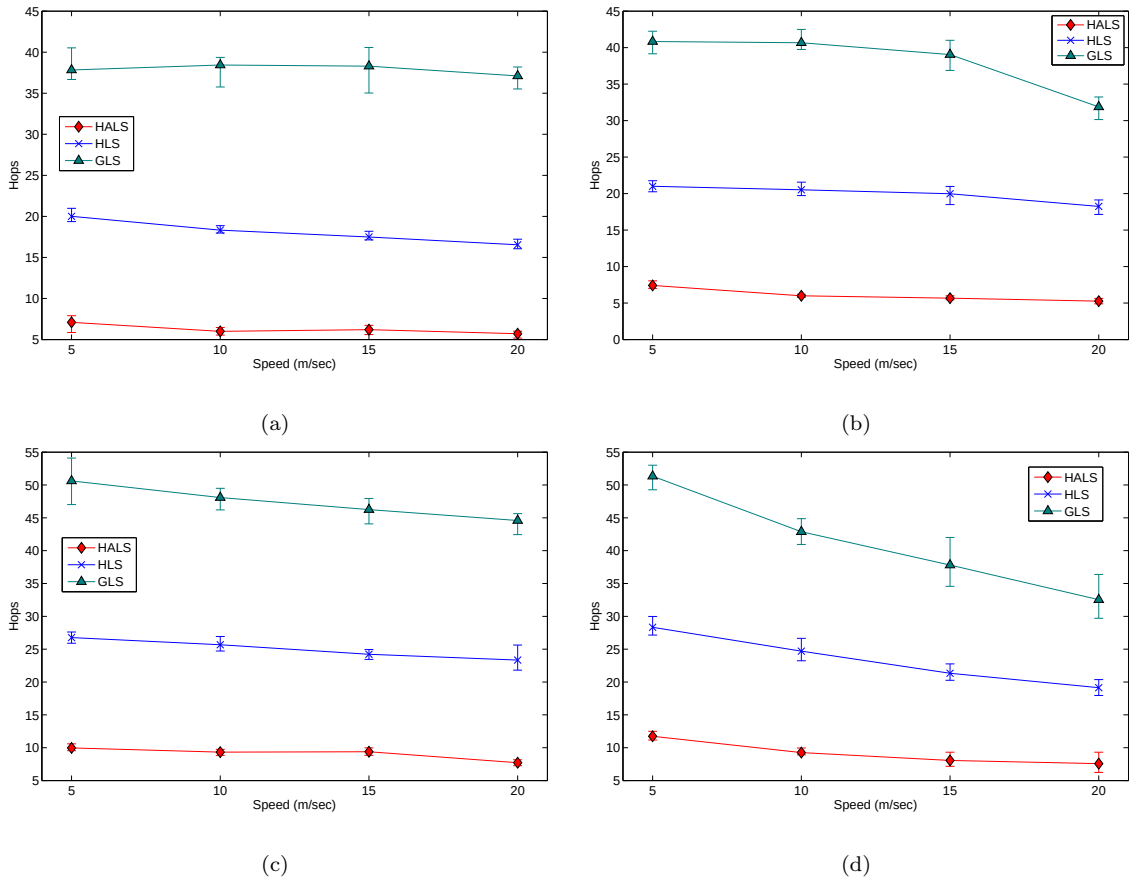


Figure 5.22: Average number of hops for location query at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

are not answered properly.

Figure 5.22(c) shows the required number of hops in the obstacle scenario, where the average number of hops increases for every location services compared to the non-obstacle scenario (Fig. 5.22(a) vs. (c)), because the packets need to travel around the obstacles - similar to what happens in the location update procedure. However, when the nodes become non-uniformly distributed due to the presence of regions of interest (Fig. 5.22(d)), the number of hops decreases rapidly for HLS and GLS as the node speed increases, since mostly the queries for nearer nodes are answered.

The delay in the query response in various location services with respect to the node mobility in different scenarios is presented in Fig. 5.23. Figure 5.23(a) shows the response delay in the free space scenario. Here it is observed that HALS results in the

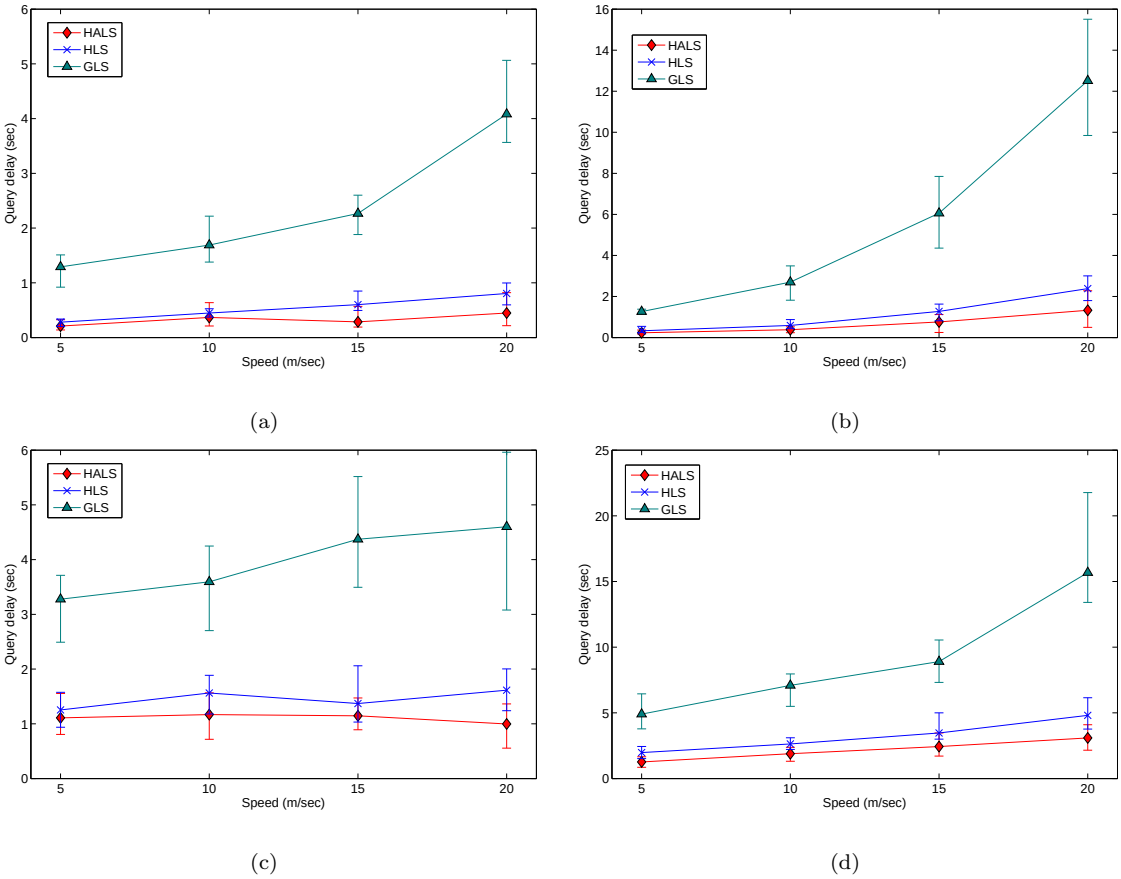


Figure 5.23: Average delay of location query at different node movement speed with 400 nodes in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

lowest delay among all the services. Though HLS requires a much higher number of hops for the query than that of HALS, the query response time is slightly higher. Since the home regions of the nodes are distributed uniformly throughout the simulation area in HLS, the query and update packets need to travel throughout the area, making the network load distributed as well. As mentioned before, unlike HLS, the home regions in HALS work more like a centralized fashion in their respective region and need to receive query packets as well as update packets. Since the number of homes is much less than that of HLS, all traffic is directed towards these regions resulting in higher network congestion. Subsequently, the delay increases and the difference in delay between HLS and HALS becomes less significant compared to the difference observed in the measurement of query hops in Fig. 5.22. The query response delay increases

for GLS as the node speed increases, mainly because of failure to locate appropriate location servers in higher node mobility.

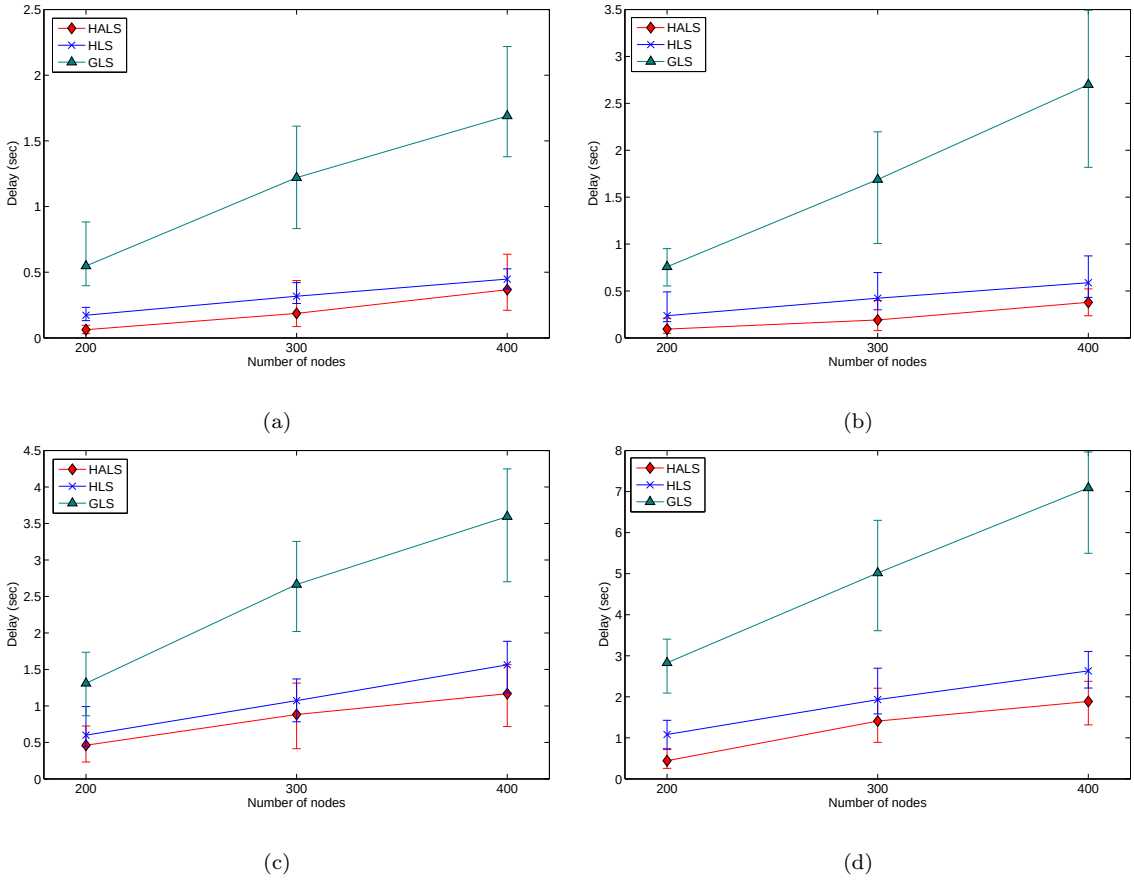


Figure 5.24: Average delay of location query at different node density (at 10 m/sec) in (a) free space scenario, (b) free space with region of interest scenario, (c) obstacle scenario based on real-world map, and (d) obstacle scenario based on real-world map with region of interest.

Figure 5.23(b) shows the query response delay in the presence of regions of interest. Regions of interest make the node concentration high and increase the network congestion. At higher node movement speed, the network congestion occurs faster and thus the delay in response becomes higher. In the presence of obstacles (Fig. 5.23(c)) for HALS and HLS, the query response delay increases slightly with the node speed. Since the packets need to travel around the obstacles, the response delay increases compared to the free space scenario (Fig. 5.23(a) vs. (c)) - a similar phenomena observed in the location update delay (Fig. 5.17). In the presence of regions of interest HALS and HLS show a slightly increasing trend in the delay as shown in Fig. 5.23(d), whereas the delay in GLS increases drastically as the node speed increases.

The query delay with respect to different node density is presented in Fig. 5.24. It is observed that the trend in the query delay is very similar to what is observed in the location update delay with respect to the node density as presented in Fig. 5.18. However, unlike the update packets, the query packets need to travel to home regions located in different hierarchies one after another, thus the delay in the query process becomes longer than that of the location update.

To test the significance of difference in the query success rate among HALS, HLS and GLS, HALS and HLS were selected as representatives of the proposed and existing models, respectively. The t-test at speed of 5, 10, 15 and 20 m/sec yielded p-values of $p < 2.47 \times 10^{-8}$, 4.73×10^{-11}, 4.77×10^{-9}, 2.92×10^{-10} for the obstacle scenario (Fig. 5.19(c)) and $p < 3.91 \times 10^{-8}$, 4.73×10^{-10}, 1.20×10^{-11}, 6.88×10^{-12} for the obstacle scenario with region of interest (Fig. 5.19(d)) at 99% confidence level , validating their performance difference being statistically significant.

5.5.3 Summary

The above analyses highlight the following key attributes of HALS:

- HALS outperforms GLS and HLS significantly in the location update and query success rates by adopting a mechanism that always maintains non-empty home regions.
- The node speed and density has less impact on the update and query success in HALS than in HLS and GLS.
- Though HALS maintains its superiority in both uniform and non-uniform node distributions, the storage overhead for the servers is not uniformly distributed among the nodes.

5.6 Conclusion

The existing location service protocols are designed assuming that the nodes will be uniformly distributed. Therefore, their performance degrades significantly in non-uniform node distributions which may arise in many practical scenarios where the nodes may

concentrate in specific regions for a particular event. To address this issue a hierarchically organized location service has been proposed in this chapter that dynamically creates, shifts and destroys the home regions making it able to cope well with empty home regions resulting from non-uniform node distribution while maintaining scalability verified through an analytical model. The characteristics of the proposed location service are analyzed through simulation in details with varying node density and distribution, and the simulation results have shown significant improvement in the location service measurement metrics.

It is important to understand that the main responsibility of a location service is to aid the geographic routing protocol by providing the location of a destination node to the source node so that the source node can send data to the destination. In order to perform the tasks, the location services use the underlying geographic routing protocol for forwarding its own messages (e.g., the update and query). The better the routing protocol, the higher the performance of the location service will be. The concerned routing protocols have been developed assuming ideal condition of environment (e.g., free space without any obstacle). As it has been observed in Chapter 4 that signal fading due to environmental attributes affects the performance of the routing protocol, it is necessary to investigate and integrate the environmental conditions in the underlying routing protocol as well so that the routing protocol can make a better decision when forwarding packets, which in turn will further improve the location service. In the following chapter this issue is addressed, and the performance of the proposed location service protocol is aimed to be improved further by extending the geographic routing protocol through the introduction of environmental awareness.

Chapter 6

Environment Aware Hierarchical Adaptive Location Service

The real-world environment impacts on the network performance in two main ways. Firstly, the mobility of the nodes is influenced by the presence of different environmental contexts, e.g., obstacles, pathways, entry points to obstacles (buildings), hot spots etc. This influence on the node mobility subsequently results in non-uniformly distributed nodes, which was analyzed in Chapter 4 and a new mobility model was proposed that captures node movement within a real environment more accurately. In Chapter 5 we focused on developing HALS, a location service that considers this realistic phenomenon which has a design philosophy that concentrates on the scalability and adaptability for non-uniformly distributed nodes.

The second impact of a real environment is experienced in the form of affecting the radio signal propagation through signal attenuation. Since HALS is designed with the aim of suitability for a real-world environment, the impact of signal propagation can not be ignored. Like other location services, HALS defines the methodology to perform the task of a location service through various control messages (e.g., update, query), and these messages are actually forwarded by the underlying routing protocol e.g., GPSR. Though a signal propagation model is proposed in Section 4.5.2 where the attenuation is incorporated based on the obstacles present between the transmitter and receiver, the combined effect of node mobility and signal attenuation has a far greater impact on the routing performance. If the impact of the attenuation in signal propagation can be modeled and utilized for selecting the forwarding nodes in the routing layer, an improvement in the performance of HALS is expected. In this way environmental awareness can be introduced to the working principles of HALS.

In this chapter, we focus on enhancing GPSR [15], one of the very promising location based routing protocols, by introducing awareness of environmental attributes and considering the path loss due to obstacles while making routing decision. This enhancement is equally suited for use in any other location based routing protocol (e.g., DREAM [12], LAR [13]). The enhancement is implemented in ns-2, analyzed and evaluated through simulation based on a university campus map, and the results show a significant improvement in the performance of GPSR. After that, environment aware GPSR is integrated with the proposed location service HALS in order to further enhance the performance of HALS.

This chapter is organized as follows: Section 6.1 presents the working principles of GPSR in a more detailed way to identify the potential aspects where the environmental impact can be introduced. How the mobility information (speed and direction) aids GPSR is explained in Section 6.2. In Section 6.3 we present the proposed environment aware GPSR. In Section 6.4, simulation results of the proposed enhanced GPSR with the existing ones are discussed. Section 6.5 integrates environment aware GPSR with our proposed location service and the outcome is presented through simulation results. Finally Section 6.6 draws the conclusion.

6.1 Revisiting GPSR

Greedy Perimeter Stateless Routing (GPSR) [15] is a routing protocol which uses the geographic location of nodes in order to forward data packets. GPSR does not need to maintain a routing table, which makes it very attractive as a routing protocol for wireless ad hoc networks. This routing protocol is comprised of three phases: 1) generating neighbor list, 2) forwarding packets in greedy mode and 3) forwarding packets in perimeter mode, as described in the following:

Generating neighbor list: As mentioned earlier, GPSR does not maintain any routing table; rather it maintains a neighbor list - a list of neighboring nodes residing within a node's radio transmission range. This is done by transmitting periodic *beacons*. It assumes that each node can identify its own location through GPS, or any other methods proposed in [16, 17, 18]. Any arbitrary node A in GPSR periodically sends a packet called beacon which contains A's ID and location.

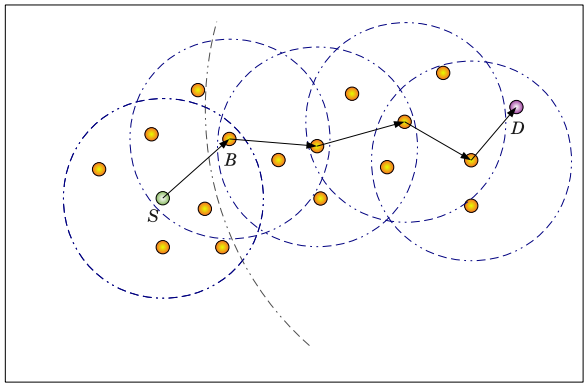


Figure 6.1: Greedy mode packet forwarding scheme.

This beacon can be heard (received) by the nodes residing within A's transmission range. Upon receiving the beacon, the neighboring nodes add A's location to their respective neighbor lists. Similarly, the beacons generated by neighboring nodes can also be *heard* by A and subsequently, A builds a neighbor list. Through this process every node becomes aware of its one hop neighbors. Whenever a node does not receive a beacon from an existing neighbor for a certain period of time, that particular neighbor is deleted from the list, and it is assumed that the neighbor has moved out of transmission range.

However, the beacon interval plays a significant role on the performance of position based routing. A larger interval results in less beacon overhead but increases the location error and consequently adversely impacts successful packet delivery. On the other hand, if the beacon interval is small, the nodes can maintain a neighbor list with more accurate location information but the MAC layer congestion increases resulting in an increased end to end delay.

Forwarding packets in greedy mode: When a data packet needs to be transmitted, GPSR starts with *greedy* forwarding. In this scheme a node forwards a packet to one of its neighboring nodes (maintained in the neighbor list) which is the closest to the destination among all its neighbors. This process continues until the destination node is close enough to receive the packet. As shown in Fig. 6.1, node S needs to send a packet to D. Thus S selects one of its neighbors, namely B which is the nearest to D and forwards the packet. Similarly, B forwards the packet to its neighbor that is closest to D. In this way the packet

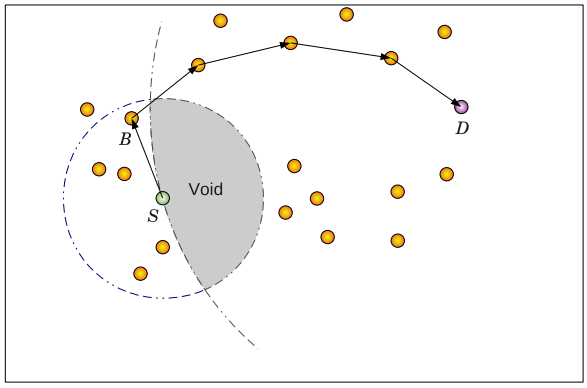


Figure 6.2: Perimeter mode packet forwarding scheme.

is finally forwarded to D.

This scheme works well in scenarios where the nodes are densely populated. However, if the node density is low, there may arise cases where a node does not have any neighbor closer to the destination than itself. In this situation greedy forwarding fails to deliver the packet. To solve this problem perimeter mode packet forwarding was introduced.

Forwarding packets in perimeter mode: In situations where a greedy path does not exist (i.e., a node can not find a neighbor closer to the destination than itself), GPSR recovers by adopting *perimeter* mode forwarding where a packet traverses successively closer faces of a planer sub-graph, until it reaches a node that is close enough to the destination to forward the packet. While a packet is being forwarded in perimeter mode, each node checks whether any of its neighbors is closer to the destination than the distance between the destination and perimeter mode initiating node. If it finds such a suitable neighbor, the packet forwarding is resumed to greedy mode again.

As shown in Fig. 6.2, the intersection of two circular areas, one being the transmission disk of S and the other circle centered at D with the radius equal to the distance between S and D, is called the *void* (marked in grey). Greedy forwarding fails when a packet reaches a node with void, as no neighbor is available there. Perimeter forwarding actually finds a route around this void (though only in cases where a route exists). In this case, S selects node B which is farther from D than

S itself. Subsequently, B forwards the packet following the perimeter until the packet is received by D.

To calculate the sequence of edges, the graph is *planarized* to a Relative Neighborhood Graph (RNG) [107] or Gabriel Graph (GG) [108]. The planarization method removes crossing edges and produces a planner graph in a distributed manner using only the local network (neighbor) information of nodes. After that by following the right hand rule, a sequence of edges creating the perimeter around the void is used to route the packet towards the destination. Perimeter mode forwarding improves the routing performance of GPSR significantly.

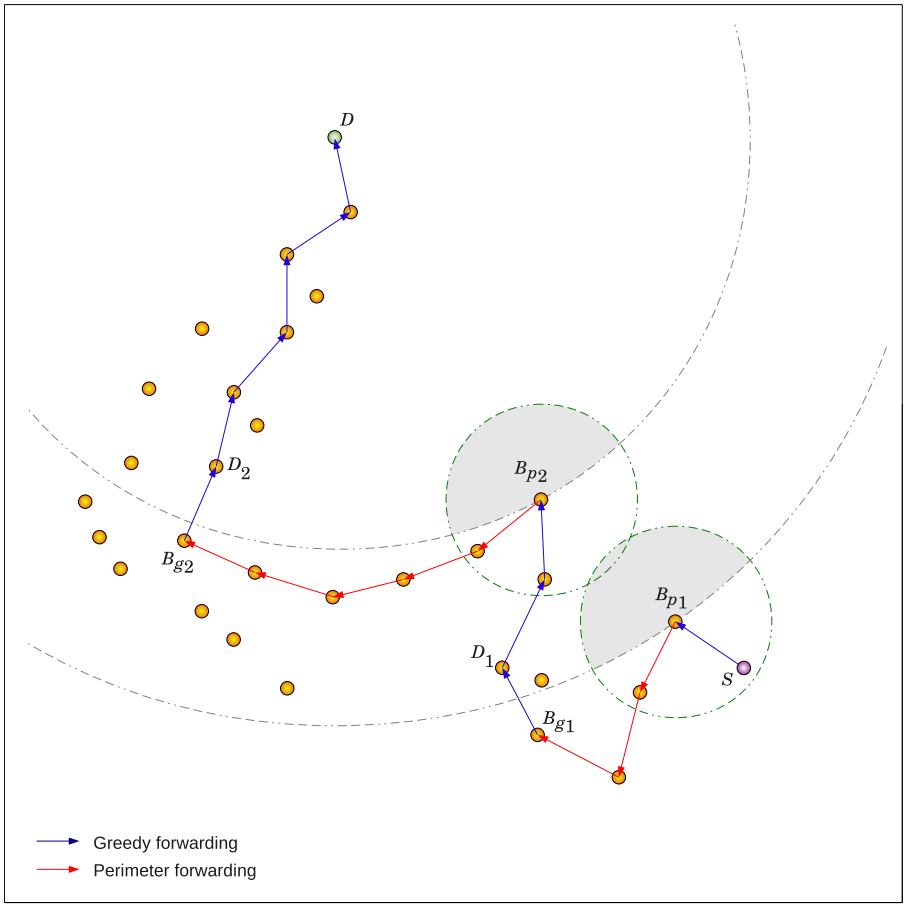


Figure 6.3: An example of packet forwarding in GPSR.

Figure 6.3 shows a routing path composed of both greedy and perimeter mode forwarding. Here, node S wants to send a packet to D and forwards it to B_{p1} in the greedy mode as B_{p1} is the nearest node towards D among all the neighbors of S. B_{p1}

however, experiences a void region and initiates perimeter mode forwarding, and the packet eventually follows the nodes making a perimeter and arrives at B_{g1}. B_{g1} finds that it has a neighbor D_1 which is nearer to D compared to the distance from D to B_{p1} (where the perimeter mode started), thus resumes the greedy mode by forwarding the packet to D_1. Following the greedy approach the packet is received by B_{p2} which again initiates perimeter mode until the packet reaches B_{g2}. In this way greedy and perimeter forwarding continues to take place one after another, and finally the packet is received by the destination node D.

6.2 Mobility Aware GPSR

Through the analysis of the working principle of GPSR in Section 6.1, it is evident that the accuracy of nodes' position is significantly important for GPSR because each node makes routing decisions based on the location of nodes stored in its neighbor list composed through periodic beacons. The main reason behind the inaccuracy in the location of a node is its mobility. As the nodes are mobile, each node can only maintain the location of its neighbors by the beacons produced by them. Thus the location of a neighbor is out dated at most by the beacon transmission interval. When a node selects a neighbor as a potential next hop destination, the decision is based on the location received in the most recent beacon. Thus while packet forwarding is taking place, the potential next hop neighbor may already have moved away a certain distance, determined by its movement speed and direction. If after moving, the neighbor stays within the sender's transmission range, the packet is successfully forwarded. However, if the neighbor moves out of the node's transmission range, the packet is dropped. Since a node in GPSR tries to minimize the number of hops, the neighbor nearest to the boundary of the transmission range has the highest chance of being selected as the potential next hop neighbor. Sometimes, this situation causes to select a neighbor that is in the neighbor list, but when the packet forwarding takes place the selected neighbor has already moved away resulting in packet drop.

To address this issue, further improvement of GPSR has been proposed in the literature. Since the mobility is the key parameter for the inaccuracy of the nodes' location, Chen *et al.* [109] and Granelli *et al.* [110] proposed to incorporate a node's speed and direction of movement in the beacons. Chen *et al.* introduced an adaptive

beacon interval selection and neighbor list maintenance scheme based on the nodes' movement speed. Granelli *et al.* improved GPSR by considering the nodes' distance from the destination, speed and direction to select a more stable path while routing data packets, mainly focused for inter vehicular communication. Both of these schemes achieve considerable improvement over GPSR.

The speed and direction of node movement received through beacons is used in selecting the next hop neighbor in the following way:

Let $\aleph$ be the set of nodes in the network and for a node $u \in \aleph$, $\{u_1, u_2, \cdots, u_k\} \subseteq \aleph$ be its set of k neighbors. When node u needs to select the next hop it recalculates each neighbor u_i's position ($1 \leq i \leq k$) according to the following equations:

$$x'(u_i) = x(u_i) + (\Gamma - t_{beacon}(u_i)) \times \nu_X(u_i), \tag{6.1}$$

$$y'(u_i) = y(u_i) + (\Gamma - t_{beacon}(u_i)) \times \nu_Y(u_i). \tag{6.2}$$

Here, for a neighbor u_i, $(x'(u_i), y'(u_i))$ represents its estimated current position. $(x(u_i), y(u_i))$ is its location and $(\nu_X(u_i), \nu_Y(u_i))$ are the speed components in the X-direction and Y-direction respectively, received through beacon originated at time $t_{beacon}(u_i)$, and Γ is the current time. After calculating the updated position of its neighbors, u uses (6.1) and (6.2) and removes u_i from the neighbor list, if

$$distance_{curr}(u, u_i) > r_t, \tag{6.3}$$

where

$$distance_{curr}(u, u_i) = \sqrt{(x(u) - x'(u_i))^2 + (y(u) - y'(u_i))^2}, \tag{6.4}$$

and r_t is the transmission range. In this way by removing potentially misleading neighbors, u maintains its neighbor list more accurately and selects a more appropriate neighbor as the next hop while forwarding packets. Thus in mobility aware GPSR, while forwarding packets each node recalculates an estimate of the current location of its neighbors based on their speed and direction. This makes it is possible to discard some of the neighbors who were in the transmission range while originating beacons, but moved away when the packet is being forwarded. Thereby, mobility aware GPSR can choose a more appropriate neighbor while making selection of next hop resulting in better packet delivery over the original GPSR.

6.3 Environment Aware GPSR

In a real-world environment, the presence of obstacles causes signal attenuation that affects the signal strength significantly. This phenomenon increases the level of inaccuracy in the nodes' location estimation one step farther. The mobility aware GPSR presented in Section 6.2 exploits the nodes' speed and direction information to maintain a more accurate neighbor list. This enhancement reduces the level of inaccuracy in the location of neighbors. However, this enhancement alone can not provide reliable accuracy in an obstacle scenario, and if the environmental influence is also taken into account a further improvement in the routing performance of GPSR is expected.

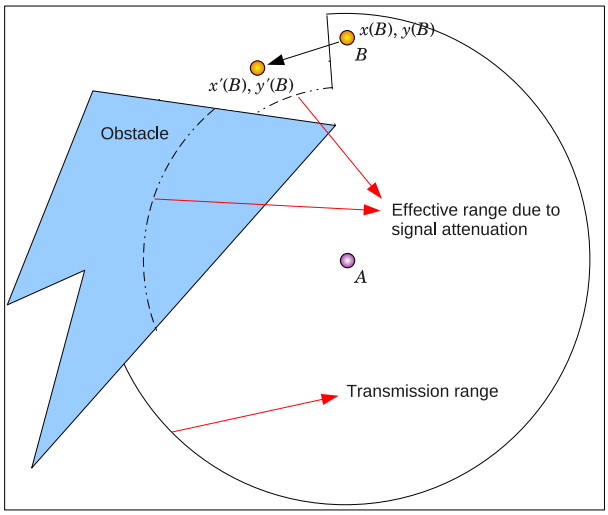


Figure 6.4: Reduced transmission range caused by signal attenuation.

As shown in Fig. 6.4, node A receives a beacon from node B. When A needs to forward a packet, A calculates B's tentative location $(x'(B), y'(B))$ using its speed and direction. But due to the presence of the obstacle the effective transmission range is reduced. If A selects B as its next hop neighbor, the packet will be dropped. For this reason, the environment should also be taken into consideration in addition to the nodes' movement. This can be achieved by considering attenuation factor in the signal propagation model and calculating the effective transmission range in the routing protocol while forwarding data. The only requirement for this is to have prior knowledge about the position of the obstacles, which is possible by incorporating the map of the deployment terrain in each node.

The proposed Environment Aware GPSR (GPSR-EA) exploits the concept of mobility aware GPSR [109, 110]. Additionally it considers any environmental influence in making the routing decision, which is based on the radio propagation model presented in Section 4.5.2. This propagation model takes into account a series of walls as obstacles and calculates the effective received signal strength as follows:

$$P_r = \begin{cases} \dfrac{P_t G_t G_r {h_t}^2 {h_r}^2}{(D_{t,r})^\alpha \varphi} & \text{no obstruction;} \\ \dfrac{P_{rn}^a G_t G_r h_t^2 h_r^2}{\left(D_{t,r} - \sum_{i=1}^{n} d_i\right)^\alpha \varphi} & \text{obstruction.} \end{cases} \tag{6.5}$$

The parameters used in the above equation have been defined in Section 4.5.2.
To recognize the received signal, P_r must fulfill

$$P_r \geq R_x \tag{6.6}$$

where R_x represents the minimum signal threshold required to receive the signal successfully. Depending on whether there exist obstacles between the transmitter (node u) and receiver (node u_i), the following two situations can occur:

Case 1: No obstruction If there is no obstacle between the transmitter and receiver, the received power is calculated as:

$$P_r = \frac{P_t G_t G_r {h_t}^2 {h_r}^2}{(D_{t,r})^\alpha \varphi}. \tag{6.7}$$

Using (6.6) and (6.7),

$$\begin{aligned} R_x &\leq \frac{P_t G_t G_r {h_t}^2 {h_r}^2}{(D_{t,r})^\alpha \varphi} \\ \Rightarrow (D_{t,r})^\alpha &\leq \frac{P_t G_t G_r {h_t}^2 {h_r}^2}{R_x \varphi} \\ \Rightarrow D_{t,r} &\leq \left(\frac{P_t G_t G_r {h_t}^2 {h_r}^2}{R_x \varphi}\right)^{\frac{1}{\alpha}}. \end{aligned} \tag{6.8}$$

After calculating the updated position of the neighbors using (6.1), (6.2) and (6.4), u uses (6.8) and removes u_i from the neighbor list, if

$$distance_{curr}(u, u_i) > D_{t,r}. \tag{6.9}$$

Case 2: Obstruction If there are obstacles present in between the transmitter and receiver, the received power is calculated as:

$$P_r = \frac{P_{rn}^a G_t G_r h_t^2 h_r^2}{\left(D_{t,r} - \sum_{i=1}^{n} d_i\right)^{\alpha} \varphi}. \tag{6.10}$$

Using (6.6) and (6.10),

$$\begin{aligned} & R_x \leq \frac{P_{rn}^a G_t G_r h_t^2 h_r^2}{\left(D_{t,r} - \sum_{i=1}^{n} d_i\right)^{\alpha} \varphi} \\ \Rightarrow \quad & \left(D_{t,r} - \sum_{i=1}^{n} d_i\right)^{\alpha} \leq \frac{P_{rn}^a G_t G_r h_t^2 h_r^2}{R_x \varphi} \\ \Rightarrow \quad & D_{t,r} \leq \left(\frac{P_{rn}^a G_t G_r h_t^2 h_r^2}{R_x \varphi}\right)^{\frac{1}{\alpha}} + \sum_{i=1}^{n} d_i \end{aligned} \tag{6.11}$$

After calculating the updated position of the neighbors using (6.1), (6.2) and (6.4), u uses (6.11) and removes u_i from the neighbor list, if

$$distance_{curr}(u, u_i) > D_{t,r}. \tag{6.12}$$

Thus, as a general rule, a node u removes its neighbor u_i from the neighbor list if

$$distance_{curr}(u, u_i) > \begin{cases} \left(\frac{P_t G_t G_r {h_t}^2 {h_r}^2}{R_x \varphi}\right)^{\frac{1}{\alpha}} & \text{no obstruction;} \\ \left(\frac{P_{rn}^a G_t G_r h_t^2 h_r^2}{R_x \varphi}\right)^{\frac{1}{\alpha}} + \sum_{i=1}^{n} d_i & \text{obstruction.} \end{cases} \tag{6.13}$$

Figure. 6.5 shows how the effective transmission range varies due to the presence of a single wall with a distance from transmitter ($1 \sim 100m$) and path loss factor ($1 \sim 100$ dB). From the figure it is clear that increased attenuation causes a drastic reduction in the effective transmission range. If the signal attenuation in not considered, a node will add unreachable nodes to its neighbor list and if any of these unreachable neighbors is selected as next hop the packet will be dropped.

Through the enhancement procedure explained in this section, a node is able to maintain a more accurate neighbor list. The following section demonstrates the outcome of this enhancement through simulation.

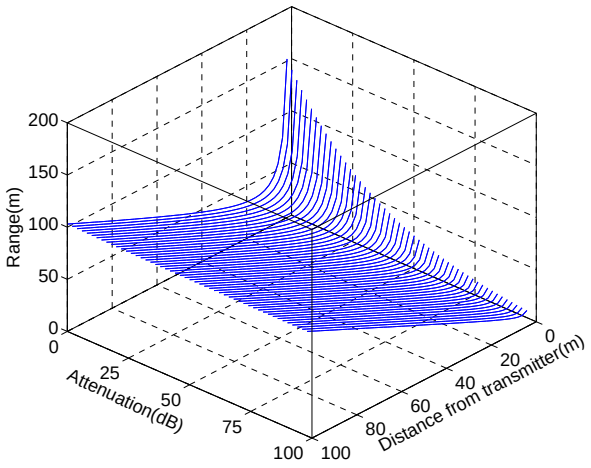


Figure 6.5: Effective transmission range through a single wall with varying attenuation factor and distance from transmitter.

6.4 Simulation Results

MobiGen (Section 4.3.4) was used to generate the scenarios, and as in the previous chapters these scenarios were based on a real context, the locations of buildings in the Clayton campus of Monash University, Australia as shown in Fig. 5.12. A portion in the map was selected keeping the simulation area size 2000m × 2000m, and within this terrain the existing pathways, buildings and the actual position of main entrances (doors) to the buildings were incorporated in MobiGen as shown in Fig. 5.13.

The ns-2.29 simulator was used for simulation where 200, 300 and 400 nodes were used in the simulation. Four maximum node movement speed of 5, 10, 15 and 20 m/sec were considered. Ten runs for each speed variation were performed and the averaged results are presented here. The parameters used in the simulation are summarized in Table 6.1.

The duration of the simulation for each run was 300 seconds. Every second 15 pairs of nodes were randomly selected, each pair consisted of one sender and one receiver, and a 128-byte packet was sent to the receiver. For routing the data packets, the original GPSR, mobility aware GPSR [110] and the proposed environment aware GPSR were used for comparison purpose, while the IEEE 802.11 protocol was used at the MAC layer.

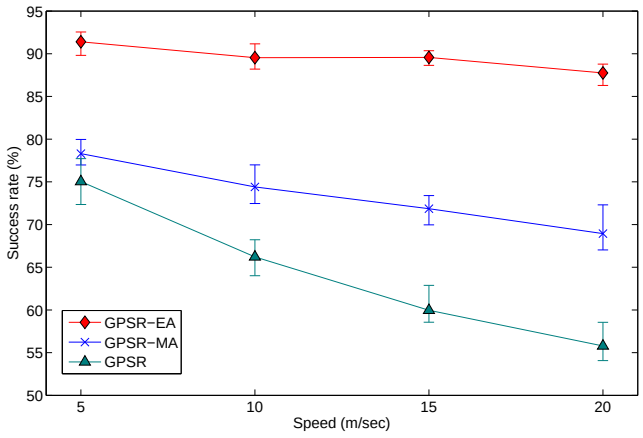
Table 6.1: Parameters used in simulation.

Notation	**Meaning**
Playground size	2000m × 2000m
Simulation duration	300 sec
Mobility model	AMM model with and without region of interest
Number of nodes N	200, 300, 400
Transmission range r_t	200m
Max node speed	5, 10, 15, 20 m/sec
Number of runs	10
MAC layer	IEEE 802.11

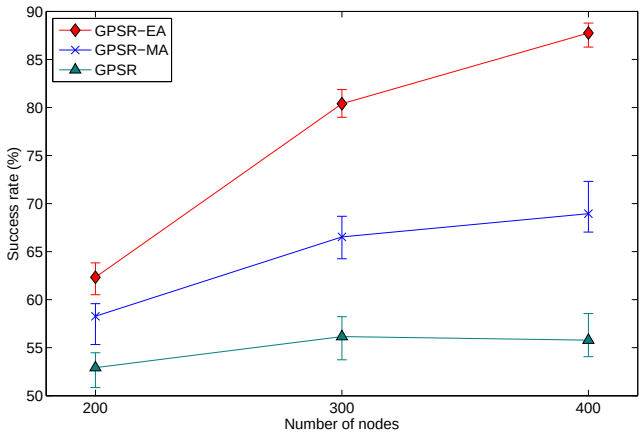
The packet delivery success rate of GPSR, Mobility Aware GPSR (GPSR-MA) and Environment Aware GPSR (GPSR-EA) at various node movement speed and node density is presented in Figs. 6.6(a) and (b), respectively. In Fig. 6.6(a) it is observed that GPSR-EA performs the best by ensuring the highest packet delivery rate. On the other hand, GPSR performs the worst, while GPSR-MA stays in between. According to this figure, it is evident that incorporating the nodes' speed and movement direction information (GPSR-MA) with the beacons results in more accurate location of the nodes, and consequently improves the packet delivery performance. Additionally, incorporating the environment information (GPSR-EA) enhances the success rate further by discarding potentially misleading neighbors from the neighbor list. Thus in GPSR-EA, a node maintains its neighbor list more accurately and can forward packets to a more appropriate neighbor. While the performance of GPSR and GPSR-MA show a declining trend with increasing node speed, GPSR-EA maintains almost the same packet delivery rate irrespective of the node movement speed.

To test the significance of the difference in packet delivery success rates between GPSR-EA and GPSR-MA, the t-test at speed 5, 10, 15 and 20 m/sec yielded p-values of $p < 4.21 \times 10^{-10}$, 1.17×10^{-10}, 3.56×10^{-11}, 1.28×10^{-10} at 99% confidence level, validating their performance difference as being statistically significant.

Figure 6.6(b) shows the packet delivery success rate at different node density. Similar to Fig. 6.6(a), here GPSR-EA maintains its superior performance. Moreover, the increase in the success rate in GPSR-EA is higher than that of GPSR-MA and GPSR



(a)

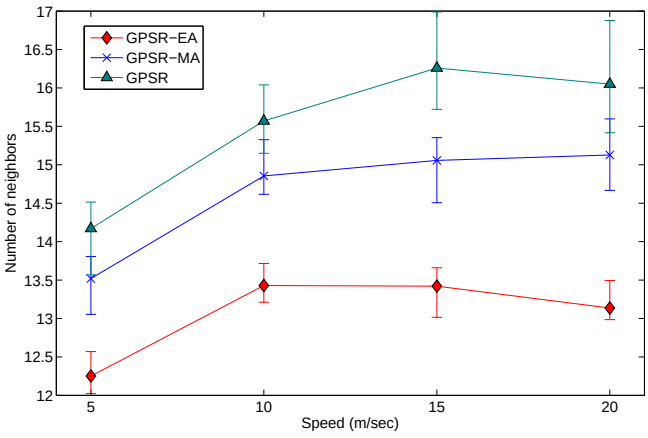


(b)

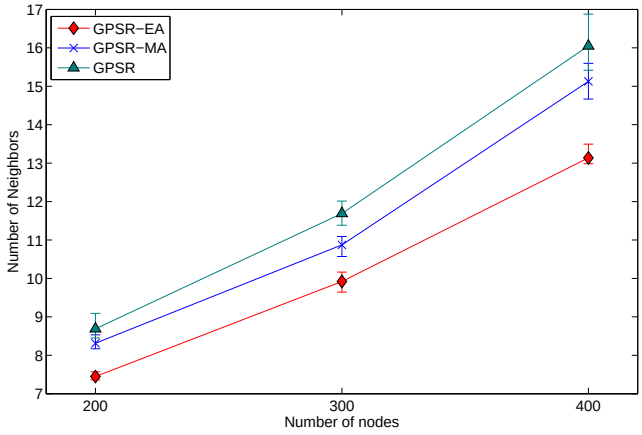
Figure 6.6: Packet delivery success rate at (a) various node movement speed (400 nodes), and (b) different node density (at 20 m/sec).

as the node density increases. This outcome highlights GPSR-EA's applicability in any variation of node density. As the node density increases, the number of neighbors for each node also increases and subsequently more packets are forwarded in greedy mode. Thus all variations of GPSR show an increasing trend in packet delivery at higher node density.

Average number of neighbors at different node movement speed and node density is presented in Figs. 6.7(a) and (b), respectively. In Fig. 6.7(a), GPSR shows the highest number of neighbors as expected. On the other hand, GPSR-EA has the least number of neighbors due to many neighbors being discarded based on both the environment and nodes' mobility. When the node density increases (Fig. 6.7(b)), all GPSR variations



(a)

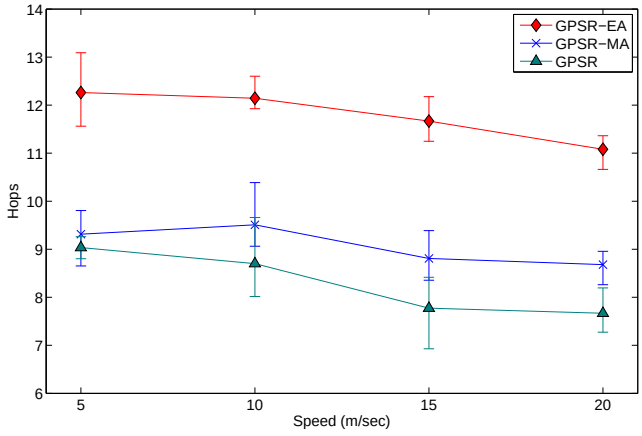


(b)

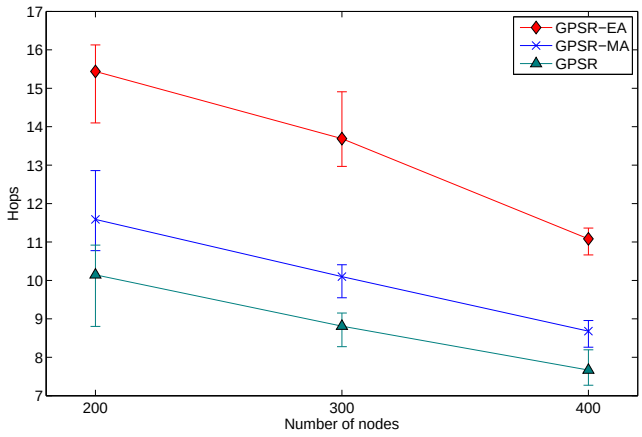
Figure 6.7: Average number of neighbors at (a) various node movement speed (400 nodes), and (b) different node density (at 20 m/sec).

show an increasing trend in the number of neighbors while their relative position in the graph remains the same, as observed in Fig. 6.7(a).

Figures 6.8(a) and (b) show the average number of hops required for GPSR, GPSR-MA and GPSR-EA at various node movement speed and node density, respectively. In Fig. 6.8(a), GPSR-EA requires the highest number of hops on average for the packets to reach their destination. Since more neighbors are discarded in GPSR-EA, packets are more often forwarded in perimeter mode resulting in more hops. In this regard we measured the number of hops that took place in perimeter mode forwarding (Fig. 6.9) which shows that perimeter mode forwarding is most utilized by GPSR-EA, irrespective of the node movement speed. Moreover, since in most cases the farther nodes in the



(a)



(b)

Figure 6.8: Average number of hops required for packet delivery at (a) various node movement speed (400 nodes), and (b) different node density (at 20 m/sec).

neighbor list are discarded, GPSR-EA uses nearer nodes as hops, which contributes to the increased number of hops as well. On the other hand GPSR needs the least number of hops as more packets are forwarded in greedy mode due to having more neighbors than GPSR-MA and GPSR-EA (Fig. 6.7).

Figure 6.8(b) shows the required number of hops at different node density. As the number of nodes increases, all variations of GPSR require fewer hops on an average. An increased node density aids in yielding a greater number of neighbors as observed in Fig. 6.7(b). This subsequently helps more packets to be forwarded in the greedy mode. Thus, at higher node density, the average number of hops to reach the destination decreases.

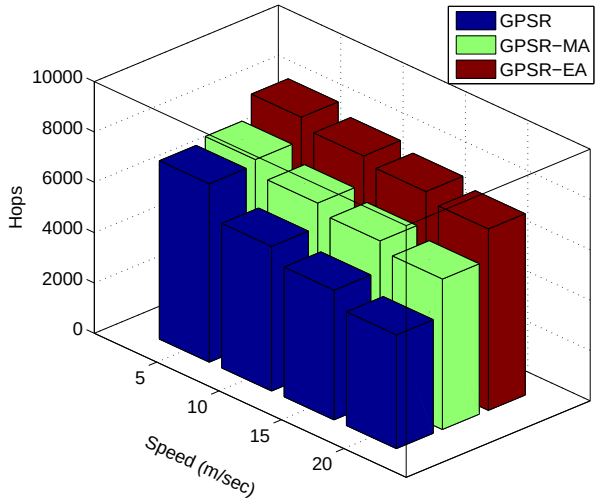
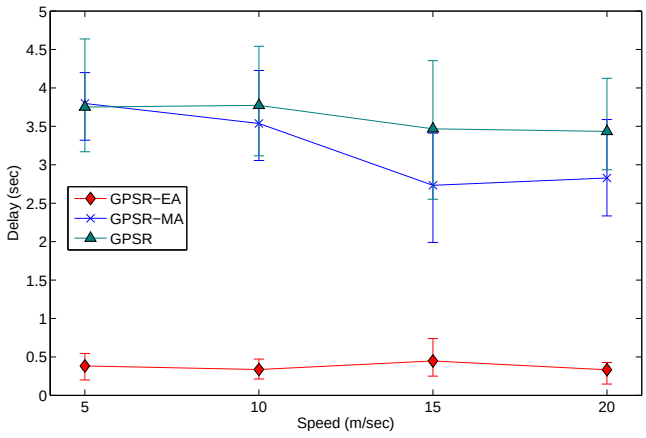


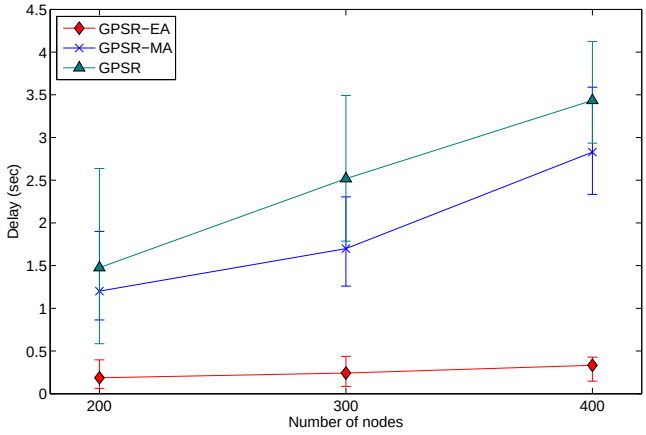
Figure 6.9: Average number of hops taking place in perimeter mode.

Figures 6.10(a) and (b) show the average packet delivery delay at various node movement speed and node density, respectively. In Fig. 6.10(a) it is observed that GPSR-EA exhibits the lowest delay in packet forwarding though it requires the highest number of hops (Fig. 6.8(a)) from the source to the destination. In order to investigate the reason behind this, we measured the average MAC layer delay experienced by the packets that were successfully delivered, as presented in Fig. 6.11. From this figure it is clear that GPSR-EA experiences the lowest MAC delay irrespective of the node movement speed. In the mobility scenario used for evaluation, the existence of roads causes more nodes to be available on the roads (Fig. 4.21(c), (d)), which results in a higher network congestion around these pathways. As GPSR-EA results in more perimeter mode forwarding (Fig. 6.9), many times the packets need to travel around obstacles and thus uses nodes that are residing in less congested areas. This phenomenon directs packets through lower network congested areas resulting in less delay in packet forwarding.

This phenomenon is observed again in Fig. 6.10(b) where the packet delivery delay is plotted against different node density. As the node density increases from 200 to 400, the average delay in GPSR and GPSR-EA increases. The rate of this increase in delay for GPSR and GPSR-MA is higher than that of GPSR-EA, as in GPSR and GPSR-MA more packets travel through more congested areas resulting in a longer delay.



(a)



(b)

Figure 6.10: Average delay for packet delivery at (a) various node movement speed (400 nodes), and (b) different node density (at 20 m/sec).

6.5 Impact on Location Service

In the previous section, by examining various metrics related to the performance measurement of GPSR it is observed that the mobility awareness improves the performance of GPSR. This improvement is more prominent when the average node mobility is higher. Incorporating environment awareness further improves the packet delivery performance of GPSR significantly over the mobility aware GPSR, as well as the original GPSR.

In this section the enhanced GPSR protocols based on mobility and environment awareness are incorporated with the location service protocol HALS proposed in the

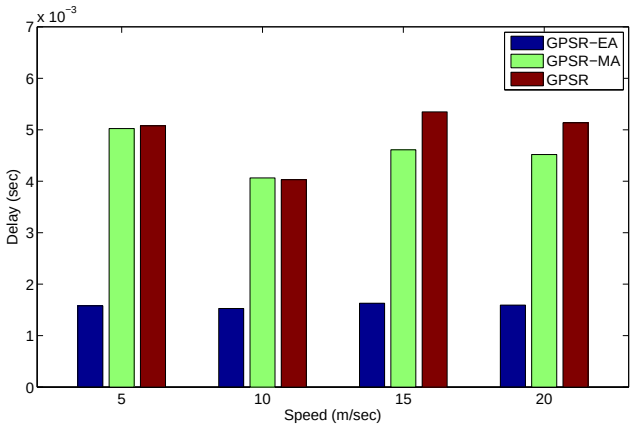


Figure 6.11: Average MAC layer delay per hop.

previous chapter. HALS using the mobility aware GPSR is termed Mobility Aware HALS (HALS-MA). Similarly, the environment aware GPSR incorporated in HALS is termed Environment Aware HALS (HALS-EA). This section mainly presents the simulation results of HALS, HALS-MA and HALS-EA focusing on different location service measurement metrics. Since each of the variations of HALS uses its own version of GPSR (i.e., original, mobility aware and environment aware), the location service related message transmission (e.g., update, query) is performed according to the respective underlying routing protocol. Thus the performance measurement metrics related to the location service reflect the nature and characteristics of the routing protocol while forwarding packets. Section 6.5.1 and 6.5.2 present the evaluation of location update and query related metrics, respectively, of HALS, HALS-MA and HALS-EA.

The parameters used in evaluation are the same as those defined in Table 6.1. The scenarios were generated by MobiGen using the map shown in Fig. 5.12.

6.5.1 Location Update

Figures 6.12 and 6.13 present the location update success rate, the average number of hops required for update, and the average update delay in a real-world context containing many obstacles without and with regions of interest, respectively.

In the obstacle scenario where there is no region of interest, Figs. 6.12(a) and (b) present the location update success rate at different node movement speed and node density, respectively. Since GPSR-EA shows the highest packet delivery success rate

(Fig. 6.6), HALS-EA also maintains a location update success rate higher than those of HALS-MA and HALS, irrespectively of the node mobility and density. On the other hand, the average number of hops for location update presented in Figs. 6.12(c) and (d) show that HALS-EA requires the highest number of hops. As the node density increases, the number of hops decreases as more packets are forwarded in the greedy mode. Figures 6.12(e) and (f) present the average delay in the location update at various node movement speed and node density, respectively. HALS-EA exhibits the lowest delay as many packets are forwarded through nodes that are in low congested area as is detailed in Section 6.4.

In the presence of regions of interest, nodes become congested along the regions of interest as shown in Section 4.6.1, and the effect on the location update is presented in Fig. 6.13. Figure 6.13(a) shows the location update success rate and similar to that shown in Fig. 6.12(a), HALS-EA maintains the highest delivery ratio. However, when the node movement speed increases, the rate of update failure increases compared to that observed in Fig. 6.12(a). Moreover, the location update remains more or less insensitive to the node density (Fig. 6.13(b)). In the presence of regions of interest, HALS-EA requires more hops (Fig 6.13(c)) than HALS-MA and HALS. As the node density increases (Fig. 6.13(d)), the required number of hops decreases steadily. However, the rate of decrease is lower than that observed in Fig. 6.12(d) as the presence of regions of interest makes most of the nodes concentrated within the regions of interest and an increase in the nodes in the already concentrated areas does not contributed significantly to the forwarding packets in greedy mode.

The average delay in the location update at various node speed is presented in Fig. 6.13(e), which shows that HALS-EA leads to the lowest delay. Increasing the number of nodes (Fig. 6.13(f)) increases the MAC layer congestion, as the nodes become more concentrated in the regions occupied by regions of interest, resulting in a higher delay in the location update in higher node density.

6.5.2 Location Query

The location query success rate, average number of hops for query and the average delay in the obstacle scenario are presented in Fig. 6.14. The location query exhibits a similar trend as observed in the location update. At various speed of node movement,

HALS-EA maintains a significantly higher location query success rate than HALS-MA and HALS (Fig. 6.14(a)). HALS-EA requires the highest number of hops (Fig. 6.14(c)) but has the lowest delay (Fig. 6.14(e)) in performing query.

Once again HALS-EA exhibits the highest query response capability among all irrespective of the node density (Fig. 6.14(b)). However, unlike what is observed in the location update (Fig. 6.12(b)), HALS-MA shows a lower query response than that of HALS when the node density is relatively low (less than 300). This indicates that at low node density in HALS-MA, the number of neighbors discarded from the neighbor list results in an inadequate number of neighbors. This adversely impacts the ability to find a potential next hop node and thus a lower query success rate is experienced than that of HALS where the node density is low.

Irrespective of the node density, HALS-EA requires the highest number of hops (Fig. 6.14(d)) but has the lowest delay (Fig. 6.14(f)). The reason for this low delay even though the packets move through the highest number of hops is explained earlier in Section 6.4. Though the number of hops in all variations of HALS remains almost constant, the delay increases due to increased network congestion in the MAC layer as the node density increases.

Figure 6.15 shows the different query related performance metrics in the presence of regions of interest. Here, the relative performance of HALS-EA, HALS-MA and HALS in the query success rate (Fig. 6.15(a)) remains the same as observed in the without region of interest scenario (Fig. 6.14(a)). In Fig. 6.15(b) HALS-EA exhibits a slight increase in the success rate, while HALS-MA remains almost the same when the node density increases. On the contrary, HALS shows a decreasing trend in the query success rate.

In the presence of regions of interest, when the nodes move at higher speed, the node concentration within the regions of interest takes place more rapidly. This aids the nodes in forwarding more packets in greedy mode, resulting in a lower number of hops in all variations of HALS (Fig. 6.15(c)). As mentioned before, higher node mobility causes a rapid concentration of nodes within the regions of interest. This event results in a higher query delay at increased node speed (Fig. 6.15(e)). When the node density increases (Fig. 6.15(f)), the network congestion also increases resulting in an increasing trend in the query delay in all variation of HALS. However, the rate of

increase in the query delay in HALS-EA remains the lowest among all.

To test the significance of the difference in the query success rate between HALS-EA and HALS-MA, the t-test at speed 5, 10, 15 and 20 m/sec yielded p-values of $p < 9.79 \times 10^{-10}$, 5.78×10^{-8}, 1.87×10^{-5}, 3.10×10^{-6} for the obstacle scenario (Fig. 6.14(a)) and $p < 1.83 \times 10^{-8}$, 2.56×10^{-7}, 1.01×10^{-6}, 2.93×10^{-9} for the obstacle scenario with region of interest (Fig. 6.15(a)) at 99% confidence level, validating their performance difference as being statistically significant.

6.6 Conclusion

Location based routing is attractive due to being scalable, having a lower routing overhead and forwarding packets efficiently by exploiting the position information of mobile nodes. The performance of a geographic routing protocol depends on how accurately the neighbor list of a node is constructed and maintained. In this regard we propose a scheme that presents an enhancement to the geographic routing protocol by exploiting knowledge of the environment while making routing decisions. By considering how different attributes of environment impact signal propagation, a more accurate neighbor list can be maintained which contributes to more successful packet forwarding; thereby environment awareness is introduced in the routing protocol and improved packet delivery is achieved. As the performance of the location service is interwoven with the underlying routing protocol, integrating an environment aware routing protocol while forwarding location service related messages results in a significantly superior outcome. A thorough performance evaluation based on a real map demonstrates the suitability and robustness of the proposed Environment Aware Hierarchical Adaptive Location Service in a real-world environment.

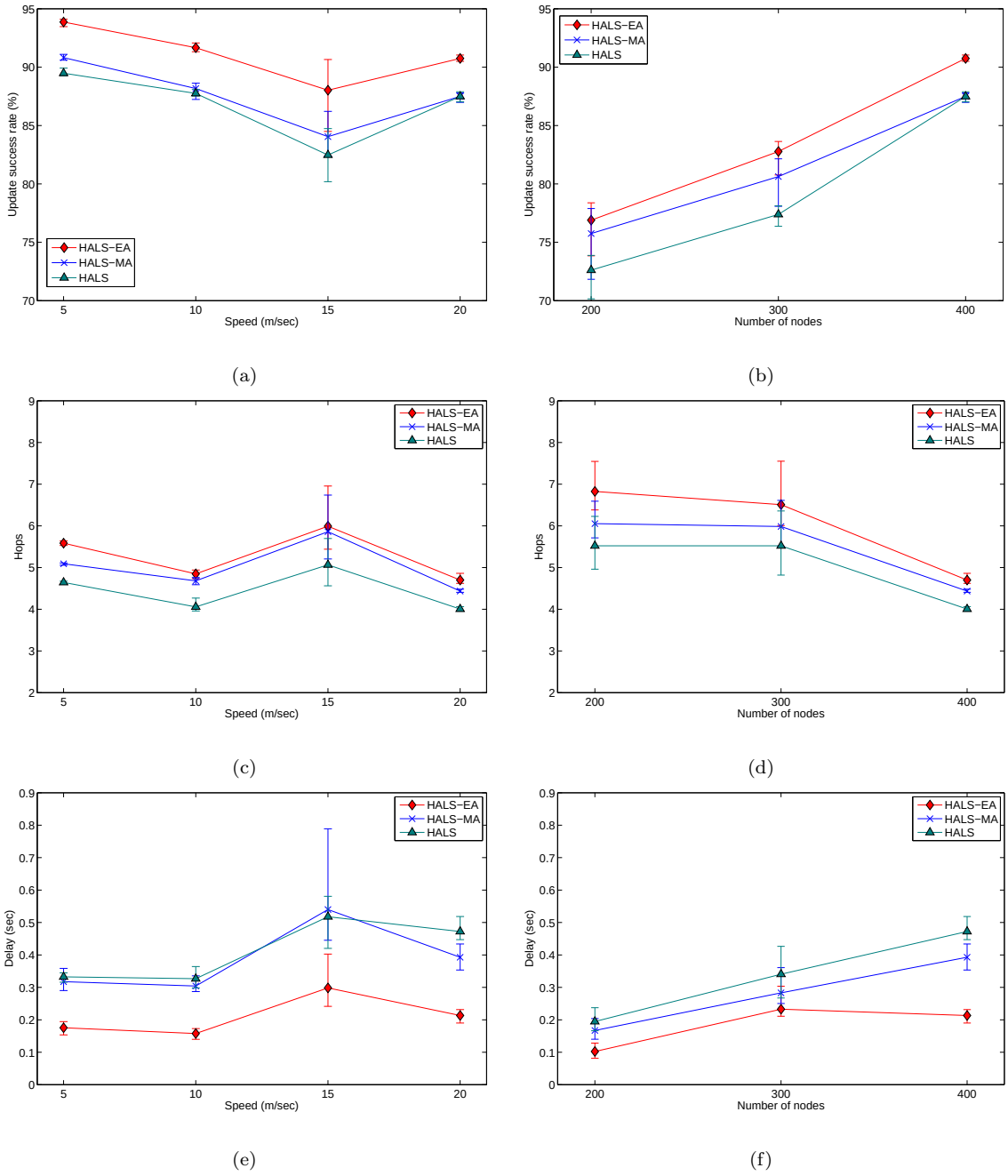


Figure 6.12: In obstacle scenario based on real-world map, (a) location update success rate at different node movement speed, (b) location update success rate at different node density, (c) average number of hops for location update at different node movement speed, (d) average number of hops for location update at different node density, (e) average delay for location update at different node movement speed, and (f) average delay for location update at different node density.

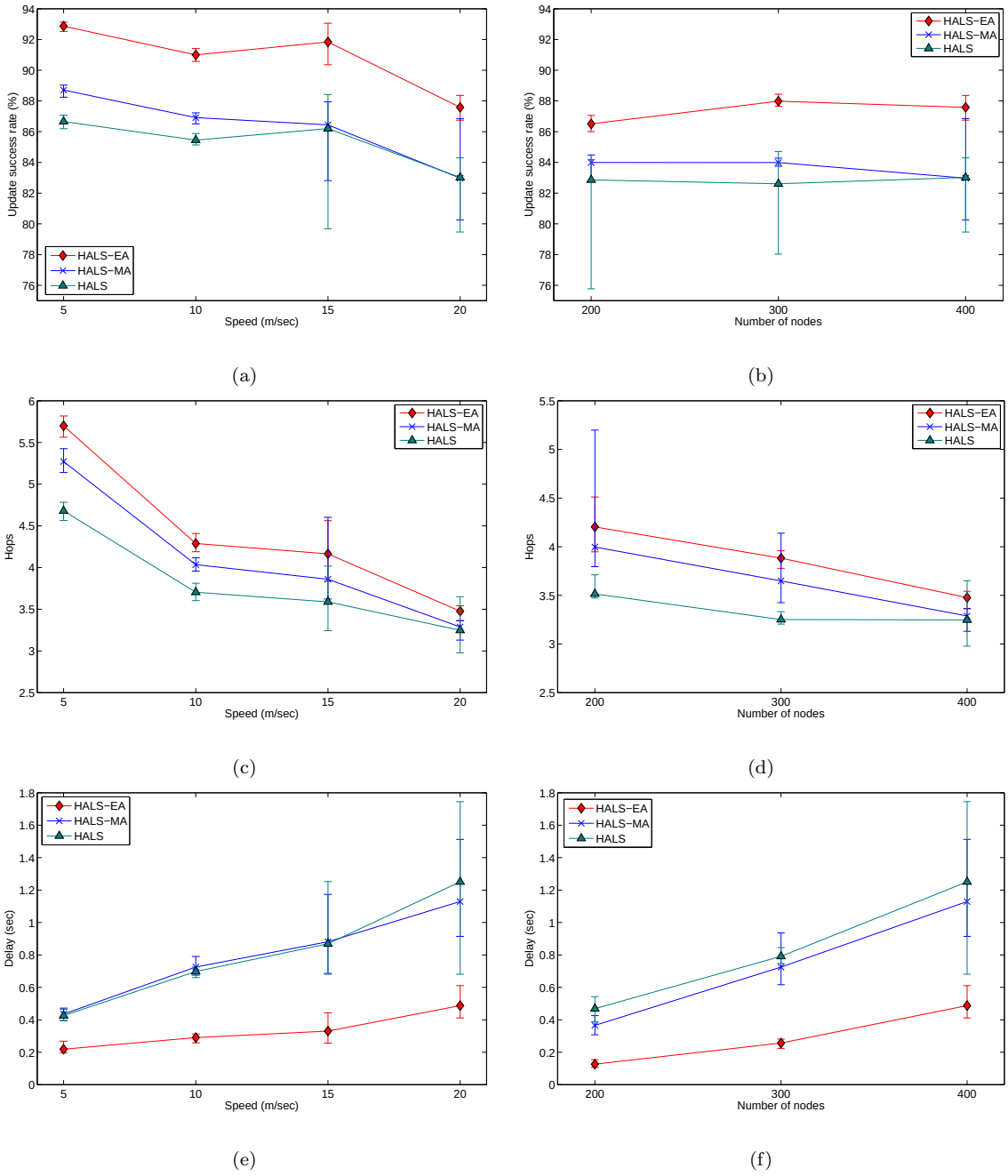


Figure 6.13: In obstacle scenario based on real-world map with region of interest, (a) location update success rate at different node movement speed, (b) location update success rate at different node density, (c) average number of hops for location update at different node movement speed, (d) average number of hops for location update at different node density, (e) average delay for location update at different node movement speed, and (f) average delay for location update at different node density.

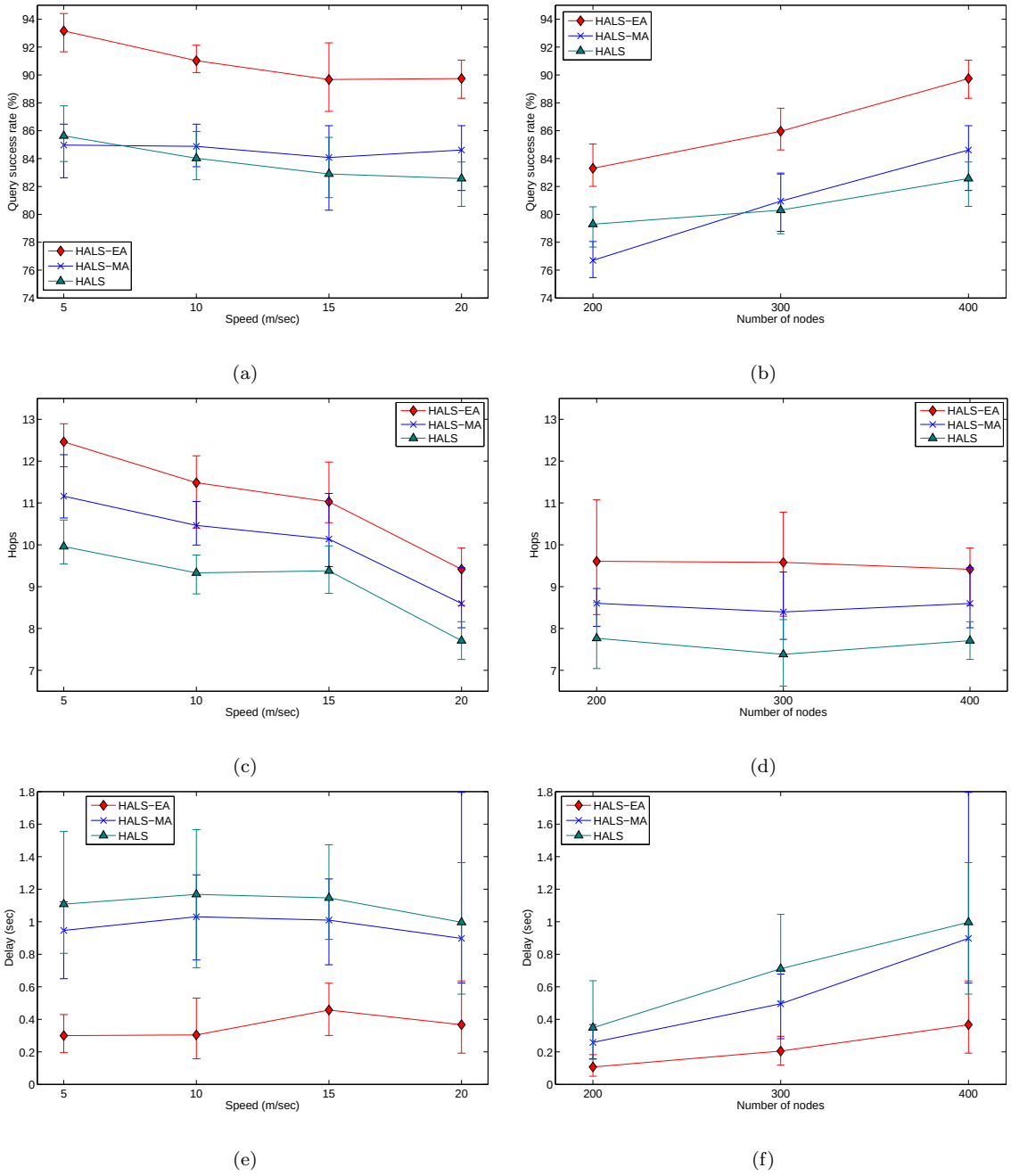


Figure 6.14: In obstacle scenario based on real-world map, (a) location query success rate at different node movement speed, (b) location query success rate at different node density, (c) average number of hops for location query at different node movement speed, (d) average number of hops for location query at different node density, (e) average delay for location query at different node movement speed, and (f) average delay for location query at different node density.

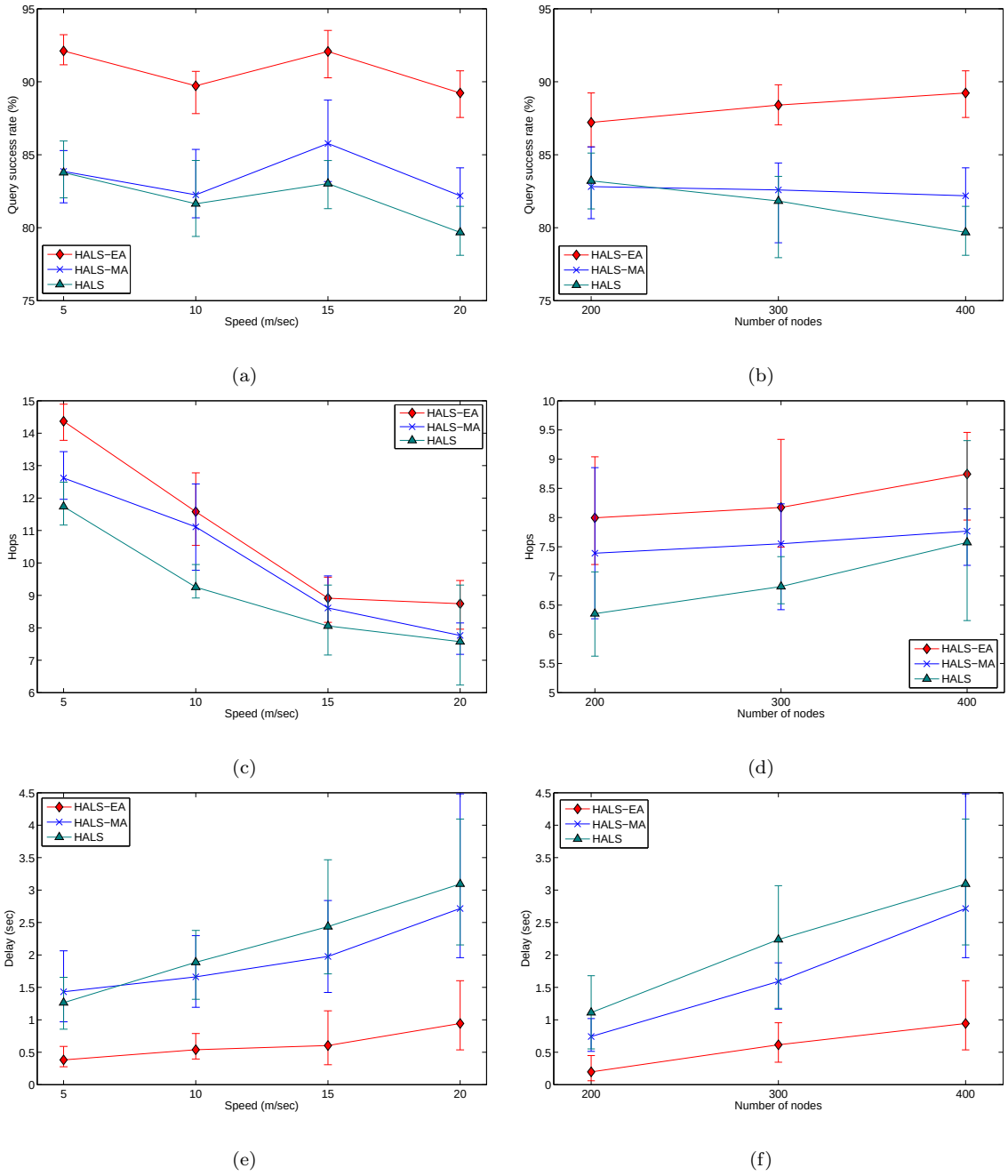


Figure 6.15: In obstacle scenario based on real-world map with region of interest, (a) location query success rate at different node movement speed, (b) location query success rate at different node density, (c) average number of hops for location query at different node movement speed, (d) average number of hops for location query at different node density, (e) average delay for location query at different node movement speed, and (f) average delay for location query at different node density.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

Ad hoc networks being one of the major branches of wireless network have gained a lot of interest over the last decades due to their capability of working in an infrastructure-less fashion, as such network can be deployed in scenarios where conventional infrastructure based network establishment is infeasible or inappropriate. The end-to-end communication between mobile devices in such network is an important factor where among the many routing protocols, especially location based protocols have become very promising because of their scalability, low routing overhead, suitability in high node mobility scenarios, etc. A location service is the main underlying component that provides the location information of mobile devices in a distributive manner and without the existence of such a service, geographic routing protocols become ineffectual.

Over the last decades huge research efforts have been committed to ensure scalable location service for geographic routing protocols. However, most of the works mainly emphasize on the theoretical development of such services, and a little or no importance has been given to their applicability in a real-world environment. Because of this many environmental aspects that have a significant impact on the network performance and workability have been ignored. In the development of such schemes, simulation environment has become the most attractive solution for validation and analysis due to its many advantages over test beds, including but not limited to time and cost effectiveness, providing the ability to repeat experiments easily, and subsequently debugging and adjusting the concerned protocol. Though possessing such advantages over test beds, simulation introduces one major challenge - to accurately mimic the environment where most likely the network would be deployed. Since most of the existing works

related to location service development are based on a very simplistic environment which is not able to represent the environment properly, it has become an utmost necessity to introduce a location service which is capable of coping with the vastly diverse real-world environments.

For this reason it became imperative to understand the impact of the environmental context, which can be reflected through a mobility model by generating node mobility in a more realistic manner. Consideration of radio signal attenuation caused by environmental attributes is also an important factor while making packet routing decisions. These challenging issues are the key motivating factors of this thesis and have been addressed through a series of innovative developments as follows:

- *Location service*: A novel location service has been developed that can adapt itself under environmental influence, and a series of innovative algorithms have been introduced to assure a seamless service in situations likely to occur in the real-world. The introduction of mechanisms like shifting, creating and destroying home regions have added very unique and useful features to the proposed location service. The location service has also been made hierarchically organized for scalability. A detail analysis on its scalability is also presented.

- *Mobility model*: Developing a realistic environment is a key factor in developing a location service aimed at handling different environment conditions well. Since the environmental influence is reflected by node mobility, a novel mobility model has been developed along with a mobility trace generation tool MobiGen (http://arrow.monash.edu.au/hdl/1959.1/109933) in which various environmental contexts can be incorporated. Moreover, a signal propagation model has been introduced that captures real-world signal fading phenomena caused by environmental objects.

- *Enhancement of routing protocol*: The influence of the environmental contexts through signal attenuation observed in the physical layer has been incorporated in the underlying routing protocol's packet forwarding mechanism which has significantly improved the routing capability through better selection of routing paths.

- *Integration of all developments*: Finally, the proposed location service with the

enhanced routing protocol is evaluated based on the realistic environment reflected by the proposed mobility model. This evaluation has shown promising outcome and more importantly ensures a seamless service in providing location information in a realistic environment model. Extensive simulations have been conducted based on a real-world map of a university campus to analyze the different aspects of the proposed location service.

While addressing each of the aspects mentioned above, extensive simulations have been performed to evaluate the proposed mechanisms. In this regard, it has been observed that our proposed mobility model is capable of representing the real-world environment more accurately by directing node mobility more realistically than the existing models. Again, the proposed location service demonstrates its significant superiority over the existing contemporary and competing location services, as it is able to cope with the real-world environmental impacts reflected by our mobility model. Moreover, consideration of signal attenuation while making routing decisions has further improved the performance of our location service and has proven itself as a potential candidate to be applicable in real-world environment to aid geographic routing protocols. We believe that the results and methods focusing on mobility modeling and location service development will make a significant contribution to advance research and applications of mobile ad hoc networks.

7.2 Future Works

The research strategies for an improved location service suitable for real-world environment presented in this thesis can be potentially extended in a number of directions, some of which are discussed in the following:

- The mobility model presented in this thesis is an entity mobility model where the movement of a node is independent of the other nodes. By introducing a group mobility feature, it is possible to introduce spatial dependency - another aspect of mobility model. Moreover, integrating this mobility model with a macro mobility model (e.g., SLAW) would definitely help to represent node mobility even more realistically.

- Using aggressive caching might improve HALS's performance, specially for low node density. When an update packet traverses, all nodes engaged in the forwarding process as well as others who can sense the packet by overhearing in MAC layer can cache the location information. A similar process can be utilized for query and reply packets as well. While implementing this type of aggressive caching, assigning a lifetime for the cached information is a challenging task and needs further investigation.

- Like other location services, a significant overhead for HALS is due to location update mechanism which constantly generates small packets. Thus collision increases which results in an increased MAC delay. For this reason piggybacking update packets may reduce the update overhead and a new procedure could be developed for attaching a node's location to an existing update packet. In this scheme, as a node attaches its own location while forwarding other nodes' update packets, the update packet generation frequency will be reduced. Since the location servers in HALS work more like in a centralized fashion, there is a high chance that the nodes participating in forwarding an update packet share the same home region.

- As mentioned earlier, in HALS the nodes that are assigned a higher rank incur a greater message and storage load than the others. Introducing a method to split the home regions into multiple homes may reduce this overhead. This challenging task needs to consider the synchronization between the split homes, and requires further investigation.

Publications from this work

Journal publications:

1. S. Ahmed, G. C. Karmakar, and J. Kamruzzaman, "An Environment Aware Mobility Model for Wireless Ad Hoc Networks," *Computer Networks*, doi:10.1016/j.comnet.2009.12.005.

Conference publications:

1. S. Ahmed, G. Karmakar, and J. Kamruzzaman, "Hierarchical Adaptive Location Service for Mobile Ad Hoc Network," in *Proc. of IEEE Wireless Communications and Networking Conference* (*WCNC*), Apr. 2009.

2. S. Ahmed, G. Karmakar, and J. Kamruzzaman, "Geographic Constraint Mobility Model for Ad Hoc Network," in *Proc. of IEEE/ACM International Symposium on Modelling, Analysis and Simulation of Computer and Telecommunication Systems* (*MASCOTS*), USA, 2008, pp. 337–346.

3. S. Ahmed, G. C. Karmakar, and J. Kamruzzaman, "Evaluating Performance of Location Service Protocols of Ad-Hoc Wireless Network in Real-World Environment Model," in *Proc. of IEEE Wireless Communications, Networking and Mobile Computing* (*WiCom*), Shanghai, China, 2007, pp. 1577–1580.

Bibliography

[1] C. E. Perkins and P. Bhagwat, "Routing Over Multi-hop Wireless Network of Mobile Computers," *Mobile Computing*, pp. 183–205, 1996.

[2] G. Pei, M. Gerla, and T.-W. Chen, "Fisheye State Routing in Mobile Ad Hoc Networks," in *Workshop on Wireless Networks and Mobile Computing (ICDCS)*, 2000, pp. 71–78.

[3] P. Jacquet, P. Mühlethaler, T. Clausen, A. Laouiti, A. Qayyum, and L. Viennot, "Optimized Link State Routing Protocol," in *IEEE International Multitopic Conference (INMIC)*. IEEE, 2001, pp. 62–68.

[4] D. B. Johnson, D. A. Maltz, and J. Broch, "DSR: The Dynamic Source Routing Protocol for Multi-Hop Wireless Ad Hoc Networks," in *Ad Hoc Networking*, 1st ed., ser. Ad Hoc Networking, C. Perkins, Ed. Addison-Wesley, 2001, ch. 5, pp. 139–172.

[5] C. E. Perkins and E. M. Royer, "Ad Hoc on Demand Distance Vector Routing," in *IEEE Workshop on Mobile Computing Systems and Applications (WMCSA)*. New Orleans, Lousiana, USA: IEEE, 1999, pp. 90–100.

[6] V. D. Park and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, vol. 3. Kobe, Japan: IEEE, 1997, pp. 1405–1413.

[7] Z. J. Haas, "A New Routing Protocol For The Reconfigurable Wireless Networks," in *IEEE International Conference on Universal Personal Communications, (ICUPC)*. San Diego, California, USA: IEEE, 1997, pp. 562–566.

[8] M. Joa-Ng and I.-T. Lu, "A Peer-to-Peer Zone-Based Two-Level Link State Routing for Mobile Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, vol. 17, no. 8, pp. 1415–1425, 1999.

[9] C.-C. Chiang, H.-K. Wu, W. Liu, and M. Gerla, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel," in *International Conference on Networks (SICON)*. Singapore: IEEE, 1997, pp. 197–211.

[10] R. Sivakumar, B. Das, and V. Bharghavan, "The Clade Vertebrata: Spines and Routing in Ad Hoc Networks," in *IEEE Symposium on Computers and Communications*, 1998.

[11] P. Misra, "Routing Protocols for Ad Hoc Mobile Wireless Networks," 1999.

[12] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, "A Distance Routing Effect Algorithm for Mobility (DREAM)," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, USA, 1998, pp. 76–84.

[13] Y.-B. Ko and N. H. Vaidya, "Location - Aided Routing (LAR) in Mobile Ad Hoc Networks," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Dallas, Texas, USA, 1998, pp. 66–75.

[14] F. Kuhn, R. Wattenhofer, and A. Zolinger, "An Algorithmic Approach to Geographic Routing in Ad Hoc and Sensor Networks," *IEEE/ACM Transactions on Networking*, vol. 16, no. 1, Feb. 2008.

[15] B. Karp and H. Kung, "GPSR: Greedy Perimeter Stateless Routing for Wireless Networks," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, USA, 2000, pp. 243–254.

[16] N. Patwari, A. O. H. III, M. Perkins, N. S. Correal, and R. J. O'Dea, "Relative Location Estimation in Wireless Sensor Networks," *IEEE Transaction on Signal Processing*, vol. 51, no. 8, pp. 2137–2148, Aug. 2003.

[17] M. Z. Rahman and L. Kleeman, "Paired Measurement Localization: A Robust Approach for Wireless Localization," *IEEE Transactions on Mobile Computing*, vol. 8, no. 8, pp. 1087–1102, Aug. 2009.

[18] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-free localization schemes for large scale sensor networks," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2003, pp. 81–95.

[19] S.-C. Woo and S. Singh, "Scalable Routing Protocol for Ad Hoc Networks," *Wireless Networks*, vol. 7, no. 5, pp. 513–529, 2001.

[20] C. Cheng, H. Lemberg, S. Philip, E. van den Berg, and T. Zhang, "SLALoM: A Scalable Location Management Scheme for Large Mobile Ad-hoc Networks," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2002, pp. 574–578.

[21] Y. Yu, G.-H. Lu, and Z.-L. Zhang, "Enhancing Location Service Scalability with HIGH-GRADE," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2004, pp. 164–173.

[22] W. Kiess and H. J. Widmer, "Hierarchical Location Service for Mobile Ad-Hoc Networks," *SIGMOBILE Mobile Computing and Communications Review*, vol. 8, no. 4, pp. 47–58, 2004.

[23] I. Stojmenovic, D. Liu, and X. Jia, "A Scalable Quorum-based Location Service in Ad Hoc and Sensor Networks," *International Journal of Communication Networks and Distributed System*, vol. 1, no. 1, pp. 71–94, 2008.

[24] J.-R. Jiang and W.-J. Ling, "SEEKER: An Adaptive and Scalable Location Service for Mobile Ad Hoc Networks," in *International Computer Symposium (ICS)*, 2006.

[25] N. Nikaein and C. Bonnet, "ALM - Adaptive Location Management Model Incorporating Fuzzy Logic for Mobile Ad Hoc Networks," in *Mediterranean Ad Hoc Networking Workshop (MED-HOC)*, Italy, 2002.

[26] I. Abraham, D. Dolev, and D. Malkhi, "LLS: A Locality Aware Location Service for Mobile Ad Hoc Networks," in *Joint Workshop on Foundations of Mobile Computing (DIALM-POMC)*, 2004, pp. 75–84.

[27] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communications and Mobile Computing*, vol. 2, pp. 483–502, 2002.

[28] E. Royer, P. Melliar-Smith, and L. Moser, "An Analysis of the Optimum Node Density for Ad Hoc Mobile Networks," in *IEEE International Conference on*

Communications (ICC), vol. 3, 2001, pp. 857–861.

[29] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri, "Real world Environment Models for Mobile Ad hoc Networks," *IEEE Journal on Special Areas in Communications - Special Issue on Wireless Ad hoc Networks*, vol. 23, no. 3, pp. 622–632, 2005.

[30] M. Kasemann, H. FuBler, H. Hartenstein, and M. Mauve, "A Reactive Location Service for Mobile Ad Hoc Network," Department of Science, University of Mannheim, Tech. Rep. TR-02-014, Nov. 2002.

[31] S. Ahmed, G. Karmakar, and J. Kamruzzaman, "Hierarchical Adaptive Location Service for Mobile Ad Hoc Network," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Apr. 2009.

[32] S. Ahmed, G. C. Karmakar, and J. Kamruzzaman, "Evaluating Performance of Location Service Protocols of Ad-Hoc Wireless Network in Real-World Environment Model," in *IEEE Wireless Communications, Networking and Mobile Computing (WiCom)*, Shanghai, China, 2007, pp. 1577–1580.

[33] S. Ahmed, G. Karmakar, and J. Kamruzzaman, "Geographic Constraint Mobility Model for Ad Hoc Network," in *IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, USA, 2008, pp. 337–346.

[34] S. Ahmed, G. C. Karmakar, and J. Kamruzzaman, "An Environment Aware Mobility Model for Wireless Ad Hoc Networks," *Computer Networks*, doi:10.1016/j.comnet.2009.12.005.

[35] J. Broch, D. A. Maltz, D. B. Johnson, Y.-C. Hu, and J. Jetcheva, "A Performance Comparison of Multi-hop Wireless Ad Hoc Network Routing Protocols," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, Dallas, Texas, United States, 1998, pp. 85–97.

[36] J. Y. L. Boudec and M. Vojnovic, "Perfect Simulation and Stationarity of a Class of Mobility Models," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2005, pp. 72–79.

[37] G. Liljana and P. Ramjee, *Ad Hoc Networking Towards Seamless Communications.* Springer, 2006, ch. 7, pp. 173–209.

[38] J. Ghosh, S. J. Philip, and C. Qiao, "Socialogical Orbit aware Location Approximation and Routing (SOLAR) in MANET," *Ad Hoc Networks*, vol. 5, no. 2, pp. 189–209, 2007.

[39] W. Hsu, K. Merchant, H. Shu, C. Hsu, and A. Helmy, "Weighted Waypoint Mobility Model and its Impact on Ad Hoc Networks," *Mobile Computing and Communications Review*, vol. 9, no. 1, pp. 59–63, 2005.

[40] D. J. Patterson, L. Liao, K. Gajos, M. Collier, N. Livic, K. Olson, S. Wang, D. Fox, and H. Kautz, "Opportunity Knocks: A System to Provide Cognitive Assistance with Transportation Services," in *International Conference on Ubiquitous Computing (UbiComp)*, ser. Lecture Notes in Computer Science, vol. 3205, 2004, pp. 433–450.

[41] M. Kim, D. Kotz, and S. Kim, "Extracting a Mobility Model from Real User Traces," in *IEEE International Conference on Computer Communications (INFOCOM)*, Barcelona, Spain, 2006, pp. 1–13.

[42] F. Bai and A. Helmy, "A Survey of Mobility Modeling and Analysis in Wireless Adhoc Networks," in *Wireless Adhoc and Sensor Networks*, ser. Wireless Ad Hoc and Sensor Networks. Kluwer Academic Publishers, 2004.

[43] C. Bettstetter, "Mobility Modeling in Wireless Networks: Categorization, Smooth Movement, and Border Effects," *ACM Mobile Computing and Communications Review*, vol. 5, no. 3, pp. 55–67, 2001.

[44] D. M. Blough, G. Resta, and P. Santi, "A Statistical Analysis of the Long-run Node Spatial Distribution in Mobile Ad Hoc Networks," *Wireless Networks*, vol. 10, no. 5, pp. 543–554, 2004.

[45] G. Lu, G. Manson, and D. Belis, "Mobility Modeling in Mobile Ad Hoc Networks with Environment-Aware," *Journal of Networks*, vol. 1, no. 1, pp. 54–63, 2006.

[46] E. W. Weisstein, *The CRC Concise Encyclopedia of Mathematics.* FL: CRC Press, 1998.

[47] B. Liang and Z. J. Haas, "Predictive Distance-Based Mobility Management for PCS Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 1999.

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

[49] M. Sanchez and P. Manzoni, "Anejos: A Java Based Simulator for Ad-Hoc Networks," *Future Generation Computer Systems*, vol. 17, no. 5, pp. 573–583, 2001.

[50] K. Lee, S. Hong, S. J. Kim, I. Rhee, and S. Chong, "SLAW: A Mobility Modelfor Human Walk," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2009.

[51] S. Lim and C. Yu, "Clustered Mobility Model for Scale-free Wireless Networks," in *IEEE Conference on Local Computer Networks (LCN)*, 2006.

[52] A. L. Barabasi and E. Bonabeau, "Scale-free Networks," *Scientific American*, vol. 288, pp. 50–59, 2003.

[53] M. F. Shlesinger, J. Klafter, and Y. M. Wong, "Random Walks with Infinite Spatial and Temporal Moments," *Journal of Statistical Physics*, vol. 27, pp. 499–512, 1982.

[54] I. Rhee, M. Shin, S. Hong, K. Lee, and S. Chong, "On the Levi-walk Nature of Human Mobility," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2008.

[55] I. Norros, P. Mannersalo, and J. L. Wang, "Simulation of Fractional Brownian Motion with Conditionalized Random Midpoint Displacement," in *Advances in Performance Analysis*, 1999, pp. 77–101.

[56] I. Norros and J. Virtamo, "Handbook of fBm formulae," European Cooperation in Science and Technology (COST), Tech. Rep. COST-257, Dec. 1996.

[57] I. Khider, W. Furong, Y. W. Hua, and Sacko, "A Survey of Geographic Restriction Mobility Models," *Journal of Applied Sciences*, vol. 7, no. 3, pp. 442–450, 2007.

[58] B. Pazand and C. McDonald, "A Critique of Mobility Models for Wireless Network Simulation," in *IEEE/ACIS International Conference on Computer and Information Science (ICIS)*, Melbourne, 2007, pp. 141–146.

[59] J. Tian, J. Haehner, C. Becker, I. Stepanov, and K. Rothermel, "Graph-based Mobility Model for Mobile Ad Hoc Network Simulation," in *Annual Simulation Symposium*, San Diego, CA, USA, 2002, p. 337.

[60] F. Bai, N. Sadagopan, and A. Helmy, "IMPORTANT: A Framework to Systematically Analyze the Impact of Mobility on Performance of Routing Protocols for Ad Hoc Networks," in *IEEE International Conference on Computer Communications (INFOCOM)*, 2003, pp. 825–835.

[61] ——, "The IMPORTANT Framework for Analyzing the Impact of Mobility on Performance of routing for Ad Hoc Networks," *Ad Hoc Networks*, vol. 1, no. 4, pp. 383–403, 2003.

[62] N. Potnis and A. Mahajan, "Mobility Models for Vehicular Ad Hoc Network Simulations," in *ACM Southeast Regional Conference*, 2006, pp. 746–747.

[63] C. Gorgorin, V. Gradinescu, R. Diaconescu, V. Vristea, and L. Ifode, "An Integrated Vehicular and Network Simulator for Vehicular Ad-Hoc networks," in *European Simulation and Modelling Conference*, 2006.

[64] D. Djenouri, W. Soualhi, and E. Nekka, "VANET's Mobility Models and Overtaking: An Overview," in *International Conference on Information and Communication Technologies: from Theory to Applications (ICTTA)*, 2008, pp. 1–6.

[65] J. Harri, F. Filali, and C. Bonnet, "Mobility Models for Vehicular Ad Hoc Networks: A Survey and Taxonomy," Eurocom, Tech. Rep., 2007.

[66] P. Johansson, T. Larsson, N. Hedman, B. Mielezarek, and M. Degermark, "Scenario-based Performance Analysis of Routing Protocols for Mobile Ad-Hoc

Networks," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 1999, pp. 195–206.

[67] A. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri, "Towards Realistic Mobility Models for Mobile Ad Hoc Networks," in *ACM International Conference on Mobile Computing and Newtorking (MobiCom)*, 2003, pp. 217–229.

[68] K. Konishi, K. Maeda, K. Sato, A. Yamasaki, H. Yamaguchi, K. Yasumoto, and T. Higashino, "MobiREAL Simulator-Evaluating MANET Applications in Real Environment," in *IEEE/ACM International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*, 2005, pp. 499–502.

[69] G. Lu, D. Belis, and G. manson, "Study on Environment Mobility Models for Mobile Ad Hoc Network: Hotspot Mobility Model and Route Mobility Model," in *International Conference on Wireless Networks, Communications and Mobile Computing*, vol. 1, 2005, pp. 808–813.

[70] Y. Lu, H. Lin, Y. Gu, and A. Helmy, "Towards Mobility-Rich Analysis in Ad Hoc Newtorks: Using Contraction, Expansion and Hybrid Models," in *IEEE International Conference on Communications (ICC)*, 2004, pp. 4346–4351.

[71] Realistic Mobility Modelling for Mobile Ad Hoc Networks Project. [Online]. Available: http://www.cs.rice.edu/~amsaha/Research/MobilityModel (last accessed Jan 10, 2010)

[72] Obstacle Mobility Model Project. [Online]. Available: http://moment.cs.uscb.edu/mobility/index.html (last accessed Jan 10, 2010)

[73] SLAW-Self Similar Least Action Walk. [Online]. Available: http://netsrv.csc.ncsu.edu/twiki/bin/view/Main/Projects (last accessed Jan 10, 2010)

[74] Mobility Models for Mobile Ad Hoc Network Simulations. [Online]. Available: http://toilers.mines.edu/twiki/bin/view/Public/CodeList (last accessed Jan 10, 2010)

[75] IMPORTANT Mobility Framework. [Online]. Available: http://nile.cise.ufl.edu/important/ (last accessed Jan 10, 2010)

[76] MobiREAL- A Realistic Network Simulator. [Online]. Available: http://www.mobireal.net/ (last accessed Jan 10, 2010)

[77] Car-to-car Cooperation for Vehicular Ad-Hoc Networks. [Online]. Available: http://aqualab.cs.northwestern.edu/projects/C3.html (last accessed Jan 10, 2010)

[78] S. A. Shaheen, "DAVIS Smart Mobility Modeling Project: Initial Scoping and Planning Project," University of California, Davis, Tech. Rep. UCD-ITS-RR-03-06, Mar. 2003.

[79] T. Camp, J. Boleng, and L. Wilcox, "Location Information Services in Mobile Ad Hoc Networks," in *IEEE International Conference on Communications (ICC)*, 2002, pp. 3318–3324.

[80] D. Liu, I. Stojmenovic, and X. Jia, "A Scalable Quorum Based Location Service in Ad Hoc and Sensor Networks," in *IEEE International Conference on Mobile Ad-hoc and Sensor Systems (MASS)*, 2006, pp. 489–492.

[81] S. M. Das, H. Pucha, and Y. C. Hu, "Performance Comparison of Scalable Location Services for Geographic Ad Hoc Routing," in *IEEE International Conference on Computer Communications (INFOCOM)*, vol. 2, 2005, pp. 1228–1239.

[82] B.-C. Seet, Y. Pan, W.-J. Hsu, and C.-T. Lau, "Multi-home Region Location Service for Wireless Ad Hoc Networks: An Adaptive Demand-driven Approach," in *International Conference on Wireless On-demand Network Systems and Services (WONS)*, 2005, pp. 258–263.

[83] J. Li, J. Jannotti, D. Couto, D. Karger, and R. Morris, "A Scalable Location Service for Geographic Ah Hoc Routing," in *ACM International Conference on Mobile Computing and Networking (MobiCom)*, 2000, pp. 120–130.

[84] M. Kasemann, H. Hartenstein, H. FuBler, and M. Mauve, "Analysis of a Location Service for Position-Based Routing in Mobile Ad Hoc Networks," in *Workshop on Mobile Ad-Hoc Networks (WMAN)*, Germany, 2002, pp. 121–133.

[85] Y. Xue, B. Li, and K. Nahrstedt, "A Scalable Location Management Scheme in Mobile Ad-Hoc Networks," in *IEEE Conference on Local Computer Networks (LCN)*. Tampa, FL, USA: IEEE Comput. Soc, 2001, pp. 102–11.

[86] P.-H. Hsiao, "Geographical Region Summary Service for Geographical Routing," in *ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, Long Beach, CA, USA, 2001, pp. 263–266.

[87] S. J. Philip and C. Qiao, "Hierarchical Grid Location Management for Large Wireless Ad Hoc Networks," *ACM SIGMOBILE Mobile Computing and Communications Review(MC2R)*, vol. 7, no. 3, pp. 33–34, 2003.

[88] S. Ratnasamy, B. Karp, Y. Li, F. Yu, R. Govindan, S. Shenker, and D. Estrin, "GHT: A Geographic Hash Table for Data-centric Storage," in *ACM International Workshop on Wireless Sensor Networks and Applications (WSNA)*, 2002, pp. 78–87.

[89] R. Flury and R. Wattenhofer, "MLS: An Efficient Location Service for Mobile Ad Hoc Networks," in *ACM international symposium on Mobile ad hoc networking and computing (MobiHoc)*, Florence, Italy, 2006, pp. 226–237.

[90] The ns-2 Network Simulator. [Online]. Available: http://www.isi.edu/nsnam/ns/ (last accessed Jan 10, 2010)

[91] P. Venkateswaran, R. Ghosh, A. Das, S. K. Sanyal, and R. Nandi, "An Obstacle Based Realistic Ad-Hoc Mobility Model for Social Networks," *Journal of Networks*, vol. 1, no. 2, pp. 37–44, 2006.

[92] A.-K. H. Souley and S. Cherkaoui, "Advanced Mobility Models for Ad Hoc Network Simulations," *Systems Communications*, pp. 50–55, 2005.

[93] L. J. Guibas and J. Hershberger, "Optimal Shortest Path Queries in a Simple Polygon," in *Annual Symposium on Computational Geometry*, 1987, pp. 50–63.

[94] F. Li and R. Klette, "Finding the Shortest Path Between Two Points in a Simple Polygon by Applying a Rubberband Algorithm," in *Advances in Image and Video Technology*, 2006, vol. 4319/2006, pp. 280–291.

[95] O. Arikan, S. Chenney, and D. A. Forsyth, "Efficient Multi-Agent Path Planning," in *Eurographic Workshop on Computer Animation and Simulation*, 2001, pp. 151–162.

[96] L. Blazevic, J.-Y. L. boudec, and S. Giordano, "A Location-Based Routing Method for Mobile Ad Hoc Networks," *IEEE Transaction on Mobile Computing*, vol. 4, no. 2, pp. 97–110, 2005.

[97] Walking. [Online]. Available: http://en.wikipedia.org/wiki/Walking

[98] B. Zalik, "Two Efficient Algorithms for Determining Intersection Points Between Simple Polygons," *Computers and Geosciences*, vol. 26, pp. 137–151, 2000.

[99] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 2nd ed. MIT Press, McGraw-Hill, 2002.

[100] F. Aurenhammer, "Voronoi Diagrams-A Survey of a Fundamental Geometric Data Structure," *ACM Computing Surveys*, vol. 23, no. 3, pp. 345–405, 1991.

[101] T. S. Rappaport, *Wireless Communications, Principles and Practice*, 2nd ed., T. S. Rappaport, Ed. Prentice Hall, Dec. 2001.

[102] F. J. Martinez, C.-K. Toh, J.-C. Cano, C. T. Calafate, and P. Manzoni, "Realistic Radio Propagation Models (RPMs) for VANET Simulations," in *IEEE Wireless Communications and Networking Conference (WCNC)*, 2009.

[103] S. Marinoni and H. H. Kari, "Ad Hoc Routing Protocol Performance in a Realistic Environment," in *International Conference on Networking and the International Conference on Systems (ICN/ICONS/MCL)*, 2006.

[104] A. Mahajan, N. Potnis, K. Gopalan, and A. Wang, "Modeling VANET Deployment in Urban Settings," in *International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM)*, 2007, pp. 151–158.

[105] J. Hahner, D. Dudkowski, P. J. Marron, and K. Rothermel, "Quantifying Network Partitioning in Mobile Ad Hoc Networks," in *International Conference on Mobile Data Management (MDM)*, 2007, pp. 174–181.

[106] L. Kleinrock and J. Silvester, "Optimum Transmission Radii for Packet Radio Network or Why Six is a Magic Number," in *IEEE National Telecommunications Conference*, 1978, pp. 4.3.1–4.3.5.

[107] G. Toussaint, "The relative Neighborhood Graph of a Finite Planar Set," *Pattern Recognition*, vol. 12, no. 4, pp. 261–268, 1980.

[108] K. Gabriel and R. Sokal, "A New Statistical Approach to Geographic Variation Analsis," *Systematic Zoology*, vol. 18, pp. 259–278, 1969.

[109] Q. J. Chen, S. S. Kanhere, M. Hassan, and K.-C. Lan, "Adaptive Position Update in Geographic Routing," in *IEEE International Conference on Communications (ICC)*, 2006, pp. 4046–4051.

[110] F. Granelli, G. Boato, D. Kliazovich, and G. Vernazza, "Enhanced GPSR Routing in Multi-Hop Vehicular Communications through Movement Awareness," *IEEE Communications Letters*, vol. 11, no. 10, pp. 781–783, 2007.