Development Of A Trace Generation And Analysis Software For Random Mobility Models

Xiang Cui

BSc., University of Northern British Columbia, 2004

Thesis Submitted In Partial Fulfillment Of

The Requirements For The Degree Of

Master Of Science

in

Mathematical, Computer, and Physical Sciences

(Computer Science)

The University of Northern British Columbia

May 2006

Library and
Archives Canada

Published Heritage
Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et
Archives Canada

Direction du
Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-28365-3
Our file Notre référence
ISBN: 978-0-494-28365-3

Dedicated to my parents,

Cui Jun and *Wang YuanYuan*

Abstract

Simulation is an integral part of most research studies in mobile computing context and building a valid, credible, and appropriately detailed simulation model is crucial for conducting accurate and meaningful simulation study. Although many simulation softwares are freely available in the Internet for mobile computing simulations, we found that there is no comprehensive software freely available for the researchers to visualize, analyze, and then generate suitable mobility trace for their simulation study. Lack of such a comprehensive software constrains the researchers to choose only from a few models supported in the generic simulation softwares and that in turn questions the validity of most simulation results. This hypothesis has been reaffirmed in a recent survey conducted on the papers published in the proceedings of ACM MobiHoc symposium between 2000 and 2005. The survey observes that the credibility of the simulation results has decreased while the use of simulation has increased. Part of this credibility crisis is clearly related to mobility model used to simulate the mobile nodes in the simulation system. We believe that the availability of a software to choose, visualize, and analyze mobility patterns before generating suitable mobility trace to use in the simulations, would resolve the mobility related concerns raised in the survey. The objective of this thesis is to develop such a software.

This thesis presents a mobility generator software called **RMobiGen** that we developed using Java. RMobiGen can be used to specify, visualize, analyze, and then generate mobility traces for various random mobility models. In addition to its rich functionality, the software has a user friendly interface which increases its appeal for wider use. We followed software engineering techniques and used UML diagrams during analysis and design phases of the development and then implemented using Java. We also surveyed the mobility models proposed in the literature and conducted various experiments on them using RMobiGen. During the experimentation, we confirmed the phenomena related to mobility models indicated in the literature and also encountered many new interesting behaviors and patterns.

Contents

List of Figures

Publications

1. Alex A. Aravind and **Xiang Cui**, RMobiGen: A Trace Generation, Scenario Visualization, and Performance Analysis Tool for Random Mobility Models, Submitted for *ACM Mobile Computing and Communication Review.*

2. Jernej Polajnar, Tyler Nielson, **Xiang Cui**, and Alex A. Aravind, Simple and Efficient Protocols for Guaranteed Message Delivery in Wireless Ad Hoc Networks, *Proc. of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications Conference (WiMob)*, 93-100, 2005.

3. Yong Sun, **Xiang Cui**, and K. Alagarsamy, Efficient Connectionless Semi-compulsory Routing Protocols for Mobile Ad-hoc Networks, *Proc. of the IASTED Intl. Conf. on Wireless Networks and Emerging Technologies (WNET)*, 484-489, 2004.

4. **Xiang Cui**, Yong Sun, and K. Alagarsamy, Simple and Efficient Connectionless Semi-Compulsory Routing Protocols for Mobile Ad-hoc Networks, *Proc. of the Intl. Conf. on Wireless Networks (ICWN)*, 906-909, 2004.

Acknowledgments

First of all, I would like to express my profound gratitude to my supervisor, Dr. Alex Alagarsamy Aravind for his invaluable support and constant encouragement. Without his guidance and endless patience this research would not have made it so far. I would also like to thank the members of the Examining Committee, Dr. Robert Tait (Dean of Graduate Studies), Dr. Waqar Haque, Dr. Jing Chen, and the external examiner, for their time and effort on my thesis. Thanks to Graduate Studies Office and the Computer Science Department of University of Northern British Columbia for financially supporting my research through Teaching and Research Assistantships.

I am thankful to former graduate students Yong Sun and Baljeet Singh Malhotra for their helpful and fruitful discussions on my research and my fellow graduate students Jeyaprakash, Srinivas, and Bharath for proofreading the draft of my thesis. I would also like to thank Paul Stokes for maintaining a reliable Tsdev server and Rob Lucas for the computing support. Taking this opportunity, I would like to thank Dr. Alex's wife, Dr. Mahi and the family for the wonderful family dinners and parties on various events and occasions. Last but not least, I would like to express my deepest gratitude to my family, who have always believed in me and supported me. Their tremendous support made me to come to Canada and earn a Master's degree.

Chapter 1

Introduction

1.1 Background

Mobile computing is a technology which enables people to connect their computing and communication devices to networks whenever and wherever they go. The maturity of technology and the falling cost of equipments make the applications of mobile computing widely available and affordable to both business executives and common users. Most of the traditional wireless networks, such as cellular telephony, personal communication systems, wireless local area networks, etc., are supported by static infrastructure (also called backbone). The infrastructure consists of fixed base stations or access points, which are connected either through wires or by long range wireless transmissions to act as gateways and bridges in the network. However, setting up of a fixed infrastructure is not always viable in ad-hoc situations such as battlefield, emergency search, rescue operation, etc. In such situations, an infrastructureless network known as Mobile Ad-hoc Network (MANET) is more attractive. A mobile ad-hoc network is a collection of mobile hosts with wireless interfaces forming a temporary network without the aid of any established infrastructure or centralized administration.

Research studies in mobile computing are carried out mainly through simulations because of their flexibility and cost effectiveness. Building a valid, credible, and appropriately detailed simulation model is crucial for conducting meaningful and accurate simulation study. In most of these simulation based research studies, modeling mobility of participating nodes plays a crucial role. The underlying mobility pattern heavily influences the behavior of the system under investigation. Random mobility models are dominantly used to model the mobility patterns of the mobile nodes in the system. A typical random mobility model is characterized as follows. A collection of mobile nodes is initially placed within a simulation area and then each node moves randomly within that area. It is generally assumed that individual nodes move completely independent of each other, which makes the implementation of this class of movement models sufficiently simple.

Recently a survey has been conducted on the MANET research papers published in MobiHoc Symposiums proceedings between 2000 to 2005 [11]. In that, it is observed that 75% of the papers used simulations in their research and in that simulations 38.5% of the papers used mobility in the study. There are many mobility models proposed in the literature, in that, the random mobility model called *random waypoint* is used dominantly (more than 73%) [11].

1.1.1 Motivation

Although many simulation softwares are freely available in the Internet for mobile computing simulations, we found that there is no comprehensive software freely available for the researchers with the following characteristics: (i) support for a wide range of random mobility models, (ii) support for visualization and analysis of the mobility trace[1] of the chosen model so that a suitable choice can be made, (iii) support for performance observations from the mobility trace to get more accurate insight about the system behavior, and (iv) support of a user friendly interface to effectively exploit the supports (i), (ii), and (iii). Lack of such a comprehensive software for generating mobility trace constrains the researchers to choose only from a few models supported in the generic simulation softwares and that in turn questions the validity of most simulation results. On the other hand, the availability of a comprehensive software to generate, analyze, and adjust various scenarios by suitably controlling the mobility parameters before applying it to the simulation would be extremely useful for the researchers in the mobile computing field.

The concern related to credibility of simulations has been reaffirmed in a recent survey showing that the credibility of the simulation results has decreased while the use of simulation has increased[11]. Part of this credibility crisis is clearly related to mobility model used to simulate the mobile nodes in the simulation system. The objective of this thesis is to develop a comprehensive mobility trace generator and analysis software.

[1] The trails of mobile nodes.

1.1.2 Contributions

This thesis presents the mobility generator software called **RMobiGen** that we developed using Java. RMobiGen can be used to specify, visualize, analyze, and then generate mobility traces for various random mobility models. In addition to its rich functionality, the software has a user friendly interface which increases its appeal for its wider use. We followed software engineering techniques and used UML diagrams during analysis and design phases of the development and then implemented using Java. We also surveyed the mobility models and some popular softwares supporting the mobility models presented in the literature. Finally, we conducted various experiments on them using RMobiGen. During the experimentation, we confirmed the phenomena related to mobility models indicated in the literature and also encountered many new interesting behaviors and patterns.

We believe that our work would serve as a guideline for researchers to choose the right model for their simulations. Also, this thesis brings many key concepts and observations scattered in the literature together under a common classification along with some new insights.

1.1.3 Organization

The rest of the thesis is organized as follows. Chapter 2 reviews random mobility models and relevant performance metrics. Chapter 3 presents the development of RMobiGen and simulation experiments using RMobiGen for various mobility models are described in Chapter 4. The thesis is concluded in Chapter 5.

Chapter 2

Random Mobility Models

2.1 Introduction

Most simulations use random mobility models to test the protocols, algorithms, or systems under study. In this section, we first describe the random mobility models and the performance metrics used to analyse the models. Then we review the random mobility models presented in the literature and the popular software tools that have implemented some of these models.

Although random mobility models have been analyzed in the literature for complex subspaces like Fish Bowl, Torus, Swiss Flag, etc.[13], the practical implementations are mostly restricted to either 2-dimensional square or 2-dimensional rectangle for ad hoc networks. We consider 2-dimensional rectangle as the region of mobility in RMobiGen.

A random mobility model can be viewed as an alternating sequence of *pause* and a continuous movement. We call the continuous movement between any two consecutive pauses as a *leg* of the mobility. That is, pause and leg are the components of a random mobility and it acquires the randomness from the computation of one or both of these two components. Probability distributions are used to generate the random values for these components. Computing a value for a pause is simple and therefore the crux of constructing a mobility model is in computing the legs.

2.2 A Classification

Although random mobility models appear to be simple, their behavior is less obvious, sometimes even counter-intuitive. In this section, we classify the random mobility models based on the way they compute leg and pause. There are five parameters, referred as *leg parameters*, that can be used to compute a leg: i) speed (S), ii) direction (θ), iii) distance (Δ), iv) time duration (T) and v) destination (D) in the simulation region R. Based on the parameters used to compute the leg, random mobility models can be classified, into five categories: (1) *random destination-speed model*, (2) *random destination-time model*, (3) *random direction-speed-time model*, (4) *random direction-speed-distance model*, and (5) *random direction-time-distance model.*

We have implemented these five models in RMobiGen. In random direction models, it is possible for a mobile node to hit the boundary during a leg travel. So, the random direction models require some kind of boundary actions. We have implemented the following three boundary actions suggested in the literature.

1. *Reflection*: The boundary is treated as a mirror to reflect the node into the mobile region when it hits the boundary.

2. *Wrap-around*: The node instantaneously reappears at the corresponding position on the opposite boundary.

3. *Restart*: The node pauses at the edge and then generates a new leg to move into the mobile region.

Three probability distributions, uniform, normal, and exponential are supported in RMobiGen to compute the parameters and the default distribution is uniform. Next we discuss the performance metrics used to analyze the mobility aspects.

2.3 Performance Metrics

Performance metrics used to analyze mobility models fall mainly into the following categories.

1. *movement metrics:* number of legs, leg distance, leg speed, leg duration, etc.

2. *connectivity metrics:* number of connections, connection duration, connection changes, connection availability, etc.

3. *coverage metrics:* node distribution, coverage, etc.

These metrics, if applicable and meaningful, may be computed for minimum, maximum, average, total, standard deviation, rate, ratio, etc., and also for individual, group, or system level. For example, connectivity may be analyzed between two nodes, between one and a group of nodes, or among all nodes. Movement metrics are easy to understand and straightforward to compute. Connectivity and coverage metrics require some elucidation.

Communication is a fundamental problem for most applications in any networked systems and achieving effective communication between the mobile nodes is challenging due to the dynamics involved in the network. In mobile networks, the nodes with transmission range form a dynamic graph called **connectivity graph**. The performance of most of the communication protocols in this context is heavily influenced by the connectivity and the coverage of this graph. We introduce some terminology related to connectivity and coverage which will be used later in our experiments to compute performance metrics.

2.3.1 Terminology

Definition 2.1 *A* **link** *is said to exist or be ON between two nodes i and j if they are within each other's transmission range. Link is a boolean function over time t and it is denoted by $link(i, j, t)$.*

Link is a communication channel and it can be generalized to path as follows.

Definition 2.2 *A* **path** *is said to exist or be ON between two nodes i and j if there is a sequence of nodes and the links between consecutive nodes in the sequence are ON. Path is also a boolean function over time t and it is denoted by $path(i, j, t)$.*

The next definition is to capture the duration of path existence. The state of the path oscillates between ON and OFF.

Definition 2.3 *We define the interval between ON and immediate OFF state of a path between the nodes i and j as a* **session**. *Session is also a boolean function on time t and it is denoted by $session(i, j, t)$.*

Path duration is an important metric for testing communication protocols. For example, some protocols referred as connection-oriented protocols require the path between source and destination to be ON throughout the communication.

Coverage is influenced by both mobility and transmission range of the nodes. Node distribution and the ratio of the area covered by transmission range to the total area are useful metrics to be analyzed for coverage. Since RMobiGen is a discrete time based simulator, the metrics are computed over discrete times.

2.4 Review

This section reviews the random mobility models proposed in the literature and the simulation softwares which support some of these models.

2.4.1 Mobility Models

Brownian Motion in science and random walk in mathematics are historically well known random mobility models. The mathematical description and analysis of Brownian motion goes back at least to 1900 by Louis Bachelier in his PhD thesis and then later by Einstein in 1905. First we briefly review these two models.

In one dimensional random walk, imagine that a walker is moving randomly on the integers [38]. The walker starts at 0 and at every integer time n the walker flips a coin and moves one step to the right if it comes up heads and one step to the left if it comes up tails. This is generalized to n-dimension as the walker is moving randomly on the n-dimensional integer grid. The model has both the time and space increments as 1. That is, the walker moves once every 1 time unit and when the walker moves, he or she moves a distance of 1 unit. The random walk model has the following interesting property [38].

Property 2.1 *A random walk on a one or two dimensional surface returns to the origin with probability 1, and when $n > 2$ the probability of returning to the origin becomes strictly less than 1.*

Brownian motion is a model of continuous random motion. One of the ways of defining Brownian motion is to look at it as a limit of a simple random walk where the time and space increments approach 0. That is, when the time and space increments are close to 0, the random walk appears as almost a continuous motion.

Next we discuss the random mobility models used in the mobile computing. In the cellular networks context, where the cells are used in the mobility model, a mobile node chooses one of its neighboring cells to move next. In such cases, the direction change is determined by a probability matrix. This is applicable to random walks on graphs also. In mobile ad hoc networks, there are two classes of models used with minor variations and they are referred as random direction/walk models and random waypoint models. We review these models next. In the review, for brevity of reference, we label the models.

Random Walk/Direction Models

This class of models are variations and in some cases generalizations of classical random walk models. We will review them next.

*RD*1: This model is introduced in [35]. In this model, the movement of each mobile node in the coverage area is characterized by its velocity vector $\vec{v} = (v,\ \theta)$, where v is the node's speed and θ is the direction, measured with respect to the positive x-axis. The position of the node and its velocity vector are updated *periodically*, by choosing the increments or decrements of θ and v from the given ranges *uniformly.* The mobile node that exits the coverage area from one side, *reenters the coverage area on the opposite side with the same velocity and same direction.*

*RD*2: This model referred as *random walk model* in [22] is described as follows. Each mobile node randomly chooses a direction in $[0, 2\pi)$, a speed between 0 and 10m/s and travels for *a fixed period* (60s). This process repeats.

*RD*3: This model also referred as *random walk model* in [22] is described as follows. Each mobile node randomly chooses a direction in $[0, 2\pi)$, a speed between 0 and 10m/s and travels *a fixed number of steps* (10 steps). This process repeats.

*RD*4: This model referred as *Brownian Motion* in [31] and renamed as *random direction model* in [25] is described as follows. Each mobile node chooses a direction in $[0, 2\pi)$ using uniform distribution, a speed value using normal distribution, and a time from fixed value or from exponential distribution. Then, the node travels in the chosen direction with the chosen speed for the chosen time period. The process repeats. When a node hits the boundary, *it bounces off the simulation border with an angle determined by the incoming direction.*

*RD*5: This model a simplified version of *RD*4 and it is introduced in [24]. In this model, each node chooses a direction and moves with *a constant speed* until it hits the boundary. When it hits the boundary, it will be *reflected* back into the coverage area immediately.

*RD*6: This model is proposed and referred as *random direction model* in [26]. It is another simplified version of *RD*4 with pause time and similar to *RD*5 without pause time. In this model, each node chooses a direction and *a random speed* to travel until it hits the boundary. When it hits the boundary, it *pauses for a fixed time* and then *chooses a direction in* $[0, \pi]$ to travel into the coverage area. This model was introduced in response to the *density wave* phenomenon observed in random waypoint model.

*RD*7: This is a variation to *RD*6 called *modified random direction* introduced in [26], and in that instead of the nodes always hitting the boundary they may choose a random destination along the chosen direction.

$RD8$: This variation called *smooth random mobility model* was introduced in [25]. It is a random direction model with the following additional feature called autocorrelation. In this model, (i) the speed is changed incrementally by the current acceleration of the mobile user, and also the direction change is smooth, and (ii) the movement patterns correlate the direction change with the speed change.

$RD9$: This model called *random drunken mobility model* is implemented in GloMoSim [9]. It is basically the simple random walk on grids, described in the beginning of this section. That is, if a node is at a position (x, y) then it can move to one of (x-1, y), (x+1, y), (x, y-1), and (x, y+1) as long as the new position is within the coverage region.

$RD10$: This model, given in [23], is a generalization of $RD9$. In this model, the nodes can choose all possible directions.

$RD11$: This model, called Gauss-Markov model, was implemented in [32] as follows: at fixed intervals of time, movement occurs by updating the speed and direction of each mobile node. The next speed and direction is computed based on the current speed and direction.

Comparing these models with the models in our classification given in Section 2.2, $RD1, RD2, RD4, RD5$, and $RD6$ are random direction-speed-time models, and

$RD8$ and $RD11$ are random direction-speed-time models with Markov property. The models $RD3$ and $RD7$ are random direction-speed-distance models. The models $RD9$ and $RD10$ are random direction-time-distance models.

Random Waypoint Models

Random waypoint model is the most popularly used model for simulations in MANET and it has been analyzed extensively in the literature.

$RW1$: This model is the first in random destination class, introduced in [37], and named as *random waypoint* in [34]. In this model, each node pauses at its current location for *a fixed period*, which is called *pause time*, and then randomly chooses a new location and velocity from a *uniform distribution* to move to. This behavior is continued, alternatively pausing and moving to the new location in the coverage region. This model has been implemented in NS2 [1].

$RW2$: This variation is introduced and used in [28]. Instead of using a constant value, the pause time is also chosen from *a uniform distribution.*

$RW3$: This variation called *steady-state random waypoint model* is introduced in [16]. In this model, the speed and destinations are generated from steady-state distributions derived in [14].

$RW4$: In [21], random waypoint model is generalized in such a way that the destination is chosen from *uniform distribution* and velocity and pause time are chosen from *arbitrary distributions.*

Comparing these models with the models in our classification, the models $RW1$ to $RW4$ are random destination-speed models.

In some models, particularly proposed in [20, 21], some of the nodes are chosen to be stationary in the beginning. Properties of random mobility models and a simple relationship to random waypoint model is given in [12].

2.4.2 Simulation Softwares Supporting some Random Mobility Models

Numerous simulation softwares are available that can be used for mobile computing simulations. Here we review only the widely used simulation softwares.

1. **NS2:** A discrete event simulator developed jointly by UCB, USC/ISI, and Xerox targeted at research over fixed and wireless (local and satellite) networks[1]. It supports random waypoint mobility model.

2. **GloMoSim:** A scalable simulation environment for wireless and wired network systems with parallel discrete-event simulation capability provided by *Parsec*, a parallel programming language. GloMoSim is developed at UCLA[2] and it supports random waypoint, random-drunken, RPGM (group mobility) models.

3. **QualNet:** The commercial version of GloMoSim with enhanced features[3].

4. **OPNET:** The largest, most comprehensive library of simulation models for the industry-level network modeling, analysis, and design[4]. The OPNET *wireless module* provides the ability to model many aspects of wireless transmissions.

5. **MATLAB:** A general purpose mathematical software package for high performance numerical computation and visualization[5]. Simulation for wireless communications is supported by an integrated interactive simulation environment called *Simulink*.

6. **SWAN:** A configurable and scalable simulator designed purely for wireless network simulations[6]. It is organized as independent components that can be composed to form a complete wireless network or a sensor network configurations. SWAN's capabilities are similar to NS2 and GloMoSim, but is claimed to be able to simulate much larger networks. SWAN is developed at Dartmouth College and it supports random waypoint, Brownian, probwaypoint (random direction with predetermined probabilities to choosing same or different directions), Gauss-Markov, Boundless (RD1) models.

In the above list, QualNet, OPNET, and MATLAB are commercial softwares and the rest are free and open source softwares and hence widely used by the academic community. Among the open source softwares, NS2 is the most widely used one. Basically, there are two techniques to incorporate node mobility into the simulators: (i) simulators having internal support for mobility and (ii) mobility trace is generated externally using other softwares and is given as an input to the network simulator.

For example, in NS2, the following command "`$node_($i) random-motion state`" is specified in the simulation script. Specifying state 1 chooses the random mobility model built in NS2 and choosing state 0 either allows mobility trace from external or keeps the nodes stationary. For GloMoSim and SWAN, the user can choose a mobility model by specifying it in the simulation configuration file. The following tools were designed specifically to generate and/or visualize mobility models.

- **nam:** A visualization tool comes with NS2 distribution allowing users to playback simulation output file generated *after* a simulation is over. It is originally designed for wired network and has the following drawbacks:
  - animation is limited to only node movements
  - no statistical insights provided
  - animation is based on a simulation output file generated *after* the simulation completion; this process can take lots of time and the output file generated can grow to be hundreds of megabytes.
- **iNSpect:** Another visualization and animation tool designed for NS2 simulator[10]. It takes a NS2 trace file as the input and displays nodes movement as well as wireless links. The transmissions are displayed with route lines and color coded nodes. Because it can animate a movement scenario without running NS2, compared to *nam*, *iNSpect* is a better tool that can be utilized with minimal overhead. However, other than visualization, no statistical analysis is supported.

- **BonnMotion:** It is a Java-based tool which generates movement scenarios of a set of random models on grids: random waypoint, ManhattanGrid, Gauss-Markov, and RPGM)[8]. Some statistical measures, such as relative mobility, average node degree, and average link duration, are provided.

- **setdest:** This also comes with the NS2 distribution which generates mobility trace for random waypoint model.

- **mobgen:** A tool developed by Tracy Camp's research group on the steady state distribution for the random waypoint model.

2.5 Summary

This Chapter, after presenting a comprehensive overview and a simple classification of various mobility models, provides a review of existing mobility models and the popular simulation softwares that implement some of the models.

As the number of nodes and movement duration increases, the mobility trace file will become overwhelmingly hard to analyze. This, in most cases, forces the researchers to adopt the mobility trace with little knowledge of what is exactly going on within the trace before applying it to the simulation. Therefore, it is desirable to have the mobility generator and analysis software independent of simulator itself. We present RMobiGen, designed to help the researchers to choose the mobility models suitable for their research, in next Chapter.

Chapter 3

RMobiGen

This Chapter presents the design and development of RMobiGen, a software of size about 8,000 lines Java code - the major contribution in this thesis. We used software engineering approach during the developmental stages of RMobiGen. We start with the requirement specification.

3.1 Specification

From the user's point of view, when the user specifies a mobility model, RMobiGen has to generate the movement trace accordingly. More importantly, it should allow the users to analyze the generated mobility trace for various aspects before exporting it in a required format. This requires a multilevel user interface for the users to navigate the system conveniently, in specifying and analyzing the mobility models.

To get the formal specification of the system, we start with use-case diagram.

Constructing an use-case diagram involves the following three steps.

- Identification of the actors (the roles played by various users while accessing the system).
- Identification of the use-cases (the ways of using the system).
- Setting the relationships between the actors and use-cases, and between the use-cases themselves.

The researcher, who is interested in using RMobiGen to analyze and generate mobility trace, is the only actor in the system. The main use-cases are:

- Generating the mobility trace,
- Analyzing and visualizing various scenarios from the trace, and
- Exporting the trace to NS2 format.

Since the pauses can be extracted from the start and end times of the legs, the mobility trace of a node can be considered as a sequence of legs. The mobility trace for all the nodes is the collection of the mobility traces of the individual nodes, as shown in Figure 3.1.

Generating the mobility trace includes generating the mobility traces from the five random mobility models discussed in the previous Chapter. Visualization and analysis of the mobility trace include, visual analysis and computing statistical metrics for movement, connectivity, and coverage aspects, discussed in Chapter 2. Adding together all these use-cases, we get the use-case diagram given in Figure 3.2.

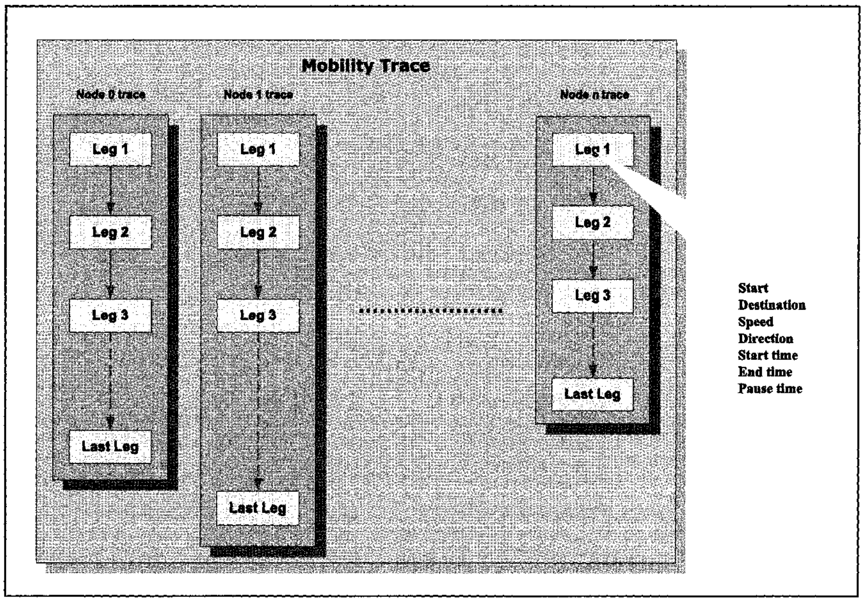


Figure 3.1: Internal Structure of Execution Trace

Next we present the higher level architecture and subsequently explain the design of its components.

3.2 Architecture

Based on the use case diagram, in a higher level, RMobiGen can be constructed by three main components.

- *Mobility Trace Generator:* Responsible for generating the mobility trace.
- *Mobility Trace Manager:* Responsible for extracting various statistical insights and providing visualizations.
- *Mobility Trace Exporter:* Responsible for converting the mobility trace into NS2 format explained in Chapter 2.

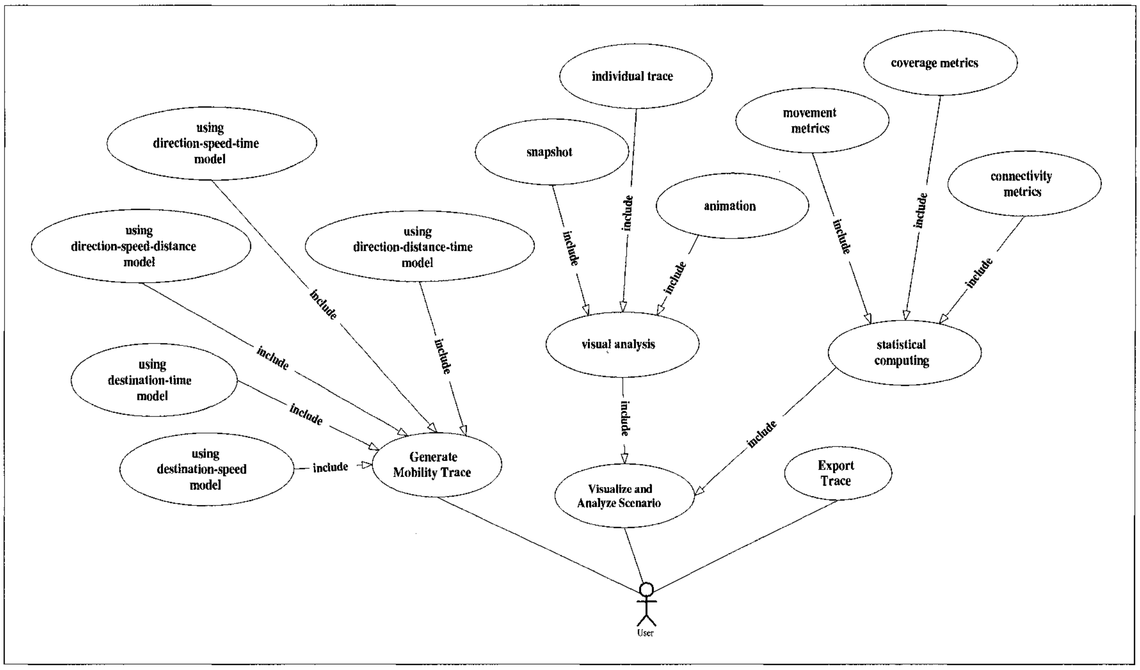

individual trace
coverage metrics
movement metrics
using direction-speed-time model
snapshot
connectivity metrics
animation
using direction-speed-distance model
using direction-distance-time model
include
visual analysis
statistical computing
using destination-time model
using destination-speed model
Generate Mobility Trace
Visualize and Analyze Scenario
Export Trace
User

Figure 3.2: Usecase Diagram

Figure 3.3 illustrates the basic architecture.

3.3 Class Diagram

Class diagram depicts the structural aspects of the system. A class essentially has three logical components: data attributes, operations that involve services from other classes, and operations to access the member attributes of the class. Identifying classes is an iterative process. Based on the insights we obtained from the above analysis, the following main classes are identified for RMobiGen.

- User specifications (Parameter)
- Mobility Trace Generator (MobilityGenerator, RandomNumberGenerator, LegGen-

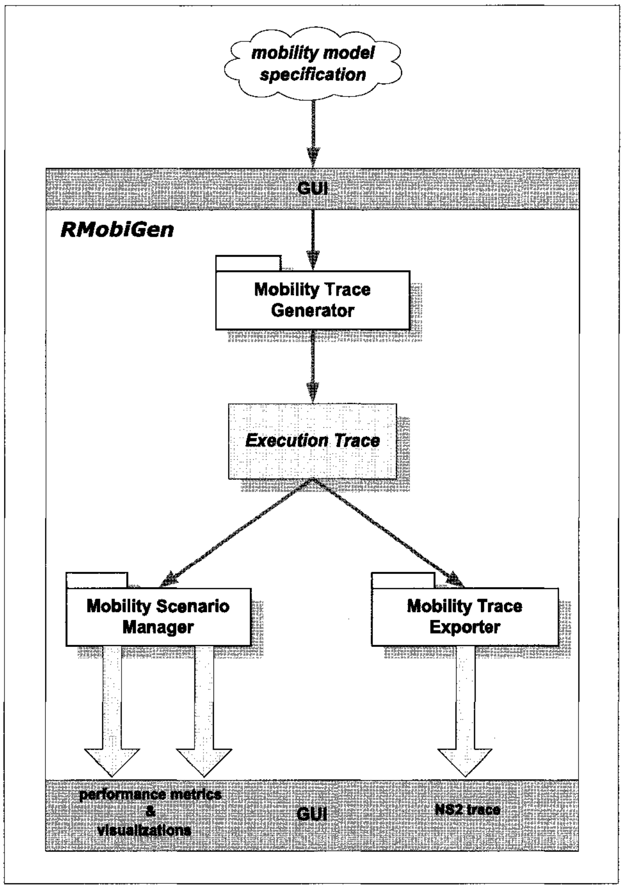

mobility model specification
GUI
RMobiGen
Mobility Trace Generator
Execution Trace
Mobility Scenario Manager
Mobility Trace Exporter
performance metrics & visualizations
GUI
NS2 trace

Figure 3.3: RMobiGen Architecture

erator)

- Mobility Trace Manager (MetricGenerator, Visualizer)

- Mobility Trace Exporter (TraceExporter)

The class diagram illustrating the classes and their relationships is given in Figure 3.4.

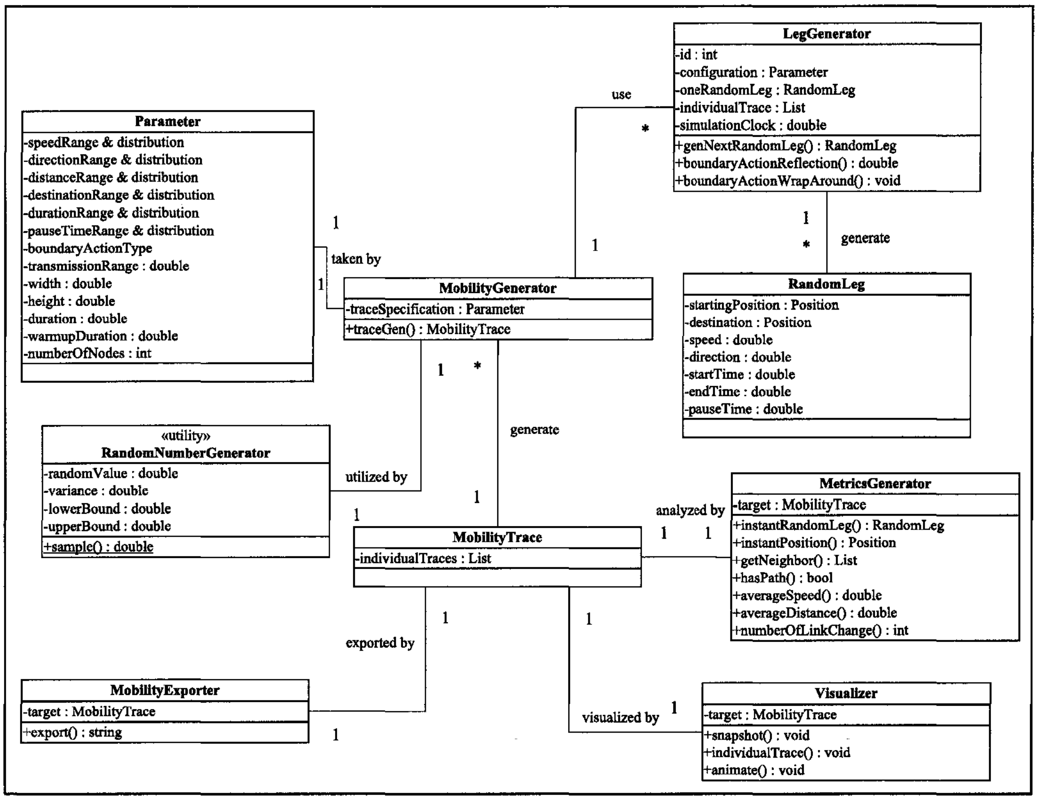


Figure 3.4: Class Diagram

3.4 Modeling Key Operations

In this section, we present some key operations implemented in RMobiGen. Building interface is a complex task and we used Java Swing and AWT package with the help of JBuilder IDE. Here, we present some selected operations of mobility trace generation and performance metrics computation.

3.4.1 Trace Generation

The trace generator uses the event-driven method to build the mobility trace for each node as follows.

1. Set simulation clock value to 0.

2. Compute the leg parameter values according to the input specification.

3. Set the leg start time to simulation clock value and leg end time to simulation clock value + leg travel time.

4. Set simulation clock value to leg end time + pause time.

5. If simulation clock value < simulation end time, then go to step 2.

Leg parameters are sampled based on its range and the probability distribution type using the `RandomNumberGenerator.sample()` operation. The different versions of `genNextRandomLeg()` operation can be described in terms of different mobility models as follows:

1. Random destination-speed model: Sample a new destination, a new speed, and a new pause time.

2. Random destination-time model: Sample a new destination, a new time duration, and a new pause time.

3. Random direction-speed-distance model: Sample a new direction, a new speed, a new distance, a new pause time.

4. Random direction-speed-time model: Sample a new direction, a new speed, a new time duration, and a new pause time.

5. Random direction-time-distance model: Sample a new direction, a new distance, a new time duration, and a new pause time.

3.4.2 Performance Metrics Computation

Performance metrics are extracted from the mobility trace by the `MetricsGenerator`. It uses time-step based technique to collect data. The time-step based data collection works as follows.

1. Set simulation clock value to 0.

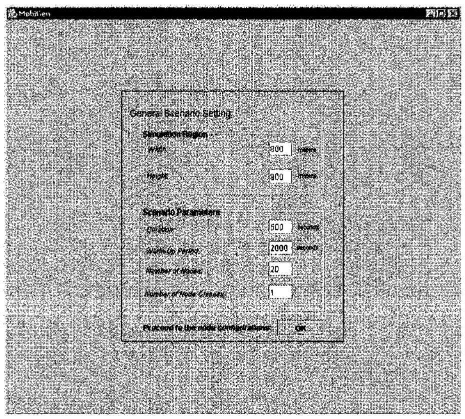
2. Collect data from the legs whose time matches the simulation clock value.

3. Increment simulation clock time by a fixed unit.

4. If simulation clock value $<$ simulation end time, then go to step 2.

The computation from the collected data may be updated as soon as the data is collected for that time step or computed at the end.

Individual metrics are collected by traversing through the `individualTrace` list at a node and the system metrics can be obtained by iterating through all the `individualTrace` lists. To compute the connectivity metrics, a series of "snapshots" of the connectivity graph is taken once for every time step using the `instantPosition()` operation.



General Scenario Setting

Simulation Region

Width: 800 meters

Height: 800 meters

Scenario Parameters

Duration: 600 seconds

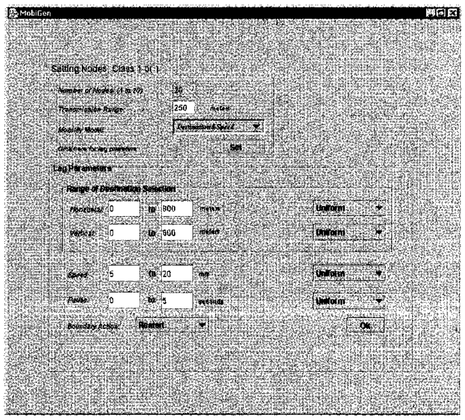
Warm-Up Period: 2000 seconds

Number of Nodes: 20

Number of Node Classes: 1

Proceed to the node configurations: OK

Figure 3.5: General Scenario Setting Panel



Setting Nodes: Class 1 of 1

Number of Nodes (1 to 20): 20

Transmission Range: 250 meters

Mobility Model: Destination & Speed

Set

Leg Parameters

Range of Destination Selection

Horizontal: 0 to 800 meters Uniform

Vertical: 0 to 800 meters Uniform

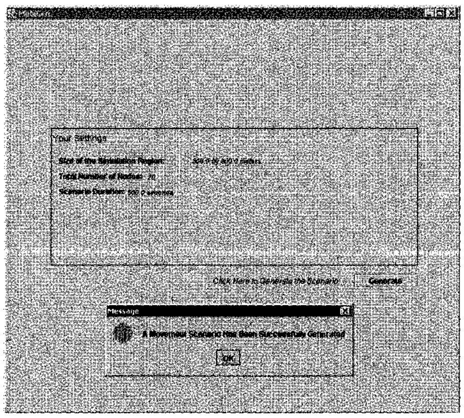
Speed: 5 to 20 m/s Uniform

Pause: 0 to 5 seconds Uniform

Boundary Action: Restart

OK

Figure 3.6: Leg Parameter Setting Panel



Your Settings

Size of the Simulation Region: 800.0 by 800.0 meters

Total Number of Nodes: 20

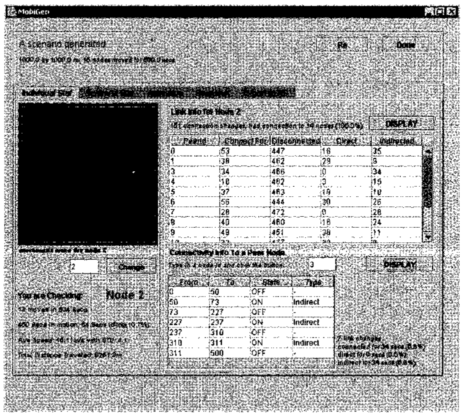
Scenario Duration: 600.0 seconds

Click Here to Generate the Scenario Generate

A Movement Scenario Has Been Successfully Generated

OK

Figure 3.7: Mobility Trace Generated



A scenario generated

Re Done

Individual Stat

Link Info for Node 2

DISPLAY

PeerId	Connect For	Disconnected	Direct	Indirected
0	53	447	18	35
1	38	462	29	9
3	34	466	0	34
4	18	482	3	15
5	37	463	19	18
6	56	444	30	26
7	28	472	0	28
8	40	460	16	24
9	49	451	38	11

Change

You are Checking Node 2

Connectivity Info To a Peer Node

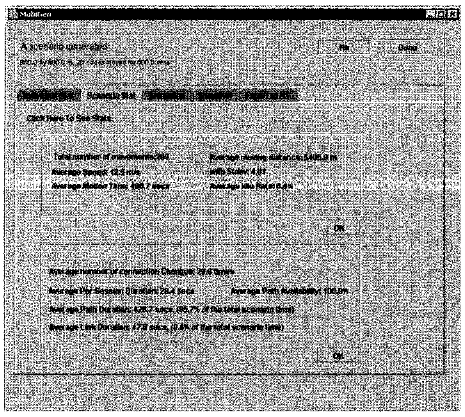
Type in a node id and click the button. 3

DISPLAY

From	To	State	Type
0	50	OFF	-
50	73	ON	Indirect
73	227	OFF	-
227	237	ON	Indirect
237	310	OFF	-
310	311	ON	Indirect
311	500	OFF	-

7 link changes

Figure 3.8: Individual Stat subpanel



A scenario generated

800.0 by 800.0 m, 20 nodes moved for 600.0 secs

Re Done

Scenario Stat

Click Here To See Stats

Total number of movements: 268

Average moving distance: 5405.8 m

Average Speed: 12.5 m/s

with Stdev: 4.01

Average Motion Time: 486.7 secs

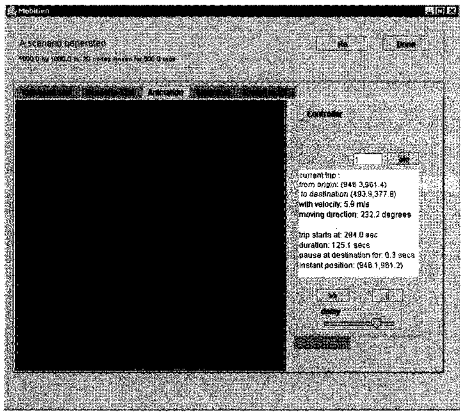
OK

Average number of connection Changes: 29.6 times

Average Path Availability: 100.0%

OK

Figure 3.9: Scenario Stat subpanel



A scenario generated

Re Done

Animation

Controller

1

current trip :
from origin: (948.3,961.4)
to destination (493.9,377.8)
with velocity: 5.9 m/s
moving direction: 232.2 degrees

trip starts at: 284.0 sec
duration: 125.1 secs
pause at destination for: 0.3 secs
instant position: (948.1,961.2)

Figure 3.10: Animation subpanel

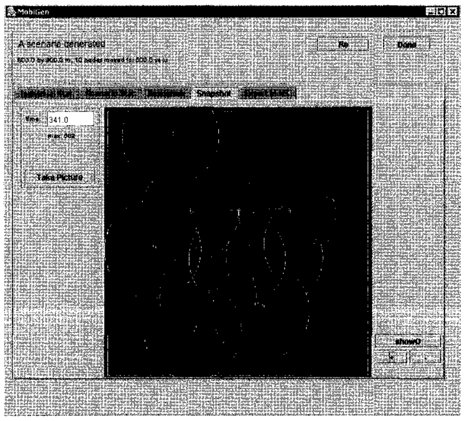
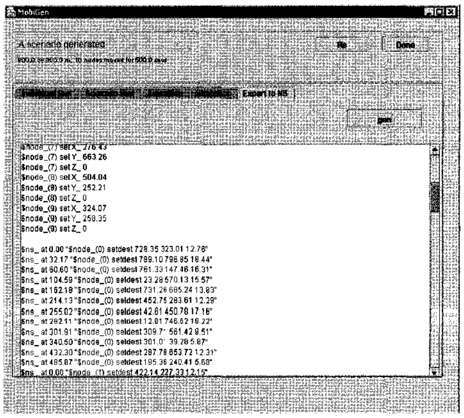


Figure 3.11: Snapshot subpanel



```
$node_(7) setX_ 276.43
$node_(7) setY_ 663.26
$node_(7) setZ_ 0
$node_(8) setX_ 504.04
$node_(8) setY_ 252.21
$node_(8) setZ_ 0
$node_(9) setX_ 324.07
$node_(9) setY_ 258.35
$node_(9) setZ_ 0

$ns_ at 0.00 "$node_(0) setdest 728.35 323.01 12.76"
$ns_ at 32.17 "$node_(0) setdest 789.10 796.85 18.44"
$ns_ at 60.60 "$node_(0) setdest 761.33 147.46 16.31"
$ns_ at 104.59 "$node_(0) setdest 23.28 570.13 15.57"
$ns_ at 162.18 "$node_(0) setdest 731.26 685.24 13.83"
$ns_ at 214.13 "$node_(0) setdest 452.75 283.61 12.29"
$ns_ at 255.02 "$node_(0) setdest 42.61 450.78 17.16"
$ns_ at 282.11 "$node_(0) setdest 12.81 746.62 19.22"
$ns_ at 301.91 "$node_(0) setdest 309.71 561.42 9.51"
$ns_ at 340.50 "$node_(0) setdest 301.01 39.28 5.87"
$ns_ at 432.30 "$node_(0) setdest 287.78 653.72 12.31"
$ns_ at 465.87 "$node_(0) setdest 195.36 240.41 5.68"
$ns_ at 0.00 "$node_(1) setdest 422.14 227.33 12.15"
```

Figure 3.12: Export to NS subpanel

3.4.3 User Interface

Here, we present snapshots of the RMobiGen graphical user interface. User inputs the specifications at *general scenario setting panel* and *leg parameter setting panel* demonstrated in figure 3.5 and 3.6 respectively. In the former case, the user specifies simulation settings such as duration, warm-up time, the size of the simulation region, the number of mobile nodes. In the latter case, the user specifies the leg parameters. After the specifications are entered, the user clicks the OK button to generate the mobility trace. A panel will pop up to indicate the success of the generation (figure 3.7). By clicking the OK button, the user can see the performance metrics and visualize the scenario in the *post-generation panel.* User can navigate through the subpanels by clicking on tabs. The contents presented at each subpanel can be described in terms of the tab:

1. **Individual Stat subpanel** (figure 3.8): Presents performance metrics of individual nodes.

2. **Scenario Stat subpanel** (figure 3.9): Provides statistical summary of over all nodes.

3. **Animation subpanel** (figure 3.10): Traces animation.

4. **Snapshot subpanel** (figure 3.11): Displays nodes topology at any instant.

5. **Export to NS subpanel** (figure 3.7): Converts the mobility trace into NS format and displays.

3.5 Summary

In this Chapter we presented the analysis, design, and implementation of RMobiGen. Also, we explained how RMobiGen can be used to generate mobility trace, visualize various performance and behavioral aspects, and then export to NS2 format. In addition, thanks to the Java2 technology, MobiGen has been made available as a Java Applet hence it can be conveniently accessed through Internet (url: `web.unbc.ca/~cuix/rmobigen`) without the need for installation. In the next Chapter, we discuss the experiments that we conducted for various mobility models using RMobiGen.

Chapter 4

Simulation Study

Numerous experiments can be conducted using RMobiGen by varying the mobility models, the probability distributions, the size of the simulation area, etc., and we conducted many such experiments during the development of RMobiGen to confirm its correctness. This Chapter presents only a selected set of simulation experiments conducted using RMobiGen mainly to study the movement, the connectivity, and the coverage metrics. The results we obtained from our experiments confirm and, in some cases, refine the past observations about random mobility models.

We will discuss movement, connectivity, and coverage in separate sections. In each section, we start with the analysis of main observation(s) made in the literature related to that section. Then we present other related observations from our experiments using RMobiGen. Since the random waypoint model is used dominantly in simulations, most observations are related to the random waypoint model.

4.1 Experimental Setup

Unless otherwise mentioned, the following are the default values used for the input parameters.

- Size of the simulation area - 1000 by 1000 meters.
- Speed range - $[0.01, 20]$, in meters per second.
- Transmission range - 250 meters. (Not relevant for node movement and distribution analysis).
- Distance range - $[0.01, 1414]$, in meters. (1414 is the diagonal length of 1000×1000).
- Time range - $[1, 100]$, in seconds.
- direction range - $[0, 359]$, in degrees.

The number of nodes and simulation duration used vary from experiment to experiment.

4.2 Movement Analysis

We start with a notorious observation regarding nodal speed in random waypoint model.

Observation 4.1 *In random waypoint model, the average nodal speed consistently decreases over time, and therefore should not be directly used for simulation [18] (***speed decay***).*

This phenomenon is true for any model which has the possibility of choosing a speed close to zero to complete a leg travel without specified time bound. When a node chooses a speed closer to zero, it will take infinitely long time for that leg travel. During this long period, it is quite possible that some other nodes also might choose speeds close to zero and that will reduce the number of nodes moving with the speed above average. From Figure 4.1, we can observe the speed decay for the models involving speed as a parameter: destination-speed and direction-speed models, when the speed is from uniform distribution.

The speed decay problem can be avoided directly by not letting any node to choose a speed less than a threshold value or simply choosing the distribution function for the speed as normal. The comparison of using normal and uniform distributions for ·destination-speed and direction speed models is shown in Figure 4.2, where the speed chosen from normal distribution has no speed decay and the average nodal speed is close to the actual average.

Experiments were conducted to observe the number of legs performed for destination-speed and direction-speed-time models for both uniform (Figures 4.3 & 4.4) and normal (Figures 4.5 & 4.6) distributions.

From Figures 4.3 & 4.4, for uniform distribution on speed, we can observe that a considerable number of nodes are stuck with only one leg while some other nodes

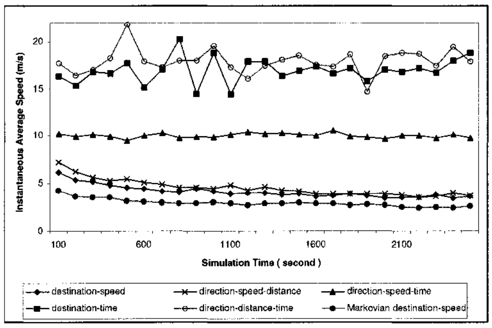


Figure 4.1: Uniform Speed

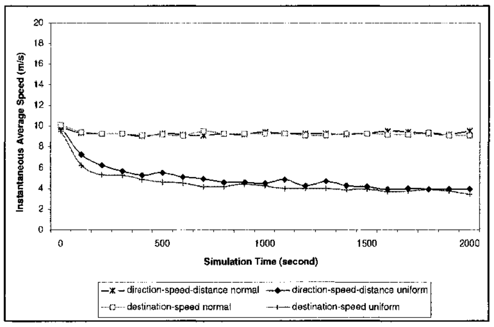


Figure 4.2: Normal and Uniform Speeds

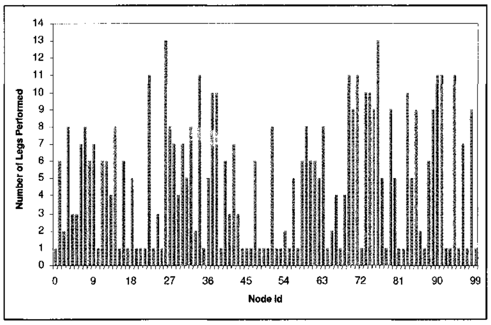


Figure 4.3: Destination-Speed Model

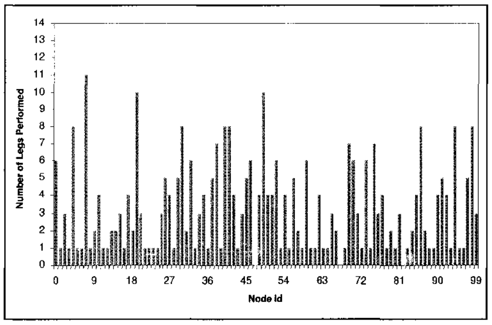


Figure 4.4: Direction-speed-distance Model

can perform as many as 10 or above legs in 1000 seconds. For normal distributions, as shown in Figures 4.5 & 4.6, the variations are small and only a few nodes perform as low as 4 legs. This explains why the speed decay occurs.

We now summarize the techniques, from which one or more can be used to choose the speed of the nodes, in order to solve the speed decay problem.

1. Do not allow the nodes to choose their speed close to zero.
2. Allow the simulation to reach the steady state before collecting data.
3. Use stationary (or steady-state) distribution for speed.
4. Use normal distribution for choosing the speed.

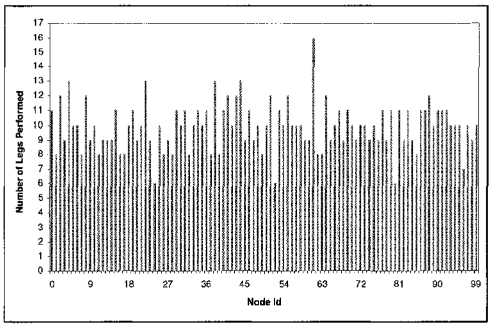


Figure 4.5: Destination-Speed Model

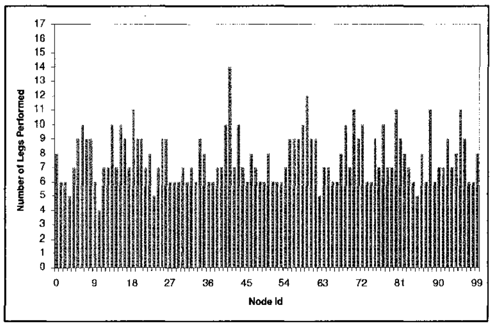


Figure 4.6: Direction-speed-distance Model

The observation related to direction and speed changes reported in the literature is the following.

Observation 4.2 *It has been criticized by Hong et al. in [30] that many researchers use a mobility model where the new choice for speed v and direction θ is not correlated to previous values (such as in the random waypoint model). This may cause unrealistic movement behavior with* **sudden speed** *and* **sharp turnings***[25].*

After presenting this observation, the authors suggested a random direction model with the next direction and speed correlated respectively with current direction and speed. We implemented another random direction model where the the next direction is chosen with some offset to current direction and the offset is derived from normal distribution in the range -180 to 180. The movement observed is shown in Figure 4.8, which is smoother compared to the movement of the same model shown in Figure 4.7 where the offset is chosen from uniform distribution.

The following experiments were conducted to analyze the number of leg movements, the average leg length, the average leg length ratio to the diagonal, and the

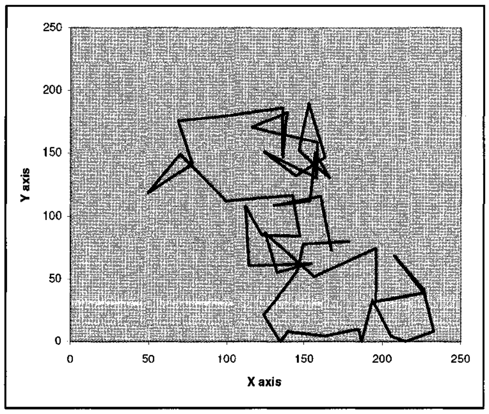


Figure 4.7: Uniform Direction

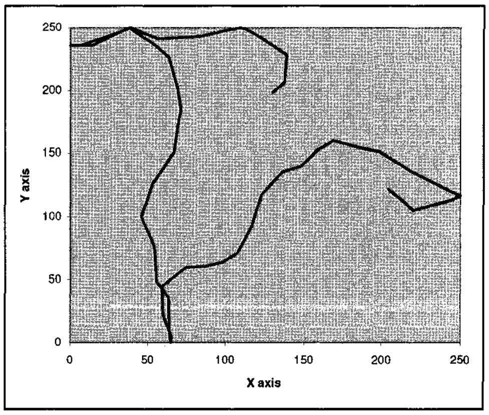


Figure 4.8: Normal Direction

average of total distance traveled. The experiments are conducted for 1000 nodes for the duration of 500 seconds. The results are presented in Figures 4.9 to 4.12.

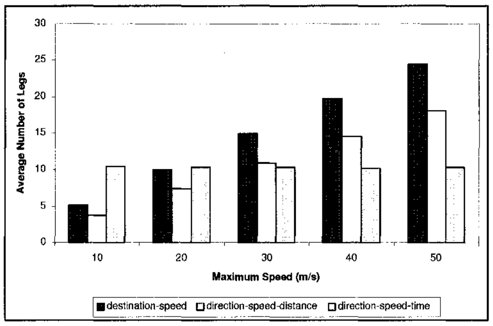


Figure 4.9: Number of Legs

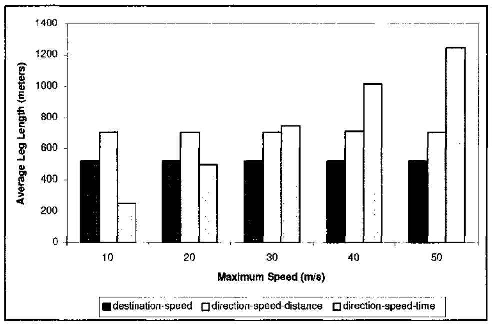


Figure 4.10: Average Leg Length

From Figures 4.9 to 4.12, we can make the following observations.

- Average number of legs traveled increases as the speed increases for destination-speed and direction-speed-distance models, whereas, the average does not change for direction-speed-time model. It is easy to see that change in speed, without proportional change in time, will not affect the number of legs traveled. However, it may increase the total distance traveled.

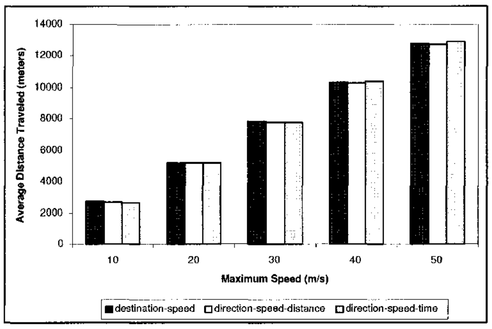


Figure 4.11: Average Distance Traveled

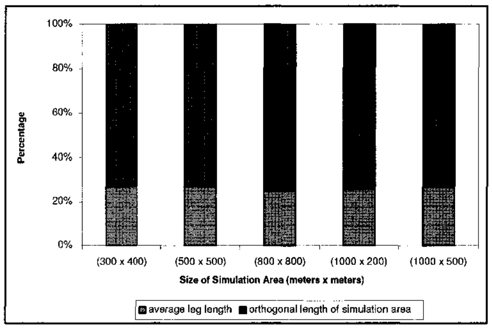


Figure 4.12: Distance Ratio for Random Waypoint

- The average leg length is not affected by change in speed in the destination-speed and direction-speed-distance models, whereas, the average leg length is proportional to the speed in direction-speed-time model.
- The average distance traveled increases proportionately to speed for all three models: destination-speed, direction-speed-distance, and direction-speed-time models.
- The ratio between the average leg length and the diagonal of the simulation area appears to be constant for random waypoint model.

4.3 Coverage Analysis

The objective in this section is to study how the nodes spread in the simulation region during the experiments. To compute this, the region is divided into small cells of equal size (10 by 10) and the nodes are counted inside each cell. Initially, 1000 nodes are uniformly placed inside the region. After the nodes move for 2000 seconds, a snapshot of the positions of the nodes is taken and the number of nodes at each cell is counted for their spatial distribution.

We start with another popularly referred observation for random waypoint model relating to coverage aspect.

Observation 4.3 *An inherent characteristics of random waypoint model is that the nodes converge in the center of the area, and then disperse, and then re-converge, etc., resulting in* **density waves** *[26].*

Since the destinations in random waypoint (destination-speed in our classification) are chosen uniformly from the simulation region, the clustering of nodes in the center is intuitive that there is a higher probability that the nodes will travel to or travel through the center region. However, we did not observe consistent density waves during our experimentation. The center clustering phenomenon observed in the destination-speed model is given in Figures 4.13 and for the destination-time model is given in Figures 4.14. Both models have similar spatial distributions as expected.

In the direction-speed models, the distributions are similar and not center clustered. However, instead of being in the center, the nodes cluster along the boundary region when the boundary action is restart (RS). For the cases of reflection (RF) and reenter, the nodes are distributed uniformly for the random direction models. It is easy to observe that reflection and restart actions have the same the effect on the nodes distribution in these models. The difference is that, in reflection action, the node is reflected from the same boundary and, in case of reenter action, the node

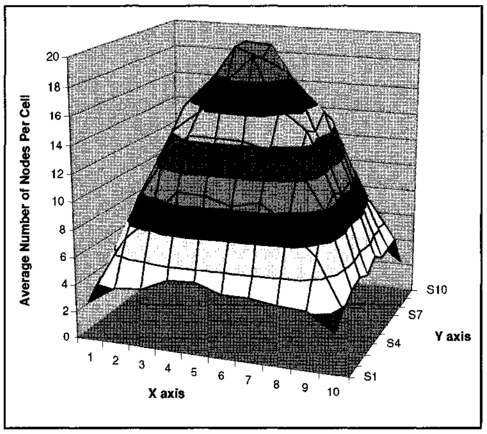


Figure 4.13: Destination-speed

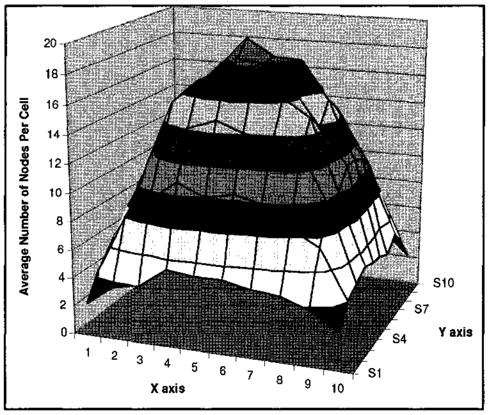


Figure 4.14: Destination-time

enters from the opposite boundary. In case of restart action, since the nodes pause at the boundary, the clustering occurs at the boundary. The coverage for the random direction models with reflection as the boundary action are shown in Figures 4.15, 4.17, & 4.19, and the coverage for the random direction models with restart as the boundary action are given in Figures 4.16, 4.18, & 4.20.

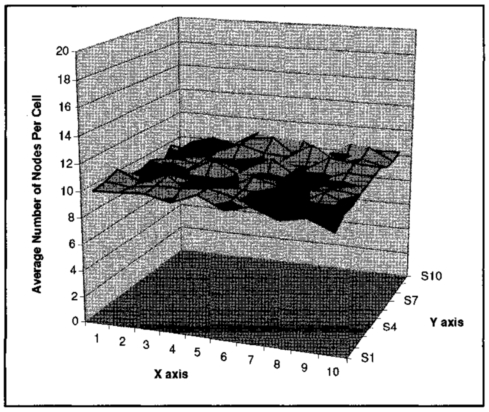


Figure 4.15: Direction-speed-distance (RF)

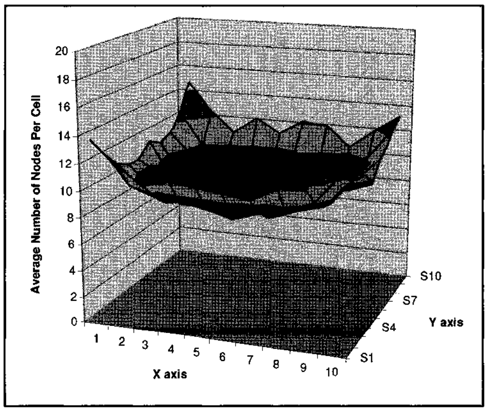


Figure 4.16: Direction-speed-distance (RS)

Center clustering, boundary clustering, or completely uniform are not realistic

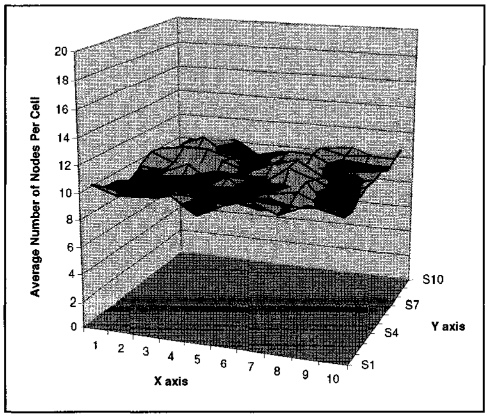


Figure 4.17: Direction-speed-time (RF)

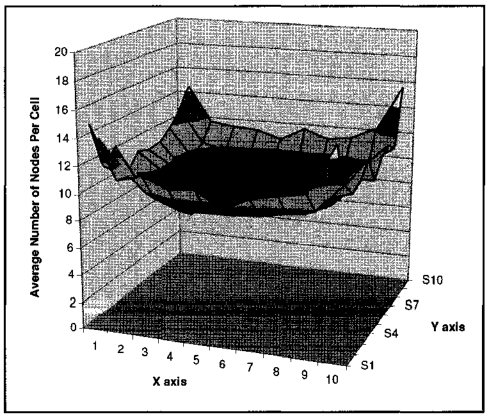


Figure 4.18: Direction-speed-time (RS)

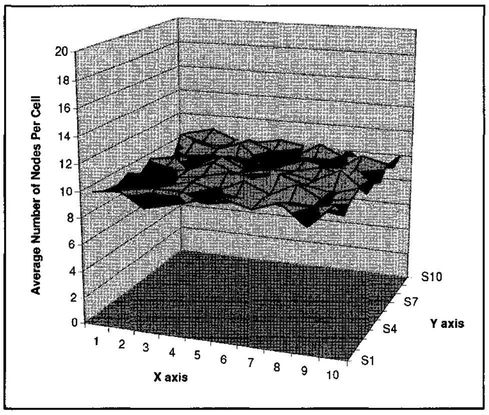


Figure 4.19: Direction-distance-time (RF)

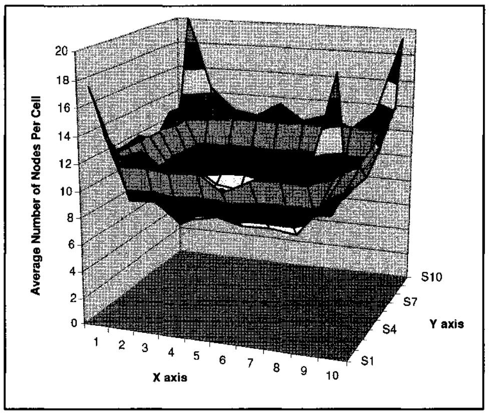


Figure 4.20: Direction-distance-time (RS)

situations in most cases. In many cases, (i) it is common that more nodes stay in the inner region compared to the boundary regions at any instance and (ii) the spatial distribution is generally less uniform. To achieve these, we developed a mobility model in which the next position is chosen in the neighborhood of current position. We call this model as **Markov-destination model**. We simulated this model using RMobiGen and we observed that it does not have the clustering phenomenon either in the center or along the boundary, as shown in Figures 4.21 & 4.22. In Figure 4.21

the destination is chosen within a rectangular neighborhood of size $\frac{1}{4}$th of the mobility region and in Figure 4.22 the destination is chosen within a rectangular neighborhood of size $\frac{1}{8}$th of the mobility region.

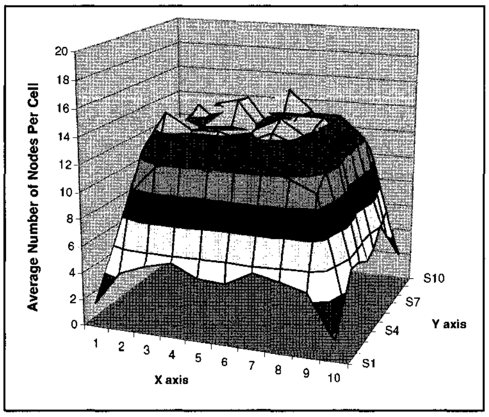


Figure 4.21: Markov-Destination ($\frac{1}{4}$)

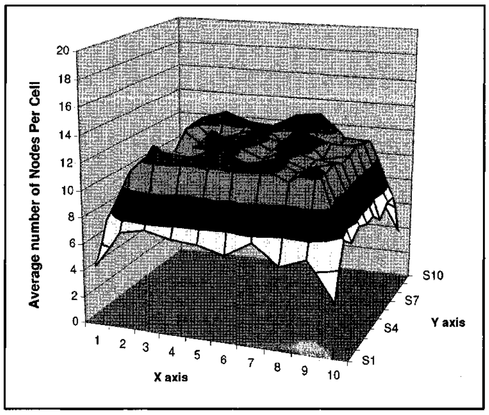


Figure 4.22: Markov-Destination ($\frac{1}{8}$)

4.4 Connectivity Analysis

For connectivity analysis, we mainly study two metrics: the connection changes and the session duration defined in Chapter 2. Three experiments were conducted by changing the transmission range, the speed, and the size of the simulation region. For each experiment, 16 nodes were simulated for 500 seconds. The observations are presented in Figures 4.23 & 4.24 by varying transmission range, in Figures 4.25 & 4.26 by varying speed, and in Figures 4.27 & 4.28 by varying size of the simulation regions.

From Figures 4.23 to 4.28, we make the following observations.

- The change in connectivity is small when the range is small (hardly get con-

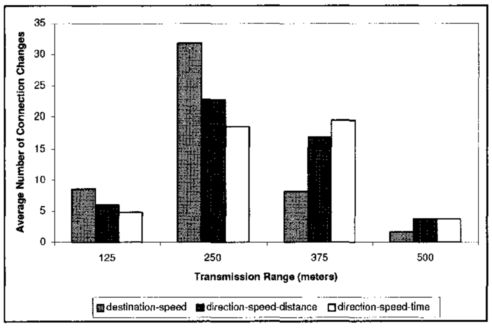


Figure 4.23: Connection Change

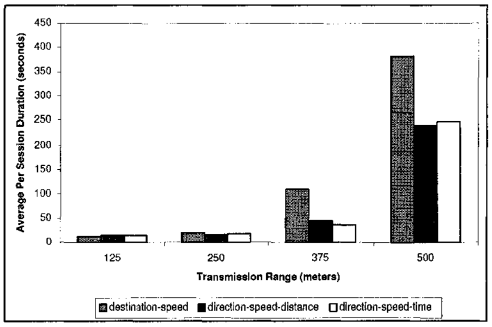


Figure 4.24: Session Duration

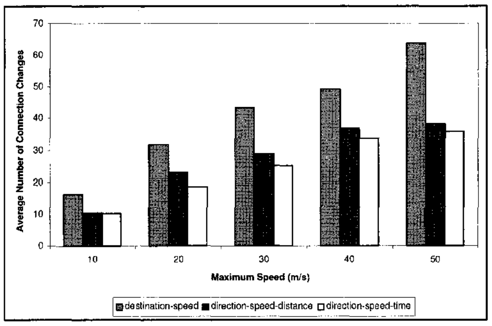


Figure 4.25: Connection Change

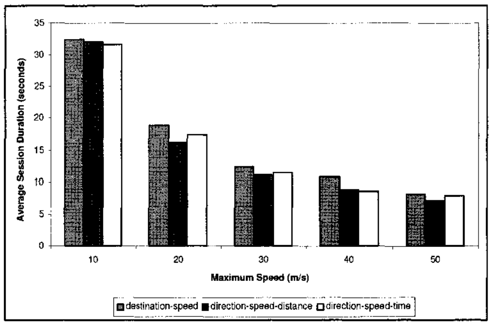


Figure 4.26: Session Duration

nected), or the range is high (often connected). The change in connectivity is high when the range is medium.

- The change in connectivity is small when the simulation area size is small, or the size is high. The change in connectivity is high when the size is medium.
- The change in connectivity increases as the speed increases.
- The average session duration increases as the transmission range increases and decreases as the size of the simulation region increases or as the speed increases.

The connectivity metrics will be useful when studying communication protocols using these models.

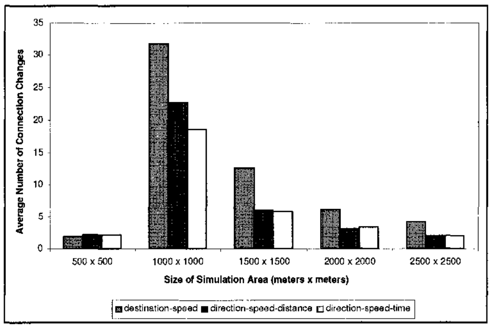


Figure 4.27: Connection Change

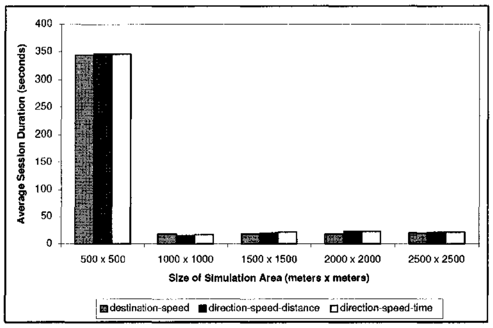


Figure 4.28: Session Duration

4.5 Implementation Complexity

Random waypoint is the most popular model heavily used for simulations. The main reason, in addition to its support in NS2, is stated in the literature as follows.

Assertion 4.1 *Random waypoint is a simple model that is easy to analyze and implement. This has, probably, been the main reason for the widespread use of this model for simulations [22].*

The assertion is true only for the particular case where the mobility region is an n-dimensional rectangle with sides aligned to coordinate axes. It loses its simplicity even for rectangles if the sides are not aligned with the coordinate axes. Also, implementing random direction models is not terribly complex compared to the implementation of random waypoint model. So the researchers should not hesitate to consider using other random mobility models if they are more relevant for their study.

4.6 Summary

In this Chapter, we presented a selected set of experiments using RMobiGen on Random mobility models. The experiments clearly illustrate the variations and their impact on the dynamics of the mobility models. Also, our experiments confirm some of the widely observed phenomena and presents some new observations.

Chapter 5

Summary and Future Directions

5.1 Summary

In recent times, mobile computing has received increasing attention from the research community. Also, many applications based on mobile networks will be realized in the near future. Research studies in mobile computing are carried out mainly through simulations due to its flexibility and cost effectiveness. In most of these simulation based research studies, modeling the mobility of participating nodes plays a crucial role. This thesis deals with mobility related issues.

The contributions in this are manyfold: First we presented a comprehensive overview of random mobility models and a simple classification. Next, the existing random mobility models and the popular softwares that support some of these models were presented. Then the design and implementation of RMobiGen, which are the major contributions of this thesis, were presented. We followed software engineering techniques and used UML diagrams during analysis and design phases of the development and then implemented using Java. RMobiGen can be used to generate, analyze, and adjust various scenarios by suitably controlling the mobility parameters before applying it to the simulation. Finally, we conducted an extensive set of experiments to analyze various characteristics of random mobility models.

We believe that our work and the insights we presented from our experiments on random mobility models would serve as guidelines for researchers to choose the right model for their simulations.

5.2 Future Directions

There are many directions in which the work presented in this thesis can be expanded. We outline some of them next.

- RMobiGen assumes the mobility region as 2-dimensional rectangle. It can be extended to accommodate other geometric regions.
- The mobility models can be extended to the regions with obstacles.
- Also, the models can be further extended to the regions with attraction points, introduced in[17], to create more realistic mobility patterns.

Bibliography

[1] The Network Simulator - ns-2, http://www.isi.edu/nsnam/ns/.

[2] Global Mobile Information Systems Simulation Library - GloMoSim, http://pcl.cs.ucla.edu/projects/glomosim/.

[3] QualNet, Network Simulation Software, http://www.scalable-networks.com.

[4] OPNET Technologies, http://www.opnet.com/.

[5] MATLAB - The Language of Technical Computing, http://www.mathworks.com/products/matlab/.

[6] SWAN Simulator for Wireless Ad-hoc Networks, http://www.eg.bucknell.edu/swan.

[7] SWAN Manual, http://www.eg.bucknell.edu/swan/doc/refman.pdf.

[8] BonnMotion - A mobility scenario generation and analysis tool, http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion

[9] L. Bajaj, M. Takai, R. Ahuja, K. Tang, R. Bagrodia, and M. Gerla, GloMoSim: A Scalable Network Simulation Environment, Report, University of California, Los Angeles.

[10] Stuart Kurkowski, Tracy Camp, and Michael Colagrosso, “A Visualization and Animation Tool for NS-2 Wireless Simulations: iNSpect”, Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, p.503-506, September 27-29, 2005.

[11] S. Kurkowski, T. Camp, and M. Colagrosso, MANET Simulation Studies: the Incredibles, *Mobile Computing and Communications Review*, 9(4):50-61, 2005.

[12] P. Nain, D. Towsley, B. Liu, and Z. Liu, Properties of Random Direction Models, *Proc. of the IEEE INFOCOM*, 2005.

[13] J.-Y. Le Boudec and M. Vojnovi, Perfect Simulation and Stationarity of a Class of Mobility Models, *Technical report*, EPFL, July 2004.

[14] W. Navidi and T. Camp, Stationary Distributions for the Random Waypoint Mobility Model, *IEEE Transactions on Mobile Computing*, 3(1):99-108, 2004.

[15] C. A. V. Campos, D. C. Otero, and L. F. M. de Moraes, Realistic Individual Mobility Markovian Models for Mobile Ad hoc Networks,

Proc. of the IEEE Wireless Communication and Networking Conference, 2004.

[16] W. Navidi, T. Camp, and N. Bauer, Improving the Accuracy of Random Waypoint Simulations through Steady-State Initialization, *Proc. of the 15th International Conference on Modeling and Simulation*, 319-326, 2004.

[17] Michael Feeley, Norman Hutchinson, and Suprio Ray, Realistic Mobility for Mobile Ad Hoc Network Simulation, *ADHOC-NOW 2004*, 324-329, 2004.

[18] J. Yoon, M. Liu, and B. Noble, Random Waypoint Considered Harmful, *Proc. of the IEEE INFOCOM*, 2003.

[19] J. Yoon, M. Liu, and B. Noble, Sound Mobility Models, *Proc. of the ACM MobiCom*, 2003.

[20] Tracy Camp, Jeff Boleng, Brad Williams, Lucas Wilcox, William Navidi, Performance Comparison of Two Location Based Routing Protocols for Ad Hoc Networks. In *Proc. of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies*, 1678-1687, 2002.

[21] G. Resta and P. Santi, An Analysis of the Node Spatial Distribution of the Random Waypoint Model for Ad Hoc Networks. *Proc. ACM Workshop on Principles of Mobile Computing*, 44-50, 2002.

[22] T. Camp, J. Boleng and V. Davies, A Survey of Mobility Models for Ad Hoc Network Research, *Wireless Communication and Mobile Computing*, 2(5):483-502, August 2002.

[23] K. Obraczka and K. Viswanath, Flooding for Reliable Multicast in Multi-Hop Ad Hoc Networks, *Wireless Networks* 7, 627-634, 2001.

[24] Z. J. Haas and M. R. Pearlman, The Performance of Query Control Schemes for the Zone Routing Protocol, *IEEE/ACM Transactions on Networking*, 9(4):427-438, 2001.

[25] C. Bettsetter, Smooth is Better than Sharp: A Random Mobility Model for Simulation of Wireless Networks, *Proc. of the 4th ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems*, 2001.

[26] E. M. Royer, P. M. Melliar-Smith, and L. E. Moser, An Analysis of the Optimum Node Density for Ad Hoc Mobile Networks, *Proc. of the IEEE International Conference on Communications*, 2001.

[27] K. Pawlikowski, H.-D.J. Jeong, and J.-S.R. Lee, On Credibility of Simulation Studies of Telecommunication Networks, *IEEE Communications Magazine*, 132-139, 2001.

[28] E.M. Royer and C.E.Perkins, Multicast Operation of the Ad Hoc On-demand Distance Vector routing protocol, *Proc. ACM MobiCom*, 1999.

[29] B. Liang and Z. Haas, Predictive distance-based mobility management for PCS networks, *Proc. of IEEE INFOCOM*, 1999.

[30] X. Hong, M. Gerla, G. Pei, and C. C. Chiang, A Group Mobility Model for Ad Hoc Wireless Networks. *Proc. ACM/IEEE MSWiM*, 53-60, 1999.

[31] C. Hartman and H.-J. Vogel, Teletraffic Analysis of SDMA-Systems with Inhomogeneous MS Location Distribution and Mobility, *Kluwer Wireless Personal Communications*, 11(1):45-62, 1999.

[32] V. Tolety, Load Reduction in Ad Hoc Networks Using Mobile Servers. Master's thesis, Colorado School of Mines, 1999.

[33] C. Chiang, Wireless Networks Multicasting. PhD thesis, Department of Computer Science, University of California, Los Angeles, USA, 1998.

[34] 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, *Proc. of the Fourth Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking*, 85-97, 1998.

[35] Z. Haas, A New Routing Protocol for the Reconfigurable Wireless Networks, *Proceedings of the ICUP*, 1997.

[36] Amotz Bar-Noy, Ilan Kessler and Moshe Sidi, Mobile Users: To Update or Not to Update?, *ACM/Baltzer Wireless Net*, vol. 1, no. 2, pp. 175-185, 1995.

[37] D. B. Johnson and D. A. Maltz, Dynamic Source Routing in Ad Hoc Wireless Networks, In *Mobile Computing*, ed. T. Imielinski and H. Horth, 153-181, Kluwer Academic Publishers, 1996.

[38] Eds: G. F. Lawler and L. N. Coyle, *Lectures on Comtemporary Probability*, American Mathematical Society, 1991.

[39] O. Balci, Credibility Assessment of Simulation Results, *Proc. of the Winter Simulation Conference*, 38-43, 1986.