Department of Computer Science
University College London
University of London

Context-aware Adaptive Routing for Delay Tolerant Networking

Mirco Musolesi
m.musolesi@cs.ucl.ac.uk

Submitted in partial fulfilment of the requirements
for the degree of Doctor of Philosophy
at the University of London

May 2007

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

I, Mirco Musolesi, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Delay tolerant networking has received considerable attention from the research community in the recent years. Advances in wireless and mobile technologies have enabled new application scenarios where intermittent disconnections are common and not exceptional. Examples include communication in sparse mobile ad hoc networks for emergency support, infostation-based systems for connectivity in remote areas and data collection in sensor networks for wildlife monitoring. At the same time, most of the existing research work in mobile networking is based on the assumption that a path exists between the sender and the receiver. Therefore, new communication paradigms and techniques have to be designed to make communication possible in these environments.

In this thesis we present the design, implementation and evaluation of the Context-aware Adaptive Routing (CAR) protocol for delay-tolerant unicast communication in intermittently connected mobile ad hoc networks. The protocol is based on the idea of exploiting nodes as carriers of messages among network partitions to achieve their delivery. The choice of the best carrier is made using Kalman filter based prediction techniques and utility theory. We argue that movement of the nodes and potential future colocation with the recipient of the messages can be used to make intelligent forwarding decisions.

We then discuss the design of a realistic mobility model based on social network theory in order to test the routing protocol. An evaluation of the proposed model using real traces and its mathematical formalisation are also presented.

The performance of the Context-aware Adaptive Routing protocol in terms of delivery ratio, delay and overhead are evaluated using simulations based on the proposed mobility model and purely random ones.

Finally, we discuss the implementation of CAR over an opportunistic networking framework (Haggle), outlining possible applications of the general principles at the basis of the proposed approach.

Acknowledgements

It is not possible to carry out and complete the preparation of a PhD thesis without the help and the presence of other people.

First of all, I would like to thank my supervisor Cecilia Mascolo for transmitting her passion for research to me, for her constant guidance and for her friendship, both in happy and difficult moments. This thesis is also the result of long discussions in front of the whiteboard and of her continuous inspiration and care during these years.

I would also like to thank Wolfgang Emmerich for his support in these years: in particular, I would like to thank him and Cecilia for accepting me as Master student for the preparation of my degree thesis in the first place. I will never forget the interview during the check-in at Bologna airport in a winter day some years ago: I have to say, everything started from there.

The initial design of CAR was also inspired by the stimulating discussions with Stephen Hailes; I would also like to thank him for his useful comments during the preparation of this manuscript.

I would also like to thank all the people sitting on the seven floor and the folks of the room 107 of the beloved Pearson building, in particular to Rami Bahsoon for his friendship and help during the first years of my PhD program. I am grateful to Franco Raimondi for his thorough comments on a draft of this work and for avoiding the transformation of myself into a node with average speed equal to 0 during the preparation of this thesis, by forcing me to the weekly Sunday runs.

This thesis is dedicated to my parents, that have supported me during all these long and wonderful years of study. I would like to thank them for teaching me the importance of knowledge and for understanding all my choices, including my movements around Europe in the recent years. This thesis is also dedicated to my brother Mauro, a person that is very important for me, also now that we are distant most of the time. He is always able to give me a different and positive perspective in everything.

Unfortunately, tragic events also occurred during the preparation of this thesis. I would like to remember three persons that are not able to see the completion of this work: my grandfather Antonino, that has been an example of determination and tenacity, also during his illness, my uncle Gianni, that supported me so much during these years and my friend Raffaella, that in her last email to was planning her visit to Camden Market. My last thought is for my grandfather: *nonno*, one day SCAR might run on your beloved sheeps: we had no time to talk and laugh about this, but, I know, now you are smiling.

Contents

List of Figures

Chapter 1

Introduction

Delay is preferable to error.

Thomas Jefferson

1.1 Background

Delay tolerant networking [Fall, 2003] has received considerable attention from the research community in recent years. Advances in wireless and mobile technologies have enabled new networked systems where intermittent disconnections are not exceptional. Examples include sparse mobile ad hoc networks [Li and Rus, 2000], deep space communication systems [IPNSIG, 2007], infostation-based systems [Frenkiel et al., 2000] and carrier based data collection in sensor networks [Shah et al., 2003].

In these types of networks, the fundamental assumption of existing TCP/IP based network model providing end-to-end communication [Saltzer et al., 1984] is not valid, as in those systems, a path between the sender and the receiver cannot always be assumed and so any *synchronous* communication paradigm is likely to show very poor performance.

From a wireless networks perspective, we define communication as being synchronous if it can only occur between hosts that are part of a set of entities that are all within radio range of at least one other host in the set[1]. In other words, synchronous communication

[1] For simplicity we assume that communication is symmetric.

between nodes can only occur when they are in radio transmission range of one another or when they can exploit an underlying synchronous routing protocol to communicate. Synchronous protocols rely on the fact that a connected path exists between the sender and the receiver of a message; the absence of such a path will, at best, lead to a failure indication to the originating host. If delivery is important, the best that can be done is for the sender to continue to poll for the receiver.

Conversely, *asynchronous* communication does not assume that the parties involved are located in the same connected portion of the system. Whilst, in the developed world, synchronous communication (in the form of telephone and Internet) is generally cheap and easy to come by, there are several real scenarios in less developed parts of the world in which different portions of a logical network are physically disconnected. Thus, for example, this is the case in recent projects established to assist nomadic communities such as the Saamis in Lapland [Doria et al., 2002] or to assist populations in rural areas of India [International Telecommunication Union, 2003, Seth et al., 2006], like the DakNet Project [Pentland et al., 2004]. In the latter, a number of villages have their own local networking infrastructure, but there is no interconnection between them nor to the Internet. A bus containing a wireless node travels between villages, picking up email in one and depositing it in others on its round. Self-evidently, although there is never a synchronous connection between sender and recipient, mail can still be delivered.

Other examples of delay tolerant networks are near-Earth satellites systems or deep space communications [Hooke, 2002] and under-water acoustic links [Partan et al., 2006]. These systems may be affected by high latencies and unpredictable disconnections based on environmental factors or predictable ones due to the movements of the entities of the systems (i.e., planets and satellites dynamics).

Finally, wireless ad hoc networks (including mobile ones and sensor networks) can be partitioned due to movement of the nodes or the inherent physical distances between the devices. In these networks, specialised nodes (also called data mules [Shah et al., 2003]) are used to provide a store-and-foward mechanism to collect and deliver data to mobile sinks. In ZebraNet [Liu et al., 2004], wireless nodes attached to animals are used to monitor wildlife in their habitat; data collected by these devices are transmitted to fixed stations when these are in reach of base stations.

With respect to mobile ad hoc networking [Perkins, 2001], most of the existing work is based on the assumption that a path exists between the sender and the receiver. Therefore, new communication paradigms and techniques have to be designed to make communication possible also in case of temporarily partitioned networks, due, for example, to host mobility. Our work addresses specifically the issues of this application domain.

A basic solution is to spread the messages to all hosts using a form of persistent flooding. In this approach, which is known as *epidemic routing* [Vahdat and Becker, 2000], a host

floods the message it wishes to send to all hosts within its connected portion of the network. Each carrier host buffers the message and if, as a result of movement, it comes into contact with hosts that do not have a copy, it transfers it to them, making them new carriers. Eventually, the message will reach all nodes in the system, provided that movement patterns allow for this. Epidemic routing is an effective approach when there is no information about the likely movement patterns of nodes in the system. In other words, when there is no basis on which to distinguish the movement pattern of a node from another, and the movement pattern of each node is individually random, the only choice is to place messages randomly or to place them everywhere, since there is no more intelligent basis for making a decision.

A small number of basic approaches have been proposed in the field of asynchronous communication for mobile ad hoc networks [Li and Rus, 2000, Chen and Murphy, 2001]. As described above, the challenge in producing an algorithm for delivering asynchronous messages derives from the deceptively simple question of determining the best carrier or carriers for each message. Clearly, leaving the message with the sender may also be inappropriate, since sender and receiver may never meet.

More recently, general solutions have been proposed to tackle the problem of routing in (possibly mobile) delay tolerant networks, based on the previous knowledge of the routes of the potential carriers [Jain et al., 2004, Zhao et al., 2004, N. Sarafijanovic-Djukic and Grossglauser, 2006] or on probabilistic approaches [Lindgren et al., 2003, Spyropoulos et al., 2005].

The routing protocol presented in this thesis is part of this effort, that has also led to the formation of an Internet Research Task Force research group [DTNRG, 2006].

In the next section, we define the research problem more precisely and we outline the contribution of this work with respect to the state of the art.

1.2 Research Problem and Thesis Contribution

This thesis focusses on the problem of supporting unicast communication in delay tolerant mobile ad hoc networks also in presence of intermittent disconnections, by means of a prediction based routing protocol, the Context-aware Adaptive Routing (CAR) protocol [Musolesi et al., 2005].

We argue that it is possible to enable the exchange of messages by exploiting nodes moving between disconnected groups of nodes (clouds) to transport them from one cloud to another. In fact, two nodes may *never* be part of the same connected cloud and, yet, may still be able to exchange delay tolerant information by making use of predicted mobility patterns as an indicator of which other nodes might make good carriers for this

information. We argue that prediction techniques can be used for an intelligent selection of the carrier. We assume that the probability that two hosts will be in reach in the future is a function of their past colocation history. Relative mobility (i.e., the rate of new encounters) is also considered in this process. These different aspects describing the mobile context are then represented by utilities that are composed using results from multi-criteria decision theory [Keeney and Raiffa, 1976]. We do not assume any previous knowledge of the routes of the hosts like other approaches, such as the Message Ferrying project [Zhao et al., 2004], that rely on the a priori knowledge of the routes of the special hosts carrying the information.

Moreover, our protocol is based on a single copy of the message in the system, instead of having multiple replicas. Other solutions are predicated on semi-epidemic algorithms like PROPhET [Lindgren et al., 2003], where the probability of replication is proportional to the time of the last encounters and their frequency. Finally, we do not exploit any geographical information such as GPS coordinates.

In general, our approach can be considered the first one exploiting forecasting techniques for carrier selection. Subsequently, in fact, our work has inspired the design of other protocols based on the study of mobility patterns such as [Leguay et al., 2006] and [Ghosh et al., 2007].

In order to test the proposed protocol, we needed a model to represent human mobility. Indeed, the large majority of the existing mobility models, such as the Random Way Point mobility model [Johnson and Maltz, 1996a], generate purely random movements that are very different from the ones observed in the real world and produces meaningless colocation patterns[2] [Sharma and Mazumdar, 2004, Chaintreau et al., 2006].

For this reason, we have designed a novel mobility model based on social network theory, the Community based mobility model [Musolesi and Mascolo, 2006a]. This relies on the simple observation that *mobile networks are social networks after all*, since mobile devices are carried by individuals. The model is able to generate movements that are based on the strength of the relationships between the people carrying the devices. To evaluate its realism, we validated it using real traces comparing some key characteristics, such as the contacts duration and inter-contact times.

To summarise, the contribution of the thesis is twofold:

- We present the design, the evaluation and the implementation of the Context-aware Adaptive Routing (CAR) protocol. This has been evaluated by means of large-scale simulations and has been implemented in the Haggle platform [Su et al., 2006].

[2]However, it is worth noting that CAR exploits not only colocation patterns but also the mobility of the hosts. Therefore, it shows good performance also in presence of random movements, since it is able to select the nodes characterised by high mobility, i.e., the ones that have a higher probability of reaching new hosts, increasing the likehood of establishing communication with the recipient of the message.

- We define the Community based mobility model using results from mathematical sociology and complex networks theory. The model has been validated using real mobility traces and it has been used for the evaluation of the CAR protocol.

The evaluation of the protocol is performed in three ways: we provide a theoretical framework, we discuss the results of simulations used for testing its performance in large scale-scenario and we present the implementation of a prototype on top of an opportunistic networking framework.

1.3 Thesis Outline

We firstly discuss the algorithmic aspects and the related mathematical foundation of the protocol and then its simulation and implementation. At the end of each chapter, we present the relevant related work, discussing the novelty of our contribution.

The remainder of this thesis is organised as follows:

- In Chapter 2 we present the Context-aware Adaptive Routing in detail, presenting the key design choices, the theoretical foundation of the protocol and the novel mechanisms that are at the basis of its implementation.
- In Chapter 3 we discuss the design of the Community based mobility model based on social networks theory and its validation using real traces.
- In Chapter 4 we offer thorough evaluation of the protocol, also using the proposed mobility model, discussing the trade-offs in terms of delivery ratio and overhead.
- In Chapter 5 we present the implementation of the protocol in the Haggle platform, analysing the key design choices and evaluating it in a real testbed.
- In Chapter 6 we summarise the contributions of this work, outlining possible current research directions.

Chapter 2

Design of the Context-Aware Adaptive Routing protocol

> Prediction is difficult, especially about the future.
>
> *Niels Bohr*

2.1 Overview of the Protocol

In this section we give an overview of the Context-aware Adaptive Routing (CAR) protocol, presenting the key design choices and the novel mechanisms that are at the basis of its implementation.

Firstly, we give an overview of the protocol, discussing the key points of its design. Secondly, we analyse the prediction theory and algorithms at the basis of this. Thirdly, we discuss its implementation, focussing on the management of routing information for synchronous and asynchronous delivery. Finally, we present the design and implementation of a component for evaluating the predictability of the time series used to describe and measure aspects of the mobile context.

As first step, we introduce the assumptions underlining the design of the protocol. We assume that the only information a host has about its position is related to its logical connectivity. In particular, we assume that a host is not aware of its absolute geographical

location nor of the location of those to whom it might deliver the message. Although this information could potentially be useful, there might also be battery implications of its use which might be unacceptable (for example, because of the energy requested to operate a GPS device).

Another basic assumption is that the hosts present in the system cooperate to deliver the message. In other words, we do not consider the case of hosts that may refuse to deliver a message or that act in a Byzantine manner. This is a typical assumption of standard mobile ad hoc routing protocols.

The design goal of CAR is to support communication in intermittently connected mobile ad hoc networks. Therefore, a path between sender and receiver may not exist when the message is sent. Store-and-forward mechanisms are necessary to allow for the delivery of the message to the destination.

The key problem is the selection of the carrier by means of an intelligent forwarding mechanism. Our solution is based on the application of forecasting techniques and utility theory for the evaluation of different aspects of the system that are relevant for making routing decisions.

Let us now consider the key aspects of the protocol. CAR is able to deliver messages synchronously (i.e., without storing them in buffers of intermediate nodes) and asynchronously (i.e., by means of a store-and-forward mechanism). The delivery process depends on whether or not the recipient is present in the same connected region of the network (cloud) as the sender. If both are currently in the same connected portion of the network, the message is delivered using an underlying synchronous routing protocol to determine a forwarding path.

If a message cannot be delivered synchronously[1], the best carriers for a message are those that have the highest chance of successful delivery, i.e., the highest *delivery probabilities*. The message is sent to the host with the highest one using the underlying synchronous mechanism.

In order to understand the operation of the CAR protocol, consider the following scenario in which two groups of nodes are connected as in Figure 2.1. As in our implementation, let us assume that the Destination-Sequenced Distance-Vector (DSDV) protocol [Perkins and Bhagwat, 1994] is used to support synchronous routing. Host H_1 wishes to send a message to H_8. This cannot be done synchronously, because there is no connected path between the two. Suppose the delivery probabilities for H_8 are as shown in Figure 2.1. In this case, the host possessing the best delivery probability to host H_8 is

[1]The recipient may be in the same connected portion of the network, but not reachable using synchronous routing, since the routing information is not available (for example, because the space in the routing tables is not sufficient to store the information related to all the hosts in the cloud or because the node has just joined the cloud). In these cases we exploit the asynchronous mechanisms.

H_4. Consequently, the message is sent to H_4, which stores it. After a certain period of time, H_4 moves to the other cloud (as in Figure 2.2). Since a connected path between H_4 and H_8 now exists, the message is delivered to its intended recipient. Using DSDV, for example, it is worth noting that H_4 is able to send the message shortly after joining the cloud, since this is when it will receive the routing information relating to H_8.

Delivery probabilities are synthesised locally from *context information* such as the rate of change of connectivity of a host (which, in our model, is used to measure the likelihood of it meeting other hosts) and its colocation with the message recipients. More specifically, we define *context* as the set of attributes that describe the aspects of the system that can be used to drive the process of message delivery. An example of context information can be the change of connectivity, i.e., the number of connections and disconnections that a host experienced over the last T seconds. This parameter measures relative mobility and, consequently, the probability that a host will encounter other hosts. Since we assume a proactive routing protocol, every host periodically sends both the information related to the underlying synchronous routing (in DSDV this is the routing tables with distances, next hop host identifier, etc.), and a list containing its delivery probabilities for the other hosts. When a host receives this information, it updates its routing tables. With respect to the table for asynchronous routing, each host maintains a list of entries, each of which is a tuple that includes the fields (*destination*, *bestHost*, *deliveryProbability*). We choose to explore the scenario in which each message is placed with only a single carrier rather than with a set, with the consequence that there is only a single list entry for each destination.

When a host is selected as a carrier and receives the message, it inserts it into a buffer. The size of this buffer is fundamental, and represents a trade-off between storage overhead and likely performance. If the buffer overflows, messages will be lost, since we assume the existence of a single replica.

What we have described is the basic model behind the CAR protocol. In the following sections we will describe the details of the algorithm, in particular the techniques exploited for the calculation of the delivery probabilities.

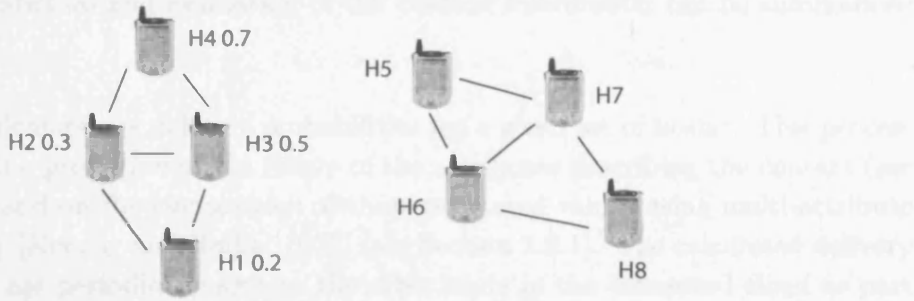

Figure 2.1: Two connected clouds, with associated delivery probabilities for message transmission between H_1 and H_8

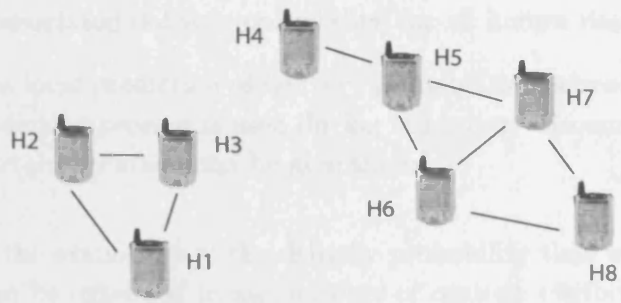

Figure 2.2: H_4, carrying the message, joins the second cloud.

2.2 Prediction and Evaluation of Context Information

The general problem from the point of view of the sender of a message (if we cannot synchronously send the message to the receiver) is to find the host with the best delivery probability among the ones in its connected portion of the network, as calculated using the predicted values describing the future evolution of a range of context attributes. We observe that it is fundamental to consider the *trend* of the variation of these attributes over time rather than their current values. For example, in the case of patterns of colocation, a host H_A currently not colocated with a host H_B may be considered of scarce utility for acting as a carrier for H_B if we evaluate only this instant of time. However, H_A may have been colocated with H_B for the past three hours and, therefore, its likelihood of being colocated again, given the assumptions of our model, are high and should be modelled accordingly. For this reason, we use a method that is able to consider the past history to make a prediction of the future of an aspect of the system, such as colocation.

In fact, instead of using the available context information as it is, CAR is optimised by using *predicted* future values of the context attributes for making routing decisions, so to have a more accurate estimation of the trend of the time series associated to each context dimension.

The process of prediction and evaluation of the context information can be summarised as follows.

- Each host calculates its delivery probabilities for a given set of hosts[2]. This process is based on the *prediction* of the future of the attributes describing the context (see Section 2.5) and on the *composition* of these estimated values using multi-attribute utility theory [Keeney and Raiffa, 1976] (see Section 2.2.1). The calculated delivery probabilities are periodically sent to the other hosts in the connected cloud as part of the update of routing information.
- Each host maintains a logical forwarding table of tuples describing the next logical

[2]We will discuss the management of this set of probabilities in Section 2.3

hop, and its associated delivery probability, for all known destinations.

- Each host uses local prediction of delivery probabilities between updates of information. The prediction process is used during temporary disconnections and is carried out until a certain accuracy can be guaranteed.

The framework for the evaluation of the delivery probability that we designed for CAR is very generic and can be extended to any number of context attributes[3].

In the remainder of this section, we will analyse more closely how delivery probability information is predicted, spread in the system, maintained, and evaluated.

2.2.1 Local evaluation of context information

Each host calculates its delivery probability utility locally, given observations related to the various context attributes. Therefore, the key problem is to *measure* and *combine* the attributes. There are several techniques for assigning an overall utility given the multiple dimensions of the context. The first possible approach is based on the use of static priorities. For example, application developers can define an order between the various attributes. In this case, the choice of the best host is based on a decision tree. These techniques are largely used in economics and form the basis of decision support systems [Russell and Norvig, 2002]. If two hosts have the same priority, we apply the results of multi-attribute decision theory, as described below.

A possible alternative to this method is to use goal programming, exploiting the so-called *preemptive methodology*. With respect to a single attribute, our goal is to maximise its value. The optimisation process is based on the evaluation of one goal at a time such that the optimum value of a higher priority goal is never degraded by a lower priority goal [Taha, 1996]. However, in general, the definition of static priorities is inflexible. For more realistic situations, a simultaneous attempt of maximisation of a range of different attributes is needed, as opposed to using a predefined hierarchy of priorities.

The priority based technique just mentioned seems too simplistic because, in general, our decision problem involves multiple conflicting objectives [Keeney and Raiffa, 1976]. For example, considering both the battery energy level and the rate of change of connectivity, it may happen that the host characterised by the highest mobility has scarce residual battery energy and vice versa. In general, maximisation across all parameters will not be possible and, instead, we must trade off the achievement of one objective (i.e., the maximisation of a single attribute) against others.

[3]However, some conditions about the mutual independence of the attributes must be verified as discussed in Section 2.2.1.

The context information related to a certain host can be defined using a set of attributes $(X_1, X_2, ..., X_n)$. Those attributes denoted with a capital letter (e.g., X_1) refer to the set of all possible values for the attribute, whereas those denoted with a lower case letter (e.g., x_1) refer to a particular value within this set. Examples of a generic attributes X_i can be the mobility of the hosts or its battery level; for instance, the value x_i of the attribute *battery level* may be 0.99 (i.e., battery almost full).

In the remainder of this section we will use the classical notation of utility theory. Our goal is to allow each host locally to associate a utility function $U(x_1, x_2, ..., x_n)$, representing the delivery probability, with every other host.

- $(x'_1, ..., x'_n) \succ (x_1, ..., x_n)$ means that $(x'_1, ..., x'_n)$ is preferred to $(x_1, ..., x_n)$.
- $(x'_1, ..., x'_n) \sim (x_1, ..., x_n)$ means that $(x'_1, ..., x'_n)$ and $(x_1, ..., x_n)$ can be chosen indifferently.
- $(x'_1, ..., x'_n) \succeq (x_1, ..., x_n)$ means that $(x'_1, ..., x'_n)$ is preferred to $(x_1, ..., x_n)$ or that they can be chosen indifferently.

Using this notation, we can associate a scalar-valued utility function $U(x_1, x_2, ...x_n)$ with each set of attributes, as follows:

$$(x'_1, ..., x'_n) \succ (x_1, ..., x_n) \Leftrightarrow U(x'_1, ..., x'_n) > U(x_1, ..., x_n)$$

$$(x'_1, ..., x'_n) \sim (x_1, ..., x_n) \Leftrightarrow U(x'_1, ..., x'_n) = U(x_1, ..., x_n)$$

$$(x'_1, ..., x'_n) \succeq (x_1, ..., x_n) \Leftrightarrow U(x'_1, ..., x'_n) \geq U(x_1, ..., x_n)$$

We use the following definitions:

Definition 1 *Given a set of attributes $X_1, X_2, ..., X_n$ partitioned into two complementary sets $\boldsymbol{Y} = (X_1, X_2, ..., X_s)$ and $\boldsymbol{Z} = (X_{s+1}, X_{s+2}, ..., X_n)$, we say that $\boldsymbol{y}$' is conditionally preferred or indifferent to $\boldsymbol{y}$ given $\boldsymbol{z}$ if and only if*

$$(\boldsymbol{y}', \boldsymbol{z}) \succeq (\boldsymbol{y}, \boldsymbol{z})$$

Definition 2 *The set of attributes $\boldsymbol{Y}$ is preferentially independent of the complementary set $\boldsymbol{Z}$ if and only if for some $\boldsymbol{z}$'*

$$[(\boldsymbol{y}', \boldsymbol{z}') \succeq (\boldsymbol{y}, \boldsymbol{z}')] \Rightarrow [(\boldsymbol{y}', \boldsymbol{z}) \succeq (\boldsymbol{y}, \boldsymbol{z})], \forall \boldsymbol{z}, \boldsymbol{y}, \boldsymbol{y}'$$

To understand this definition, consider the case of three attributes X_1, X_2, X_3: two attributes X_1 and X_2 are preferentially independent of a third attribute X_3 if the preference

between (x_1, x_2, x_3) and (x'_1, x'_2, x_3) does not depend on the particular value x_3 for the attribute X_3.

Definition 3 *The attributes $X_1, X_2, ..., X_n$ are mutually preferentially independent if every subset $\boldsymbol{Y}$ of these attributes is preferentially independent of its complementary set of attributes.*

Given these definitions, an interesting result of the multi-attribute decision theory is the following theorem demonstrated by Debreu in 1959 [Debreu, 1959].

Theorem 1 *Given attributes $X_1, X_2, ...X_n$, an additive function of the following form exists if and only if the attributes are mutually preferentially independent*

$$U(x_1, x_2, ..., x_n) = \sum_{i=1}^{n} U_i(x_i)$$

where U_i is a utility function over X_i.

Thus, in the case of mutually preferentially independent attributes, that is to say those characterised by the same degree of significance, the sum of the attributes is adequate as a means of combining those attributes. However, the case of attributes that have different relative importance is more interesting. In this case, we use the theory of goal programming, a branch of mathematics that has been studied since 1960 in the operational research community. More specifically, we use the so-called *Weights method* [Keeney and Raiffa, 1976].

Our aim is to maximise each attribute, in other words, to choose the host that presents the best trade-off between the attributes representing the relevant aspects of the system for the message delivery. Analytically, considering n attributes, the problem can be reformulated in terms of n goals where each goal is given as

$$Maximise\{U_i(x_i)\}, i = 1, 2, ..., n$$

The combined goal function used in the Weights method can be defined as

$$Maximise\{f(U_i(x_i)) = \sum_{i=1}^{n} w_i U_i(x_i)\}$$

where $w_1, w_2, ...w_n$ are *significance weights* reflecting the relative importance of each goal.

We exploit these results for the composition of the utilities in CAR related to the different context dimensions (given their mutual independence).

It is worth noting that, in our case, the solution is very simple, since it consists in the evaluation of the function $f(U_1, ..., U_n)$ using the values predicted for each host and in the selection of the host i with the maximum such value.

The detailed description of the calculation of these utilities is presented in the next subsection.

2.2.2 Definition of the Attributes of the Utility Functions

Knowledge about the current values of these context attributes is helpful, but only to a limited extent. What really matters are the values the attributes are likely to assume in the future.

We compute these *predicted* values using techniques based on Kalman filters [Kalman, 1960]. These techniques do not require the storage of the entire past history of the system and are computationally lightweight, making them suitable for a resource-scarce mobile setting. The details of the mathematical model for time series forecasting used in CAR is presented in Section 2.5. In this section, instead, we focus on the use of the predicted values of the attributes for the calculation of the utility of each host as message carrier.

In the implementation of CAR, we focus on two attributes, the change degree of connectivity and the future host colocation, because these are the attributes most relevant to the ad hoc scenario taken into consideration. However, the framework is general and open to the inclusion of any other context attribute, given the underlying assumption of their mutual independence. For example, in the adaptation of CAR for sensors (SCAR) [Mascolo and Musolesi, 2006], we also consider an attribute describing the battery level, a fundamental aspect for that specific application domain.

The change degree of connectivity of a host h is[4]:

$$U_{cdc_h}(t) = \frac{|n(t-T) \cup n(t)| - |n(t-T) \cap n(t)|}{|n(t-T) \cup n(t)|}$$

where $n(t)$ is h's neighbour set at time t. The formula yields the number of hosts that became neighbours or disappeared in the time interval $[t-T, t]$, normalised by the total number of hosts met in the same time interval. A high value means that h recently changed a large number of its neighbours.

[4]In the remainder of the thesis, we simplify the notation used for indicating the utility function that we adopted in the previous subsection. We omit the attribute from the utility and, instead, we indicate explicitly the time dependence. For example, $U_{cdc_h}(x_{cdc_h}(t)) = U_{cdc_h}(t)$. This choice is motivated by the fact that the time intervals are essential aspects of the model, especially with respect of the h-steps ahead prediction that we present in Section 2.5.5.

The colocation of h with a host i is calculated as follows:

$$U_{col_{h,i}}(t) = \begin{cases} 1 & \text{if the host } h \text{ is colocated with host } i; \\ 0 & \text{otherwise} \end{cases}$$

A value of 1 means that h has been colocated with i at time t.

These values are fed into Kalman filter predictors, which yield the predictions $\widehat{U}_{cdc_h}$ and $\widehat{U}_{col_{h,i}}$ of these utilities at time $t+T$. These are then composed into a single utility value using results from multi-criteria decision theory described above, as follows:

$$U_{h,i} = w_{cdc_h}\widehat{U}_{cdc_h} + w_{col_{h,i}}\widehat{U}_{col_{h,i}}$$

which represents how good of a node h is for delivering messages to i.

We observe that the effectiveness of the choice of using predicted values and not current values of the attributes is evident in the case of colocation. For example, let us consider two hosts that have disconnected for just ten seconds after being connected for a long period of time. If we only considered the current status, the value of the utility function related to colocation would be 0. Instead, since the hosts have been colocated for long time in the past, according to our assumptions, they will be likely colocated again the future. The value 0 does not provide a correct measure of the probability of future colocation of the two hosts. On the contrary, the output of the Kalman filter will be close to 1.

The weights w denote the relative importance of each attribute. Their value depends on the application scenario, and in Chapter 4 we show their impact on performance. The values of these weights are the same for every host; in other words, the utility composition function is the same for all the nodes of the system.

2.2.3 Automatic Adaptation of the Utility Functions

As it stands, the utility function weights are fixed in advance, reflecting the relative importance of the different context attributes. However, such a formulation is still too static, since it fails to take into account the values of the attributes. Thus, for example, a small drop in battery voltage may be indicative of the imminent exhaustion of the battery; consequently, it would be useful to reduce the weight of this attribute non-linearly to reflect this.

In general, we wish to adapt the weights of each parameter *dynamically* and in ways that are dependent on the values of those parameters. In other words, we need a runtime self-adaptation of the weightings used for this evaluation process that could be categorised as

a typical autonomic mechanism [Bantz et al., 2003]. A simple solution to this problem is the introduction of adaptive weights a_i into the previous formula, in order to modify the utility function according to the variation of the context.

$$Maximise\{f(U(x_i)) = \sum_{i=1}^{n} a_i(x_i) w_i U_i(x_i)\}$$

$a_i(x_i)$ is a parameter that may itself be composite. For our purposes, we define it to have three important aspects that help to determine its value, though the model could easily be expanded to incorporate other aspects deemed to be of importance:

- Criticality of certain ranges of values, $a_{range_i}(x_i)$
- Predictability of the context information, $a_{predictability_i}(x_i)$
- Availability of the context information, $a_{availability_i}(x_i)$

We now compose the a_i weights as factors in the following formula:

$$a_i(x_i) = a_{range_i}(x_i) \cdot a_{predictability_i}(x_i) \cdot a_{availability_i}(x_i)$$

We now describe each of these aspects in detail.

Adaptive Weights Related to the Ranges of Values Assumed by the Attributes

We can model the adaptive weights $a_{range}(x_i)$ as a function in the domain $[0, 1]$. For example, with respect to the battery energy level (modelled using the percentage of residual battery energy), we would use a monotonically decreasing (though not necessarily linear) function to assign a decreasing adaptive weight that is, in turn, used to ensure that the corresponding utility function decreases as the residual energy tends towards zero.

Adaptive Weights Related to the Predictability of the Context Information

It may happen that the forecasting model is not able to provide accurate predictions for a certain time series related to a given attribute. There are many different methods to evaluate the predictability of a time series [Chatfield, 2004].

$$a_{predictability_i} = \begin{cases} 1 & \text{if the context information is currently predictable} \\ 0 & \text{if the context information is not currently predictable} \end{cases}$$

We prefer to adopt an approach based on two discrete values (0 and 1) rather than one based on continuous values (i.e. an interval between 0 and 1), since the latter would be only based on a pure heuristic choice and not on any sound mathematical basis. In other words, it is very difficult to map different scales of predictability into the values of $a_{predictability_i}$.

The problem of the analysis of the predictability of the time series is discussed in Section 2.5.

Adaptive Weights Related to the Availability of Context Information

It is unreasonable to assume that all context attributes have the same degree of availability. Thus, we expect to have a time-varying set of attributes available whose values are known or predictable. Attributes may drop out of this set if meaningful values can no longer be predicted for them, since the information on which the prediction would have been based is too old. The simplest approach to this problem is to ensure that missing context information carries an adaptive weight a_i equal to 0:

$$a_{availability_i} = \begin{cases} 1 & \text{if the context information is currently available} \\ 0 & \text{if the context information is not currently available} \end{cases}$$

Formally, to date, we have implicitly assumed that a static set of attributes is defined. However, it is worth noting that, using this approach, we can dynamically incorporate new attribute values, simply by assuming that they were always there, but had zero weight for $a_{availability_i}$. For example, if an operating system is not able to provide information about the current battery level of a device, the value of $a_{availability_i}$ is set to 0. This may also be due to an erroneous reading of a parameter (for example, in the case of the change degree of connectivity, because the wireless interface has been switched off temporarily).

2.3 Routing Tables

We have seen how each host calculates its delivery probability by assembling predictions related to different context attributes. We now describe how this information is circulated in the network.

2.3.1 Format of the Routing Table Entries

The delivery probability information is piggybacked on the synchronous routing table information.

Each host maintains a *routing and context information table* used for asynchronous and synchronous (DSDV) routing. Each entry of this table has the following structure:

`(targetHostId, nextHopId, distance, bestHopHostId, deliveryProb)`

The first field is the recipient of the message, the second and the third are the typical values calculated in accordance with the DSDV specification, whereas the fourth is the identifier of the host with the best delivery probability, the value of which is stored in the last field.

It is worth noting that these routing tables are used both for synchronous and asynchronous delivery. In fact, they store information used for routing messages inside a cloud (i.e., the fields `nextHopId` and `distance`) and for the selection of the best carrier (i.e., the fields `bestHopHostId` and `deliveryProb`). A distance equal to 16 is considered infinite and the host is treated as unreachable using DSDV. We choose 16 since this is the classic Routing Information Protocol (RIP) infinite [Malkin, 1999]. However, this is a parameter that can be tuned according to user requirements. In a scenario characterised by high average host speed a lower value may be used, since the probability that the route will be broken or stale is potentially high.

The value of the field `deliveryProb` is updated using the last received value. However, the values received by the neighbours are also used to update a corresponding Kalman Filter predictor, one for each entry of the table. The state of the filter is updated using the last received utility from the host `bestHopHostId` (this utility is calculated by `bestHopHostId` as described in the previous section).

The filter is used if one or more updates are not received, due for example to a temporary disconnection, to transmission errors (for example interference) or simply because the host has moved away. If an update is not received in a given refresh interval of the filter (that is equal to the routing table transmission interval), the previous output of the filter is used as input (i.e., the filter is, in a sense, short-circuited). This is predicated on a particular mathematical property of the Kalman filter, i.e., that it is possible to calculate predicted values also in case of missing observations as explained in Section 2.5.5.

The entries are removed after a certain number of updates are not received, since the accuracy of the prediction is clearly decreasing; a discussion about the accuracy of the estimation of the h-step prediction can be found in [Brockwell and Davis, 1996]. This technique alleviates the CAR scalability issue in terms of routing table size.

2.3.2 Local Utilities and Update of Routing Tables

Each node keeps local utilities related to the colocation with other nodes. The routing tables are exchanged periodically with a given transmission interval. When a host receives a routing table, it checks its entries against the ones stored in its routing table.

The update of the information related to the synchronous protocol is the standard one of every table-driven protocol: an entry in the routing table of the host is replaced if one related to the same `targetHostId` and a lower or equal distance is received. It is important to note that we also replace the entry if the distance is the same in order to have fresh information about the route.

Instead, as far as the asynchronous delivery protocol is concerned, an entry is replaced only if one related to the same `targetHostId` and higher or equal delivery probability is received. As said before, the entry is removed after a number of missing updates: this is also avoids the problem that entries with high probabilities persist in the routing table even if they are stale.

When the routing table is full, the entries are replaced starting from the one corresponding to the nodes that are not in reach anymore (i.e. that have a value of the `distance` field equal to 16). Among these entries, the one with the lowest delivery probability is selected.

We now give an example of routing information distribution and management.

2.3.3 Example of Routing Table Management

Let us consider the network scenario depicted in Figure 2.3.

We assume that H_1, H_2, H_4 were in reach of H_5 but not of H_3 in the past. H_3 was in reach of H_5 too.

The routing table stored by H_1 is the following:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	0	H_1	1
H_2	H_2	1	H_1	0.7
H_4	H_2	2	H_2	0.75
H_5	–	16	H_4	0.65

The routing table stored by H_2 is:

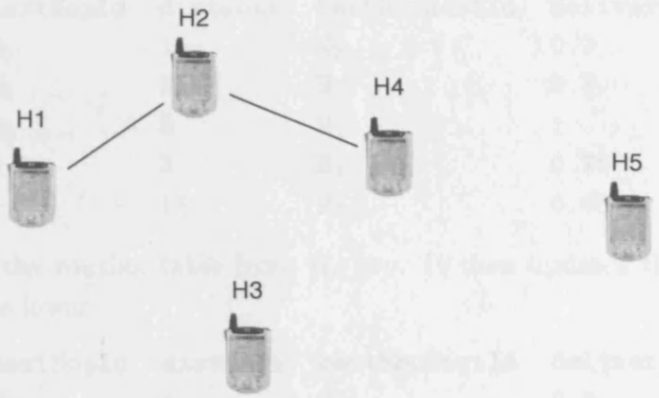

Figure 2.3: Example of network topology used for explaining routing table management

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	1	H_2	0.9
H_2	H_2	0	H_2	1
H_4	H_4	1	H_2	0.75
H_5	–	16	H_4	0.65

The routing table stored by H_4 is:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_2	2	H_2	0.9
H_2	H_2	1	H_1	0.7
H_4	H_4	0	H_2	1
H_5	–	16	H_4	0.65

Let us assume that H_3 is approaching the cloud composed by H_1, H_2 and H_4.

Let us also assume that the routing table of H_3 before joining the cloud is the following:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_3	H_3	0	H_3	1
H_5	–	16	H_3	0.3

Then, let us suppose that H_3 joins the cloud. The new topology of the network is shown in Figure 2.4. Please note that it has an entry about H_5 (that is currently not reachable) expressing the fact that it is the best carrier.

Let us assume that H_3 receives the routing table firstly from H_1. H_3 will insert the information about H_1, H_2 and H_4.

With respect to the asynchronous routing, it receives an update about the best carrier for H_5 and then it replaces the entry with H_4 that has a probability higher than its own. The new routing table is:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	1	H_2	0.9
H_2	H_1	2	H_1	0.7
H_3	H_3	0	H_3	1
H_4	H_1	3	H_2	0.75
H_5	–	16	H_4	0.65

Then, H_3 receives the routing table from H_4 too. It then updates the entry about H_4, since the distance is lower.

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	1	H_2	0.9
H_2	H_4	2	H_1	0.7
H_3	H_3	0	H_3	1
H_4	H_4	1	H_2	0.75
H_5	–	16	H_4	0.65

H_1, H_2 and H_4 receives the routing table from H_3 inserting the information about this node for synchronous routing. The value of `bestHopHostId` is not updated since it is the same.

2.3.4 Routing Table Transmission Interval

The routing table transmission interval is another fundamental parameter of CAR. In fact, routing tables are not only used to exchange routing information, but also for discovery. Routing tables are employed as a sort of beaconing mechanism at application level to keep information about the presence of neighbours[5]. In fact, a host is considered colocated (i.e., the input of the Kalman filter is set to 1), if a routing table related to that host has been received in the last routing table transmission interval. Therefore, we assume that the frequency of the transmission of routing tables is relatively high in order to provide correct information to the colocation predictor.

The update interval of the Kalman filter is set to the routing table transmission interval in our work. The update interval of the Kalman filter (i.e., its sample interval) is another fundamental parameter of the protocol: this value should be carefully selected in order to detect changes in the observed context attribute. For example, in the case of host colocation, a low sampling interval in a very dynamic mobile scenario may lead to the fact that hosts passing by will not be detected. For instance, if the relative speed of the two hosts is 20 m/s and the transmission range is 200 m/s, an update interval greater than 20 s may lead to the fact that some hosts will not be discovered.

[5]As we will describe in Chapter 5, we integrated CAR in the Haggle platform [Su et al., 2006]. Haggle provides a discovery mechanism so, in that case, routing tables are not used to update colocation information.

Since CAR relies on DSDV, it shows the same limitations and potential issues in terms of routing table convergence. The retransmission interval should be adequately small if the speed of the hosts is high (i.e., the variations of the topology are frequent). Results about the impact of the duration of the transmission interval are reported in the evaluation in Chapter 4.

2.4 Message Delivery

2.4.1 Synchronous Delivery

When a message has to be sent, if the recipient is reachable synchronously (i.e. an entry with the field `TargetHostId` exists in the routing table and the associated distance is less than 16), the message is forwarded to the next hop indicated by `nextHopId`. This forwarding mechanism is the typical one of distance vector protocols.

It may happen that the a path to a certain host is broken, but, at the same time, the routing table has not yet been updated with the information related to this change, given the propagation delay of routing tables. In this case, the message is forwarded until it reaches the host that has been already notified about the disconnections. This host will then check if the message can be sent using the asynchronous delivery mechanism (i.e., an entry for the selection of the best carrier exists in its routing table). If not, the host stores the message in its buffer and tries to resend it periodically. We discuss this mechanism in Section 2.4.3.

2.4.2 Asynchronous Delivery

If a connected path to the recipient does not exist (i.e., the value of `distance` is equal or greater than 16), the message is forwarded to the host with the highest value of delivery probability (expressed by `deliveryProb`). In order to reach the carrier, DSDV is used. In other words, the entry having the value of the key `targetHostId` equal to `bestHopHostId` is used to forward the message.

As the network is dynamic, it may happen that the carrier is unreachable, since, in the meanwhile, it has left the connected cloud. In this case, if the information about the disconnection has reached the sender, the entry related to the best carrier is removed (set to an invalid state designated by 0).

If this information has not been propagated yet to the sender, the intermediate host aware of the topology change will try to resend the message (see Section 2.4.3).

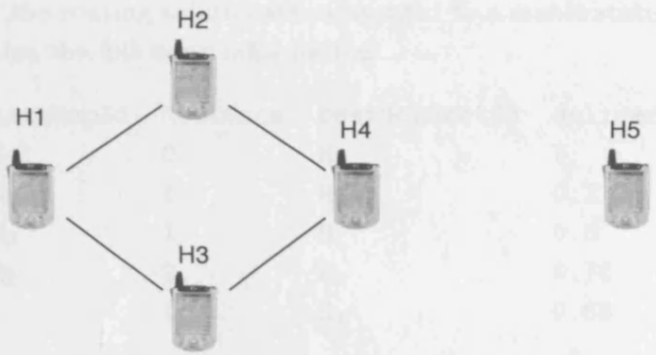

Figure 2.4: Example of network topology used for explaining routing table management (H_3 joins the cloud composed of H_1, H_2 and H_4).

2.4.3 Retransmissions

Periodically, for each message in its buffer, a host checks its routing table. The message is then forwarded synchronously to the recipient or to a carrier if a corresponding entry is present in the routing table. If no entry is present, the message stays in the buffer.

The number of the retransmissions is another key configuration value that we have measured and tested during the performance evaluation of the protocol. The results are reported in Section 4.2.6.

Each node also maintains a list of its utilities for a certain set of hosts. In particular, each node keeps a list of the local utilities related to the colocation with other hosts[6] and one related to its change degree of connectivity. Periodically, these utilities are composed and the resulting ones are checked against the utilities stored in the local routing table. If the utility of the host is higher of the one currently maintained in the table, the latter is replaced. The value of the utilities is updated before comparing it with the entries of the local routing table.

2.4.4 Example of Message Delivery

In order to provide a detailed description of the mechanisms used to manage and update the routing tables, we now discuss a toy example considering the scenario depicted in Figure 2.4 composed of 5 hosts. We assume that H_5 was previously colocated with the other hosts. This fact is reflected in the contents of its routing table.

[6]The size of this list is another parameter that can be tuned during the deployment. It is important to note that the memory occupancy of a single Kalman filter predictor is very limited since it consists in storing the value of the last observation and the current state.

Let us assume that the routing tables have converged to a stable state. The routing table stored by H_1 contains the following information:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	0	H_1	1
H_2	H_2	1	H_1	0.7
H_3	H_3	1	H_1	0.8
H_4	H_2	2	H_2	0.75
H_5	–	16	H_4	0.65

Let us now consider the other routing tables. The routing table stored by H_2 is:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	1	H_2	0.9
H_2	H_2	0	H_2	1
H_3	H_1	2	H_1	0.8
H_4	H_4	1	H_2	0.75
H_5	–	16	H_4	0.65

The routing table stored by $H3$ is:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_1	1	H_2	0.9
H_2	H_4	2	H_1	0.7
H_3	H_3	0	H_3	1
H_4	H_4	1	H_2	0.75
H_5	–	16	H_4	0.65

The routing table stored by H_4 is:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	H_2	2	H_2	0.9
H_2	H_2	1	H_1	0.7
H_3	H_3	1	H_1	0.8
H_4	H_4	0	H_2	1
H_5	–	16	H_4	0.65

Finally, the routing table stored by the isolated node $H5$ is the following:

targetHostId	nextHopId	distance	bestHopHostId	deliveryProb
H_1	–	16	H_2	0.6
H_2	–	16	H_1	0.5
H_3	–	16	H_5	0.4
H_4	–	16	H_5	0.68
H_5	H_5	0	H_5	1

First of all, we observe that the best carrier for H_5 is the same for all the hosts in the connected cloud (i.e. H_4). Secondly, we note that the value of `nextHopHostId` related to the hosts that are not currently in reach is set to an undefined value (i.e., the value – in the tables above). It is also interesting to note that, in this example, even if two hosts are in reach, the `deliveryProb` field is different from 1, since it is used to store the current utility that is calculated using the predicted values of the attributes (i.e., the outputs of the Kalman filter predictors). In fact, the Kalman filter predictors have 1 as asymptotic value of the outputs.

Let us consider a case of synchronous message delivery. Let us suppose that H_1 has to send a message to H_4. The forwarding process is the typical one of table-driven protocol. H_1 will check its routing table and then forward the message to H_2. This host will then forward the message immediately to H_4 without storing it temporarily in its buffer.

Let us consider now a more interesting case. Let us suppose that H_1 wants to send a message to H_5. H_5 is not reachable synchronously, since the field distance is set to 16. So H_1 will forward the message to the best carrier (i.e., H_4) synchronously through H_2. More precisely, H_1 will retrieve the best carrier identifier and it will check the next hop to reach it using DSDV. When H_4 will receive the message, it will store the message in its buffer until it will eventually be in reach of H_5. More precisely, the final delivery will happen when H_4 will receive the routing table of H_5 (i.e., when H_4 will realise that H_5 is reachable synchronously).

2.5 Prediction of the Context Information Attributes using Kalman Filter Techniques

In this section, we present the details of the prediction model used to estimate the delivery probability of the potential carriers discussed in Section 2.2. We used Kalman filter forecasting techniques [Kalman, 1960] for the prediction of the future values of the context attributes and of the delivery probabilities in the local routing tables, if updates are not received.

2.5.1 Overview

Kalman filter prediction techniques were originally developed in automatic control systems theory. These are essentially a method of discrete signal processing that provides optimal estimates of the current state of a dynamic system described by a *state vector*. The state is updated using periodic observations of the system, if available, using a set of *prediction recursive equations*.

Kalman filter theory is used in CAR both to achieve a more realistic prediction of the evolution of the context of a host and to optimise the bandwidth usage. As discussed above, the exchange of context information that allows the calculation of delivery probabilities is a potentially expensive process, and unnecessarily so where such information is relatively predictable. If it is possible to predict future values of the attributes describing the context, we update the delivery probabilities stored in the routing tables, even if fresh information is unavailable. Fortunately, this prediction problem can be expressed in the form of a state space model. Starting from a time series of observed values that represent context information, we derive a prediction model based on an inner state that is represented by a set of vectors, and to add to this both trend and seasonal components [Brockwell and Davis, 1996]. One of the main advantages of the Kalman filter is that it does not require the storage of the entire past history of the system, making it suitable for a mobile setting in which memory resources may potentially be very limited.

In the following subsections, we give a general introduction to state space models and then we present how we have applied these concepts to the analysis and the prediction of context information, discussing three cases according to the different behaviour of the time series.

Finally, we also prove the equivalence of our model with EWMA and ARMA models [Brockwell and Davis, 1996]. In fact, the state space formulation of the prediction problem can be also recast in these alternative theoretical frameworks posing simple conditions to the values assumed to factors of the model. We also discuss the relationship of our model with Bayesian forecasting [West and Harrison, 1997]Kto provide a theoretical comparison with other alternative techniques.

2.5.2 State Space Models

A state space model for a time series $\mathbf{Y}_t$ consists of two equations. The first one called the *observation equation* is the following

$$\mathbf{Y}_t = G_t\mathbf{X}_t + \mathbf{W}_t \quad t = 1, 2, ...$$

with $\mathbf{W}_t$ defined as [7]

$$\mathbf{W}_t = WN(0, R_t)$$

[7]WN stands for White Noise, a term that derives from telecommunication engineering. A white noise is a sequence of uncorrelated random variables X_t, each with the same mean and variance σ^2. Therefore, white noise is also an example of stationary time series. More specifically, the notation $WN(0, \{R_t\})$ indicates white noise with zero mean and variance R_t.

This equation defines the w-dimensional observation $\{\mathbf{Y}_t\}$ as a linear function of a v-dimensional state variables $\{\mathbf{X}_t\}$ and a noise term. The second one is the *state equation* defined as follows

$$\mathbf{X}_{t+1} = F_t\mathbf{X}_t + \mathbf{V}_t \quad t = 1, 2, ...$$

with $\mathbf{V}_t$ defined as

$$\mathbf{V}_t = WN(0, Q_t)$$

This equation determines the state $\mathbf{X}_{t+1}$ at time $t+1$ in terms of the previous state $\mathbf{X}_t$ and a noise term. Let w as the dimension of $\mathbf{Y}_t$ and v as the dimension of $\mathbf{X}_t$, $\{G_t\}$ is a sequence of $w \times v$ matrices and $\{F_t\}$ is a sequence of $v \times v$ matrices. We assume that $\{\mathbf{V}_t\}$ is uncorrelated with $\{\mathbf{W}_t\}$, even if a more general form of the state space model allows for correlation between these two variables. Analytically, we can rewrite this condition as follows

$$E(\mathbf{W}_s\mathbf{V}_t^T) = 0 \quad \forall s, t$$

We also assume that the initial state $\mathbf{X}_1$ is uncorrelated with all of the noise terms $\{\mathbf{V}_t\}$ and $\{\mathbf{W}_t\}$.

2.5.3 Kalman Filter Prediction

With the notation of $P_t(\mathbf{X})$ we refer to the best linear predictor (in the sense of minimum mean-square error) of $\mathbf{X}$ in terms of $\mathbf{Y}$ at the time t. $P_t(\mathbf{X})$ is defined as follows

$$P_t(\mathbf{X}) \equiv \begin{bmatrix} P_t(X_1) & ... & P_t(X_v) \end{bmatrix}^T$$

where

$$P_t(X_i) \equiv P(X_i|\mathbf{Y}_0, \mathbf{Y}_1, ..., \mathbf{Y}_t)$$

$P(X_i|\mathbf{Y}_0, \mathbf{Y}_1, ..., \mathbf{Y}_t)$ indicates the best predictor of X_i given $\mathbf{Y}_0, ..., \mathbf{Y}_t$. We can also observe that $P_t(\mathbf{X})$ has the following form

$$P_t(\mathbf{X}) = A_0\mathbf{Y}_0 + ... + A_t\mathbf{Y}_t$$

since it is a linear function of $\mathbf{Y}_0, ..., \mathbf{Y}_t$. It is possible to prove [Brockwell and Davis, 1996] for the state space model discussed in the previous section that the one-step predictor

$$\widehat{\mathbf{X}}_t \equiv P_{t-1}(\mathbf{X}_t)$$

and their error covariance matrices

$$\Omega_t = E[(\mathbf{X}_t - \widehat{\mathbf{X}}_t)(\mathbf{X}_t - \widehat{\mathbf{X}}_t)^T]$$

are determined by these initial conditions

$$\widehat{\mathbf{X}}_1 = P(\mathbf{X}_1|\mathbf{Y}_0)$$

$$\Omega_1 = E[(\mathbf{X}_1 - \widehat{\mathbf{X}}_1)(\mathbf{X}_1 - \widehat{\mathbf{X}}_1)^T]$$

and these recursive equations

$$\widehat{\mathbf{X}}_{t+1} = F_t\widehat{\mathbf{X}}_t + \Theta_t\Delta_t^{-1}(\mathbf{Y}_t - G_t\widehat{\mathbf{X}}_t)$$

$$\Omega_{t+1} = F_t\Omega_t F_t^T + Q_t - \Theta_t\Delta_t^{-1}\Theta_t^T$$

where

$$\Delta_t = G_t\Omega_t G_t^T + R_t$$

$$\Theta_t = F_t\Omega_t G_t^T$$

2.5.4 Estimation Model

As estimation model, we use a basic state space model composed of the following two scalar equations

$$Y_t = X_t + W_t \quad t = 1, 2, ...$$

with

$$W_t = WN(0, Q_t)$$

and

$$X_{t+1} = X_t + V_t \quad t = 1, 2, ...$$

with

$$V_t = WN(0, R_t)$$

With respect to the Kalman filter prediction, we can consider a mono-dimensional system with

$$G_t = [1]$$

$$F_t = [1]$$

Therefore, we can derive the recursive equations of the Kalman filter for the prediction of the values of this series. Given the previous observed value Y_t and the predicted value at time t, $\widehat{X}_t$, the recursive equation for the determination of the predicted value at time $t + 1$ is

$$\widehat{X}_{t+1} = \widehat{X}_t + \frac{\Omega_t}{\Omega_t + R_t}(Y_t - \widehat{X}_t)$$

with

$$\Omega_{t+1} = \Omega_t + Q_t - \frac{\Theta_t^2}{\Omega_t + R_t}$$

Since in this case

$$\Omega_t = \Theta_t$$

we can also write

$$\Omega_{t+1} = \Omega_t + Q_t - \frac{\Omega_t^2}{\Omega_t + R_t}$$

In Appendix A we present more complex estimation models that take into account local trends and seasonal behaviour of the time series. The tuning of these models requires a knowledge of the application scenarios, for example to set the magnitude of the seasonal period.

2.5.5 Subsequent Predictions

It is also possible to make subsequent predictions, in the case that observation values are not available, by using the model previously discussed [Brockwell and Davis, 1996, Durbin and Koopman, 2001]. In other words, we can predict h consecutive estimations $\mathbf{Y}_{t+1}, ..., \mathbf{Y}_{t+h}$, using the predictor calculated at time t without exchanging new context and routing information[8]. Now we derive the expression of the h-step prediction of $\mathbf{Y}_t$ using recursive equations.

Observing that

$$\widehat{\mathbf{X}}_t = P_t(\mathbf{X}_{t+1}) = F_t P_{t-1}(\mathbf{X}_t) + \Theta_t \Delta_t^{-1}(\mathbf{Y}_t - P_{t-1}(\mathbf{X}_t))$$

we can write $\widehat{\mathbf{X}}_{t+h}$ as

[8] We can evaluate the maximum number n of predictions that it is possible to make with a given required accuracy. However, we do not present these results in detail, since they are outside the scope of this work.

$$\begin{aligned}
&\widehat{\mathbf{X}}_{t+h} = P_t(\mathbf{X}_{t+h}) = \\
&= F_{t+h-1} P_t(\mathbf{X}_{t+h-1}) = \\
&\vdots \\
&= (F_{t+h-1} F_{t+h-2} ... F_{t+1}) P_t(\mathbf{X}_{t+1}) \qquad t = 2, 3, ...
\end{aligned}$$

and $\widehat{\mathbf{Y}}_{t+h}$ as

$$\widehat{\mathbf{Y}}_{t+h} = P_t(\mathbf{Y}_{t+h}) = G_{t+h} P_t(\mathbf{X}_{t+h})$$

2.5.6 Relations with Alternative Prediction Models

In this section we discuss how the Kalman filter model that we use in CAR can be recast and studied using alternative theoretical frameworks, namely EWMA [Le Boudec, 2006], ARMA [Brockwell and Davis, 1996] and Bayesian forecasting models [West and Harrison, 1997].

The goal of this analysis is to prove that the model itself can be studied using alternative popular theoretical frameworks. In fact, our model can be considered as a refined version of an EWMA or a particular instance of an ARMA model. A comparison in terms of accuracy or better suitability is unnecessary given the formal results presented in this section.

Equivalence with ARMA model

We now demonstrate how our model can be considered as a particular instance of ARMA models. In general, it is possible to prove that all the ARMA models can be interpreted as state space models [Chatfield, 2004]. In particular, we demonstrate that our model is equivalent to an ARMA(1,1) model.

Let us reconsider our state space prediction model

$$Y_t = X_t + W_t \quad t = 1, 2, ...$$

with

$$W_t = WN(0, Q_t)$$

and

$$X_{t+1} = X_t + V_t \quad t = 1, 2, ...$$

with

$$V_t = WN(0, R_t)$$

Let us consider a model with the following generic form

$$X_{t+1} = \phi X_t + V_{t+1}$$

with ϕ constant. This can also be written as follows

$$V_{t+1} = X_{t+1} - \phi X_t$$

Let

$$D_{t+1} = Y_{t+1} - \phi Y_t$$

We can write

$$D_{t+1} = X_{t+1} + W_{t+1} - \phi(X_t + W_t) = V_t + W_{t+1} - \phi W_t$$

The autocorrelations of $\{D(t)\}$ are 0 at lags greater than 1. So $\{D(t)\}$ is MA(1) and $\{Y(t)\}$ is ARMA(1,1).

Our model can be considered an ARMA model with $\phi = 1$.

Equivalence with EWMA model

We now derive the equivalence of our model with an EWMA model. The derivation of the equivalence of our model and EWMA models was inspired by an observation by Le Boudec in [Le Boudec, 2006].

Given the implicit model

$$Y_t = m_t + W_t$$

the one step predictor $\widehat{m}(t) = (Y_{t+1}|Y_1, ..., Y_t)$ is

$$\widehat{m}_{t+1} = \alpha \sum_{s=0}^{\infty} (1-\alpha)^s Y_{t-s}$$

with $Y_t = Y_1$ for $t \leq 1$.

The EWMA model is characterised by the following recursive computation of $\widehat{m}(t)$

$$\widehat{m}_{t+1} = \alpha Y_{t+1} + (1-\alpha)\widehat{m}_t$$

with initial condition $\widehat{m}_1 = Y_1$.

Let us now consider again the recursive equation used to update the value of the state $\{X_t\}$

$$\widehat{X}_{t+1} = \widehat{X}_t + \frac{\Omega_t}{\Omega_t + R_t}(Y_t - \widehat{X}_t)$$

This can be rewritten as follows

$$\widehat{X}_{t+1} = (1 - \frac{\Omega_t}{\Omega_t + R_t})\widehat{X}_t + \frac{\Omega_t}{\Omega_t + R_t} Y_t$$

Let

$$\alpha = \frac{\Omega_t}{\Omega_t + R_t}$$

we can rewrite the update equation using the same form of the EWMA model

$$\widehat{X}_{t+1} = (1-\alpha)\widehat{X}_t + \alpha Y_t$$

Kalman Filter prediction models and Bayesian forecasting

Bayesian forecasting [West and Harrison, 1997] relies on the so-called *dynamic linear model* that can be considered closely related to the general class of state space models.

According to a Bayesian formulation, Kalman filters are regarded as a way of updating the prior probability distribution of $\mathbf{X}_t$ when a new observation become available to give

a revised (posterior) distribution. For example, the model used in CAR for the prediction of the host colocation can be seen a Bayesian model representing the probability of being colocated with a given host.

The specification of the model in this framework is based on the assumption that the observations $\mathbf{Y}_t$ given an unobserved state $\mathbf{X}_t$ are independent with specified conditional distribution. More details about the application of Bayesian forecasting to generalised state space models can be found in [Brockwell and Davis, 1996].

2.6 Context Predictability

Dealing with the variability and uncertainty is one of the major issues in many networked systems such as mobile ad hoc and delay tolerant networks [Fall, 2003]. The decentralisation of the control and the movement of the hosts have a great impact on systems topology and, more generally, on their conditions.

CAR heavily relies on the accuracy of the prediction model as we show in the evaluation section (see Section 4.2.7). There are situations, however, where context cannot be predicted. In these cases, using any prediction based techniques to improve performance of the system is completely ineffective.

For this reason, we designed a predictability component that is used to measure the accuracy of the prediction of context information [Musolesi and Mascolo, 2006c]. The technique that we adopted is predicated on the analysis of the time series representing the context information [Brockwell and Davis, 1996].

2.6.1 Design of the Prediction Component

The component receives in input the observed value (at time t) and the corresponding predicted value (calculated at time $t-1$) of a particular context indicator (such as the colocation value with a given host) and is able to decide if the prediction model (in the case of our prototype system, a Kalman filter based predictor [Kalman, 1960]) is able to forecast the next value of the time series with a given accuracy that can be set by the developer.

To estimate the predictability of context, we exploit a technique based on the analysis of the time series of the prediction errors [Chatfield, 2004].

Residuals Analysis

We are interested in the analysis of the so-called *residuals*, defined as the differences between the observed values at time t and the one-step-ahead forecasts for these same values at time t. More formally, given the observed value $y(t)$ and the predicted value $\widehat{y}(t)$, we define the residual at time t as follows:

$$z(t) = \widehat{y}(t) - y(t)$$

Since we are considering time series, the residuals are ordered in time and can therefore be treated as a time series too.

In particular, our analysis is based on the *correlogram* (also called sample autocorrelation function) of residuals. A correlogram is a diagram in which the sample autocorrelation coefficients r_k are plotted against the lag k for $k = 0, 1..., M$, where M is usually much less than the number of samples N taken into consideration.

The autocorrelation coefficient of $z(t)$ at lag k, r_k, is defined as follows

$$r_k = \frac{\sum_{t=1}^{N-k}(z(t) - \mu)(z(t+k) - \mu)}{\sum_{t=1}^{N}(z(t) - \mu)^2}$$

with $\mu = \sum_{t=1}^{N} \frac{y(t)}{N}$

Values of r_k close to zero indicate that the values of the time series $z(t)$ are scarcely correlated at a distance k. When the prediction model provides a good estimation of the future trend of the context information, the standardised forecast errors are serially uncorrelated. Therefore, the correlogram of forecast errors should reveal insignificant serial correlation. In other words, if we have a good prediction model, we expect the residuals to be random (i.e., the time series $\{z(t)\}$ forms a random process) and close to zero. If the residuals $\{z(t)\}$ form a purely random process, their correlogram is such that each autocorrelation coefficient is approximately normally distributed, with mean 0 and variance $\frac{1}{N}$ for sufficiently large values of N[9].

The analysis of the predictability of the time series can be performed periodically. In other words, it is not necessary to store the last N values of all the time series considered in the system. The analysis of the predictability of the time series can be performed sequentially, considering all the time series one after the other, avoiding to store all the values of all

[9]The accuracy of the calculation increases as N increases. The choice of N is therefore based on the evaluation of the trade-off between the desired accuracy and the resource consumption in terms of memory. In our simulations, we set N equal to 60.

the time series, but only the ones related to a single series[10]. Storing all the information would also contradict the design philosophy behind the choice of adopting Kalman filters.

Let us then suppose that z_1, z_2, ..., z_N are observations on independent and identically distributed random variables with arbitrary mean. It can be shown that r_k is asymptotically normally distributed and that

$$E(r_k) \approx -\frac{1}{N}$$

$$Var(r_k) \approx \frac{1}{N}$$

We can use this result to *check the randomness* of the time series of the residuals. Let us suppose that we are interested in 95% confidence interval in our calculation. In this case the range of the confidence interval is

$$[-\frac{1}{N} - \frac{2}{\sqrt{N}}, -\frac{1}{N} + \frac{2}{\sqrt{N}}]$$

This interval can be further approximated as

$$[-\frac{2}{\sqrt{N}}, \frac{2}{\sqrt{N}}]$$

Observed values of r_k which fall outside these limits are significantly different from 0 at the 5% level. Therefore, considering a degree of confidence of 95%, no more than 5% of the values should be outside the range $[-\frac{2}{\sqrt{N}}, \frac{2}{\sqrt{N}}]$. For example, if the first 20 values of r_k (i.e., r_1, r_2, ..., r_k) are taken into consideration, one value outside the range may be expected, even if the data are really random. If we obtain more than one of the values of r_k outside the range, then we can deduce that the time series of the residuals is not random. Therefore, the prediction model that we are using is not sufficient to forecast the future behaviour of the time series representing the context information taken into consideration.

At the same time, the values of $z(t)$ should be close to zero. In order to test this, we consider the normalised value of $z(t)$ by dividing it by 100. We then define an acceptable percentage error ζ (for example equal to 0.1%). The value of ζ can be chosen according to the developer's application requirements. If the $z(t) > \zeta$, once again we derive that the prediction model does not provide good forecast results.

[10]Clearly, considering the available memory space, optimisations are possible: for example, a certain subset of time series can be stored and evaluated in parallel.

To summarise, the prediction component will return true if the $z(t) > \zeta$ and the time series of the last M values of the residuals is random. If there are not enough samples, the component is not able to derive any information about the predictability of the time series and, therefore, it will notify this fact to the system.

2.6.2 Implementation and Use of the Prediction Component in CAR

Given a certain number of measurements of the predictability of a time series, we define *predictability level* of a context attribute as the percentage of samples for which the component returns true, in other words, the percentage of samples for which the prediction model is sufficiently accurate given a predefined acceptable error.

If the predictability level is under a given threshold, alternative protocols can be used for example pure epidemic approaches or more efficient ones in terms of overhead such as the one that we presented in [Musolesi and Mascolo, 2006b]. The design of these automatic selection mechanisms is outside the scope of the present work.

2.6.3 Routing Table Transmission Interval and Predictability

The right choice of the routing transmission interval is of key importance for the predictability of the time series and, to this end, for the performance of the CAR protocol.

For example, as we show in Section 4.2.6, in presence of a scenario characterised by high mobility, the routing table transmission interval should be small in order to have frequent updates of the information. This also ensures a rapid convergence of the filter; moreover, the routing tables are used to beacon in order to detect the presence of a given host. If the routing table interval is high, hosts may also get undetected and, therefore, the trend of colocation may appear as random.

2.7 Related Work

2.7.1 Mobile Ad Hoc Networking

In the context of mobile ad hoc networks, the synchronous routing problem [Perkins, 2001] is essentially the distributed version of the shortest-path problem [Lynch, 1996]. The protocols used to solve this problem are usually classified into two main classes: link-state and distance-vector, depending on how routing information is distributed and route tables are calculated. Synchronous protocols for mobile ad hoc networks are further classified into two categories, but these depend instead on *when* routing information is exchanged

(i.e, *proactively* or *reactively*). This distinction is somewhat simplistic; there are protocols that do not fit neatly into either category.

Proactive protocols are similar to routing protocols for traditional wired networks in that they attempt to maintain up-to-date routing tables for the entire network. A large number of possible solutions have been proposed for this space; the most basic of these is the Destination Sequenced Distance Vector Routing protocol (DSDV) [Perkins and Bhagwat, 1994], though there are many others, notably WRP [Murthy and Garcia-Luna-Aceves, 1996], GSR [Iwata et al., 1999] and FishEye [Pei et al., 2000]. Reactive protocols were designed as an alternative approach for situations of lower call to mobility ratio (CMR). Routing information is obtained only when one node wishes to initiate a connection with another; this information is typically cached rather than recalculated on a per-message basis since, in most situations, routes are not expected to change significantly over short timescales. Examples of reactive protocols are the Dynamic Source Routing (DSR)[Johnson and Maltz, 1996a] and Ad Hoc On Demand Distance Vector (AODV)[Perkins and Royer, 1999] protocols, which are essentially based on reverse-path techniques, and the Temporally Ordered Routing Algorithm (TORA)[Park and Scott Corson, 1997], based on the Gafni-Bertsekas algorithm [Gafni and Bertsekas, 1981] and on the link-reversal concepts.

There are also hybrid approaches that combines the characteristics of proactive and reactive routing protocols, e.g., the Zone Routing Protocol (ZRP) [Haas and Pearlman, 1998]. An excellent survey about synchronous routing protocols, even if slightly outdated, is [Royer and Toh, 1999].

Another class of protocols is the one of the position-based routing ones. A complete survey about these protocols can be found in [Mauve et al., 2001]. Examples of position based routing protocols are the one used in the Terminodes Project [Blazevic et al., 2001], DREAM [Basagni et al., 1998] and EASE.

However, the EASE protocol [Grossglauser and Vetterli, 2003] is based not only on geographical information but also on the history of the encounters and, for this reason, is quite relevant to our work. Each node maintains a list of the time and location of its last encounter with every node in the network. This information is used to calculate an optimal path to a remote host; the authors assume that the probability of finding a host in a position near the position of the last encounter decreases as the time elapses. In other words, the elapsed time after the encounter provides a good measure of the reliability of the stored information about the position of a certain host. This protocol eliminates the traffic overhead due to this type of messages, relying only on the information that is able to collect during its movement. This algorithm is loop-free, since each host sends data towards a specific direction and robust, since messages are transmitted using multiple paths. Predictions techniques are not used in EASE; each node maintains only the last positions of the nodes without predicting the future ones.

However, all the approaches simply fail if a connected path between the sender and the receiver does not exist since they do not use store-and-forward mechanisms. In the evaluation in Chapter 4, we will show a comparison between CAR and the flooding protocol that provides an upper bound for the performance of this class of synchronous protocols in terms of delivery ratio.

2.7.2 Delay Tolerant Newtorking

A number of approaches have been proposed to enable asynchronous communication in intermittently connected mobile ad hoc networks.

The seminal paper analysing the problem and containing a first solution to it is [Li and Rus, 2000]. The authors propose an approach that guarantees message transmission in minimal time. However, the proposed algorithm relies on the fact that mobile hosts actively modify their trajectories to transmit messages. The authors developed algorithms that minimise the trajectory modifications under two different possible scenarios, one in which the movements of all the nodes in the system are known and one in which they are not.

In [Grossglauser and Tse, 2001], the authors demonstrate that the use of asynchronous delivery mechanisms, by means of intermediate carriers, increases the performance in terms of delivery ratio in ad hoc settings. The authors study a model of an ad-hoc network where n mobile nodes communicate in random source-destination pairs. They examine the per-session throughput for applications with loose delay constraints, such that the topology changes over the time-scale of packet delivery. They prove that, under this assumption, the per-user throughput can increase dramatically when nodes are mobile rather than fixed.

We give now an overview of the most interesting algorithms for routing in delay tolerant networks. A survey of routing protocols for intermittently connected delay tolerant mobile networks can be found in [Zhang, 2006].

Epidemic and Semi-epidemic Approaches

In [Fall, 2003], Fall proposes the Delay Tolerant Network architecture to solve the internetworking issues in scenarios where partitions are frequent and a connected path between message senders and receivers may be not present (such as satellite and interplanetary communication systems).

The simplest way of enabling communication in intermittently connected networks is by means of replicating messages on all the hosts (epidemic algorithms) or in a certain num-

ber of them (semi-epidemic algorithms). Epidemic algorithms were first devised in the context of distributed database systems in an attempt to guarantee data consistency after disconnections [Demers et al., 1988]. Interesting theoretical results show that, using random data exchanges, all updates are seen by all the hosts of the system in a bounded amount of time, given reasonable assumptions about connectivity. The epidemic routing protocol [Vahdat and Becker, 2000] forms the basis of much of the work in this field, even if buffer issues are not considered in their solution.

In [Chen and Murphy, 2001] the epidemic protocol was refined, presenting the so-called Disconnected Transitive Communication paradigm. This approach is based on the use of utility functions that are calculated evaluating context information, in particular, the history of the previous connections of the hosts (by storing a certain number of previous values). The authors provide a general framework rather than a detailed instantiation, and so the investigation about the aspects related to the composition of calculated delivery probabilities are almost entirely missing. Moreover, no prediction mechanisms are used.

In [Lindgren et al., 2003] the authors propose PRoPHET, a probabilistic routing protocol to enable asynchronous communication among intermittently connected portions of a network. Their approach is based on the fact that the exploited communication model is typically transitive and, for this reason, the probability of message delivery must be calculated accordingly: if, for example, a host H_A is able to communicate with H_B through H_C, the overall delivery probability is derived by the multiplication of the probability that H_A becomes a neighbour of H_B, with the probability that H_B becomes a neighbour of H_C. The calculation of the delivery probabilities is based on the period of time of colocation of two hosts, weighted by an ageing factor that is used to decrease the overall probability with the increasing age of the information on which it was based.

Another typical semi-epidemic approach is Spray&Wait [Spyropoulos et al., 2005]: this algorithm is based on the dissemination of a certain number of replicas in the system (the *spray* phase). Then the messages are finally delivered to the recipients of the messages when the carriers are in reach of the recipient. The authors claim that their approach outperforms other semi-epidemic protocols in terms of overhead. However, the design (and the evaluation) of the protocol does not take into account the overhead due to the discovery process of the neighbours. Recently, the authors also propose a variation of this mechanism (Spray & Focus) [Spyropoulos et al., 2007] based on the calculation of delivery probability along the lines of PRoPHET.

In [Sasson et al., 2003] a possible application of percolation theory (that studies the probability of transition between two states in fluids) to improve information dissemination based on the flooding of messages in ad hoc settings is presented. Another interesting epidemic model for mobile ad hoc networks is presented in [Khelil et al., 2002], where the similarities between flooding-based approaches for the information dissemination in mobile ad-hoc networks and the epidemic spreading of diseases are investigated.

Finally, multiple copies approaches can also benefit from the exploitation of erasure coding techniques [Byers et al., 1998] to improve the performance of the algorithm in terms of delivery ratio given a certain degree of redundancy in the system [Wang et al., 2005, Jain et al., 2005].

With respect to the these approaches, CAR relies on a single copy of the message in the network, optimising memory space and transmission overhead. For this reason, CAR is also suitable for scenarios composed of resource-constrained devices, whereas epidemic and semi-epidemic solutions are too expensive in terms of overhead.

Knowledge-based Routing Protocols

In [Jain et al., 2004] the authors present a set of protocols for routing in ad hoc networks based on a partial or complete knowledge of the structure of the network using a time-varying network graph representation. The design of this protocol is based on a modified version of Djikstra's algorithm, minimising the delivery delays and queuing times. The authors use linear programming techniques to derive a solution to this flow problem.

Zhao et alii in [Zhao et al., 2004] discuss the so-called Message Ferrying approach for message delivery in mobile ad hoc networks. The authors propose a proactive solution based on the exploitation of highly mobile nodes called ferries. These nodes move according to pre-defined routes, carrying messages between disconnected portions of the network. This approach has inspired many works based on similar mechanisms, including some based on the knowledge of the geographical coordinates of all the potential carriers such as [Li et al., 2006].

Island Hopping [N. Sarafijanovic-Djukic and Grossglauser, 2006] is another example of store-and-forward approach for delay tolerant network in a wide-area; in particular, the authors consider an urban setting and the evaluation of the system is based on traces of movement of taxis in the city of Warsaw. This protocol is predicated on the assumption that the network possesses *concentration points*, that are regions where the node density is much higher than on average and, consequently, where they have a higher probability of being in reach with the other hosts. The authors discuss an algorithm to infer the topology of the graph in a collaborative way and to exploit for efficient mobility-assisted forwarding of the messages.

Another example of routing protocols based on knowledge about the system is MoVe by Lebrun et alii [Lebrun et al., 2005]. This protocol has been specifically designed for delay tolerant vehicular networks. It relies on the evaluation of the geographical coordinates of the current position and the final destination of the vehicles. In particular, messages are forwarded to the hosts that are moving towards the closest point to the destination.

With respect to this class of protocols, we have introduced a general framework for the prediction of the evolution of the delay tolerant network scenarios which does not rely on knowledge of the routes of the mobile nodes, but is able to infer them from connectivity patterns. At the same time, we used lightweight mechanisms, because we believe that routing algorithms that are complex from a computational point of view are unsuitable for mobile devices, usually characterised by scarcity of resources. One example is the use of Kalman filter techniques that do not necessitate storing all the history of the evolution of the context information. Our techniques could be integrated with these approaches, since they address orthogonal aspects of the problem.

Data Collection and Delay Tolerant Sensor Networks

Delay tolerant protocols have also been developed for data collection in sparse sensor ad hoc networks using mobile carriers (or Mules like in [Shah et al., 2003]).

A major research effort in this direction is the ZebraNet Project [Liu et al., 2004, Juang et al., 2002] at Princeton University: researchers used collars with sensors to collect environmental information and to study the movement of animals. Data are replicated using a modified epidemic protocol based on priorities considering the originator of the information. The data created locally are the last to be deleted.

In [Small and Haas, 2003], Small and Haas describe another very interesting application of epidemic routing protocols to a problem of cost-effective data collection, using whales as message carriers. Fixed buoys are used to collect data that are copied epidemically, stored and spread among the whales.

DTN Technologies for Developing Countries and Rural Communities

Finally, as seen with ZebraNet, some actual deployments of these technologies already exist. For instance, Daknet [Pentland et al., 2004] is an example of a system for delay tolerant networking deployed in India and Cambodia. Daknet transmits data over short point-to-point links between *digital kiosks* and portable storage devices called MAPs (Mobile Access Points). These are mounted on and powered by buses, motorcycles or even bicycles with small generators. Wi-Fi transceivers automatically transfer the data stored in the MAPs, when they are in reach.

A more advanced system for supporting communication in rural areas of India using vehicles as a *mechanical backhaul* has been developed and deployed by the group lead by Keshav at the University of Waterloo [Seth et al., 2006]. Cars and buses are used to carry data to and from villages and Internet gateways.

Another project to support communication in areas not covered by traditional telecommunication systems is DieselNet [Burgess et al., 2006] that exploits buses to provide intermittent connectivity also outside urban areas by means of buses. To enhance connectivity between mobile nodes, they designed *throwboxes* that act as stationary routers. Throwboxes are untethered from any power-supply, or backend wireless or wired connectivity. This makes throwboxes easily deployed ad-hoc into a challenged environment using batteries for short-term deployment, or solar for longer-term use.

CAR could be potentially used also in presence of fixed hosts, possibly equipped with larger buffer. In fact, the definition and the calculation of the change rate of connectivity and host colocation are independent form the fact that a host if physically moving. Only relative movement matters: a fixed host may be *relatively* highly mobile. This sounds like an oxymoron, but a similar scenario can be realistic in the case of fixed nodes that get in reach of many different devices because of their geographical strategic location, such as the one that may be located in tube or train stations.

2.7.3 Prediction Techniques Applied to Delay Tolerant Systems

In the recent years, prediction techniques have been applied in different areas of computer science and, in particular, in the field of dynamic distributed systems and mobile networking.

Systems for delay tolerant communication based on the prediction of movements or other resource indicators include [Lindgren et al., 2003, Wang and Wu, 2006]. However, these works do not rely on the analysis of time series like CAR and do not take into consideration the problem of the predictability of the indicators used to improve the system performance.

In [Kim and Noble, 2001] the authors analyse the application of various types of filters to forecast network performance; however, they do not take into consideration the problem of evaluating the actual predictability of the observed systems. More recently, in [Kim et al., 2006], Kalman filter techniques have been used to extract the movement of the users. Also in this work, the aspects related to the predictability of users' mobility are not directly considered.

We believe that these systems based on the movement prediction can benefit from the integration of our component in order to make alternative decisions when prediction is not possible. Our approach is lightweight, since it requires to store only a small number of the past values of the context information. Moreover, it is easy to integrate with virtually all possible prediction techniques based on the analysis of time series. Furthermore, we believe that it may be applied in many other fields in computer science where similar prediction techniques are (or can be) used such as peer-to-peer systems and grid computing.

With respect to the general problem of designing a prediction model, many techniques have been proposed in the literature. However, we do not discuss the several approaches tackling the problem of time series forecasting, since it is outside the scope of this thesis. Comprehensive introduction to the field can be found in [Brockwell and Davis, 1996, Durbin and Koopman, 2001, Chatfield, 2004]. We only remark again here that our choice was motivated by the fact that the calculation of the Kalman filter models are recursive so that, although the current estimates are based on the whole past history of measurements, there is no need for an ever-expanding or, in general, very large memory. The second evident advantage is that it converges fairly quickly where there is a constant underlying model, but it can also follow the variations of the observed system when the model is changing over time [Meinhold and Singpurwalla, 1983, Aoki, 1990].

As far as the predictability problem is concerned, we adopted one of the most widely used solutions in forecasting based on state space models [Durbin and Koopman, 2001] Comparison of accuracy with the alternative predictability model called *portmanteau lack-of-fit test* can be found in [Davies and Newbold, 1979]. A modified version of this test that is often used is the *Ljung-Box-Pierce statistic* [Box et al., 1994]. However, it has been proven that the performance of this model are in general quite poor [Davies and Newbold, 1979]. An alternative way of testing the residuals is the *Durbin-Watson statistic* [Granger and Newbold, 1986]. This method is usually exploited for measuring the predictability of auto-regressive models.

2.7.4 Context-aware Systems

Even if, in the recent years, a large number of so-called context-aware platforms have been presented [Chen and Kotz, 2000], we discuss only two interesting examples of systems, since these are related to the design and implementation of the autonomic mechanisms in the CAR protocol.

One of the most famous examples of context-aware systems is Odyssey [Satyanarayan et al., 1994], a platform designed and implemented at Carnegie Mellon University by Satyanarayan and his group. More specifically, Odyssey provides a support for the dynamic adaptation to the current level of available resources (memory, computation power, etc.) in a transparent way. Recently, an extension of Odyssey [Dushyanth et al., 2000] was proposed in order to improve this kind of adaptation using the previous context history. The goal of this new release of Odyssey is the optimisation of the so-called fidelity factor that is defined as the quality of results (for example, the definition of an image) presented to users using some prediction mechanisms based on machine learning theory. The technique chosen by the authors is based on the use of linear regression functions that must be provided by application developers. A drawback is related to the fact that developers must provide these functions that are usually

difficult or impossible to define with accuracy. Furthermore, programmers may not have the necessary know-how to provide them, since they might not know the details of the devices and the deployment scenario in advance. Finally, Odyssey is not suitable for an ad hoc scenario, since the architecture is composed of hosts with different capabilities (i.e., a collection of powerful servers and clients with limited resources.).

In [Poladian et al., 2004] the authors propose a novel approach to provide adaptation to the context at run time. They exploit optimisation techniques to maximise user's utility with respect to a specific task (for example watching a video or buying a product online) in order to configure dynamically the system. Each task requires the use of different services (such as web-browsing or text-editing). Users must define their utility functions by means of preference functions that map from a multidimensional capability space that is correlated to (and constrained by) the available resources. The optimised configuration is obtained by a search in the space of the possible configurations that are derived taking in consideration the required services in order to perform the required task. To implement this computationally complex algorithm, the authors uses a third-part operation research library to approximate the optimal result.

Chapter 3

Design of a Social Network Founded Mobility Model

> The sciences do not try to explain, they hardly try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, described observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work.
>
> *John von Neumann*

> Of all things the measure is man.
>
> *Protagoras*

3.1 Introduction

The definition of realistic mobility models is one of the most critical and, at the same time, difficult aspects of the simulation of applications and systems designed for mobile environments. Currently, there are very few and very recent public data banks capturing node movement in real large-scale mobile ad hoc environments.

For example, researchers at Intel Research Laboratory and the University of Cambridge distributed Bluetooth devices to people, in order to collect data about human movements and to study the characteristics of the colocation patterns among people. These experiments were firstly conducted among students and researchers in Cambridge [Chaintreau et al., 2005] and then among the participants of INFOCOM 2005 [Hui et al., 2005b]. Other similar projects are the Wireless Topology Discovery project at the UCSD [McNett and Voelker, 2005] and the campus-wide WaveLan traffic measurement and analysis exercises that have been carried out at Dartmouth College [Henderson et al., 2004]. At this institution, a project with the aim of creating a repository of publicly available traces for the mobile networking community has also been started [Kotz and Henderson, 2005].

Until now, in general, real movement traces have been rarely used for evaluation and testing of protocols and systems for mobile networks, with the only exception of [Tuduce and Gross, 2005] and [Hsu et al., 2005], in which the authors used, respectively, the movement traces collected from a campus scenario and direct empirical observations of pedestrians in downtown Osaka as a basis of the design of their models.

In general, synthetic models have been largely preferred [Camp et al., 2002]. The reasons of this choice are many. First of all, as mentioned, the available traces are limited. Second, these traces are related to very specific scenarios and their validity is difficult to generalise. However, as we will discuss later in the chapter, these data show surprising common statistical characteristics, such as the same distribution of the duration of the contacts and inter-contacts intervals. We define *contact duration* as the time interval in which two devices are in radio range. We define *inter-contacts time* as the time interval between two contacts. These indicators are particularly important in ad hoc networking and, in particular, in *opportunistic mobile networks*, such as delay tolerant mobile ad hoc networks [Musolesi et al., 2005, Harras et al., 2005]: inter-contacts times define the frequency and the probability of being in contact with the recipient of a packet or a potential carrier in a given time period. Third, the available traces do not allow for sensitivity analysis of the performance of algorithms, since the values of the parameters that characterise the simulation scenarios, such as the distribution of the speed or the density of the hosts, cannot be varied. Finally, in some cases, it may be important to have a mathematical model that underlines the movement of the hosts in simulations, in order to study its impact on the design of protocols and systems.

Many mobility models for the generation of synthetic traces have been presented (a survey can be found in [Camp et al., 2002]). The most widely used of such models are based on random individual movement; the simplest, the Random Walk mobility model (equivalent to Brownian motion), is used to represent pure random movements of the entities of a system [Einstein, 1956]. A slight enhancement of this is the Random Way-Point mobility model [Johnson and Maltz, 1996a], in which pauses are introduced between changes in direction or speed. More recently, a large num-

ber of more sophisticated mobility models for ad hoc network research have been presented [Le Boudec and Vojnovic, 2005, Jardosh et al., 2005, Maeda et al., 2005].

However, all synthetic movement models are suspect because it is quite difficult to assess to what extent they map reality. It is not hard to see, even only with empirical observations, that the random mobility models generate behaviour that is most unhuman-like. This analysis is confirmed by the examination of the available real traces. As we will discuss later in this chapter, mobility models based on random mechanisms generate traces that show properties (such as the duration of the contacts between the mobile nodes and the inter-contacts time) very different from those extracted from real scenarios.

Our model is based on a simple observation. In mobile ad hoc networks, mobile devices are usually carried by humans, so the movement of such devices is necessarily based on human decisions and socialisation behaviour. For instance, it is important to model the behaviour of individuals moving in groups and between groups, as clustering is likely in the typical ad hoc networking deployment scenarios of disaster relief teams, platoons of soldiers, groups of vehicles, etc. In order to capture this type of behaviour, we define a model for group mobility that is heavily dependent on the structure of the relationships among the people carrying the devices. Existing group mobility models fail to capture this social dimension [Camp et al., 2002][1].

Fortunately, in recent years, social networks have been investigated in considerable detail, both in sociology and in other areas, most notably mathematics and physics. In fact, in the recent years, various types of networks (such as the Internet, the World Wide Web and biological networks) have been studied by researchers especially in the statistical physics community. Theoretical models have been developed to reproduce the properties of these networks, such as the so-called small worlds model proposed in [Watts, 1999] or various scale-free models [Newman, 2003, Vasquez et al., 2003][2]. Excellent reviews of the recent progress in complex and social networks analysis may be found in [Albert and Barabasi, 2002] and [Newman, 2003].

However, as discussed in [Newman and Park, 2003], social networks appear to be fundamentally different from other types of networked systems. In particular, even if social networks present typical small-worlds behaviour in terms of the average distance between

[1]These mobility models can also be test other types of networks. Within the emerging field of sensor networks, mobile hosts are not necessarily carried directly by humans. However, sensor networks are usually embedded in artefacts and vehicles (such as cars or planes or clothing) or are spread across a geographical area (such as environmental sensors). In the former case, the movements of the sensors embedded in a car or in aeroplane, for instance, are not random but are dependent on the movements of the carriers; in the latter, movement is not generally a major issue.

[2]Scale-free networks are characterised by a degree distribution that shows the following power law tail shape:

$$P(k) = k^{-\gamma}$$

A function $f(x)$ is scale-free if it remains unchanged to within a multiplicative factor under a re-scaling of the independent variable x (i.e., it has a power law form) [Newman, 2003]

pairs of individuals (the so-called *average path length*), they show a greater level of clustering. In [Newman and Park, 2003] the authors observe that the level of clustering seen in many non-social systems is no greater than in those generated using pure random models. Instead in social networks, clustering appears to be far greater than in networks based on stochastic models. The authors suggest that this is strictly related to the fact that humans usually organise themselves into *communities*. Examples of social networks used for these studies are rather diverse and include, for instance, networks of coauthorships of scientists [Newman, 2001] and the actors in films with Kevin Bacon [Watts, 1999].

We propose a new mobility model that is founded on social network theory. One of the inputs of the mobility model is the social network that links the individuals carrying the mobile devices based on these results in order to generate realistic synthetic network structures [Watts, 1999]. The model allows collections of hosts to be grouped together in a way that is based on social relationships among the individuals. This grouping is only then mapped to a topographical space, with topography biased by the strength of social ties. The movements of the hosts are also driven by the social relationships among them. The model also allows for the definition of different types of relationships during a certain period of time (i.e., a day or a week). For instance, it might be important to be able to describe that in the morning and in the afternoon of weekdays, relationships at the workplace are more important than friendships and family one, whereas the opposite is true during the evenings and weekends.

We evaluate our model using real mobility traces provided by Intel Research Laboratory and we show that the model provides a good approximation of real movements in terms of some fundamental parameters, such as the distribution of the contacts duration and inter-contacts time. In particular, the data show that an approximate power law holds over a large range of values for the inter-contacts time. Instead, contacts duration distribution follows a power law for a more limited range. These characteristics of distribution are also very similar to those observed by the researchers at the University of California at San Diego and Dartmouth College [Chaintreau et al., 2005].

3.2 Design of the Mobility Model

In this section we present the design of the Community based mobility model. The description of the model, mirroring its conceptual steps, is organised as follows:

- Firstly, we describe how we model social relationships and, in particular, how we use *social networks as input* of the mobility model.
- Secondly, we present the *establishment of the model*: we discuss how we identify communities and groups in the network and how the communities are associated to

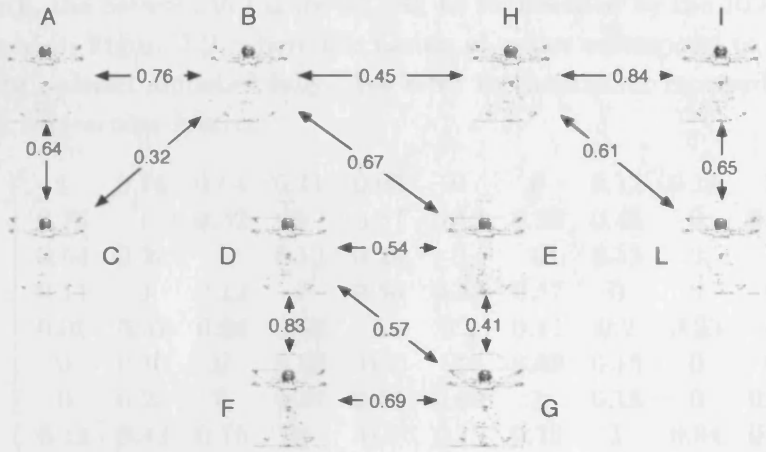

Figure 3.1: Example of social network.

a geographical space. Our observation here is that people with strong social links are likely to be geographically colocated often.

- Thirdly, we describe the algorithm that is at the basis of the *dynamics* of the nodes, that, again, is based on the strength of social relationships. We argue that individuals with strong social relationships move towards (or within) the same geographical area.

3.2.1 Using Social Networks as Input of the Mobility Model

Modelling Social Relationships

One of the classic ways of representing social networks is *weighted graphs*. An example of social network is represented in Figure 3.1. Each node represents one person. The weights associated with each edge of the network are used to model the strength of the interactions between individuals [Scott, 2000]. It is our explicit assumption that these weights, which are expressed as a measure of the strength of social ties, can also be read as a measure of the likelihood of geographic colocation, though the relationship between these quantities is not necessarily a simple one, as will become apparent. We model the degree of social interaction between two people using a value in the range $[0, 1]$. 0 indicates no interaction; 1 indicates a strong social interaction. Different social networks can be valid for different parts of a day or of a week[3].

[3] Let us consider a family of three people, with one child. During the days, when the child is at school and the parents at their workplaces, their social relationship is weak (i.e., represented with low values in the matrix). During the evening, the social ties are stronger as the family members tend to be colocated (i.e., high values in the matrix). The relationship between two colleagues sharing the same office will be represented with a value higher than these family relationships during the working hours in week days.

As a consequence, the network in Figure 3.1 can be represented by the 10×10 symmetric matrix $\mathbf{M}$ showed in Figure 3.2, where the names of nodes correspond to both rows and columns and are ordered alphabetically. We refer to the matrix representing the social relationships as *Interaction Matrix*.

$$\mathbf{M} = \begin{bmatrix} 1 & 0.76 & 0.64 & 0.11 & 0.05 & 0 & 0 & 0.12 & 0.15 & 0 \\ 0.76 & 1 & 0.32 & 0 & 0.67 & 0.13 & 0.23 & 0.43 & 0 & 0.05 \\ 0.64 & 0.32 & 1 & 0.13 & 0.24 & 0 & 0 & 0.15 & 0 & 0 \\ 0.11 & 0 & 0.13 & 1 & 0.54 & 0.83 & 0.57 & 0 & 0 & 0 \\ 0.05 & 0.67 & 0.24 & 0.54 & 1 & 0.2 & 0.41 & 0.2 & 0.23 & 0 \\ 0 & 0.13 & 0 & 0.83 & 0.2 & 1 & 0.69 & 0.15 & 0 & 0 \\ 0 & 0.23 & 0 & 0.57 & 0.41 & 0.69 & 1 & 0.18 & 0 & 0.12 \\ 0.12 & 0.43 & 0.15 & 0 & 0.2 & 0.15 & 0.18 & 1 & 0.84 & 0.61 \\ 0.15 & 0 & 0 & 0 & 0.23 & 0 & 0 & 0.84 & 1 & 0.65 \\ 0 & 0.05 & 0 & 0 & 0 & 0 & 0.12 & 0.61 & 0.65 & 1 \end{bmatrix}$$

Figure 3.2: Example of an Interaction Matrix representing a simple social network.

The generic element $m_{i,j}$ represents the interaction between two individuals i and j. We refer to the elements of the matrix as the *interaction indicators*. The diagonal elements represent the relationships that an individual has with himself and are set, conventionally, to 1. In Figure 3.1, we have represented only the links associated to a weight equal to or higher than 0.25.

The matrix is symmetric since, to a first approximation, interactions can be viewed as being symmetric. However, we are using a specific measure of the strength of the relationships. It is probable that by performing psychological tests, the importance of a relationship, such as a friendship, will be valued differently by the different individuals involved; in our model, this would lead to an asymmetric matrix. We plan to investigate this issue further in the future.

The Interaction Matrix is also used to generate a *Connectivity Matrix*. From matrix $\mathbf{M}$ we generate a binary matrix $\mathbf{C}$ where a 1 is placed as an entry c_{ij} if and only if $m_{i,j}$ is greater than a specific threshold t (i.e., 0.25). The Connectivity Matrix extracted by the Interaction Matrix in Figure 3.2 is showed in Figure 3.3. The idea behind this is that we have an *interaction threshold* above which we say that two people are interacting as they have a strong relationship. The Interaction Matrix (and, consequently, the Connectivity Matrix) can be derived by available data (for example, from a sociological investigation) or using mathematical models that are able to reproduce characteristics of real social networks. As we will discuss in Section 3.3.3, the default implementation of our model uses the so-called Caveman model [Watts, 1999] for the generation of synthetic social networks with realistic characteristics (i.e, high clustering and low average path length). However, this is a customisable aspect and, if there are insights on the type of scenarios to be tested, a user-defined matrix can be used as input.

$$\mathbf{C} = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

Figure 3.3: Example of a Connectivity Matrix representing a simple social network.

Detection of Community Structures

The simulation scenario is established by mapping groups of hosts to certain areas in geographical space. After the definition of the social graph described above, groups, i.e., the highly connected set of nodes in the graph, need to be isolated. Fortunately, existing algorithms can be exploited for this purpose.

We use the algorithm proposed in [Newman and Girvan, 2004] to detect the presence of community structures in social networks represented by matrices, like the Connectivity Matrix that we have defined in the previous section. This algorithm is based on the calculation of the so-called *betweenness* of edges. The value of the betweenness represents one of the possible ways of evaluating the centrality of an edge. For example, considering two communities connected by few inter-community edges, all the paths through the nodes in one community to nodes in the other must traverse one of these edges, that, therefore, will be characterised by a high betweenness. Intuitively then, one of the possible estimation of the centrality of an edge is given by the number of shortest (geodesic) paths between all pairs of vertices that run along it. In other words, the average distance between the vertices of the network has the maximum increase when the edges with the highest betweenness are removed.

Therefore, in order to extract the communities from the network, edges characterised by high values of centrality are progressively detected in subsequent rounds. At each round, one of the edges of the host with the highest centrality is removed. The final result is a network composed of (fairly distinguishable) groups of hosts (i.e., the communities).

The complexity of this algorithm is $O(mn^2)$, considering a graph with m edges and n vertices. The calculation of the shortest path between a particular pair of vertices can be performed using a breadth-first search in time $O(m)$ and there are $O(n^2)$ vertices. However, in [Newman and Girvan, 2004], the authors proposed a faster algorithm with a complexity equal to $O(mn)$.

As we said, the algorithm can be run on the graph a number of times, severing more and more links and generating a number of distinguishable communities. However, a mechanism to stop the algorithm when further cuts would decrease the quality of the results is also needed: this would mean that the graph has reached a state when meaningful communities have already been identified.

We adopted a solution based on the calculation of an indicator defined as *modularity* Q [Newman and Girvan, 2004]. This quantity measures the proportion of the edges in the network that connect vertices within the same community minus the expected value of the same quantity in a network with the same community division but random connections between the vertices. If the number of edges within the same community is no better than random, the value of Q is equal to 0. The maximum value of Q is 1; such a value indicates very strong community structure. In real social networks, the value of Q is usually in the range $[0.3, 0.7]$. The analytical definition of the modularity of a network division can be found in [Newman and Girvan, 2004]. At each run the algorithm severs one edge and measures the value of Q. The algorithm terminates when the obtained value of Q is less than the one obtained in the previous edge removal round. This is motivated by the fact that Q presents one or, more rarely, two local peaks at most: therefore, the algorithm can be stopped when the first local peak is reached. This is clearly an approximation since the value of the other possible local peak (if it exists) may be higher, but it has been observed that the quality of the division obtained is, in the vast majority of the cases, very good [Newman and Girvan, 2004]. Also, by adopting this technique, we considerably simplify the computational complexity of the algorithm.

In order to illustrate this process, let us now consider the social network in Figure 3.1. Three communities (that can be represented by sets of hosts) are detected by running the algorithm: $C_1 = \{A, B, C\}$, $C_2 = \{D, E, F, G\}$ and $C_3 = \{H, I, L\}$. Now that the communities are identified given the matrix, they need to be associated with locations.

3.2.2 Establishment of the Model: Placement of the Communities in the Simulation Space

After the communities are identified, each of them is randomly associated to a specific location (i.e., a square) on a grid[4]. We use the symbol $S_{p,q}$ to indicate a square in position p, q. The number of rows and columns are inputs of the mobility model.

Going back to the example, in Figure 3.4 we show how the communities we have identified can be placed on a 3x4 grid (the dimension of the grid is configurable by the user and influences the density of the nodes in each square). The three communities C_1, C_2, C_3 are

[4]A non random association to the particular areas of the simulation area can be devised, for example by deciding pre-defined *areas of interest* corresponding for instance to real geographical space. However, this aspect is orthogonal to the mechanisms in this thesis.

placed respectively in the grid in the squares $S_{a,2}$, $S_{c,2}$ and $S_{b,4}$. Each node of a certain community is placed in random positions inside the assigned square.

Once the nodes are placed on the grid, the model is established and the nodes move around according to social-based attraction laws as explained in the following section.

3.2.3 Dynamics of the Mobile Hosts

As described in the previous section, a host is initially positioned in a certain square in the grid. Then, in order to drive movement, a goal is assigned to the host. More formally, we say that a host i is associated to a square $S_{p,q}$ if its goal is inside $S_{p,q}$. Note that host i is not necessarily always positioned inside the square $S_{p,q}$, despite this association (see below).

The goal is simply a point on the grid which acts as *final destination* of movement like in the Random Way-Point model, with the exception that the selection of the goal is not as random.

Selection of the first goal

When the model is initially established, the goal of each host is randomly chosen inside the square associated to its community (i.e, the first goals of all the hosts of the community C_1 will be chosen inside the square $S_{a,2}$).

Selection of the subsequent goals

When a goal is reached, the new goal is chosen according to the following mechanism. A certain number of hosts (zero or more) are associated to each square $S_{p,q}$ at time t. Each square (i.e., place) exerts a certain *social attractivity* to a certain host. The social attractivity of a square is a measure of its importance in terms of the social relationships for the host taken into consideration. The social importance is calculated by evaluating the strength of the relationships with the hosts that are moving towards that particular square (i.e., with the hosts that have a current goal inside that particular square). More formally, given $CS_{p,q}$ (i.e., the set of the hosts associated to square $S_{p,q}$), we define *social attractivity* of that square towards the host i $SA_{(p,q)_i}$, as follows

$$SA_{(p,q)_i} = \frac{\sum_{j \in CS_{p,q}} m_{i,j}}{|CS_{p,q}|} \tag{3.1}$$

In other words, the social attractivity of a square in position (p, q) towards a host i is defined as the sum of the interaction indicators that represent the relationships between i and the other hosts that belong to that particular square, normalised by the total number of hosts associated to that square. If $|C_{S_{p,q}}| = 0$ (i.e., the square is empty), the value of $SA_{(p,q)_i}$ is set to 0. We consider the average values of the interaction indicators and not the sum of them, since we want to model the fact that a destination is selected not only because there are many people with whom the host has relationships (that may also be weak), but because, in average, the people that are in that square have strong relationships with it. In other words, by doing this, the selection is based not on the number of people but exclusively on the strength of the relationships.

The mobility model allows for two alternative mechanisms for the selection of the next goal that are described in the following two subsections, a *deterministic* one based on the selection of the square that exerts the highest attractivity and a *probabilistic* one based on probability of selection of a goal in a certain square proportional to their attractivities. Using the first one, the goals are chosen only inside the squares associated to the community, whereas with the second, the hosts may also randomly select their goals in other squares of the simulation area, with a certain non zero probability. In other words, the second mechanism allows for the selection of the destinations not only based on social relationships adding more realism to the model.

Deterministic Selection of the Goal in the Area with the Highest Social Attractivity According to this mechanism, the new goal is randomly chosen inside the square characterised by the highest social attractivity; it may be again inside the same square or in a different one. New goals are chosen inside the same area when the input social network is composed by loosely connected communities (in this case, hosts associated with different communities have, in average, weak relationships between each others). On the other hand, a host may be attracted to a different square, when it has strong relationships with both communities. From a graph theory point of view, this means that the host is located between two (or more) clusters of nodes in the social network[5].

Let us suppose, for example, that host A has reached its first goal inside the square $S_{a,2}$. The new goal is chosen by calculating the social attractivities of all the squares that compose the simulation space and then by choosing the highest. If, say, square $S_{c,2}$ exerts the highest attractivity (for example, because a host with strong relationship with node A has joined that community), the new goal will then be selected inside that square.

Probabilistic Selection of the Goal Proportional to the Social Attractivity An alternative mechanism is based on a selection of the next goal proportional to the attrac-

[5]This is usually the case of hosts characterised by a relatively high betweenness that, by definition, means that they are located *between* two (or more) communities.

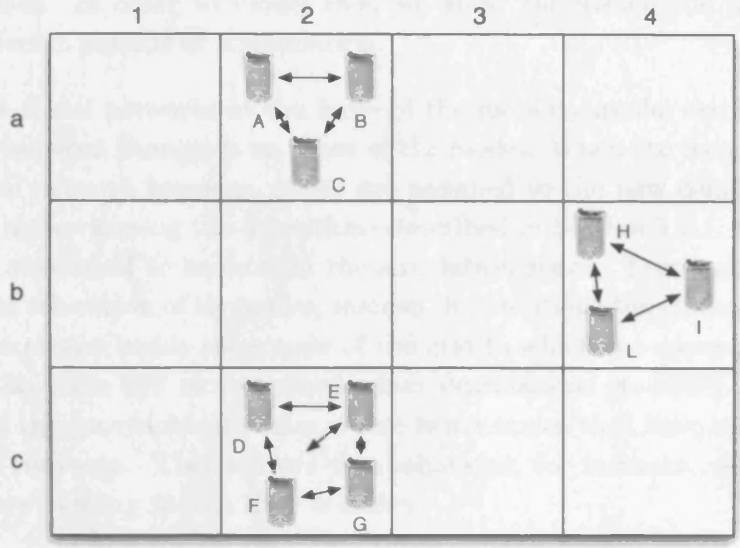

Figure 3.4: Example of initial simulation configuration.

tivity of each square.

In other words, we assign a probability $P(s = S_{p,q_i})$ of selecting the square S_{p,q_i} as follows:

$$P(s = S_{p,q_i}) = \frac{SA_{(p,q)_i} + d}{\sum_{j=1}^{p \times q}(SA_{p,q_j} + d)} \tag{3.2}$$

where d is a random value greater than 1 in order to ensure that the probability of selecting a goal in a square is always non zero[6].

The parameter d can be used to increase the randomness of the model in the process of selection of the new goal. This may be exploited to increase the realism of the generated scenario, since in real situations, humans also move to areas without people or for reasons not related to their social sphere.

Social Network Reconfigurations and their Effects on the Dynamics of Mobile Hosts

Like in everyone's life, the movement during the day are governed by different patterns of mobility which depend on the people we need to interact with. For example, most people spend a part of their day at work, interacting with colleagues, and another part at home

[6]The role of d is similar to the *damping factor* used in the calculation of the Google PageRank [Brin and Page, 1998]. In fact, the transitions between squares can also be similarly represented using a Markov Chain model with $P(s = S_{p,q_i})$ as probability of transitions between states (squares).

with their families. In order to model this, we allow the association of different social networks to different periods of a simulation.

Periodically, the social networks at the basis of the mobility model can be changed. The interval of time between changes is an input of the model. When the reconfiguration of the underlying social network happens, nodes are assigned to the new communities that are detected in the network using the algorithms described in Section 3.2.1. Communities are then randomly associated to squares in the simulation space. This assignment does not imply immediate relocation of the nodes, instead, it conditions the choice of the next goal. In fact, goals are chosen inside the square of the grid to which the community they belong to is assigned. So hosts will move towards their destinations gradually. The nodes start moving towards the geographical region where other nodes that have strong interactions with them will converge. This mirrors the behaviour, for instance, of commuters who travel home every evening to join their families.

3.3 Implementation and Evaluation

In order to evaluate our model we have performed a number of tests, in particular, we have analysed real mobility traces collected by Intel Research. We have then tested our model using realistic social networks and compared the mobility patterns with the Intel traces. In this section, we will present and discuss the results of our simulations comparing them with these data from real scenarios.

3.3.1 Implementation of the model

We implemented a movement patterns generator that produces primarily traces for the ns-2 simulator [ns2, 2006], one of the most popular tool in the ad hoc network research community. However, the generator is also able to produce traces in a XML meta-format that can be parsed and transformed into other formats (for example, by using XSLT) such as the one used by GlomoSim/Qualnet [Zeng et al., 1998]. The model is available for downloading at the following URL: `http://ww.cs.ucl.ac.uk/staff/m.musolesi/mobilitymodels`.

3.3.2 Validation of the Model using Real Movement Traces

In this section, we present a comparison of the properties of the movement patterns generated by our mobility model with those of the real traces provided by Intel Research. The description of this measurement exercise is presented in [Chaintreau et al., 2005]. In that paper, the authors also compare their results with other publicly available data sets

Table 3.1: Simulation parameters

Number of hosts	100
Simulation area	5 Km x 5 Km
Propagation model	free space
Antenna type	omnidirectional
Transmission range (radius)	250 m
Square area	200 m x 200 m
Node speed	m/s (unif. distribution)
Number of groups (Caveman model)	10
Rewiring probabiliity (Caveman model)	0.0/0.05/0.1/0.2
Simulation duration	1 day (86400 s)
Reconfiguration interval	8 hours

provided from University of California at San Diego [McNett and Voelker, 2005] and from Dartmouth College [Henderson et al., 2004] showing evident similarities between the patterns movements collected by the three different groups. For this reason, we decided to compare the traces obtained by using our mobility model only with the data provided by Intel Research [7].

The traces were collected by Intel researchers using iMotes (a modified version of the Berkeley Motes) [Balazinska and Castro, 2003] equipped with Bluetooth. The iMotes were then given to members of the staff of Intel Research and University of Cambridge. The iMotes were packed in keyfobs in order to make sure that people carried them around. Each iMotes logged contacts data in a flash memory using the standard Bluetooth Baseband layer inquiry procedure. Every contact was stored as a tuple composed of three fields, the MAC address of the other device, the start and the end of the interval of time of the contact. Every iMotes collected information, not only on the other samplers, but also on the other Bluetooth devices in reach. The iMotes were programmed to perform an inquiry for 5 seconds every $2+\Delta$ minutes with Δ randomly chosen in the range $[-12, 12]$ seconds. This correction was introduced to avoid undesired synchronisation effects, i.e., to avoid that the iMotes performed inquiries at the same time. iMotes are not able to perform and reply to inquiries at the same time.

[7]The measurement exercise was also repeated among the participants of INFOCOM 2005 [Hui et al., 2005b]. In our comparison, we used the traces related to the first experiment. However, the results obtained in the two different studies show remarkable similarities.

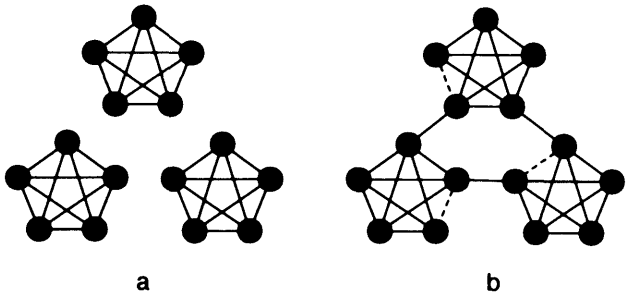

Figure 3.5: Generation of the social network in input using the Caveman model: (a) initial configuration with 3 disconnected 'caves'. (b) generated social network after the rewiring process.

3.3.3 Description of the Simulation

We tested our mobility model using several runs generating different mobile scenarios and we compared the results with the real movement patterns provided by Intel and synthetic traces generated using a Random Way-Point model.

We tested our model considering a scenario composed of 100 hosts in a simulation area of $5\ km \times 5\ km$, divided into a grid composed of 625 squares of 200 m (i.e., the numbers of rows and columns of the grid were set to 25). We chose a relatively large simulation scenario, with a low population density, in order to better see the differences in the results obtained with a Random Way-Point model. In fact, in small simulation areas, the limited possible movements and the higher probability of having two nodes in the same transmission range may affect the simulation results introducing side-effects that are not entirely due to the mobility model.

We also assumed that each device was equipped with an omnidirectional antenna with a transmission range of 250 m, modelled using a free space propagation model. The speeds of the nodes were randomly generated according to a uniform distribution in the range $[1 - 6]\ m/s$. The duration of the simulation was one day and the reconfiguration interval was equal to 8 hours. These values were not chosen to reproduce the movements described by the traces provided by Intel, rather, we were more interested in observing if similar patterns could be detected in synthetic and real traces. In other words, our goal has mainly been to verify whether the movement patterns observed in Intel traces were generated by our mobility model. We ran simulations with both probabilistic and deterministic selection mechanisms. The resulting distributions of contacts duration and inter-contacts time are very similar. We show all the results related to the deterministic selection mechanism, since this one generates longer colocation and inter-contacts time intervals that are closer to the ones observed in the Intel traces in terms of duration. However, we also report results obtained using the probabilistic selection mechanism, to

show that this also produces power-law distributions. The value of the damping factor d for the probabilistic selection mechanism was set equal to 0.01; this lead to a very small (but non null) probability of selecting any empty squares in the simulation space. This, together with the fact the selection of the squares was based on probabilities proportional to the attractivities, lead, in average, to shorter colocation time.

A key aspect of the initialisation of our model is the selection of the social network in input. We implemented a generator of synthetic social networks using the so-called Caveman Model proposed by Watts [Watts, 1999]. The social network is built starting from K fully connected graphs (representing communities living in isolation, like primitive men in caves). According to this model, every edge of the initial network in input is re-wired to point to a node of another cave with a certain probability p. The re-wiring process is used to represent random interconnections between the communities. Figure 3.5.a shows an initial network configuration composed by 3 disconnected communities (*caves*) composed by 5 individuals; a possible social network after random rewiring is represented in Figure 3.5.b.

Individuals of one cave are closely connected, whereas populations belonging to different caves are sparsely connected. Therefore, the social networks generated using this model are characterised by a high clustering coefficient and low average path length. It has been proved that this model is able to reproduce social structures very close to real ones [Watts, 1999]. We generated social networks with different rewiring probabilities, also considering the case of disconnected communities (i.e., $p = 0$).

We also implemented a movement patterns generator based on the Random Way-Point model. We generated traces with the same simulation scenarios in terms of size of the area and characteristics of the mobile devices, with hosts that move with a speed uniformly distributed in the range $[1 - 6]$ m/s and stop time equal to $[1 - 6]$ s.

3.3.4 Simulation Results

The emergent structure of the network derived by analysing the Intel traces is typically exponential [Albert and Barabasi, 2002]; in fact, the *degree of connectivity* shows a local peak near the average. Our mobility model (indicated with CM) produces a similar type of distribution as shown in Figure 3.6. The peak shifts to the right as the density of the squares increases. We analysed two further properties of the movement patterns, the contact duration and the inter-contacts time. We adopt the same definitions used by the authors of [Chaintreau et al., 2005] in order to be able compare the results.

Figures 3.7 and 3.8 show the comparison between the inter-contacts time and the contact duration cumulative distributions[8] using log-log coordinates. These distributions are ex-

[8]Cumulative distributions are generally used instead of frequency distributions to avoid the issues

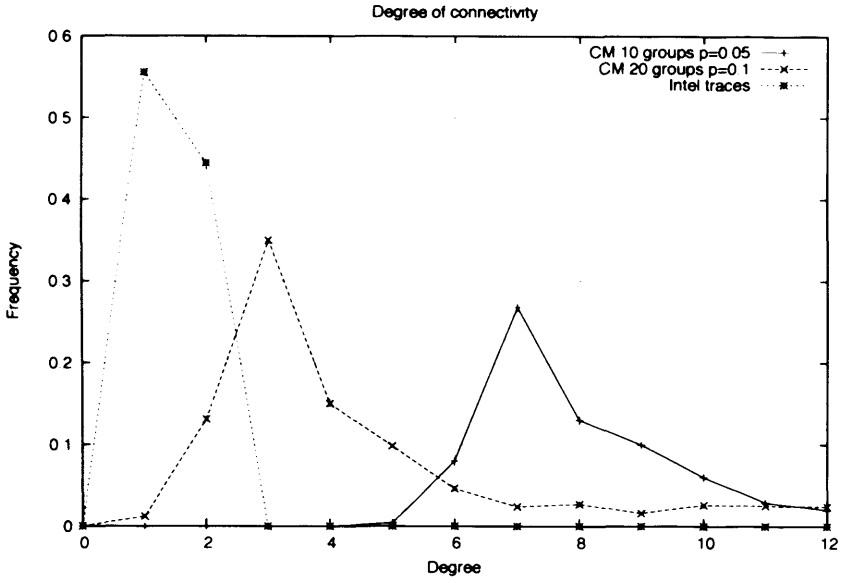

Figure 3.6: Distribution of the degree of connectivity.

tracted from the real and synthetic traces generated by the Random Way-Point (indicated with RWP) and our Community based mobility model with different rewiring probabilities p.

With respect to the inter-contacts time, our traces (excluding the case with $p = 0$ that we will discuss separately) show an approximate power law behaviour for a large range of values like those extracted from Intel data. A similar pattern can be observed in UCSD and Dartmouth traces [Chaintreau et al., 2005]. The cumulative distribution related to Random Way-Point, instead, shows a typical exponential distribution. The same behaviour can be observed for the traces generated using our Community based mobility model with a probability of rewiring equal to 0. In fact, in this case, the only movements of the hosts outside the assigned square happen when a reconfiguration takes place (i.e., a new generation of the social networks takes place and a consequent new assignment to different squares in the grid are performed). However, the case of disconnected and isolated communities is not so realistic. As far as the contacts time distribution is concerned, we observe a power law behaviour for a much more limited range of values and, in general, with a lower angular coefficient of the interpolating line. The traces from Dartmouth College and UCSD also show a power law distribution with different angular coefficients [Chaintreau et al., 2005]. It seems that data related to different scenarios are

related to the choice of the bins of the plot. It is possible to prove that if a set of data shows a power law behaviour using a frequency histogram, its cumulative distribution also follows the same pattern.

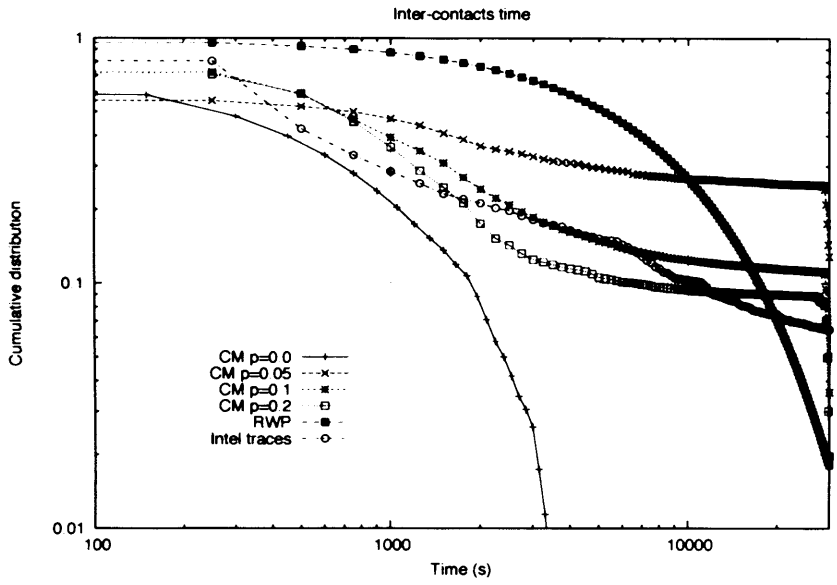

Figure 3.7: Comparison between synthetic and real traces (log-log coordinates) : cumulative distribution of inter-contacts time in seconds.

characterised by different types of power law distributions.

By plotting the same distributions using semi-log coordinates (see Figures 3.9 and 3.10), the differences between the curves corresponding to real traces and those generated using the Random Way-Point mobility model are even more evident. The exponential nature of the cumulative distribution of the inter-contacts time[9] extracted by the latter is clearly reflected by the approximated straight line that is shown in the figure. The results that we obtained using the probabilistic selection mechanism are shown in Figures 3.11 and 3.12. As it is possible to observe in Figure 3.12, by using this mechanism, we obtain shorter contacts durations. This is caused by the higher probability of leaving the current square.

Figure 3.13 and 3.14 show the influence of the speed respectively on the cumulative distributions of the inter-contacts time and contacts duration. We simulated scenarios with host speed uniformly distributed in the range $[1-6]$, $[1-10]$ and $[1-20]m/s$. The cumulative distributions related to all these scenario can be approximated with a power law function for a wide range of values.

In many of our experiments, the coefficient of the power law of the distribution of the Intel traces is different from those related to synthetic traces generated using our model.

[9]This behaviour has been theoretically studied and predicted by Sharma and Mazumdar in [Sharma and Mazumdar, 2004].

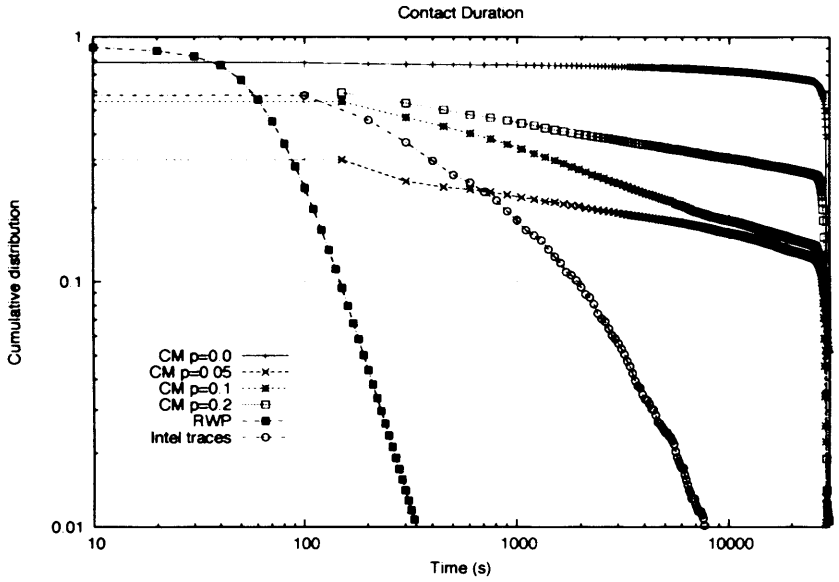

Figure 3.8: Comparison between synthetic and real traces (log-log coordinates) : cumulative distribution of contacts duration in seconds.

Different coefficients can be observed in the available sets of real traces. In a sense, it seems that the values of these coefficients characterise the various mobile settings. It is worth noting that currently there are not available theoretical models that justify the emergence of these distributions.

The impact of the density of the population in the simulation scenario is presented in Figures 3.15 and 3.16. We simulated scenarios composed of 100, 200, 300 nodes with a starting number of groups for the Caveman model, respectively equal to 10, 20, 30, and a rewiring probability of 0.2. Also in these scenarios, the inter-contacts time and contacts duration distributions follow a similar pattern. As discussed previously, our aim was not to exactly reproduce the traces provided by Intel. However, quite interestingly, we observe that the inter-contacts time distribution lie in between the curves representing the scenario composed of 100 and 200 nodes. The number of nodes recorded in the Intel experiments was in fact 140. Instead, the contacts duration distribution is bounded by the curves extracted by these two synthetic traces for a smaller range of values. Finally, in Figures 3.17 and 3.18 we consider a scenario composed of 100 hosts connected by a social network generated using different initial numbers of groups (i.e., caves) as input for the Caveman model (with a re-wiring probability equal to 0.1). By varying the number of groups, the density of the squares of the grid changes. The power law patterns can be observed in all the scenario, also with a large number of small initial groups.

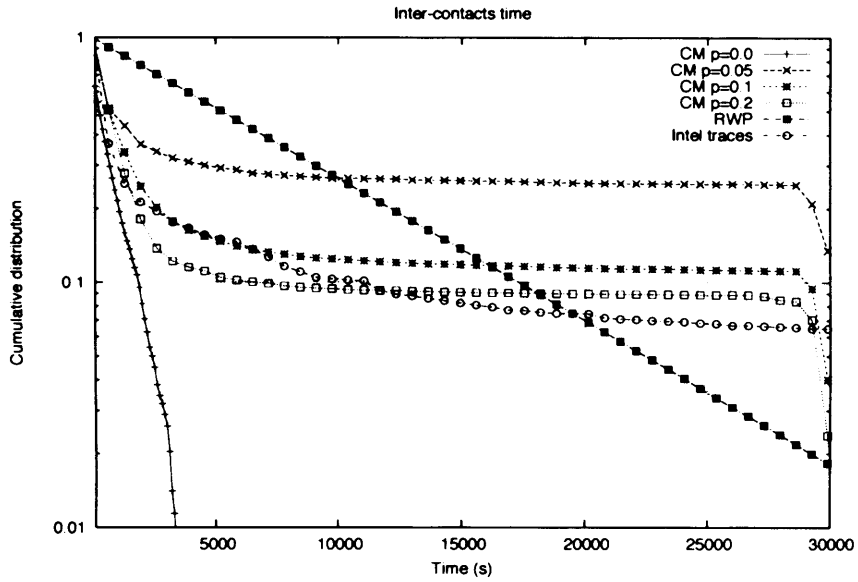

Figure 3.9: Comparison between synthetic and real traces (semi-log coordinates): cumulative distribution of inter-contacts time in seconds.

3.3.5 Influence of the Choice of the Mobility Model on Routing Protocols Performance

Simulation Description

In order to show the impact of the choice of this mobility model on the testing of protocols for mobile ad hoc networks, we simulated a scenario composed of 50 hosts and we compared the performance in terms of delivery ratio of the AODV [Perkins and Royer, 1999] and DSR [Johnson and Maltz, 1996a] protocols using the ns-2 simulator (using the reference implementation provided by the authors). In Chapter 4, we will present a thorough analysis of CAR and other protocols for delay tolerant communication in detail.

We used a $1000m \times 1000m$ simulation area with a maximum node transmission range equal to $250m$. We chose the two-ray pathloss model as propagation model and at the MAC layer, the IEEE 802.11 DCF protocol was used with a bandwidth equal to 2 Mbps. We started 10 sessions between randomly chosen hosts[10] using CBR traffic with data packet size and sending rate respectively equal to 512 bytes and 4 packets/second. The simulation

[10]This kind of traffic can be considered as a worst case scenario; in reality, it is probable that sessions will be between hosts of the same community. We plan to investigate this aspect in the future, developing a social networks founded traffic generator.

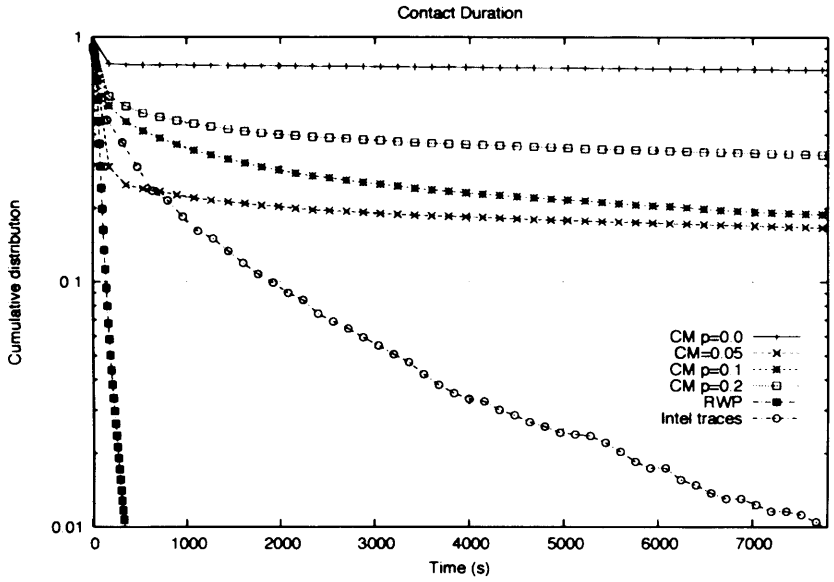

Figure 3.10: Comparison between synthetic and real traces (semi-log coordinates): cumulative distribution of contacts duration in seconds.

time was equal to 2 hours.

We studied the influence of the speed on the performance comparing the results obtained by using the Random Way-Point model and the Community based mobility model. Every node in the simulation is moving at the same speed. With respect to the Random Way-Point model, the stopping times are chosen randomly in the interval $[1-10]m/s$. As far as our mobility model is concerned, the reconfiguration interval was set to 1 hour. The social network in input was generated with the Caveman model with 5 groups of 10 individuals and a re-wiring probability equal to 0.1. The simulation scenario was divided into a 5×5 grid. We performed a number of runs sufficient to achieve a 10% confidence interval with an error equal to 5%.

Simulation Results

Using the Random Way-Point mobility model, as expected, and confirming the results obtained by the authors of these protocols [Perkins and Royer, 1999, Johnson and Maltz, 1996a], the delivery ratio decreases as the speed increases. The simulations are presented in Figure 3.19 and 3.20, respectively. Instead, using our model, the decreasing trend of the delivery ratio is less evident, since the emerging structure is composed of groups of hosts moving in limited areas (i.e., the square of the grids) that

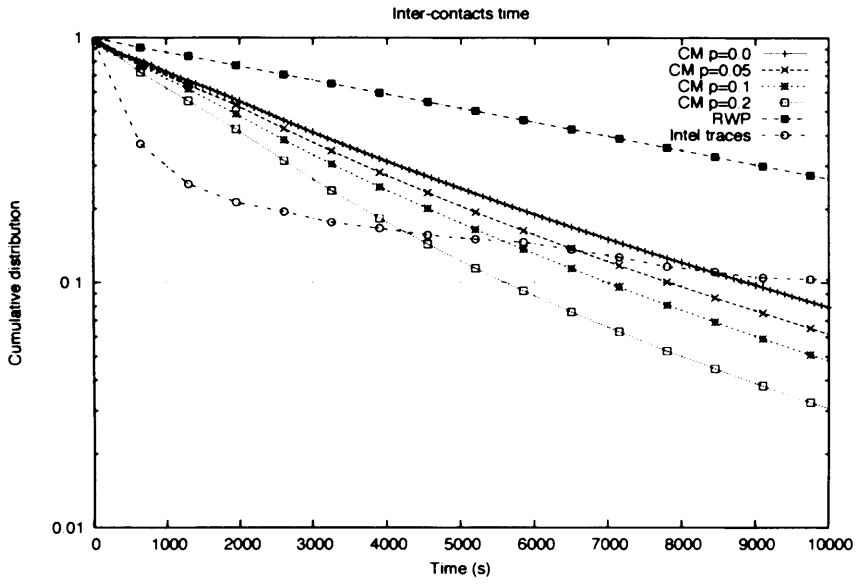

Figure 3.11: Comparison between synthetic and real traces (semi-log coordinates): cumulative distribution of inter-contacts time in seconds with probabilistic selection mechanism.

are 'bridged' by hosts roaming among them. In other words, the movement of most of the hosts is constrained in geographical terms so topology changes are less frequent than in the case of a pure random model. As we observe in Figure 3.20, the difference in terms of performance using the two mobility models is more evident for the DSR protocol.

In case of fixed hosts (i.e., with a speed equal to 0), the delivery ratio that we obtained using our mobility model is lower than in the scenarios with a speed greater than 0, since in the former case, there may be disconnected communities, whereas in the latter, hosts move between communities, providing a link between them.

3.4 Related Work

3.4.1 Mobility Modelling for Ad Hoc Network Research

There are essentially two possible types of mobility patterns that can be used to evaluate mobile ad hoc network protocols and algorithms *in vitro*: traces and synthetic models [Camp et al., 2002]. Traces are a useful method of empirical investigation, but they are not in general available in large numbers for many reasons: retrieving these information is extremely difficult from a technological point of view and expensive, since this process

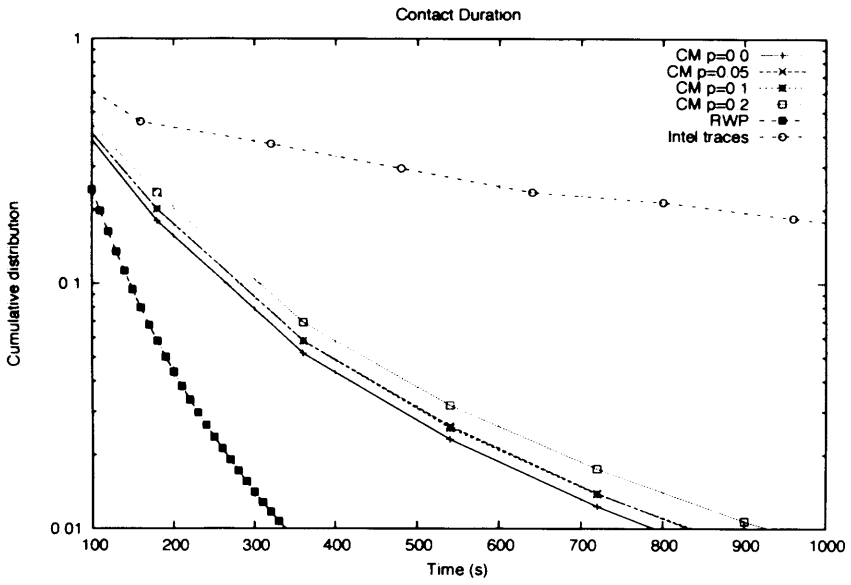

Figure 3.12: Comparison between synthetic and real traces (semi-log coordinates): cumulative distribution of contacts duration in seconds with probabilistic selection mechanism.

implies that the movements of a representative number of users are traced in large areas (indoor and outdoor). Moreover, the collection of mobility data is not possible in certain deployment scenarios such as dangerous environments.

The simplest mobility model is the Random Walk mobility model [Einstein, 1956], also called Brownian motion; it is a widely used model to represent pure random movements of the entities of a system in various disciplines. However, it cannot be considered as a suitable model to simulate wireless environments, since movements are never random in the real world. Another example of random mobility model is the Random Way-Point mobility model [Johnson and Maltz, 1996a] that can be considered an extension of the previous one, with the addition of pauses between changes in direction or speed.

These models and similar existing ones are used to model the movements of single mobile nodes, however, in some situations the behaviour of mobile hosts that move together, such as platoons of soldiers, group of students or colleagues and so on also need to be modelled.

Despite the simple intuitions at the basis of these considerations, it is interesting and, at the same time, surprising to note that even the best solutions and approaches have only been tested using completely random models such as the Random Way-Point model, without grouping mechanisms, or using other simple groups mobility models, like [Hong et al., 1999].

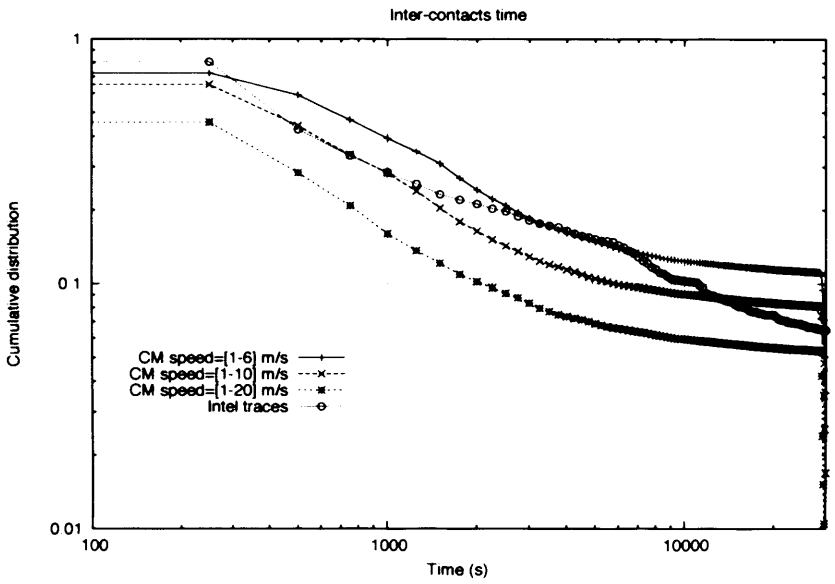

Figure 3.13: Influence of the hosts speed: cumulative distribution of inter-contacts time in seconds.

To our knowledge, the only notable example of mobility model founded on the social relationships between the individuals carrying the mobile devices is presented in [Hermann, 2003]. This work is based on assumptions similar to ours, but is considerably more limited in scope. Hosts are statically assigned to a particular group during the initial configuration process whereas our model accounts for movement between groups. Moreover, the authors claim that mobile ad hoc networks are scale-free, but the typical properties of scale-free networks are not considered in the design of the model presented by the authors. The scale-free distribution of mobile ad hoc networks is still not proven in general, since practical measurements are not currently available and it is worth noting that the scale-free properties are strictly dependent on the movements of hosts and therefore are dependent on the actual application scenarios [Glauche et al., 2003].

In the recent years, many researchers have tried to refine existing models in order to make them more realistic. In [Jardosh et al., 2005], a technique for the creation of more realistic mobility models that include the presence of obstacles is presented. The specification of obstacles is based on the use of Voronoi graphs in order to derive the possible pathways in the simulation space. This approach is orthogonal to ours, and our framework can be modified in order to include the presence of obstacles using similar techniques; only the equations used to update the positions of the entities need changing.

In [Camp et al., 2002] Camp, Boleng and Davids provide an excellent review of the most

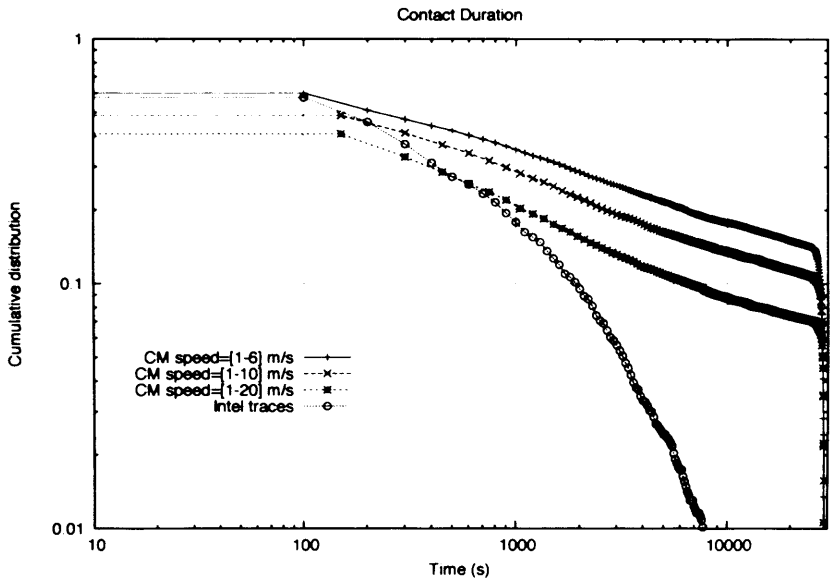

Figure 3.14: Influence of the hosts speed: cumulative distribution of contacts duration in seconds.

relevant and popular mobility models used in ad hoc network research.

In recent years, many researchers have tried to refine existing models in order to make them more realistic by studying the available mobility traces [Kotz and Henderson, 2005]. Various measurement studies have been conducted both in infrastructure-based and infrastructure-less environments. Extensive measurements about the usage of a WLANs have been conducted, for instance, in [Tang and Baker, 2000], in [Balachandran et al., 2002] and in [Balazinska and Castro, 2003]. A detailed analysis of the usage of the WLAN of the Dartmouth Campus College is presented in [Henderson et al., 2004].

In [Tuduce and Gross, 2005] a mobility model based on real data from the campus wireless LAN at ETH in Zurich is presented. The authors use a simulation area divided into squares and derive the probability of transitions between adjacent squares from the data of the access points. Also in this case, the session duration data follows a power law distribution. This approach can be a refined version of the Weighted Way-Point mobility model [Hsu et al., 2005], based on the probability of moving between different areas of a campus using a Markov model. Moreover, Tuduce and Gross' model represents the movements of the devices in an infrastructure-based network and not ad hoc settings. In [Maeda et al., 2005], the authors reproduce the movements of pedestrians in downtown Osaka by analysing the characteristics of the crowd in subsequent instants of time and

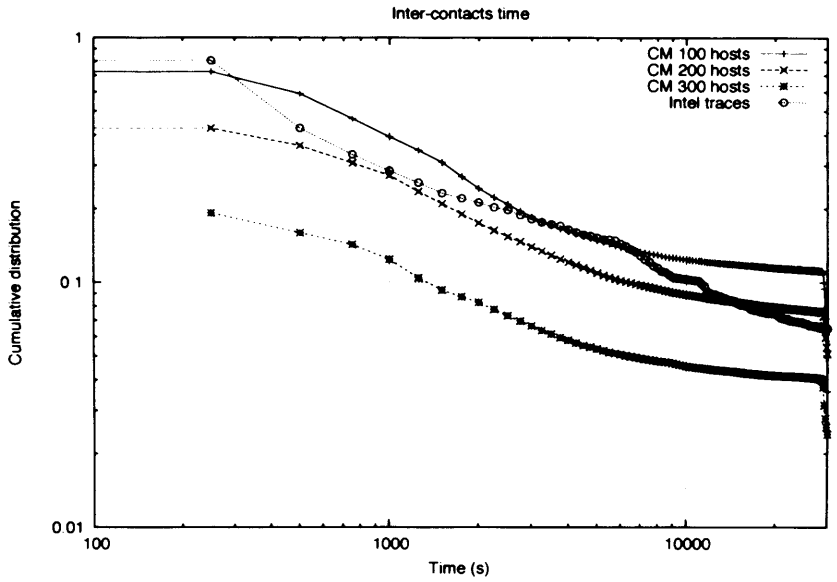

Figure 3.15: Influence of the density of population: cumulative distribution of inter-contacts time in seconds.

maps of the city using an empirical methodology. In general, the main goal of these works is to reproduce the specific scenarios with a high degree of accuracy. We focus, instead, on the cause of these movements, trying to capture the social dimensions that lead to general emergent human movement patterns.

In [Lelescu et al., 2006] the Model T++ is proposed, based on a study on the correlation between the number of sessions per access point and the time spent at each location. The authors also define the concept of popularity gradient between different access points and its influence on users' movement. Similarly to ours, their model is extracted and validated using real traces of movements, more specifically the Dartmouth College campus traces.

In [Yoon et al., 2006] a model extracted from real traces based on the study of probability of transitions between different locations is discussed. Connectivity is one of the emergent properties of the model, rather than an input of the patterns generator. Moreover, the evaluation of the model is essentially based on the matching of the geographical movements and density of users, rather than on the analysis of the patterns of connectivity among them.

In terms of more analytical work, a key study in the field is [Chaintreau et al., 2006], where the authors analyse the distribution of inter-contacts time and the duration of contacts considering different data sets from various measurements exercises. All of these exhibit

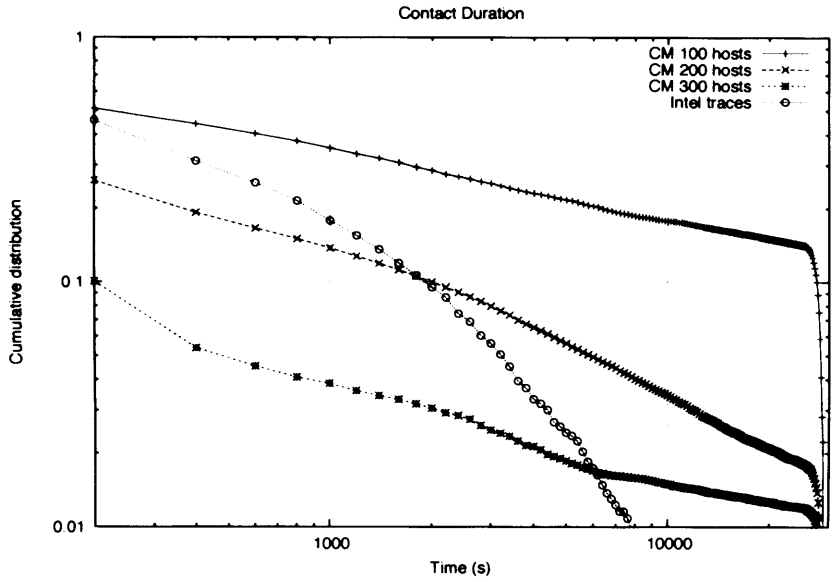

Figure 3.16: Influence of the density of population: cumulative distribution of contacts duration in seconds.

a similar heavy tail distribution that can be approximated using a power-law function over a large range of values. The work confirms the results from other studies conducted at Dartmouth [Henderson et al., 2004], UCSD [McNett and Voelker, 2005] and University of Toronto [Su et al., 2004]. At the same time, it is interesting to observe that these observed patterns are at odds with the ones that can be extracted from random mobility models that show an exponential decay [Sharma and Mazumdar, 2004]. In a previous work [Hui et al., 2005a], similar connectivity patterns have also been observed among the participants of INFOCOM'05.

3.4.2 Social and Complex Networks Analysis

Since mobile ad hoc networks are usually composed of mobile devices carried by human beings. Therefore, we can study the formation and the evolution of these network from social networks analysis point of view.

A social network is a set of people or groups of people with some patterns of contact or interaction among them [Scott, 2000]. Research studies in the area of social networks started in the 1920s [Freeman, 1996]. However, the first significant quantitative results were presented in [Rapoport, 1957] in the 1950s and 1960s in a series of papers in which the statistics of epidemic diffusion in populations characterised by different social structures

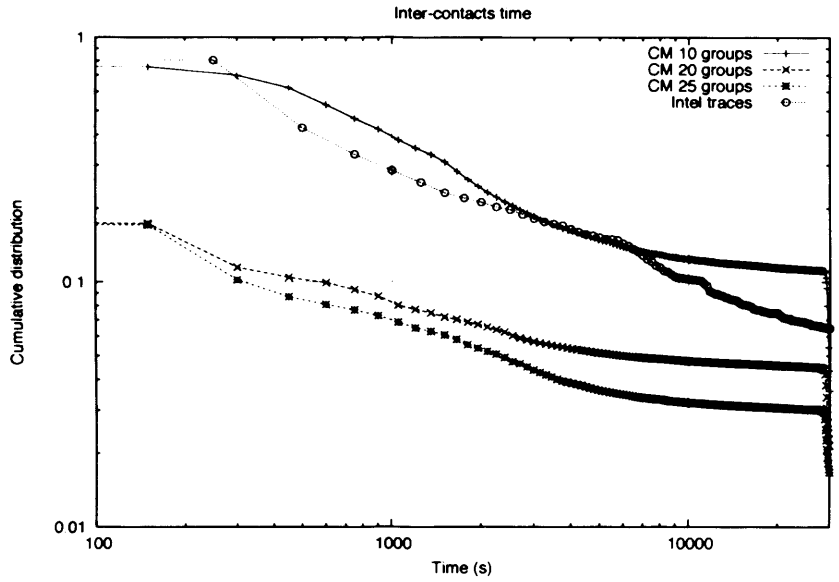

Figure 3.17: Influence of the number of initial number of groups in input to the network generator based on the Caveman model: cumulative distribution of inter-contacts time in seconds.

are analysed. In particular, they focused on the correlation between the average *degree* of the network and the general properties of the investigated environment (such as spread of information and rumours). In social (and in complex) networks theory, the average degree of a network is the average number of acquaintances of the people present in the network. Considering a graphical representation, where the individuals are the vertices and the relationships connecting them are the edges, the average degree of the network is clearly the average number of edges per node.

Another famous example of empirical investigation of social network is the series of experiments carried out by Stanley Milgram [Milgram, 1967] in the late 1960s. He tried to reconstruct a portion of the social network of the population of United States using letters sent to people in Nebraska and Kansas with instructions for the delivery of these to their actual recipients in Massachusetts, with the constraint that the letters could only be sent to people that were personal acquaintances and not directly to the recipients (if not known personally). The intermediate receivers had to record their personal details on the letters. Using this empirical method, Milgram traced the chain of people between the senders and the receivers. He claimed that, on average, five intermediates were necessary to deliver the letter, leading to the popular concept of the *six degrees of separation* that was coined decades after by the playwriter John Guare [Guare, 1990].

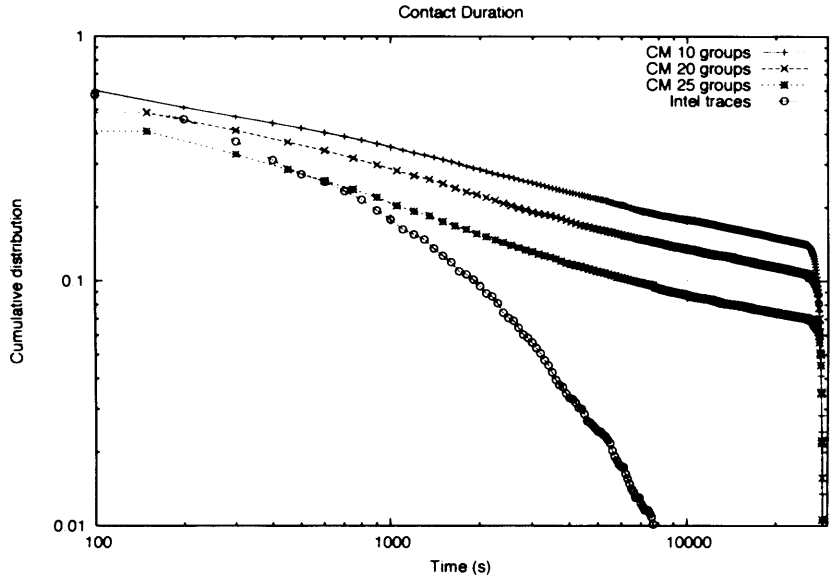

Figure 3.18: Influence of the number of initial number of groups in input to the network generator based on the Caveman model: cumulative distribution of contacts duration in seconds.

Whilst this was pioneering exploratory work, it was not rigourous from a scientific point of view. However, in that period, a renewed interest in graph theory led to the definition of the so-called random graphs [Erdos and Renyi, 1959, Erdos and Renyi, 1960]. These theoretical studies were applied to many disciplines including, biology, ecology, linguistics and sociology. This was the beginning of the complex networks research area, investigating properties such as their topology, average diameter and degree of connectivity, as well as the presence of clusters. Many analytical models of real networks have been proposed and interesting experiments have been conducted to assess the validity of the proposed theories. For example, the so-called scale-free model was used to represent and predict some properties of very large networks of computers, such as the Internet, with surprising precision [Vasquez et al., 2003].

With respect to the application of complex network theory to the analysis of social networks, one of the most interesting approaches is the so-called small world model [Watts, 1999], that is, in a sense, a formalisation of the results discovered empirically by Milgram. Watts and Strogatz [Watts, 1999] study, in particular, the characteristic path length[11] of real networks.

[11]The characteristic path length (L) of a graph G is the median of the means of the shortest path lengths connecting each vertex $v \in V(G)$ to all other vertices.

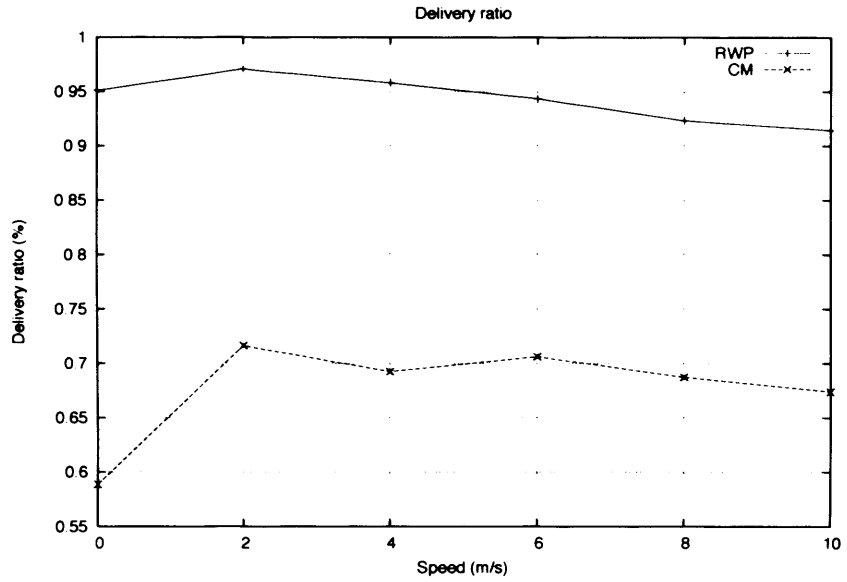

Figure 3.19: Influence of the mobility model on the AODV protocol performance (delivery ratio vs speed).

In the recent years, various types of social networks have been investigated by many researchers in order to verify the properties predicted by the theoretical models; example of social networks used for these studies are rather diverse and include the actors in films with Kevin Bacon [Watts, 1999], the characters of Anna Karenina by Tolstoy [Knuth, 1993] and the Marvel superheros [Alberich et al., 2002].

Excellent reviews of the recent progress in complex and social networks analysis may be found in [Albert and Barabasi, 2002] and [Newman, 2003].

Some interesting studies have been recently carried out on the connectivity of ad hoc networks with respect to complex networks theory. For example, Glauche et al. in [Glauche et al., 2003] discuss some network properties using percolation theory [Stauffer and Aharony, 1992]; that is, an application of complex networks theory derived by the investigation of physical phenomena such as phase transitions in molecular lattices. In [Sarshar and Chowdhury, 2003], the authors presents mathematical results about the possible emergence of scale-free structures in ad hoc networks. However, the authors consider only *fixed* ad hoc networks (such as peer-to-peer networks), without analysing the influence of movement in the definition of their model.

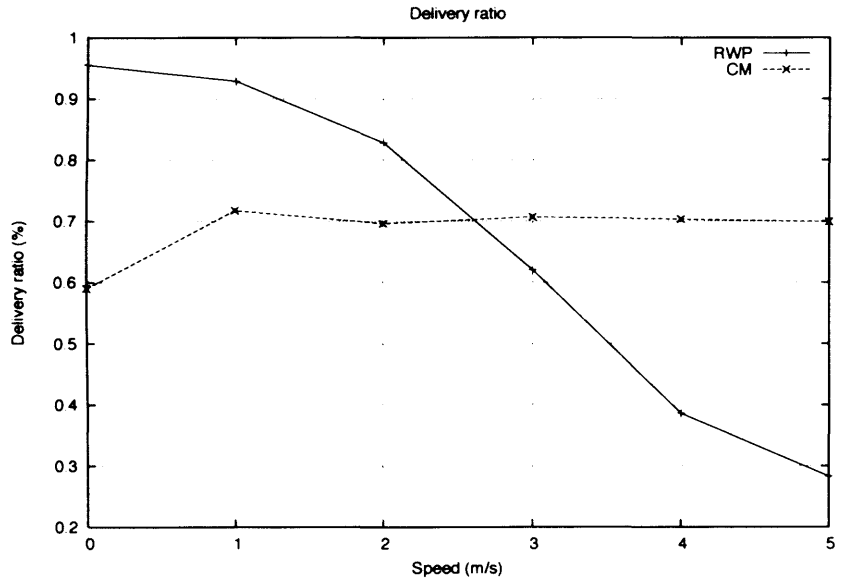

Figure 3.20: Influence of the mobility model on the DSR protocol performance (delivery ratio vs speed).

Chapter 4

Evaluation

> It's clearly a budget. It's got a lot of numbers in it.
>
> *George W. Bush*

4.1 Description of the Simulations

In this section, we present the performance evaluation of CAR, discussing the choice of the parameters of the forwarding algorithm by means of an extensive sensitivity analysis.

One of the classic problems of network research is the evaluation of algorithms and protocols in large-scale scenarios due to their complexity in terms of number of entities and interactions. Deployments of large number of nodes are in general not feasible and multiple experiments aimed at sensitivity analysis of the parameters that characterise and influence the performance of the protocol are quite hard to carry out. We have decided to evaluate CAR through simulation in terms of its large-scale performance. In Chapter 5 we also show how we have implemented the protocol into a middleware framework and tested some functional performance as well.

In the following, we firstly describe the simulation settings and then we discuss the results that we have obtained.

4.1.1 Preliminary Considerations

The first step of any simulation exercise is the choice of an appropriate tool to support the work. Various network simulators are available for the evaluation of protocols and systems of mobile ad hoc networks; the most popular are ns-2 [ns2, 2006] with the so-called Monarch extension [Johnson and Maltz, 1996b], Glomosim [Zeng et al., 1998] and Opnet [OPNET Technologies Inc., 2004]. Another class of tools for simulation of generic complex systems (not only computer systems, but also economic, biological, industrial, etc.) are the so-called *discrete-event simulators*[1] that only provide primitives for the concurrent execution of multiple entities and communication among them usually by means of message passing based paradigms.

We have chosen OMNet++ [Vargas, 2001], an open-source modular discrete-event simulation environment. The reasons behind our choice are several. First of all, it provides a generic and flexible architecture for experiments. Secondly, many plug-ins and extensions, for example, for the statistical analysis of data are available. Thirdly, OMNeT++ is free for academic use: this is extremely important in order to allow all the researchers interested in our work to repeat the experiments.

OMNeT++ does not provide any real support for networking simulation, but only a minimal set of primitives to schedule events and message passing primitives between entities. For this reason, we built the abstraction of movement, space and connectivity.

We abstract the mobile scenario at network level: in fact, the aspects that are of our interest are the colocation and connectivity of the hosts. We do not consider issues related to radio and MAC layers such as interference or packet loss, since these are secondary aspects of our problem. The fact of being in reach is the primary element of our study. We assume that the transmission of messages may happen and be completed when two hosts are in radio range. We do not model retransmission of packets.

In our simulations, we use values for the parameters generally adopted in the literature[2];

[1] It is worth noting that the results of simulations performed with different simulators may show significant divergence; this may be explained by the different modelling techniques and assumptions and by the different levels of details offered by these simulators. In [Cavin et al., 2002] the authors show and discuss the divergent results obtained by using OPNET Modeler, ns-2 and Glomosim. In general, the choice of a more sophisticated simulation tool does not increase the accuracy of the simulation results automatically.

[2] Unfortunately, the choice of values for parameters of simulations for ad hoc networks research is extremely variable. In fact, the ad hoc and delay tolerant research communities lack of consistent scenarios to validate and to benchmark the different solutions. For example, in [Kurkowski et al., 2005] Kurkoswski, Camp and Colagrosso reported an analysis of the performance evaluation of papers published at MobiHoc from 2000 to 2005, showing evident flaws of a large number of works from a scientific point of view in terms of simulation methodology. We would like to underline that in this work, we have tried to address the shortfalls that are usually pointed out by the members of the community, such as the problem of the repeatability of the experiments (the code of our simulations will be released for comparisons), the exclusive use of simulations for validation without any implementation (see instead the integration of CAR in Haggle described in Chapter 5), the use of a sound number of runs of experiments to ensure statistical

in terms of mobile scenarios, we reproduce possible realistic deployments of our protocol in small and medium size areas.

4.1.2 Simulation Scenarios Parameters

We considered two simulation scenarios composed of 50 and 100 nodes in a 1 km $\times$ 1 km and 2 km $\times$ 2 km areas, respectively. The first one identifies a small and dense scenario whereas the second is characterised by a lower degree of connectivity. The resulting topology of the latter is also clearly more sparse. We note that the results related to these two specific scenarios have not general validity from an absolute quantitative point of view, but they provide information about the relative performance of the protocols, the effectiveness of the prediction mechanisms and the limitations of CAR. We also believe that the results obtained in our simulations help to identify the key parameters of the protocol and their impact on its performance. Moreover, the evaluation methodology followed in this chapter is very general and could be adopted to tune the parameters of the protocol in other scenarios (for example, characterised by different node density or speed, etc.).

The movements of the simulated hosts were generated using the Community based mobility model presented in Chapter 3. The speed of the nodes was uniformly distributed in the range $[1 - 6]$ m/s. The size of the square sides was set to 200 m for both scenarios. The underlying social network was generated using 5 and 10 communities for the 50 and 100 hosts scenario respectively. The rewiring probability was set to 0.1. The selection mechanism that we adopted was the probabilistic one with a damping factor equal to 0.01 (i.e., the probability of moving towards a square without hosts is very low).

We also ran simulations using the Random Way-Point model in the same simulation scenarios, with host speeds in the range $[1 - 6]$ m/s and stop times equal to 0 seconds.

In the remainder of this chapter, we assume that the movement of hosts is based on the Community based mobility model if not otherwise stated (i.e. the expression n *hosts scenario* refers to a scenario composed of n hosts moving according to our Community based mobility model).

With respect to the radio technology, we assumed a free space propagation model with 200 m range and the use of omnidirectional antennas.

We evaluated the performance of each protocol by sending 1000 messages with a simulation time equal to 2400 seconds for the 50 hosts scenario and 4400 for the 100 hosts scenario. The messages sent have an expiration time of 2000 seconds for the 50 hosts scenario and

validity to the results, the definition of confidence intervals and a thorough sensitivity analysis, considering a large number of dimensions of the problem.

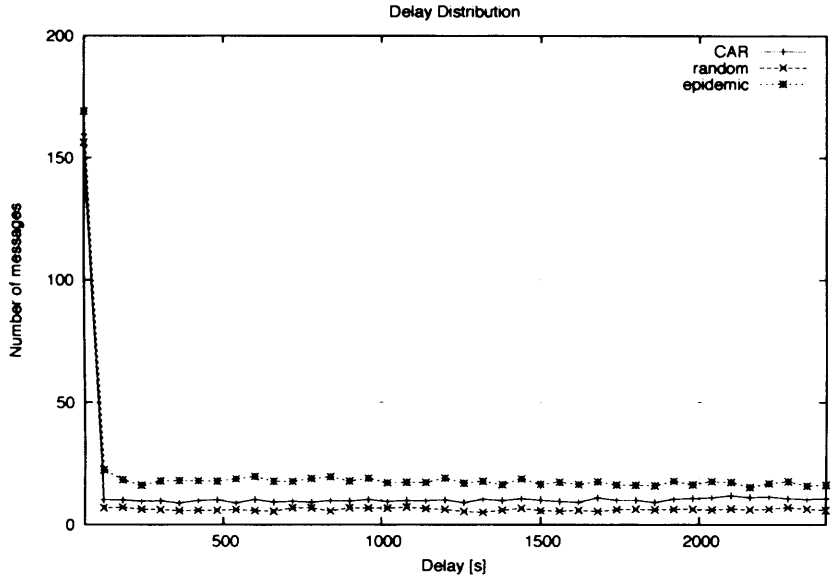

Figure 4.1: Delivery delay distribution (scenario with 50 hosts and Community based mobility model).

4000 seconds for the 100 hosts scenarios.

The message buffer size was set to 1000 slots (i.e., infinite), except for the simulations related to the study of the impact of the memory size. We assume that each host is able to store a certain number of messages, one per slot. The messages were sent after 300 seconds, in order to allow for the convergence of the routing tables after the initial exchanges, the intervals between each message were modelled as a Poisson process, with an average interval between the generation of two subsequent messages equal to 0.1 seconds. In other words, all the messages are generated in the first 400 seconds.

The sender and receiver of each message were chosen randomly. This choice is clearly unrealistic. However, we believe that this is a sort of pessimistic case scenario, where communication happens between any nodes in the network and not only between people with strong social ties[3]. In fact, if communication happens between people that are members of the same community, the probability of direct contact is higher and, in general, the number of potential carriers moving between them is also higher in average. This is a characteristic of real life settings that is reproduced by our Community based mobility model.

[3]We plan to investigate the problem of designing realistic traffic models based on the same principles on which the mobility model is predicated in the future. The main problem of designing such a model is the lack of real data for validating it, especially for delay tolerant scenarios.

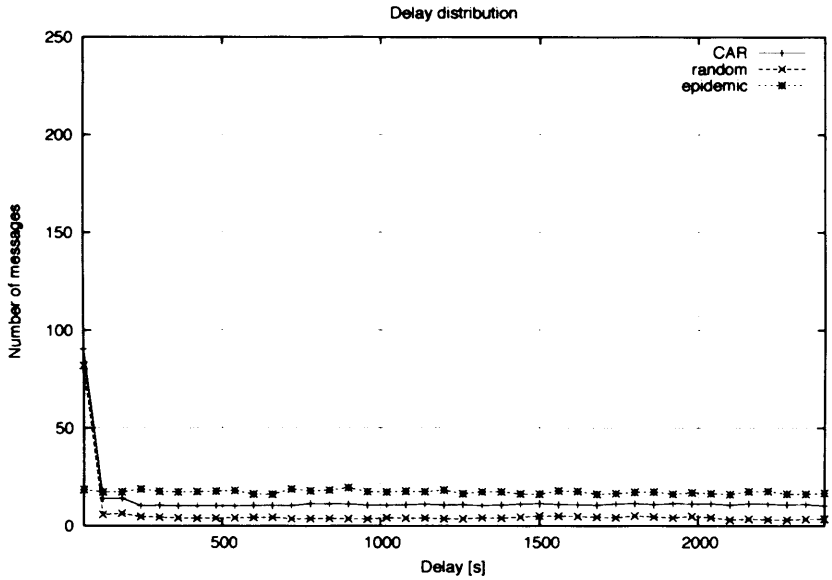

Figure 4.2: Delivery delay distribution (scenario with 100 hosts and Community based mobility model).

The number of runs for each particular configuration was set to 25 for the Community based mobility model scenarios and 50 for the Random Way-Point scenarios (that present an inherent higher variance of the results). The diagrams in this section show error bars corresponding to 5% confidence intervals.

4.1.3 Choice of Parameters of CAR

We implemented all the features of CAR described in Chapter 2 in the simulation code. We simulated CAR using a utility function based on the evaluation of two attributes: (i) the change rate of connectivity and (ii) the probability of being located in the same cloud as the destination.

We assumed that all the possible values in the range had the same importance (i.e, $a_{range_i} = 1$) and that the values of attributes are always available during the simulation (i.e., $a_{availability_i} = 1$). The values of the weights associated to the change degree of connectivity (w_{cdc_h}) and colocation ($w_{col_{h,i}}$) for all the pair of hosts (h, i) are set to 0.25 and 0.75, respectively. These values ensure the best performance in terms of delivery ratio in the scenario based on the Community based mobility model, as we will show in Section 4.2.4.

Each message has a *time to live* field that is decreased each time a message is transferred

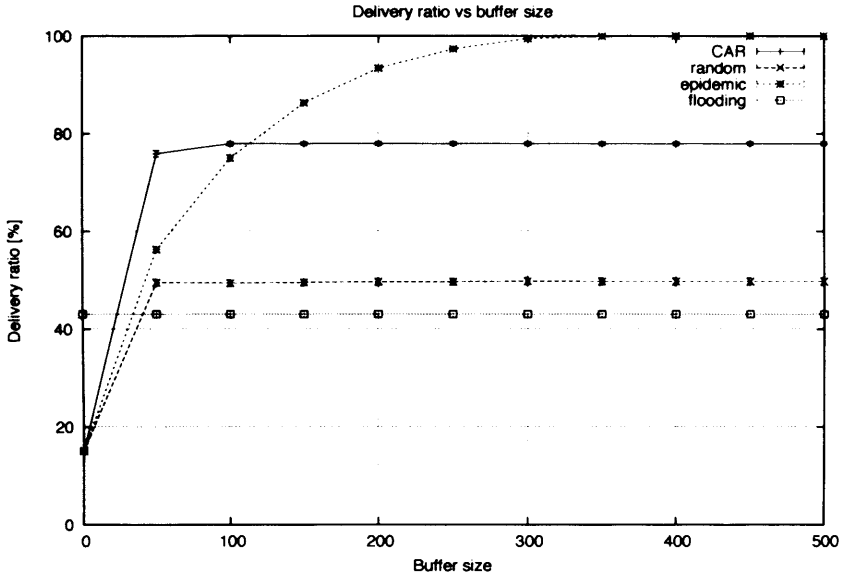

Figure 4.3: Delivery ratio vs buffer size (scenario with 50 hosts and Community based mobility model).

to another host (the initial value being 10). Moreover, in this case, we also introduced a *split horizon* mechanism to prevent messages from being retransmitted unnecessarily (i.e., hosts do not advertise a route back to the node from which it was learned). The buffer for each node was set to 100 messages, unless otherwise specified.

The number of retransmissions (i.e., the number of times a message is forwarded) for the 50 hosts scenario was set to 10; instead, for the 100 hosts this was set to 20. The values of the message retransmission and the routing table transmission intervals were set to 30 s. The local utilities and the routing tables are updated every 30 s. The routing table size was set to 20 and 40 hosts for the 50 and 100 hosts scenarios respectively (i.e., it is equal to 40% of the number of the hosts and sufficient to store information about all the hosts of two initial communities). This limited size of the routing table is used to study the replacement mechanisms in the buffer and to reproduce possible limitations in terms of memory of small devices.

We set the values of the variances of the White Noise R_t of Y_t and Q_t of the White Noise of X_t to 0.1 and 0.01, respectively (see 2.5.2). This choice is motivated by the fact that the values of the observations and the states of the model are in the range $[0, 1]$. The value of R_t corresponds to 10% of the range. There is no univocal and standard way of setting and tuning the parameters of the filter [Durbin and Koopman, 2001]. The values that we selected are appropriate and general enough for the range of inputs (and its variations)

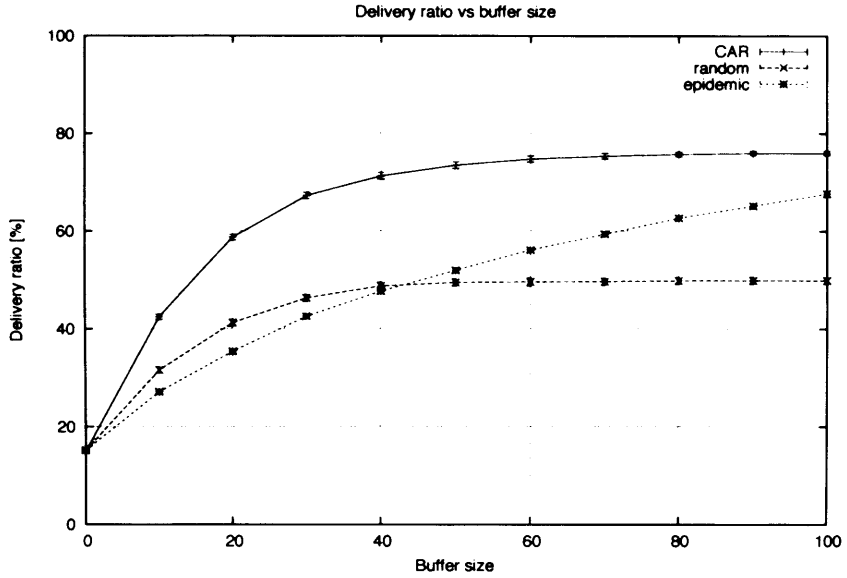

Figure 4.4: Delivery ratio vs buffer size – analysis with very small buffer (scenario with 50 hosts and Community based mobility model).

that we are considering.

Table 4.1 summarises the simulation parameters that are common to all the simulations, unless otherwise specified. Tables 4.2 and 4.3 report the values of the parameters for the evaluation of the specific scenarios generated using the Community based mobility model composed of 50 and 100 hosts, respectively.

4.1.4 Description of the Protocols Used for Performance Comparison

In order to have benchmarks to evaluate the performance of CAR, we also implemented the flooding routing protocol, an Epidemic one (according to the Vahdat and Becker definition) and a version of CAR where the selection of the best carrier is not based on the delivery probabilities but on a random choice. We will refer to the latter as *Random Choice* protocol.

We now briefly describe the algorithms and the parameters used in the simulations of these protocols.

Table 4.1: Simulation parameters for all the scenarios

Propagation model	free space
Antenna type	omnidirectional
Transmission range	200 m
Number of messages sent	1000
Message buffer size	1000
Max number of hops DSDV	10
w_{cdc_h}	0.25
$w_{col_{h,i}}$	0.75
Message retransmission interval	30 s
Routing table transmission interval	30 s
Refresh of the routing table interval	30 s
a_{range_i}	1
$a_{availability_i}$	1
R_t	0.1
Q_t	0.01

Table 4.2: Simulation parameters specific for the 50 hosts scenario

Simulation area	1 Km × 1 Km
Number of communities	50
Square area	200 m × 200 m
Node speed	1-6 m/s
Max number of retransmissions	10
Routing table size	20 entries

Table 4.3: Simulation parameters specific for the 100 hosts scenario

Simulation area	2 Km × 2 Km
Number of communities	5
Square area	200 m × 200 m
Node speed	1-6 m/s
Max number of retransmissions	20
Routing table size	40 entries

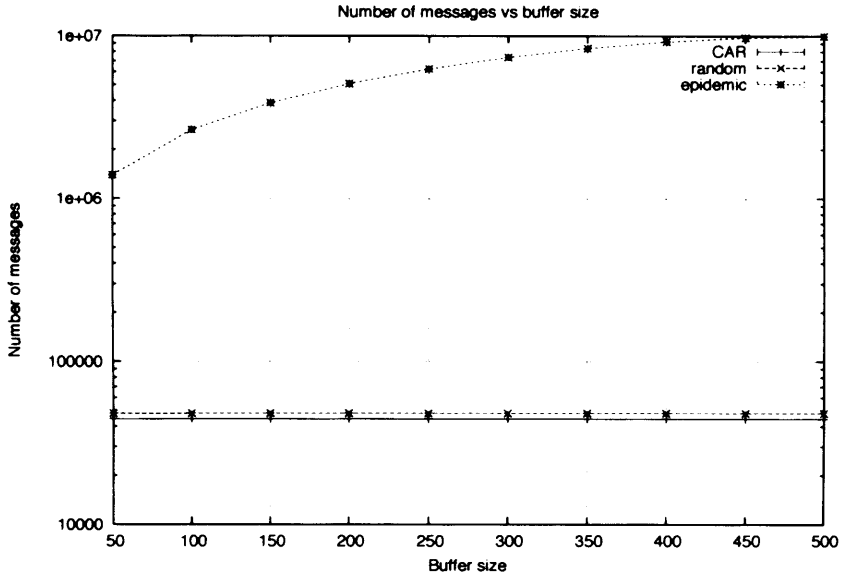

Figure 4.5: Number of messages vs buffer size (scenario with 50 hosts and Community based mobility model).

Flooding We elected to compare our approach with flooding. This decision may seem strange, since flooding only works in a fully connected environment. However, since communications patterns are random in the simulations, many messages will be passed between hosts that are in connected portions of the network, even when assessing the performance of the epidemic algorithm and of the CAR algorithm. In order to see the difference in delivery rates that result from the algorithms' ability to handle partial connectivity, it is therefore essential to compare against a synchronous protocol with optimum delivery ratio.

Epidemic Routing The implementation of the epidemic protocol follows the description presented in [Vahdat and Becker, 2000]. The only assumption made by the authors is a periodic pair-wise connectivity, since the protocol relies on the transitive distribution of messages for delivery. When two hosts become neighbours (in other words, they are within each other's radio range), they determine which messages each possesses that the other does not, using summary vectors that index the list of messages stored at each node; they then exchange them. Each message is characterised by a unique message identifier. The simulation prepared by the authors shows that the algorithm achieves total delivery of messages sent after a limited period of time, but at the cost of very substantial overheads.

The epidemic approach represents an example of an asynchronous protocol and, in partic-

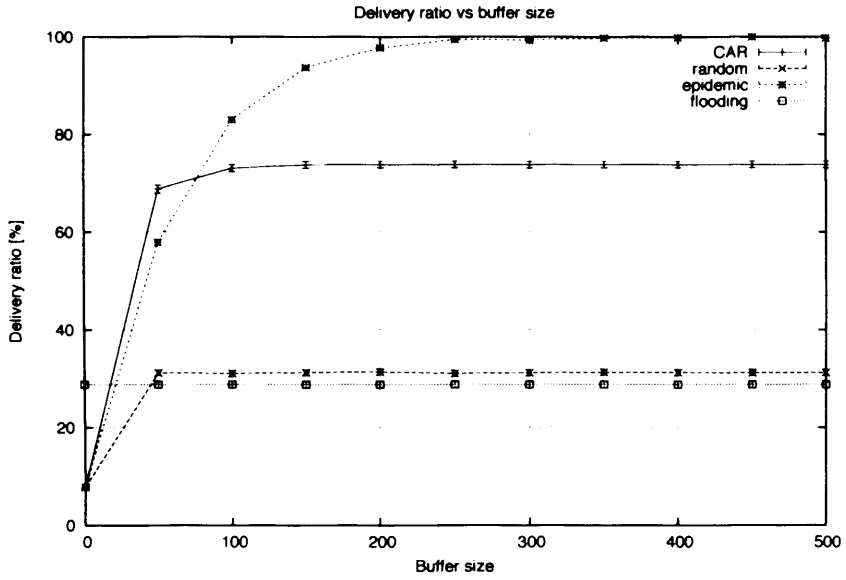

Figure 4.6: Delivery ratio vs buffer size (scenario with 100 hosts and Community based mobility model).

ular, it provides the theoretical upper bound in terms of delivery ratio with infinite buffer. In this case, given a probability different from zero of having a contact between any pair of hosts in a non infinite period of time T, the protocol is able to reach 100% delivery. The retransmission interval of the epidemic routing was set to 30 seconds (like in CAR).

Random Choice We implemented the Random Choice protocol to compare the performance of the prediction based best carrier selection mechanism in CAR. This is a modified implementation of CAR where the selection of the carriers is done randomly instead of choosing the host with the highest delivery probability in the connected portion of the network. All the routing mechanisms of the CAR protocol are implemented (routing table management and updates, synchronous routing, etc.), *except* the selection of the best carrier based on the utility functions.

More precisely, a message is periodically retransmitted to a host in the same cloud or it is maintained in the buffer according to a uniform distribution: every host has the same probability of being selected. The delivery to the hosts in the same connected portion of the network is based on DSDV, like for CAR. The management of the routing table is identical to CAR and we use the same retransmission interval of CAR, since we want to isolate the effect of the utility-based selection mechanism.

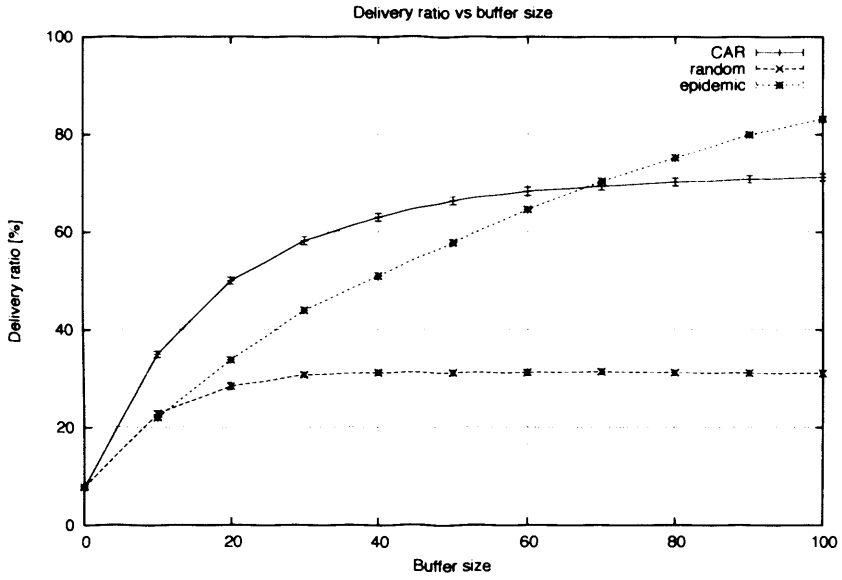

Figure 4.7: Delivery ratio vs buffer size – analysis with very small buffer (scenario with 100 hosts and Community based mobility model).

4.2 Simulation Results

In this section, we present several simulations results that describe the performance of the protocol in large-scale scenarios.

4.2.1 Evaluation Metrics

The metrics used in this evaluation are defined as follows:

- **Delivery delay** The delay is calculated as the time between the generation of the message and the delivery to the final recipient of the message.
- **Delivery ratio** The delivery ratio is given by the ratio between the number of messages received and the total number of messages sent.
- **Number of messages** The number of messages indicated in the simulation results include data and control messages.
- **Predictability level** As discussed in Section 2.6.2, given a certain number of measurements of the predictability of a time series, we define *predictability level* of a

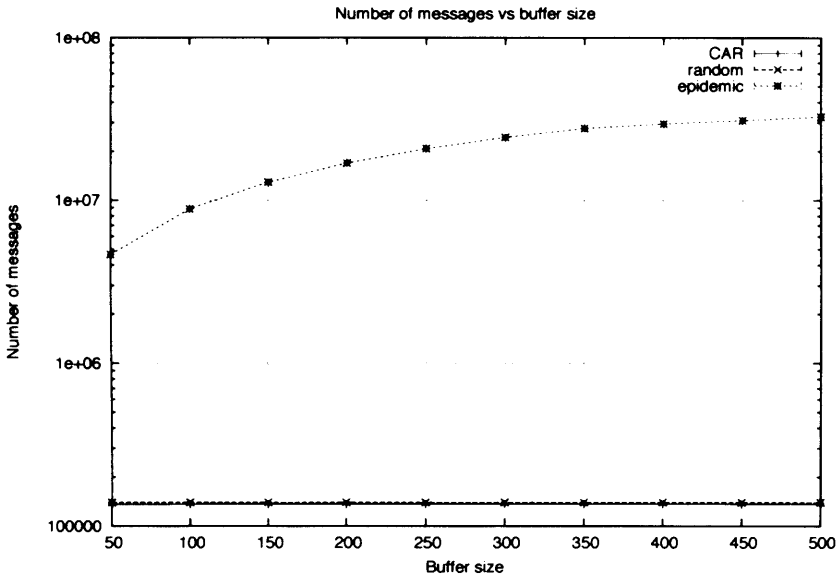

Figure 4.8: Number of messages vs buffer size (scenario with 100 hosts and Community based mobility model).

context attribute as the percentage of samples for which the component returns true, in other words, the percentage of samples for which the prediction is sufficiently accurate given a predefined acceptable error. We consider a prediction error of 0.15 as the maximum acceptable error in the current evaluation. This value was chosen also considering the value of the variance of the White Noise R_t of the observation in the prediction model is 0.1.

4.2.2 Delay Distribution

The first interesting aspect that we analyse is the distribution of the delivery delays, a characterising aspect of a protocol for delay tolerant mobile networks.

Figures 4.1 and 4.2 show the delay distribution measured in the 50 and 100 hosts scenarios respectively. In these simulations, we consider an infinite buffer size. As expected, a percentage of messages are delivered immediately, since the recipients are in the same connected portion of the network of the hosts when the message is sent (and the sender has a routing table entry related to the recipient of the message for synchronous delivery), whereas the majority of the messages are delivered with a variable, possibly long, delay. The amount of messages delivered in the time interval considered slowly decreases as

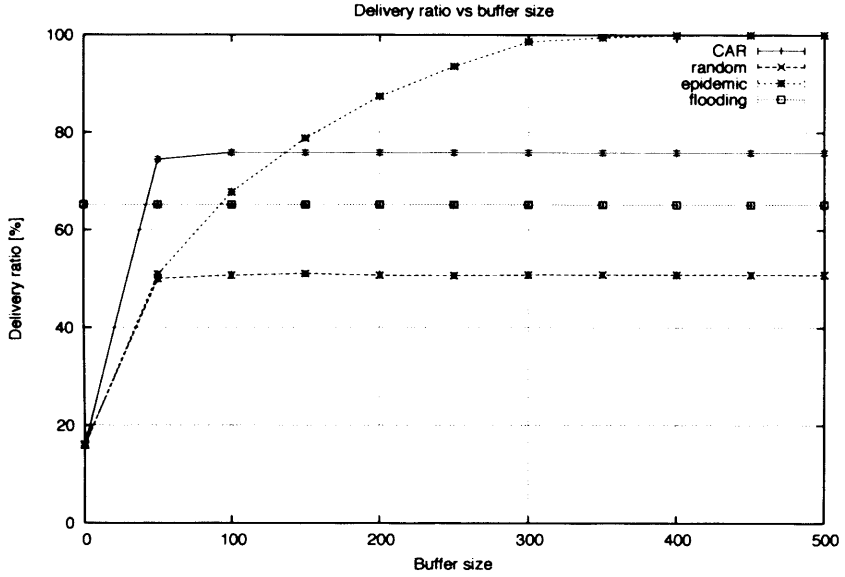

Figure 4.9: Delivery ratio vs buffer size (scenario with 50 hosts and Random Way-Point model).

time passes. This is due to the fact that, in the figures, the average number of messages delivered per time slot is shown. In a single simulation, messages are delivered in particular instants of time and, in some cases, in batches, when carriers reach the communities of recipients. The interval between these events is determined by the distance between the communities, which are randomly placed in the simulation space, and by the speed of the carriers. By averaging the results of different runs, characterised by different placements of the communities, speed of the hosts and underlying social networks, we obtained the "flat" curve showed in Figures 4.1 and 4.2.

We assume a retransmission delay equal to 0.001 seconds. The distribution delay of the flooding protocol is not reported given the scale of the graph since it is lower than 10 milliseconds.

We note that the number of messages delivered by the epidemic protocol synchronously is lower than the other two, since the message is replicated to the neighbouring nodes at each replication step that is equal to the retransmission interval in CAR and in the Random Choice protocols. In other words, the synchronous delivery mechanism in CAR allows for a more efficient routing in presence of connectivity.

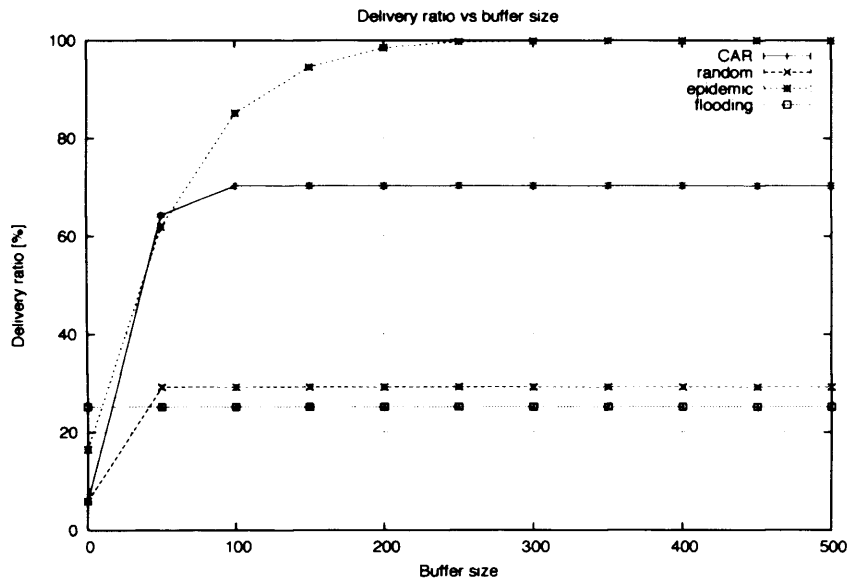

Figure 4.10: Delivery ratio vs buffer size (scenario with 100 hosts and Random Way-Point model).

4.2.3 Influence of the Buffer Size

Another critical aspect we investigated is the size of the buffers of the hosts. Figures 4.3 and 4.6 show the impact of the buffer size on the delivery ratio. We also report a more detailed curve for values of buffer size in the range $[0, 100]$ to show the impact on the performance with very small buffers in Figure 4.4 and 4.7.

First of all, we note that the performance of the flooding protocol is strictly correlated to the connectedness of the graph. In particular, the value of its delivery ratio also gives an estimation of the number of hosts in the same connected component of the resulting instantaneous network graph. This value is around 40% for the 50 scenarios and around 25% for the 100 scenarios. It is worth noting that, given the delays in the routing table updates and the routing table size limitation, the fact that the sender and the receiver are in the same connected component does not imply that CAR is able to support communication between the two, since the former may not store the entry related to that recipient.

As expected, the epidemic protocol is able to deliver all the messages if the buffer size is large enough to avoid the deletion of certain messages. The epidemic protocol reaches 100% with a buffer size greater than 300 and 250 for the 50 and 100 scenarios respectively.

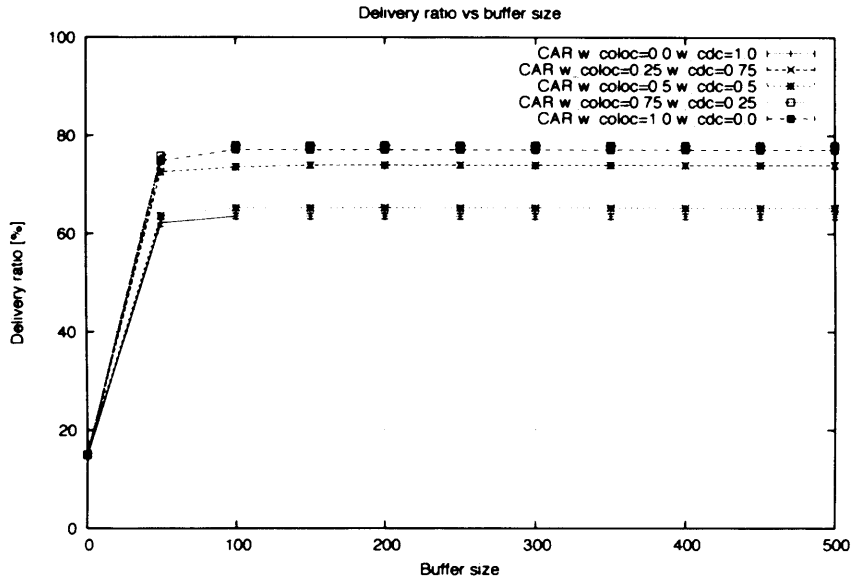

Figure 4.11: Influence of the choice of the weights of the utility function on the delivery ratio (scenario with 50 hosts and Community based mobility model).

This buffer size is sufficient to allows recipients to receive a replica of the messages.

Let us consider the scenario composed of 50 hosts in Figure 4.3, where CAR outperforms the Random Choice considerably. Quite interestingly, CAR shows delivery ratio higher than the epidemic protocol for buffer size equal or smaller than 70. In fact, with small buffers, the epidemic protocol shows its evident limitations due to the replication mechanism. We also note an asymptotic behaviour of the curves related to CAR and Random Choice for buffer size greater than 70 and 35, respectively. The protocols reaches its best performance, given a buffer size sufficient to contain all the messages. The fact that CAR does not reach 100% is due to the number of contacts opportunities in the given amount of time, to the limited number of retransmissions and also to the limitations of the prediction model. In fact, the predictability level for this scenario is about 85%. Moreover, as far as the CAR protocol is concerned, by analysing Figure 4.4, we also deduce that some hosts will carry up to 70 messages. These hosts are the ones characterised by high mobility between different communities. From the point of view of the social network in input, these hosts are characterised by strong links with multiple communities.

Let us now observe the results related to the 100 hosts scenario reported in Figure 4.6. Also in this case, the performance of the CAR protocol shows the effectiveness of the prediction based forwarding mechanism. The gap between the curves describing the performance of CAR and the Random Choice model is larger than in the previous scenario. This is due to

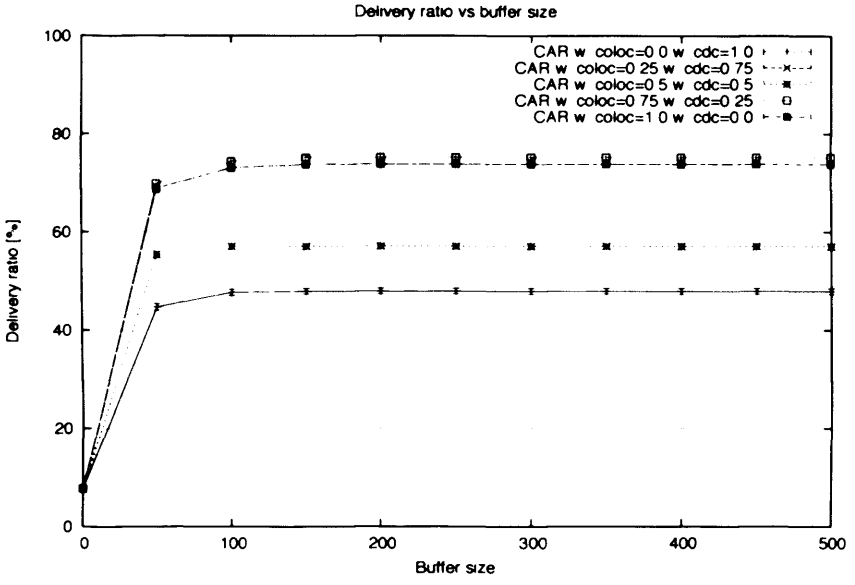

Figure 4.12: Influence of the choice of the weights of the utility function on the delivery ratio (scenario with 100 hosts and Community based mobility model).

the fact that in a more dense scenario, like the one composed of 50 hosts, the probability of getting in reach of the recipient by chance is higher. It is interesting to note that CAR is able to reach good performance also in this scenario. The performance are slightly worse than in the previous one, as in more sparse networks the probability of being in reach of a connected path is lower. The predictability level of this scenario is about 82%, a value close to the one measured for the 50 hosts scenario. In Figure 4.7 we can observe an asymptotic behaviour for values of the buffer size higher than 70 and 30 for CAR and the Random Choice protocol respectively.

We also ran simulations with infinite routing table size (i.e., equal to 50 and 100 entries): CAR is able to reach a delivery ratio closed to 80% and 75% (an improvement of about 4 %). In other words, it seems that removing routing table limitations does not have a strong impact on the performance of the protocols in the scenarios that we considered. This is explained by the fact that the nodes are moving between a limited number of communities. We observe again that in the case of the scenario composed of 50 hosts, a buffer size of 20 hosts allows for storing information about all the hosts of two initial communities of 10 nodes. Instead, with half size routing tables (i.e., 10 and 20 entries) we observe a 18% and 24% reduction of the delivery ratio (with a standard deviation of 1%). The reduction is more evident for the 100 hosts scenario, where the network topology is more sparse.

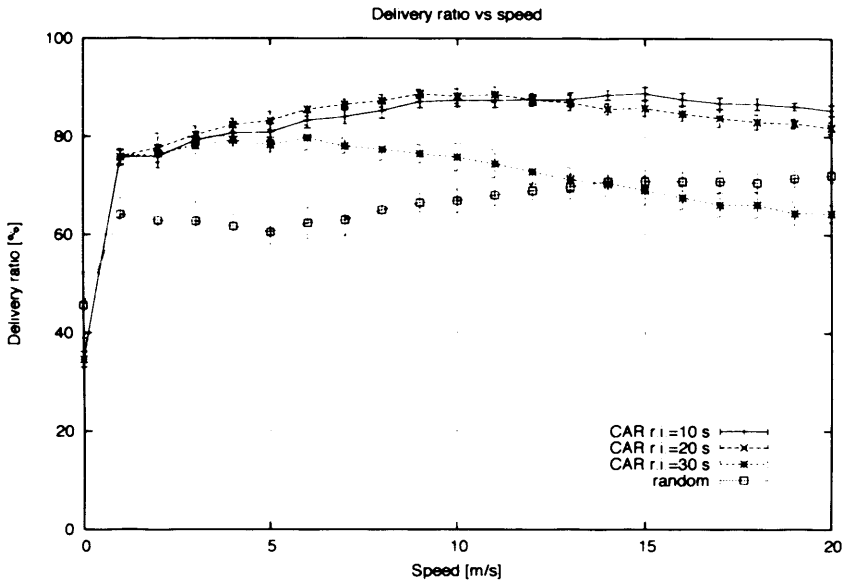

Figure 4.13: Influence of the host speed on the delivery ratio (scenario with 50 hosts and Community based mobility model).

Another interesting aspect is the overhead in terms of number of messages. The results for the 50 hosts scenario and 100 hosts scenario are shown in Figures 4.5 and 4.8 respectively. CAR shows the best performance in this case, also with respect to the Random Choice protocol. In fact, the number of forwarded messages by CAR is lower. The overhead in terms of control traffic (i.e., transmission of routing tables) is the same for both protocols (in fact, the Random Choice protocol exploits DSDV information like CAR).

Figures 4.9 and 4.10 we show the performance of CAR, Random Choice and the Epidemic protocol using the Random Way-Point model. Quite interestingly, the performance of CAR is still good. We observe, in general, that the connectivity of the two resulting graphs is higher due to the distribution of the hosts in the simulation space (instead of the clustered one resulting from the Community based mobility model). More specifically, the good performance is mainly related to the fact that CAR is able to select the hosts with the highest mobility (i.e., highest change degree of connectivity); in the composition of the utility function, the part related to the colocation attribute is very similar for all the hosts and can be modelled as a sort of random noise. Instead, the utility function associated to the change degree of connectivity is clearly higher for highly mobile hosts and, therefore, these are selected as carriers. As expected, the best combination of the weights for this scenario is $w_{cdc_h} = 1.0$ and $w_{col_{h,i}} = 0.0$, since the latter nullifies the inaccurate contribution related to the host colocation.

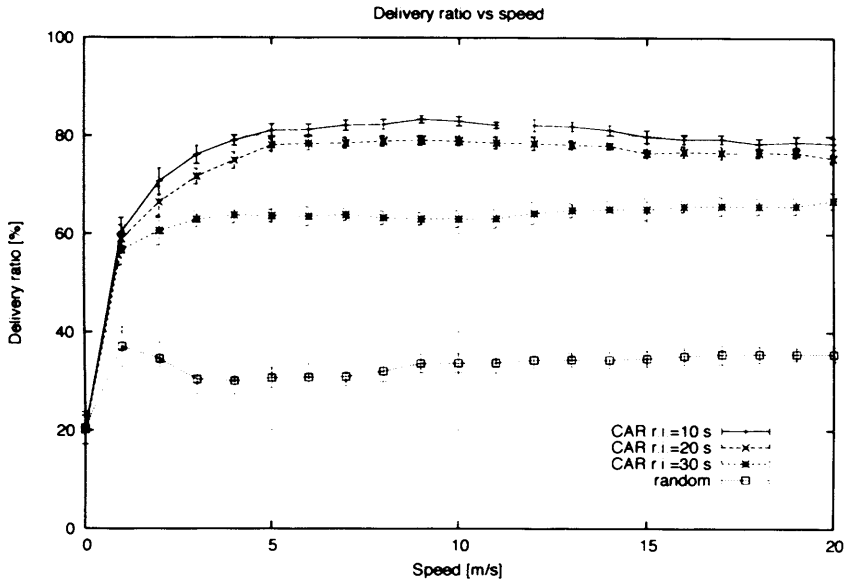

Figure 4.14: Influence of the host speed on the delivery ratio (scenario with 100 hosts and Community based mobility model).

To summarise, these experiments show that CAR is able to guarantee good performance also in presence of small buffers with a limited overhead in terms of number of messages sent, in comparison to the other protocols taken into consideration, thanks to the effectiveness of the utility based prediction algorithm.

4.2.4 Influence of the Choice of the Values of the Weights of the Utility Function

We now analyse the influence of the choice of the values of weights of the utility function. The results are reported in Figures 4.11 and 4.12. We observe that the best combination of the weights is $w_{cdc_h} = 0.25$ and $w_{col_{h,i}} = 0.75$.

From these diagrams, we can deduce that the prediction of future colocation is fundamental for the performance of CAR. At the same time, we also note that both attributes are important and need to be considered in the calculation of the utility function. The best performance is not obtained when we consider the colocation utility exclusively, but when both are evaluated. Moreover, the worst performance is obtained when the colocation factor is not used.

These results are related to the mobility model that we used. In fact, the change degree of

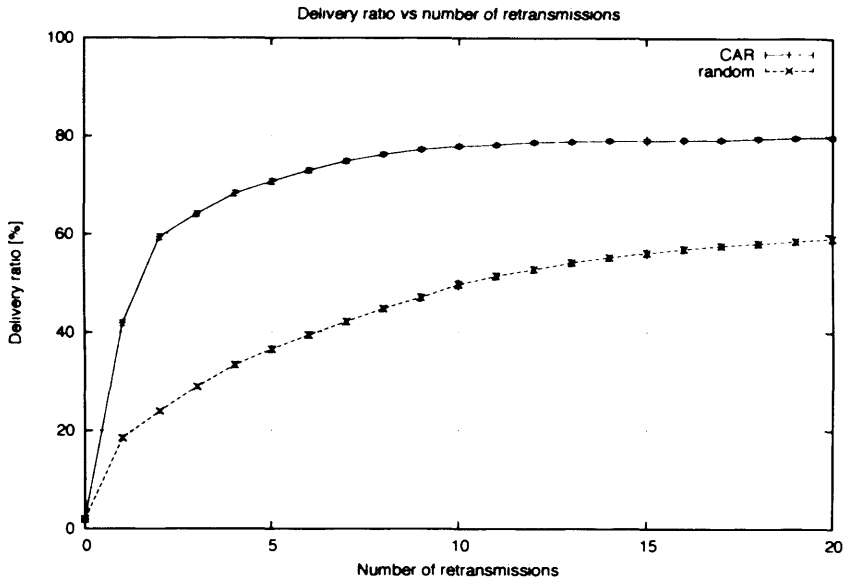

Figure 4.15: Influence of the number of retransmissions on the delivery ratio (scenario with 50 hosts and Community based mobility model).

connectivity attribute is particularly important when no information about colocation is available, i.e. when there are no potential carriers that have met the recipient in the past. In these cases, the best possible choice is to select a node characterised by high mobility, that potentially may be in contact with the recipient or, at least, with a better carrier that has been in reach of the recipient previously. This situation is not modelled by the Community based mobility model and, therefore, in our simulations, the most important attribute is the colocation one, that is very effective for selecting a host that is periodically in contact with the host of a particular community.

From a real deployment perspective, the selection of the values of the weights may be based on preliminary observations of the mobility patterns of the hosts. Since the protocol is not based on feedback mechanisms (like acknowledgement messages), it is not possible to adapt dynamically the values of these weights at run-time.

4.2.5 Influence of the Speed of the Hosts and Routing Table Transmission Interval

In Figures 4.13 and 4.14 the influence of the speed of the hosts on the delivery ratio for the 50 and 100 scenarios is shown considering different routing interval retransmission

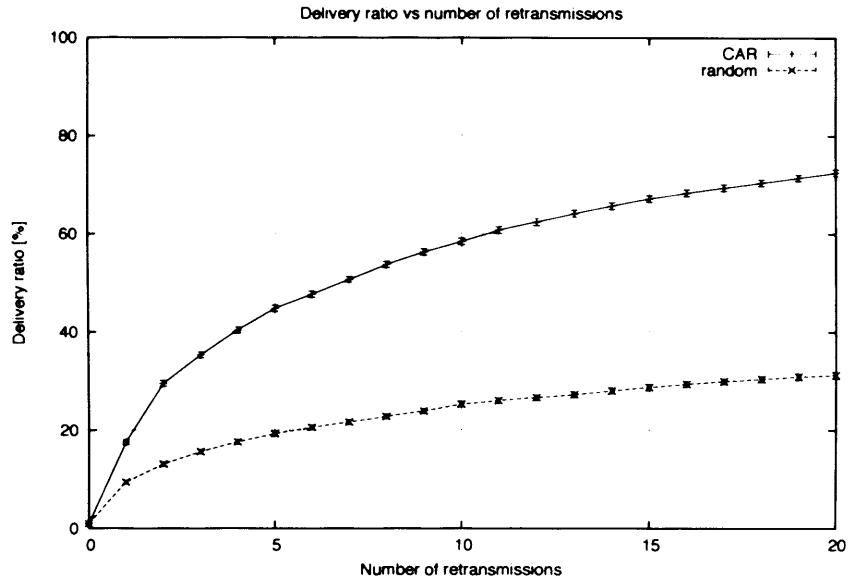

Figure 4.16: Influence of the number of retransmissions on the delivery ratio (scenario with 100 hosts and Community based mobility model).

intervals. In these experiments all the nodes have the same speed. In these experiments, the update interval of the prediction model is equal to the routing table retransmission interval.

Let us consider the interval equal to 30 seconds. We observe that the delivery ratio decreases as the speed increases. This is due to the fact that some routes become stale[4], but also because the quality of the prediction deteriorates as the mobility of the nodes increases as we will show in Section 4.2.7. If the calculation of the delivery is wrong, in scenarios composed of isolated clusters the probability that the messages are copied to the carriers that are moving between the communities is lower.

As the routing table transmission increases, the delivery ratio increases as expected, since the updates of the routes increases and the inputs of the filter are more frequent (and, therefore, the prediction is more accurate).

We also observe that the Random Choice protocol performs better as the speed of the nodes increases. In fact, as the speed increases, the probability of being in reach of the final recipient of the message is also higher. We also note a small decrement of the performance of the Random Choice protocol for values in the range $[1-5]$ m/s for

[4]This is a well know problem of distance vector protocols for mobile ad hoc networks [Perkins and Bhagwat, 1994].

the 50 hosts scenario and in the range $[1 - 3]$ m/s for the 100 hosts scenario. In this case two concurrent phenomena are present: as the speed increases, the percentage of the contents of routing tables that becomes stale increases[5] and, at the same time, the probability of meeting the recipients also increases. However, with low speed the effect of the latter is nullified by the effect of increasingly stale routing information. The curve for the Random Choice protocol is generated using a transmission interval equal to 30 seconds. By increasing the transmission interval, the improvement of the performance of the Random Choice protocol is minimal, since it does not exploit delivery probability information. The improvement is due to the fact that there is an increased probability that a neighbour will be discovered and, then, randomly selected. In fact, routing tables are also used for neighbour discovery.

4.2.6 Influence of the Number of Retransmissions

The influence of the number of the retransmissions on the delivery ratio for the 50 and 100 hosts is shown in Figures 4.15 and 4.16.

A limited number of retransmissions impacts on the possibility of transferring messages from a carrier to another one with a higher delivery probability. As shown in Figure 4.15, the number of hops that is sufficient to achieve the maximum delivery ratio in this scenario, given the time constraints, is around 10. This saturation of the performance is less visible for the 100 hosts scenario (Figure 4.16), where the impact of imposing a limited number of retransmissions is more evident. In this scenario, the higher number of nodes also increases the probability of being in reach of better carriers.

Clearly, the impact of the number of retransmissions is more evident for the Random Choice protocol, since it increases the chances of selecting a host that is moving between the communities (i.e., it increases the probability of being in reach of the final recipient of the message).

The influence on the number of messages sent is instead shown in Figures 4.17 and 4.18. A higher number of retransmissions has also an impact on the number of messages sent. However, in percentage, this is really limited: for this reason, from these experiments, we conclude that the benefit of increasing the number of retransmissions exceeds the cost in terms of overhead.

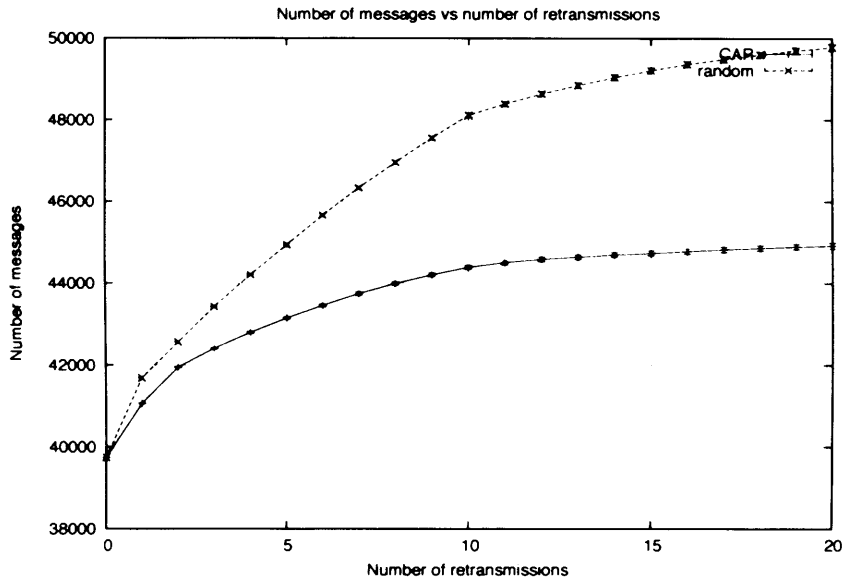

Figure 4.17: Influence of the number of retransmissions on the number of messages (scenario with 50 hosts and Community based mobility model).

4.2.7 Predictability Level and Protocol Performance

An analysis of the prediction level for both scenarios with different routing table transmission intervals is reported in Figures 4.19 and 4.20.

In general, we expect that the prediction level decreases as the retransmission level decreases. We observe that for transmission intervals equal to 10 and 20 seconds a reduction is visible for values in the range $[0, 6]$ and $[0, 9]$ for the 50 and 100 scenarios, respectively. Then, we note an increment of the prediction level: this is probably due to the fact that, since the routing tables are also used to detect the presence of the neighbours, with high speeds the node goes undetected. If a node is undetected, it is clearly considered as non colocated. In general, the predictor calculates low values of the utility associated to the colocation attribute (close to 0, i.e., value that corresponds to *non colocated*). The prediction error will be very low and the prediction will be considered correct. In other words, the inputs of the predictors are wrong, whereas the predictions are correct. This is a limitation of the proposed protocol in case of very dynamic environments, since it needs frequent updates of colocation attributes to achieve good performance.

[5]We underline again the fact that the freshness of routing information needs to be considered also for the synchronous delivery of the messages.

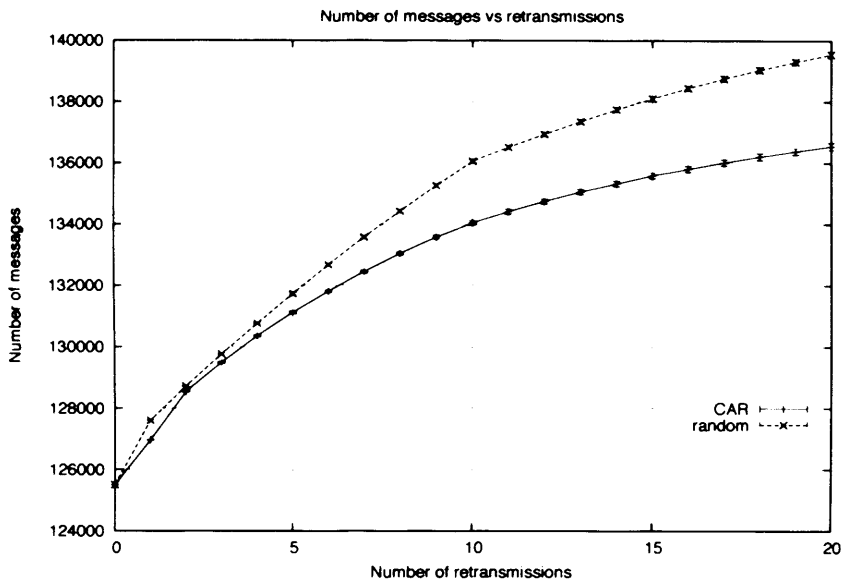

Figure 4.18: Influence of the number of retransmissions on the number of messages (scenario with 100 hosts and Community based mobility model).

With frequent updates (i.e., routing tables retransmission interval equal to 10 seconds), a decrement of the prediction level is observed, as the speed increases for all the range of values shown in the diagram. We also note that the decrement of the prediction level (in the range $[0, 6]$ and $[0, 9]$) is more evident for lower transmission frequency, since this leads to less frequent sampling of the time series.

From these experiments, we deduce that the update interval should be (approximately) inversely proportional to the speed of the hosts.

4.3 Critical Summary

The simulation experiments presented in this chapter have shown that CAR is able to guarantee good performance, also in presence of small buffers, with a limited overhead compared to the other protocols taken into consideration.

Limiting factors of the performance of CAR are small buffer size and the frequency of routing table transmission interval also in relation to the host speed. A trade-off has been identified between the performance of the protocol in terms of delivery ratio and the overhead associated to the routing tables transmission. In order to maintain high

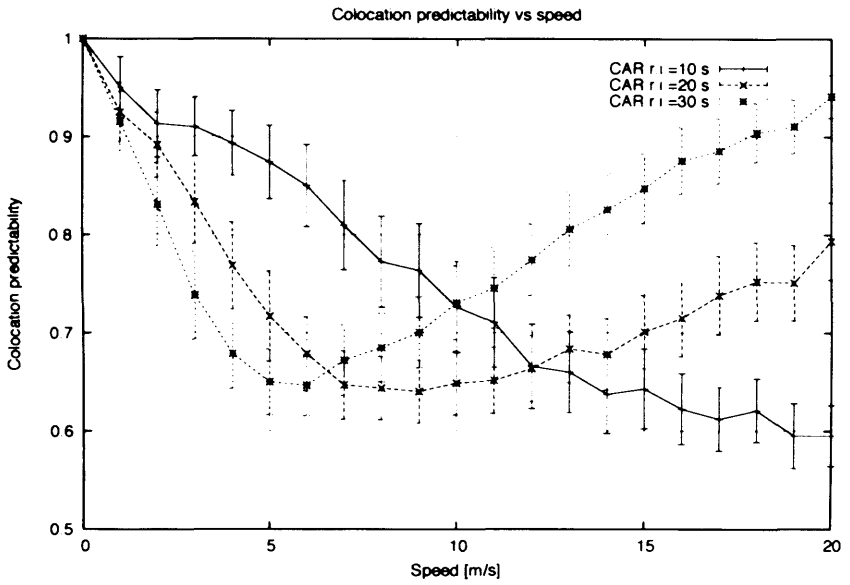

Figure 4.19: Influence of the host speed on the colocation predictability (scenario with 50 hosts and Community based mobility model).

predictability levels, the frequency of the transmission must be increased, as the speed increases.

The choice of the weights of the utility function has also a considerable impact on the performance of the protocol. From our analysis, it is evident that both context dimensions influence the message delivery ratio; as expected, host colocation has a primary importance in the best carrier selection process.

The number of retransmissions is also another key parameter of the protocol; the additional overhead associated to the number of retransmissions is low compared to the corresponding performance increment and, therefore, the choice of a relatively high number of possible retransmissions seems appropriate. This is necessary especially in scenarios composed of a high number of nodes.

The use of a fixed (small) size of routing table limits the performance and the scalability of CAR. However, the exploitation of the mechanisms for the intelligent replacement of the entries seems to alleviate the impact on the performance of the protocol.

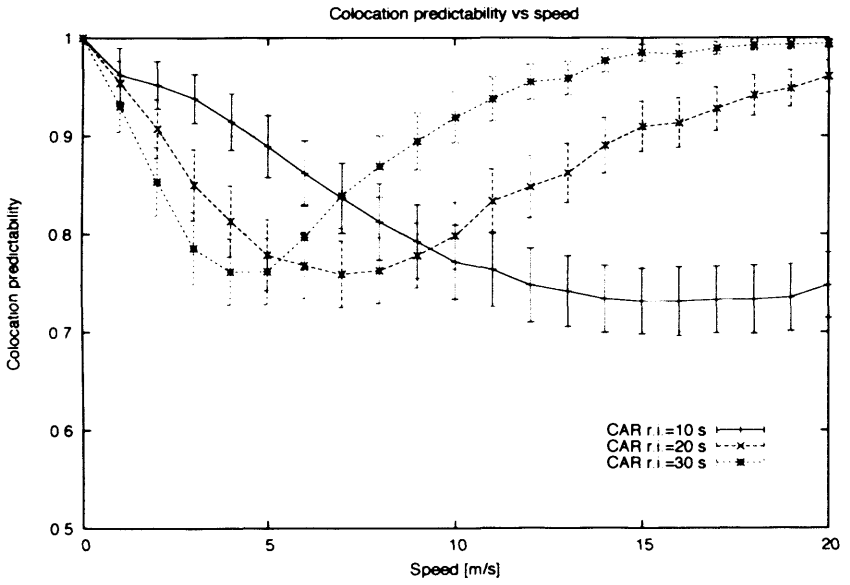

Figure 4.20: Influence of the host speed on the colocation predictability (scenario with 100 hosts and Community based mobility model).

Chapter 5

Implementation of the Context-Aware Adaptive Routing protocol

> The process of preparing a program for a digital computer is especially attractive, not only because it can be economically and scientifically rewarding, but also because it can be an aesthetic experience much like composing poetry or music.
>
> *Donald E. Knuth*

5.1 Overview of the Implementation

Haggle [Su et al., 2006] is a framework for opportunistic communication among mobile devices in infrastructure-based and infrastructure-less environments. Haggle allows for the use of different networking technologies at the same time such as Ethernet, 802.11, Bluetooth and so on. The goal of Haggle is to support communication in the form of data objects exchanges. These objects can also be stored temporarily in the local repositories providing the support for the implementation of store-and-forward routing mechanisms.

The architecture of Haggle is open and flexible and allows for the integration of routing protocols and other functionalities such as data storage related ones into the framework.

The design philosophy underlying the Haggle platform is inspired by Plutarch [Crowcroft et al., 2003], a proposal for a novel inter-network architecture supporting multiple heterogeneous interfaces and different routing mechanisms.

Two very simple routing algorithms have been implemented in the current release (0.5) of the Haggle platform: a one hop direct transmission protocol (i.e., data are sent to the recipient only if this is currently in reach) and a pure epidemic one. We implemented the Context-aware Adaptive Protocol as a *forwarding algorithm* for this platform. In this chapter, we present the details of the implementation and discuss the general design issues.

5.2 The Haggle Platform

We now briefly present the aspects of the Haggle platform that are relevant to the implementation of CAR.

5.2.1 Overview of the Haggle Architecture

In the current version of the Haggle platform two very simple routing algorithms have been implemented: a one hop direct transmission protocol (i.e., messages are sent to the recipient only if this is currently in reach) and a pure epidemic one (i.e., messages are sent to all the neighbours, without summary lists as in [Vahdat and Becker, 2000], using a time to live field to limit the spread of the information).

The architecture of Haggle is composed of six independent *Managers*:

- **Data Manager** This entity is responsible for managing the data that are stored in a searchable repository of *Data Objects* (implemented by the class `DO`), an abstraction used in Haggle to store data with a set of attributes. Queries for particular data based on matching on the values of the attributes using regular expressions can be performed through the Data Manager. Data Objects can be linked together to build more complex data structures. In the current version, the Data Manager is implemented using MySQL back-end [MySQL, 2007]. However, more lightweight solutions are currently being developed removing the need of a relational database; this is essential to run the platform on resource-constrained devices like PDAs.

  In addition to the ability of retrieving Data Objects by means of unique identifiers, the Data Manager also supports searching of Data Objects by means of filters.

- **Name Manager** This entity is responsible for the mapping between different names identifying the user-level end-points. The names of entities are stored as *Name Objects* (implemented by the class `Name`). A Name Object is a particular type of

Data Object. It can be linked to others referring to the same entity to form a *Name Graph*, that is used to store the information about all the known names of an entity collectively (e.g., all the identifiers for all the interfaces).

- **Forwarding Manager** This manager is responsible for delivering Data Objects to other hosts. It encapsulates a certain number of Forwarding Algorithms that are used to deliver the data to the final destinations. CAR is implemented as one of these algorithms.

  Applications can start a data transfer by specifying a set of Data Objects and a set of Name Objects. The Forwarding Manager then constructs a *Forwarding Object* that is a wrapper of a Data Object containing metadata needed to the forwarding operation. This is linked to Name Graphs containing the identifiers of the sender and the recipient(s).

  The metadata can also include expiration times and hop counts. More details about Forwarding Algorithms are provided in Section 5.2.2.

- **Protocol Manager** This manager is only responsible for encapsulating and managing a set of *Protocols*. According to the Haggle terminology, a Protocol is defined as a method used to transfer a Forwarding Object between two nodes (i.e., to enable point-to-point communication). Examples are SMTP [Klensin, 2001], HTTP [Fielding et al., 1999] or a direct peer-to-peer protocol.

- **Connectivity Manager** This manager is responsible for encapsulating a certain number of so-called *Connectivities*, representing the available network interfaces. It provides support for neighbour discovery and for managing communication channels between two hosts. The underlining communication mechanisms are implemented using Java sockets.

  The Connectivity Manager provides the abstraction of *Neighbour* that is a potential next-hop by which a Protocol may know how to send data of certain types to specific Name Objects.

- **Resource Manager** This manager schedules the various *Tasks* (i.e., operations) according to a predefined cost-benefit algorithm[1]. All the outgoing and incoming network connections are proposed for the scheduling to the Resource Manager and executed according to their priorities based on the evaluation of their possible benefits.

All managers provide interfaces which are used for the communication and coordination among each others and to be accessed by other entities, including the Forwarding Algorithms. In particular, our CAR implementation interacts with the Data Manager in order to store and retrieve Data Objects, with the Protocol Manager to obtain the list of the

[1] Different algorithms can be plugged in by developers.

hosts that are currently reachable and with the Name Manager to manage the identities of the hosts and the users.

In the following section we discuss the role of the forwarding algorithm and, in particular, we introduce the integration of the component implementing the core functionalities of the CAR protocol in the Haggle framework.

5.2.2 Forwarding Algorithms

The role of Forwarding Algorithms is to route the Forwarding Objects to the final destination(s). For each Forwarding Object that has to be sent, the Forwarding Algorithm creates a *Forwarding Task* that is executed by the Resource Manager according to priorities evaluated with a decision algorithm that calculates the benefit of performing a certain task. In our implementation, we set the benefit to the maximum to guarantee the immediate scheduling of the task[2]. When executed, the Forwarding Task causes the associated Protocol to send the Forwarding Object to the specified recipient.

From a more practical point of view, Forwarding Algorithms must implement the `setForwardingTasks()` method that receives in input the list of the pending Forwarding Objects. These Forwarding Objects may have been created by the host or may have been received from other hosts and stored temporarily. Periodically, the Forwarding Manager invokes the `setForwardingTasks()` method. In the current implementation of Haggle, this interval is fixed, but this can be easily modified by developers.

In Figure 5.1, the UML sequence diagram shows the interactions between the Forwarding Manager that invokes (periodically) the `setForwardingTasks()` of the instance of the class `CARForwardingAlgorithm`. Then this method calls the static method `setForwardingTask()` of the class `ForwardingTask` for each Forwarding Object that has to be sent. If no objects have to be sent, because there are no recipients of pending Forwarding Object or better carriers for them, it returns without interacting with any other class.

[2]Clearly, many tasks with the same priority may be in the queue waiting to be executed. In this case, the order of execution is random. The tasks related to the execution of the operations of the CAR protocol have the highest possible priority. A refinement may be to associate lower priority to transmissions to intermediate carriers (for example assigning a benefit of 90 over 100) with respect to the task of transmitting the data directly to the recipient if this is in reach. The transmission of routing tables may have a lower priority than the forwarding of objects containing data. It is worth noting that the prioritisation of the tasks is an aspect of the system still under investigation by the researchers working on the Haggle Project.

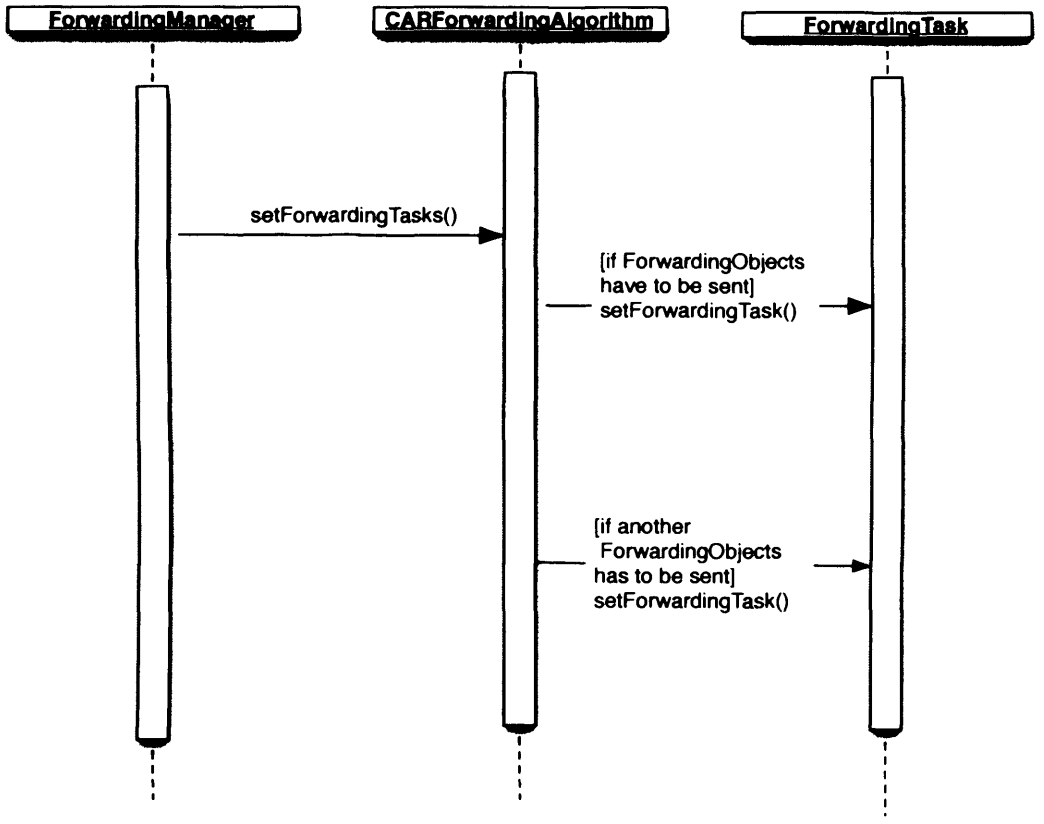

Figure 5.1: Sequence diagram illustrating the interactions between the Forwarding Manager and the class implementing the CAR protocol.

5.3 Integration of CAR in Haggle

5.3.1 Overview

In this section, we present the implementation of CAR in Haggle. As mentioned above, CAR is implemented as a forwarding algorithm of Haggle. We designed a prototype implementation of CAR supporting one hop synchronous communication.

The CAR implementation is shipped as a Java package called `haggle.forwarding.CAR` containing three classes:

- `CARForwardingAlgorithm` This class encapsulates the logic of the algorithm implementing the interface `ForwardingAlgorithm`. The constructor of this class launches three threads, responsible for forwarding the data and control messages for creating and updating the predictors.
- `KalmanPredictor` This class implements the prediction algorithm presented in Sec-

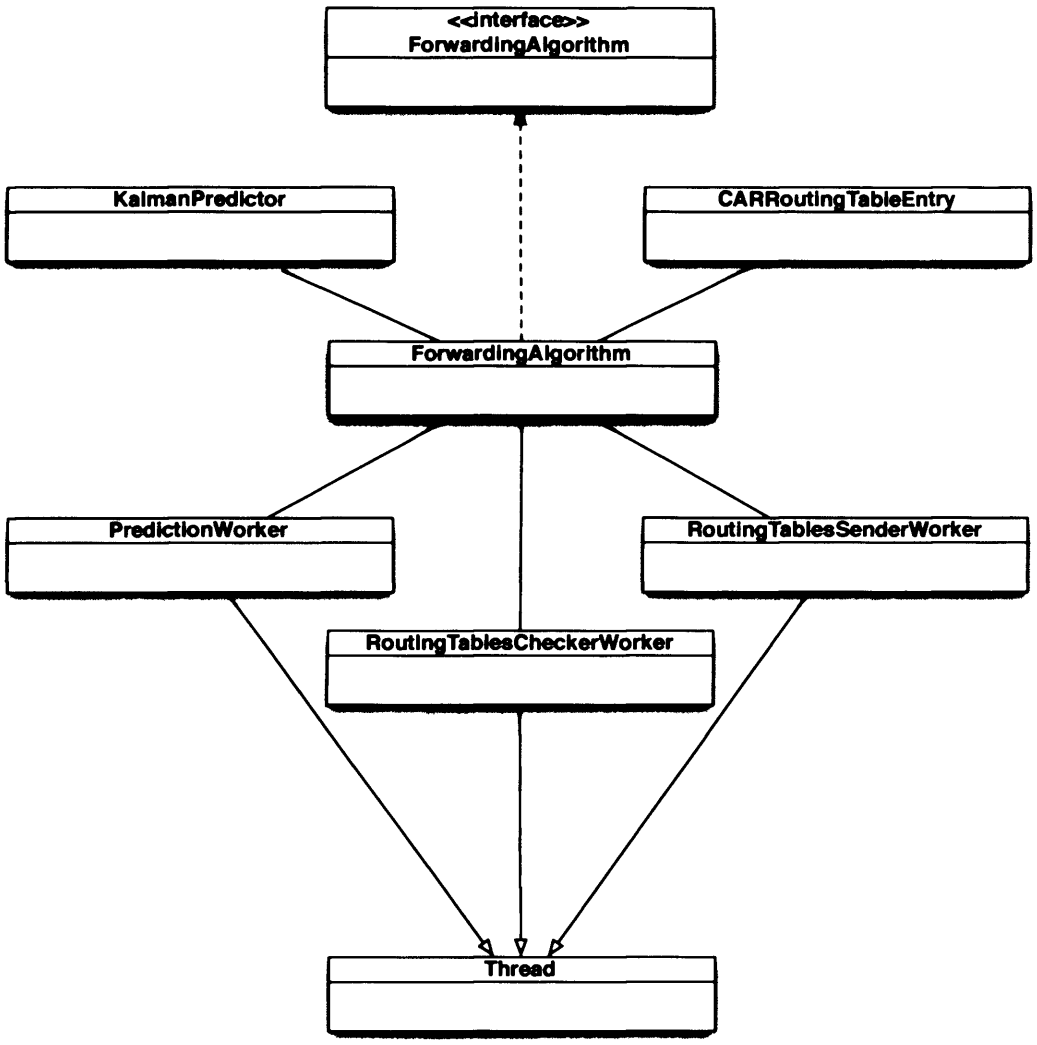

Figure 5.2: Class diagram of the implementation of CAR for Haggle.

tion 2.5. The default values of the filter are R_t=0.01 and Q_t=0.1. These values can be set by developers. The implementation of the algorithm used in this class is functionally identical to the one used in the simulations.

- `CARRoutingTableEntry` This class implements an entry of the routing table, providing different constructors for its initialisation (using `String` objects or instances of the `Name` class). Routing tables are implemented as a `HashMap` of `CARRoutingTableEntry` objects.

A UML class diagram showing all the interdependencies of the classes used for the integration of CAR in Haggle is shown in Figure 5.2.

From a practical point of view, in order to run Haggle with CAR, this needs to be added to the list of the routing protocols available for the forwarding process in the configuration file of Haggle: we assume that only CAR is used as a forwarding algorithm[3]. No modifications to the Haggle codebase is necessary.

When the `setForwardingTasks()` is invoked, the recipient of each Forwarding Object passed as a parameter is extracted. Then, the protocol verifies if the recipient is currently a neighbour (i.e., reachable at least through one of the available interfaces) and, if this the case, the object is forwarded to it.

If a synchronous delivery is not possible, the host checks if it is currently the best carrier for the message. If this is the case, the message is added to the list of Forwarding Objects that is managed by the Resource Manager. Otherwise, the message is forwarded to the best carrier currently in reach. We note that we had to provide support for *multi-regions routing*. In other words, a host may be reachable through different network interfaces. Different cost metrics can be assigned to different interfaces.

Haggle does not provide any mechanism for exchanging information periodically, like routing tables; however, this is essential for proactive protocols. We implemented the support for it by means of the classic threads synchronisation techniques available in Java and the available Haggle components and abstractions. In particular, the routing table messages are implemented as a special type of Forwarding Object. More precisely, we added a field in the Data Object that distinguishes these objects from the ones carrying data called `ProtocolCode`. Forwarding Object containing routing tables are characterised by a specific value of this variable. This field is used for searching among the Forwarding Objects stored in the Data Manager. Since the current version of Haggle does not support binary fields in the Data Object but only Java strings, we implemented two methods to serialise and deserialise the routing tables into/from strings (i.e., the `encodeRoutingTableIntoString` and the `encodeStringIntoRoutingTable` methods).

Each host maintains a routing table that is implemented as a `HashMap` containing entries that are instances of the class `CARRoutingTableEntry`. Routing tables are indexed using the field `targetHostId`. The fields of the routing table entries `targetHostId`, `nextHopHostId` and `bestProbHostId` are implemented as instances of `Name`.

5.3.2 Multi-threaded structure

Our goal was to encapsulate all the logic of the protocol in one class in order to simplify the integration of CAR in the platform. The multi-threaded structure of the class implementing the forwarding algorithm is presented in Figure 5.3.

[3]However, we observe that CAR includes the direct forwarding algorithm implicitly.

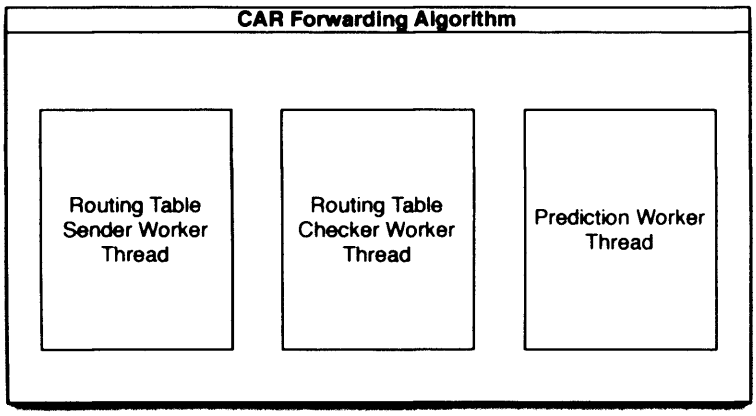

Figure 5.3: Multithreaded structure of the class implementing the CAR forwarding algorithm in Haggle.

We now describe the role of each thread in detail:

- **Prediction worker** The role of the `PredictionWorker` is to update the delivery probabilities by retrieving information about the context from other Haggle managers and by calculating the next step prediction using the Kalman filter based local predictors. More specifically, the `PredictionWorker` firstly updates the colocation predictor by checking if the host is a neighbour (i.e., it is reachable through at least one of the available interfaces). This is performed by requesting the list of Protocols to the Name Manager and then by checking all the reachable neighbours through a specific protocol. This logic is implemented by the method `isColocated()` that receives a `Name` object as parameter. If a new node is discovered, a new Kalman filter predictor is initialised and added to the pool of the predictors[4]. The thread also updates the current change degree of connectivity and the related Kalman filter. Then, for each host, the combined weighted utility is calculated. The estimated delivery probabilities of all the nodes that are present in the routing tables are also updated.

  The other role of the thread is to check if the delivery probability of the host is higher than the one stored in the routing table and, if this is the case, to insert its identifier as best carrier and record its corresponding current delivery probability.

  Finally, this thread also checks if new names of the host have been added to the Name Manager, inserting these names to the routing tables for the delivery of the messages. A host can have different entries in the routing tables, one for each name. A different metric can be associated to different types of names (i.e., communication

[4]In the current implementation, there is not a predefined size of the pool. However, a possible choice is to enforce a fixed size by removing the hosts of the pool with the lowest colocation values. This is clearly necessary for resource-constrained devices.

domains). An alternative choice is to keep a name graph for all the names indicating the same host, but this does not allow for implementing specific calculation of the best path according to a given metric or to particular cost/benefit mechanisms[5].

- **Routing tables sender worker** The `RoutingTablesSenderWorker` thread is responsible for sending the routing tables periodically to the other nodes that are currently in reach.

  The routing tables are implemented as a special type of Forwarding Objects that are generated by the Forwarding Algorithm itself by constructing them as Data Objects. They are then sent as standard Forwarding Objects to all the neighbours on all the interfaces.

- **Routing tables checker worker** The `RoutingTableCheckerWorker` is responsible for checking if Forwarding Objects containing routing tables have been received. This is necessary, since an architecture based on listeners for particular types of messages (based on message filters) is not currently available in Haggle.

  This thread periodically checks if the Data Manager is currently storing routing tables by means of a filter on the `protocolCode` field. In theory, this should be done using a persistent filter that notifies an event handler, but this has not yet been implemented in Haggle.

  If Forwarding Objects containing routing tables have been received, these are deserialised and the information is used to update the local routing table.

These threads are started as daemons (according to the Java terminology) when Haggle is launched. The interval of retransmissions of routing tables and updating of the predictors can be set by developers.

5.4 Testing and Evaluation

In order to perform a functional test of the integration of CAR in Haggle, we have set up a testbed of four desktop computers equipped with 108 Mbps Netgear WG113T Wireless Adapters. The testbed is represented in Figure 5.4.

The testing was limited by the current implementation of the Haggle prototype, since it is not possible to stop and restart the platform to simulate a disconnection. For this reason, we simulate this in software, by limiting the reachability of the hosts building a sort of virtual network and allowing only asynchronous routing, even if a connected path exists.

[5]In order to add the aggregation of names related to the same entity is sufficient to check if another name of the same entity is already present in the routing table before adding a new entry.

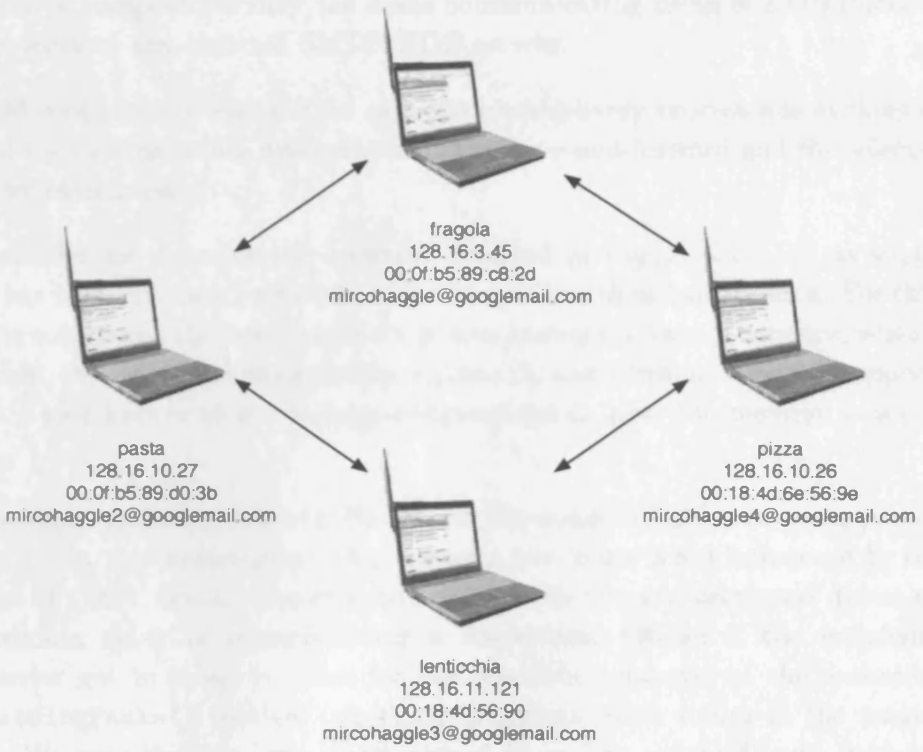

Figure 5.4: Topology and addresses of the hosts of the Haggle testbed.

We tested different *virtual* topologies; one of these is shown in Figure 5.4, where even if the four hosts are physically in radio range, the host `pasta` is not able to communicate using synchronous routing to `pizza`.

We used a SMTP/POP application to test the platform. This application is composed of two components: a SMTP/POP proxy for the interaction with an email client (like Thunderbird) and SMTP and POP protocols (implementing the `Protocol` interface) inside Haggle that communicate with email servers. The SMTP proxy translates the email sent by the client into a Forwarding Object with one or more Data Objects, one containing the email itself and the others if attachments have to be sent. The POP proxy listens on port 110 for incoming connections from the user's client. When the client checks for email for the first time, the POP proxy inserts a new known name (the email address of the host) in the Name Manager. CAR is periodically checking if new names have been registered requesting the list of known names of the host to the Name Manager, as described above. The new names are then inserted in the routing table. In Figure 5.4 we indicated all the names associated to the hosts (DNS name, IP address, Ethernet 802.11 address and email address of the user).

The POP proxy interacts with the Data Manager for searching for Data Objects containing email; it then reconstructs the original email and return it to the client. We assume that

the network is composed of only the hosts communicating using 802.11b interfaces in ad hoc mode without any external SMTP/POP servers.

We ran different test to observe if the asynchronous delivery process was working correctly. We tested the routing tables dissemination, the store-and-forward and the selection of the best carrier mechanisms[6].

Let us consider for example the scenario depicted in Figure 5.4. Let us suppose that `fragola` has been colocated with `pizza` for longer time than `lenticchia`. For this reason, `fragola` is considered the best carrier for `pizza` among the two. Therefore, when an email is sent from the user `mircohaggle2@googlemail.com` running its email application on `pasta` to a user known as `mircohaggle4@googlemail.com`, the message is sent through `fragola`.

Since our implementation relies on Haggle for the communication between pairs of nodes in ad hoc mode, the transmission delay between two hosts is not influenced by our implementation of CAR. Some measurements of these delays are presented [Su et al., 2006]. The maximum delay in retransmitting a Forwarding Object if the recipient or if a better carrier get in reach is given by the scheduling interval of the execution of the `SetForwardingTasks()` method (we run experiments with values in the range $[0.1, 10]$ seconds). We note that also this parameter is dependent on the Haggle implementation itself.

[6] In order to perform this test easily, we also implemented a *carrier* mode. A host in carrier mode will have delivery probability set to 1 for all the hosts and then it will always be chosen as best carrier.

Chapter 6

Conclusions

Now this is not the end.

It is not even the beginning of the end.

But it is, perhaps, the end of the beginning.

Winston Churchill

6.1 Summary of the Thesis

We have presented the design, the evaluation and the implementation of the Context-aware Adaptive Routing protocol which supports communication in delay tolerant mobile ad hoc networks.

We have shown that prediction techniques can be used to design store-and-forward mechanisms to deliver messages in intermittently connected mobile ad hoc networks, where a synchronous path between the sender and receiver may not exist. We have designed a generic framework for the evaluation of multiple dimensions of the mobile context in order to select the best message carrier. We have demonstrated that Kalman filter based forecasting techniques can be applied effectively to support intelligent message forwarding. We have shown that these techniques are lightweight and, therefore, suitable for resource-constrained devices.

In order to be able to validate our protocol, we have developed a novel mobility model based on recent results in social networks theory. We have validated the model using real traces, showing that it is able to reproduce the host colocation patterns in terms of inter-contacts time and contacts duration.

The simulation experiments have shown that CAR is able to guarantee good performance, also in presence of small buffers, with a limited overhead in terms of number of messages sent, in comparison to the other protocols taken into consideration. We have also highlighted the differences in terms of delivery ratio between random and prediction based routing. We have presented a sensitivity analysis of the parameters of the protocol, underlining the trade-offs in terms of delivery ratio and additional overhead. Furthermore, we have discussed the causes and the implications of inaccurate predictions, proposing possible countermeasures in case of highly dynamic systems and an algorithm for the selection of alternative protocols, like epidemic-style ones, if necessary.

Finally, we have presented an implementation of CAR on Haggle, an opportunistic communication framework, discussing the issues related to the design of the system prototype.

6.2 General Contributions of the Thesis and their Applicability to Other Areas of Computer Science

We now consider three key contributions of the present work in detail, outlining their possible general applicability to other problems in networking and, more in general, in other areas of computer science.

- **Routing based on Predictable Mobility Patterns Founded on Social Relationships** With respect to the delay tolerant networking research area, we believe that our key contribution resides in the evaluation of the evolution of different dimensions that describe the context where communication happens in order to make routing decisions. Our work has inspired the design of other forwarding mechanisms based on predictable colocation patterns founded on social relationships, such as [Leguay et al., 2006] and [Ghosh et al., 2007].

- **Integrated Routing Table Management for Synchronous and Asynchronous Delivery** Another contribution of this work is the design of routing table management mechanisms to store, update and distribute information to support synchronous and asynchronous message delivery. Up to our knowledge, this is the first attempt to provide a unifying solution to this problem in challenged networks.

  More generally, the approach used in CAR can be adopted to support the integration of deterministic routing mechanisms (i.e., based on the existence of known routing

paths, unless failures happens, like in DSDV) and probabilistic ones (i.e., based on the likelihood that a path will exist in the future between the sender and the recipient). In general, it can be used to enable communication in hybrid networks composed of infostations connected to the Internet and mobile nodes moving around connected intermittently to the backbone.

- **Application of Social Networks Theory to the Design of Realistic Mobility Models** The observation that mobile networks are social networks after all and, then, that their evolution over time can be modelled considering the relationships among the people carrying the devices is another key contribution of this thesis. This idea can be used to develop more refined models of human mobility and connectivity, but also to design mobile systems that take social relationships into account for data routing and sharing of information. Our work has been a starting point for other projects in this area such as [Legendre et al., 2006] and [Resta and Santi, 2006].

- **Utility Based Framework for Context-Aware Systems** The utility based framework presented in Section 2.2 provides a simple and extensible way of measuring and combining different aspects of context that can be measured and represented by means of utilities.

  The applications of such framework are potentially large, including, for instance, the definition of Quality of Service evaluation model [Chalmers and Sloman, 1999] (for example, for the selection of transmission links considering error rates, bandwidth and power consumption) and the selection of service providers [Raverdy et al., 2006].

- **Prediction Techniques for Dynamic Distributed Systems** We envisage many different applications of the prediction framework based on Kalman filter forecasting techniques to other problems in networking and distributed systems, for example to the problem of bandwidth [Song et al., 2006] and load balancing [Dinda and O'Hallaron, 1999]. Furthermore, the utility and prediction framework presented in this work was also successfully applied by us to the problem of scheduling of jobs in Grid Computing [Chapman et al., 2007], where the current load of computational units is evaluated for the choice of the best server for a given job.

6.3 Current Research Directions

The theoretical results and the practical solutions presented in this work can represent a foundation for a wide research agenda in the area of delay tolerant networks and, more in general, of mobile systems.

In this section, we outline our most promising research directions founded in the work presented in this thesis.

6.3.1 Sensor Context-aware Adaptive Routing

We are adapting the core routing mechanisms of CAR for data collection in sensor networks [Mascolo and Musolesi, 2006, Mascolo et al., 2006].

Sensor devices are now present in virtually all sorts of items, from vehicles and furniture, and deployed on humans and animals. This generates networks of wireless connected devices with topologies which could be very dynamic. The monitoring abilities of these devices range from pollution and temperature to health-care and mobility. The amounts of data generated by these applications are usually quite large, but, fortunately, the data is also, in most cases, *delay tolerant*, in the sense that it can be stored in the network for a certain period of time before being collected. The scenario we envisage is one where mobile sensor nodes (e.g., animals, vehicles or humans) route data through each others in order to reach sink nodes, which can be either mobile or fixed. The fixed nodes are intended as nodes connected to a backbone network and therefore able to forward the data to the appropriate destination.

Different techniques could be employed for mobile sensor data gathering. A basic strategy would be to only allow data delivery when sensors are in direct proximity of the sinks. This technique has very little communication overhead, given that messages are only sent directly from the sensor node generating messages to the sink. However, depending on how frequently sensor nodes meet the sinks, the delivery of the data might be very poor. This is particularly true if the sinks are very few and spread out. More refined techniques include epidemic-style approaches, which would spread the data over the sensor network, so that eventually a sink could be reached. This approach has very good delivery ratio if buffers are sufficiently large, however the overhead in terms of communication and, therefore, energy is quite high. Some solutions have been presented in literature, including the use of data mules with scheduled routes [Jea et al., 2005].

For these reasons, the spreading of the messages needs to be carefully controlled and traded off for the delivery ratio. This is even more true if the nodes have limited memory so that the buffer size is small and very few messages can be stored. Moreover, we consider scenarios where the routes of the potential message carriers may not be known a priori, as in the case of systems where devices attached to animals or humans are exploited to deliver the data to the sinks.

Starting from these considerations, we have developed SCAR (Sensor Context-Aware Routing), a routing approach which uses the prediction techniques over context of the sensor nodes to foresee which of the sensor neighbours are the best carriers for the data messages. These mechanisms are inspired by the ones used in CAR.

In fact, our proposed solution relies on the analysis of the history of the movement pattern of the nodes and their colocation with the sinks and on the evaluation of the current

available resources of the sensors. In particular, each node evaluates its change rate of connectivity, colocation with sinks, and battery level. The forecasted values of the attributes describing the context are then combined to define a delivery probability $P(s_i)$ for each sensor s_i to deliver bundles to sinks.

While moving, the sensors will transfer their data to other sensors only if these have a higher probability to deliver the data to sinks (i.e., they are better carriers). The calculation of the delivery probability is *local* and it does not involve any distributed computation. Nodes exchange information about their current delivery probability and their available buffer space with the neighbours only periodically.

The delivery probability of the nodes also keeps into account the energy level of the nodes, so to avoid that some best carriers become strong attractors and run into low battery problems more quickly than others. However, as the battery level decreases, the probability of being selected also decreases.

We are currently working on an implementation of SCAR for Contiki [Dunkels et al., 2004] and we plan to deploy a prototype of the protocol for wildlife monitoring.

6.3.2 Improvement of the Community based Mobility Model

A number of features can be added to the mobility model presented in this paper in order to increase its realism. Many researchers in the ad hoc network community have been focussing on different aspects of this problem. Some of the techniques are orthogonal to this work and can be integrated in our model. We plan to explore these refinements in the future. More specifically, some of the most important improvements can be summarised as follows:

- **Non random assignment of the nodes to the geographical locations** A possible improvement of the model may be the assignment of the communities based on real mapping between groups of people and geographical locations (such as students moving around lecture rooms and halls in a campus, etc.). An example of this kind of mobility models is [Hsu et al., 2005]. In the current implementation, the placement of the communities is random. This allows for multiple runs with different automatically generated social networks and mobile scenarios. However, the current implementation can be easily modified and replaced by a custom initialisation of the simulation settings.
- **Movement determined by pre-defined trails and presence of obstacles** Many existing mobility models are based on the definition of *trails* or *paths* that are used to define the movements of the mobile nodes in the simulation scenarios. Examples are the Manhattan model [Camp et al., 2002] or other models used to sim-

ulate protocols and systems for vehicular ad hoc networks [Saha and Johnson, 2004]. In these models hosts move between different locations following precise paths that represent roads or motorways. We plan to study the effects of the introduction of pre-defined trails in our model, in particular to characterise the movements between different communities (i.e., squares of the grid) with better accuracy. Instead of using a straight line for reaching the goals, hosts may move on the shortest path between the current point and the goal given a grid of roads constraining their movements.

Another improvement may consist in the insertion of obstacles that limit the radio propagation in the system. For example, in [Jardosh et al., 2005] the authors propose a modified version of the Random Mobility Model that allows for the insertion of obstacles in the simulation space. The definition of obstacles can also be easily integrated in our mobility model: this aspect is orthogonal with respect to the mechanisms at the basis of our proposed model.

- **Study of the connectivity of the resulting topology of the network** We plan to study the connectivity of the topology generated by the model also in relation to the social networks given in input using results from graph theory studies [Bollobas, 2001]. We believe that some interesting results of graph theory can be used to investigate properties of the network at run-time. For example, the detection of the formation of one fully connected group in the system, a *giant cluster* (i.e., each pair of the graph is connected by a link) or several networks partitions, can be done using spectral properties of the connectivity matrix of the mobile network. This is the adjacency matrix of a graph where the mobile nodes are the vertices and the radio links are the edges[1].

6.3.3 Development of Models of Human Connectivity

We plan to continue our study on human connectivity, starting from the analysis of real traces like the one collected by the CRAWDAD Project [Kotz and Henderson, 2005].

We believe that this is a very general problem that may have applications not only for protocol testing and simulation but also for the design of novel systems that exploit the properties of human connectivity patterns for the dissemination (and retrieval) of information in mobile networks. A key point is the study of the factors that lead to the emergence of power-law distributions in human networks [Newman, 2005].

[1] This adjacency matrix can be used for the calculation of the so-called $n \times n$ Laplace matrix which indicates the difference between the *node degree matrix* and the adjacency matrix. The node degree matrix is a diagonal matrix where each diagonal element of the node degree matrix counts the neighbours of the respective node. If this matrix has only one eigenvalue equal to 0, the network is connected. If the number of the eigenvalues equal to 0 is greater than 1, then the network is partitioned in disconnected groups. The number of partitions is equal to the number of eigenvalues equal to 0.

6.3.4 Traffic Models

Another problem in evaluating protocols for delay tolerant networks is the definition of realistic traffic patterns. Social networks models may be used to design more realistic traffic models for evaluating mobile systems, and in particular opportunistic ones. However, the lack of traffic measurements is currently the main obstacle to the definition of such models. In fact, synthetic traffic models have been successfully derived for fixed networks (see, for example, the World Wide Web traffic model presented in [Crovella and Bestavros, 1997]), but, unfortunately, the small number of deployments of delay tolerant systems limit the availability of traces providing a quantitative representation of their usage. The other problem is that the traces that can be extracted are usually very specific and, therefore, can be used to model particular and not generic scenarios.

A possibility is to try to infer communication patterns from the use of other communication devices such as mobile phone calls [Eagle and Pentland, 2006].

6.3.5 Predictive Publish/Subscribe based on CAR

Finally, the CAR protocol can be applied to support more complex communication paradigms than simple unicast such as publish-subscribe systems [Costa and Picco, 2005].

The CAR point-to-point communication protocol could be easily extended to support 1-to-N communication for event notification dissemination in intermittently connected mobile ad hoc networks, by evaluating the probability of being colocated with the subscribers of a certain topic. Another open question is the definition of a set of primitives for probabilistic communication in mobile ad hoc networks. The idea is to provide a middleware layer based on the CAR routing protocol for asynchronous communication in delay tolerant mobile systems [Musolesi et al., 2004], similarly to the primitives of EMMA [Musolesi et al., 2006], a message oriented middleware implementing the JMS interface [Hapner et al., 2002] based on the epidemic replication of messages.

Appendix A

Estimation Models

A.1 Model with Trend Component

A more complex model can be obtained adding a trend component. We adopt a model composed by the following equations:

$$\begin{cases} Y_t = M_t + W_t \\ M_{t+1} = M_t + B_t + V_t \\ B_{t+1} = B_t + U_t \end{cases}$$

where

$$W_t = WN(0, \sigma_w^2)$$

$$V_t = WN(0, \sigma_v^2)$$

$$U_t = WN(0, \sigma_u^2)$$

In this model the state vector is the following

$$\mathbf{X}_t = \begin{bmatrix} M_t \\ B_t \end{bmatrix}$$

We can write

$$\begin{bmatrix} M_{t+1} \\ B_{t+1} \end{bmatrix} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} M_t \\ B_t \end{bmatrix} + \begin{bmatrix} V_t \\ U_t \end{bmatrix}$$

Setting

$$\mathbf{V}_t = \begin{bmatrix} V_t \\ U_t \end{bmatrix}$$

we can also rewrite this equation in a more compact way

$$\widehat{\mathbf{X}}_{t+1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \widehat{\mathbf{X}}_t + \mathbf{V}_t$$

Using the same notation that we have adopted for the Kalman filter, in this case we have

$$F_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$G_t = \begin{bmatrix} 1 & 0 \end{bmatrix}$$

$$Q_t = \begin{bmatrix} \sigma_V^2 & 0 \\ 0 & \sigma_U^2 \end{bmatrix}$$

$$R_t = \sigma_W^2$$

Therefore, we can rewrite the Kalman filter prediction equations for this model. Firstly, we consider the initial conditions that, in this case, can be calculated using the following formulae

$$\widehat{\mathbf{X}}_1 = P(\mathbf{X}_1 | Y_0)$$

$$\Omega_1 = E((\mathbf{X}_1 - \hat{\mathbf{X}}_1)(\mathbf{X}_1 - \hat{\mathbf{X}}_1)^T) =$$
$$= E\left(\begin{bmatrix} M_1 - \widehat{M}_1 \\ B_1 - \hat{B}_1 \end{bmatrix} \begin{bmatrix} M_1 - \widehat{M}_1 & B_1 - \hat{B}_1 \end{bmatrix} \right) =$$
$$= E\begin{pmatrix} (M_1 - \widehat{M}_1)(M_1 - \widehat{M}_1) & (M_1 - \widehat{M}_1)(B_1 - \hat{B}_1) \\ (M_1 - \widehat{M}_1)(B_1 - \hat{B}_1) & (B_1 - \hat{B}_1)(B_1 - \hat{B}_1) \end{pmatrix}$$

With respect to the recursive equations of the filter we obtain

$$\hat{\mathbf{X}}_{t+1} = \begin{bmatrix} \widehat{M}_{t+1} \\ \hat{B}_{t+1} \end{bmatrix} =$$
$$= \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} \widehat{M}_t \\ \hat{B}_t \end{bmatrix} + \Delta_t^{-1}(Y_t - M_t) \begin{bmatrix} \Theta_{M_t} \\ \Theta_{B_t} \end{bmatrix} =$$
$$= \begin{bmatrix} \widehat{M}_t + \hat{B}_t \\ \hat{B}_t \end{bmatrix} + \Delta_t^{-1}(Y_t - M_t) \begin{bmatrix} \Theta_{M_t} \\ \Theta_{B_t} \end{bmatrix}$$

that can be decomposed as follows

$$\widehat{M}_{t+1} = \widehat{M}_t + \hat{B}_t + \Delta_t^{-1}(Y_t - M_t)\Theta_{M_t}$$

$$\hat{B}_{t+1} = \hat{B}_t + \Delta_t^{-1}(Y_t - B_t)\Theta_{B_t}$$

with

$$\Delta_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \Omega_{11} & \Omega_{12} \\ \Omega_{21} & \Omega_{22} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \end{bmatrix} + [\sigma_w^2] = [\Omega_{11}] + [\sigma_w^2]$$

A.2 Model with Trend and Seasonal Components

It is possible to derive a more general model, adding another component in order to allow consideration of possible seasonal behaviour in the time series. Therefore, we introduce the term S_t in the observation equation, which can be rewritten as follows

$$Y_t = M_t + B_t + S_t + W_t$$

To define this seasonal component we have to analyse its properties. In general, we consider

a time series γ_t representing a seasonal component such that

$$\gamma_{t+d} = \gamma_t$$

and

$$\sum_{i=1}^{d} \gamma_t = 0$$

Therefore, it is possible to derive the following expression for the determination of γ_{t+1}

$$\gamma_{t+1} = -\gamma_t - ... - \gamma_{t-d+2} \qquad t = 1, 2, ...$$

A more general expression of the seasonal component S_t allowing for random deviations from strict periodicity is obtained by adding a term V_t to the right hand side of the previous expression

$$S_{t+1} = -S_t - ... - S_{t-d+2} + V_t \qquad t = 1, 2, ..$$

Considering only the seasonal effect, in order to obtain a state space representation, we introduce the $(d-1)$-dimensional state vector $\mathbf{X}_t$

$$\mathbf{X}_t = \begin{bmatrix} S_t & S_{t-1} & ... & S_{t-d+2} \end{bmatrix}^T$$

The series S_t is given by the observation equation

$$S_t = \begin{bmatrix} 1 & 0 & 0 & 0 & ... & 0 \end{bmatrix} \mathbf{X}_t \qquad t = 1, 2, ...$$

where $\mathbf{X}_t$ satisfies the state equation

$$\mathbf{X}_{t+1} = F\mathbf{X}_t + \mathbf{V}_t \qquad t = 1, 2, ...$$

with

$$\mathbf{V}_t = \begin{bmatrix} Z_t & 0 & 0 & 0 & ... & 0 \end{bmatrix}^T$$

and

$$F = \begin{bmatrix} -1 & -1 & \cdots & -1 & -1 \\ 1 & 0 & \cdots & 0 & 0 \\ 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

To derive a prediction state space model with a trend and a seasonal components, it is sufficient to add this state equation to that which we have discussed in the previous section. In other words, we have to consider the following state vectors

$$\mathbf{X}_t^1 = \begin{bmatrix} M_t & B_t \end{bmatrix}^T$$

$$\mathbf{X}_t^2 = \begin{bmatrix} S_t & S_{t-1} & \ldots & S_{t-d+2} \end{bmatrix}^T$$

Defining

$$\mathbf{X}_t = \begin{bmatrix} X_t^1 \\ X_t^2 \end{bmatrix}$$

we can derive a general form of the state equation that can be used to take in consideration both trend and seasonal components as follows

$$\mathbf{X}_t = F\mathbf{X}_t + \mathbf{V}_t$$

with

$$\mathbf{X}_t = \begin{bmatrix} M_t & B_t & S_t & S_{t-1} & \ldots & S_{t-d+2} \end{bmatrix}^T$$

$$F = \begin{bmatrix} 1 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 1 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & -1 & -1 & \cdots & -1 & -1 \\ 0 & 0 & 1 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \cdots & 1 & 0 \end{bmatrix}$$

$$\mathbf{V}_t = \begin{bmatrix} V_t & U_t & Z_t & 0 & \ldots & 0 \end{bmatrix}^T$$

The observation equation will be the following

$$Y_t = \begin{bmatrix} 1 & 0 & 1 & 0 & \ldots & 0 \end{bmatrix} \mathbf{X}_t + W_t$$

Bibliography

[Alberich et al., 2002] Alberich, R., Miro-Julia, J., and Rossello, F. (2002). Marvel Universe looks almost like a real social network. Research note, Departament de Matematiques i Informatica Universitat de les Illes Balears.

[Albert and Barabasi, 2002] Albert, R. and Barabasi, A.-L. (2002). Statistical mechanics of complex networks. *Review of Modern Physics*, 74:47–97.

[Aoki, 1990] Aoki, M. (1990). *State Space Modeling of Time Series*. Springer-Verlag.

[Balachandran et al., 2002] Balachandran, A., Voelker, G. M., Bahl, P., and Rangan, P. V. (2002). Characterizing user behavior and network performance in a public wireless LAN. In *Proceedings of SIGMETRICS'02*, pages 195–205, New York, NY, USA. ACM Press.

[Balazinska and Castro, 2003] Balazinska, M. and Castro, P. (2003). Characterizing Mobility and Network Usage in a Corporate Wireless Local-Area Network. In *1st International Conference on Mobile Systems, Applications, and Services (MobiSys'03)*, San Francisco, CA.

[Bantz et al., 2003] Bantz, D. F., Bisdikian, C., Challener, D., Karidis, J. P., Mastrianni, S., Mohindra, A., Shea, D. G., and Vanover, M. (2003). Autonomic personal computing. *IBM Systems Journal*, 42(1).

[Basagni et al., 1998] Basagni, S., Chlamtac, I., Syrotiuk, V. R., and Woodward, B. A. (1998). A Distance Routing Effect Algorithm for Mobility (DREAM). In *Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, MobiCom'98*.

[Blazevic et al., 2001] Blazevic, L., Giordano, S., and Le Boudec, J.-Y. (2001). Self organized routing in wide area mobile ad-hoc networks. In *IEEE Symposium on Ad-Hoc Wireless Networks (Globecom'01)*.

[Bollobas, 2001] Bollobas, B. (2001). *Random Graphs*. Cambridge University Press, Second edition.

[Box et al., 1994] Box, G., Jenkins, G., and Reinsel, G. (1994). *Time Series Analysis, Forecasting and Control*. Prentice-Hall.

[Brin and Page, 1998] Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual web search engine. In *Proceedings of the 7th International Conference on World Wide Web*, pages 107–117.

[Brockwell and Davis, 1996] Brockwell, P. J. and Davis, R. A. (1996). *Introduction to Time Series and Forecasting.* Springer.

[Burgess et al., 2006] Burgess, J., Gallagher, B., Jensen, D., and Levine, B. N. (2006). MaxProp: Routing for Vehicle-Based Disruption-Tolerant Networks. In *Proceedings of IEEE INFOCOM'06.*

[Byers et al., 1998] Byers, J. W., Luby, M., Mitzenmacher, M., and Rege, A. (1998). A digital fountain approach to reliable distribution of bulk data. In *Proceedings of SIGCOMM'98*, pages 56–67.

[Camp et al., 2002] Camp, T., Boleng, J., and Davies, V. (2002). A survey of mobility models for ad hoc network research. *Wireless Communication and Mobile Computing Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502.

[Cavin et al., 2002] Cavin, D., Sasson, Y., and Schiper, A. (2002). On the Accuracy of MANET Simulators. In *Proceedings of the Workshop on Principles of Mobile Computing (POMC'02)*, pages 38–43.

[Chaintreau et al., 2005] Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R., and Scott, J. (2005). Pocket Switched Networks: Real-world mobility and its consequences for opportunistic forwarding. Technical Report UCAM-CL-TR-617, University of Cambridge, Computer Laboratory.

[Chaintreau et al., 2006] Chaintreau, A., Hui, P., Crowcroft, J., Diot, C., Gass, R., and Scott, J. (2006). Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proceedings of INFOCOM'06*, Barcelona, Spain.

[Chalmers and Sloman, 1999] Chalmers, D. and Sloman, M. (1999). A survey of quality of service in mobile computing environments. *IEEE Communications Surveys and Tutorials*, 2(2).

[Chapman et al., 2007] Chapman, C., Musolesi, M., Emmerich, W., and Mascolo, C. (2007). Predictive Resource Scheduling in Computational Grids. In *Proceedings of the 21st International Parallel and Distributed Processing Symposium*, Long Beach, CA. IEEE Computer Society Press.

[Chatfield, 2004] Chatfield, C. (2004). *The Analysis of Time Series An Introduction.* Chapman and Hall CRC.

[Chen and Kotz, 2000] Chen, G. and Kotz, D. (2000). A survey of context-aware mobile computing research. Technical Report TR2000-381, Dept. of Computer Science, Dartmouth College.

[Chen and Murphy, 2001] Chen, X. and Murphy, A. L. (2001). Enabling Disconnected Transitive Communication in Mobile Ad Hoc Networks. In *Proceedings of the Workshop on Principles of Mobile Computing (POMC'01)*, pages 21 23.

[Costa and Picco, 2005] Costa, P. and Picco, G. P. (2005). Semi-probabilistic content-based publish-subscribe. In *Proceedings of the 25th International Conference on Distributed Computing Systems (ICDCS'05)*, pages 575–585, Columbus (OH, USA). IEEE Computer Society Press.

[Crovella and Bestavros, 1997] Crovella, M. E. and Bestavros, A. (1997). Self-similarity in World Wide Web traffic: evidence and possible causes. *IEEE/ACM Transactions on Networking*, 5(6):835–846.

[Crowcroft et al., 2003] Crowcroft, J., Hand, S., Mortier, R., Roscoe, T., and Warfield, A. (2003). Plutarch: an argument for network pluralism. In *FDNA'03: Proceedings of the ACM SIGCOMM Workshop on Future Directions in Network Architecture*, pages 258 266, New York, NY, USA. ACM Press.

[Davies and Newbold, 1979] Davies, N. and Newbold, P. (1979). Some power studies of a portmanteau test of time series model specification. *Biometrika*, 66.

[Debreu, 1959] Debreu, G. (1959). Topological methods in cardinal utility theory. In Arrow, K. J., Karlin, S., and Suppes, P., editors, *Mathematical Methods in the Social Sciences*. Stanford University Press.

[Demers et al., 1988] Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker, S., Sturgis, H., Swinehart, D., and Terry, D. (1988). Epidemic algorithms for replicated database maintenance. *ACM SIGOPS Operating Systems Review*, 22(1).

[Dinda and O'Hallaron, 1999] Dinda, P. A. and O'Hallaron, D. R. (1999). An evaluation of linear models for host load prediction. In *Proceedings of the The Eighth IEEE International Symposium on High Performance Distributed Computing (HPDC '99)*, page 10, Washington, DC, USA. IEEE Computer Society.

[Doria et al., 2002] Doria, A., Uden, M., and Pandey, D. P. (2002). Providing connectivity to the Saami nomadic community. In *Proceedings of the Second International Conference on Open Collaborative Design for Sustainable Innovation*.

[DTNRG, 2006] DTNRG (2006). Delay tolerant networking research group homepage. http://www.dtnrg.org.

[Dunkels et al., 2004] Dunkels, A., Grnvall, B., and Voigt, T. (2004). Contiki - a Lightweight and Flexible Operating System for Tiny Networked Sensors. In *Proceedings of the First IEEE Workshop on Embedded Networked Sensors (Emnets-I)*, Tampa, Florida, USA.

[Durbin and Koopman, 2001] Durbin, J. and Koopman, S. J. (2001). *Time Series Analysis by State Space Methods*. Oxford University Press.

[Dushyanth et al., 2000] Dushyanth, N., Flinn, J., and Satyanarayanan, M. (2000). Using history to improve mobile application adaptation. In *Proceedings of the 3rd IEEE Workshop on Mobile Computing Systems and Applications*.

[Eagle and Pentland, 2006] Eagle, N. and Pentland, A. S. (2006). Reality mining: sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268.

[Einstein, 1956] Einstein, A. (1956). *Investigations on the Theory of the Brownian Movement*. Dover Publications.

[Erdos and Renyi, 1959] Erdos, P. and Renyi, A. (1959). On random graphs. *Publicationes Mathematicae*, 6:290–297.

[Erdos and Renyi, 1960] Erdos, P. and Renyi, A. (1960). On the evolution of random graphs. *Publications of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17 61.

[Fall, 2003] Fall, K. (2003). A delay-tolerant network architecture for challenged internets. In *Proceedings of SIGCOMM'03*.

[Fielding et al., 1999] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and Berners-Lee, T. (1999). RFC2616 - Hypertext Transfer Protocol - HTTP/1.1.

[Freeman, 1996] Freeman, L. C. (1996). Some antecedents of social network analysis. *Connections*, 19:39–42.

[Frenkiel et al., 2000] Frenkiel, R. H., Badrinath, B., Borres, J., and R.D. Yates, R. (2000). The infostations challenge: balancing cost and ubiquity in delivering wireless data. *IEEE Personal Communications*, 7(2):66–71.

[Gafni and Bertsekas, 1981] Gafni, E. M. and Bertsekas, D. P. (1981). Distributed algorithms for generating loop-free routes in networks with frequently changing topology. *IEEE Transactions on Communications*, 29(1).

[Ghosh et al., 2007] Ghosh, J., Philip, S. J., and Qiao, C. (2007). Sociological Orbit aware Location Approximation and Routing (SOLAR) in MANET. *Elsevier Ad Hoc Networks Journal*, 5(2):189–209.

[Glauche et al., 2003] Glauche, I., Krause, W., Sollacher, R., and Greiner, M. (2003). Continuum percolation of wireless ad hoc communication networks. *Physica A*, 325:577–600.

[Granger and Newbold, 1986] Granger, C. and Newbold, P. (1986). *Forecasting Economic Time Series.* Academic Press.

[Grossglauser and Tse, 2001] Grossglauser, M. and Tse, D. (2001). Mobility increases the capacity of ad-hoc wireless networks. In *Proceedings of INFOCOM'01*, pages 1360–1369.

[Grossglauser and Vetterli, 2003] Grossglauser, M. and Vetterli, M. (2003). Locating Nodes with EASE: Last Encounter Routing in Ad Hoc Networks through Mobility Diffusion. In *Proceedings of IEEE Infocom'03.*

[Guare, 1990] Guare, J. (1990). *Six Degrees of Separation: A Play.* Vintage.

[Haas and Pearlman, 1998] Haas, Z. J. and Pearlman, M. R. (1998). The performance of query control schemes for the Zone Routing Protocol. In *Proceedings of SIGCOMM'98.*

[Hapner et al., 2002] Hapner, M., Burridge, R., Sharma, R., Fialli, J., and Stout, K. (2002). *Java Message Service Specification Version 1.1.* Sun Microsystems, Inc. http://java.sun.com/products/jms/.

[Harras et al., 2005] Harras, K., Almeroth, K., and Belding-Royer, E. (2005). Delay Tolerant Mobile Networks (DTMNs): Controlled Flooding Schemes in Sparse Mobile Networks. In *IFIP Networking 2005*, pages 1180–1192.

[Henderson et al., 2004] Henderson, T., Kotz, D., and Abyzov, I. (2004). The changing usage of a mature campus-wide wireless network. In *Proceedings of MOBICOM'04*, pages 187–201, New York, NY, USA. ACM Press.

[Hermann, 2003] Hermann, K. (2003). Modeling the sociological aspect of mobility in ad hoc networks. In *Proceedings of MSWiM'03*, pages 128–129, San Diego, California, USA.

[Hong et al., 1999] Hong, X., Gerla, M., Pei, G., and Chiang, C.-C. (1999). A group mobility model for ad hoc networks. In *Proceedings of the 2nd International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems (MSWiM'99)*, pages 53 60.

[Hooke, 2002] Hooke, A. J. (2002). Towards an interplanetary internet: a proposed strategy for standardization. In *Proceedings of the Space Ops 2002 Conference.*

[Hsu et al., 2005] Hsu, W., Merchant, K., Shu, H., Hsu, C., and Helmy, A. (2005). Weighted Waypoint Mobility Model and its Impact on Ad Hoc Networks. *ACM Mobile Computer Communications Review (MC2R)*, pages 59–63.

[Hui et al., 2005a] Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J., and Diot, C. (2005a). Pocket switched networks and human mobility in conference environments. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay Tolerant Networking (WDTN'05)*, pages 244–251, New York, NY, USA. ACM Press.

[Hui et al., 2005b] Hui, P., Chaintreau, A., Scott, J., Gass, R., Crowcroft, J., and Diot, C. (2005b). Pockets Switched Networks and Human Mobility in Conference Environments. In *Proceedings of ACM SIGCOMM'05 Workshops*, pages 244–251.

[International Telecommunication Union, 2003] International Telecommunication Union (2003). Connecting remote communities. *Documents of the World Summit on Information Society*. http://www.itu.int/osg/spu/wsis-themes.

[IPNSIG, 2007] IPNSIG (2007). InterPlanetary Internet Special Interest Group. http://www.ipnsig.org.

[Iwata et al., 1999] Iwata, A., Chiang, C.-C., Pei, G. Y., Gerla, M., and Chen, T.-W. (1999). Scalable routing strategies for ad hoc wireless networks. *IEEE Journal on Selected Areas in Communications, Special Issue on Ad-Hoc Networks*, pages 1369–1379.

[Jain et al., 2005] Jain, S., Demmer, M., Patra, R., and Fall, K. (2005). Using redundancy to cope with failures in a delay tolerant network. In *Proceedings of SIGCOMM'05*, pages 109–120, New York, NY, USA. ACM Press.

[Jain et al., 2004] Jain, S., Fall, K., and Patra, R. (2004). Routing in a delay tolerant network. In *Proceedings of SIGCOMM'04*.

[Jardosh et al., 2005] Jardosh, A., Belding-Royer, E. M., Almeroth, K. C., and Suri, S. (2005). Real world Environment Models for Mobile Ad hoc Networks. *IEEE Journal on Special Areas in Communications - Special Issue on Wireless Ad hoc Networks*, 23(3).

[Jea et al., 2005] Jea, D., Somasundara, A. A., and Srivastava, M. B. (2005). Multiple Controlled Mobile Elements (Data Mules) for Data Collection in Sensor Networks. In *Proceedings of DCOSS'05*, pages 244–257.

[Johnson and Maltz, 1996a] Johnson, D. B. and Maltz, D. A. (1996a). Dynamic Source Routing in ad hoc wireless network. In Imielinski, T. and Korth, H., editors, *Mobile Computing*, chapter 5, pages 153–181. Kluwer Academics Publisher.

[Johnson and Maltz, 1996b] Johnson, D. B. and Maltz, D. A. (1996b). Protocols for adaptive wireless and mobile networking. *IEEE Personal Communications*, 3(1):34–42.

[Juang et al., 2002] Juang, P., Oki, H., Wang, Y., Martonosi, M., Peh, L. S., and Rubenstein, D. (2002). Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. *SIGOPS Operating Systems Reviews*, 36(5):96–107.

[Kalman, 1960] Kalman, R. E. (1960). A new approach to linear filtering and prediction problems. *Transactions of the ASME Journal of Basic Engineering.*

[Keeney and Raiffa, 1976] Keeney, R. L. and Raiffa, H. (1976). *Decisions with Multiple Objectives: Preference and Value Tradeoffs.* Wiley Series in Probability and Mathematical Statistics. John Wiley and Sons.

[Khelil et al., 2002] Khelil, A., Becker, C., and Tian, J. a. (2002). An epidemic model for information diffusion in MANETs. In *Proceedings of the the Fifth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile System (MSWiM'02).*

[Kim et al., 2006] Kim, M., Kotz, D., and Kim, S. (2006). Extracting a mobility model from real user traces. In *Proceedings of INFOCOM'06*, Barcelona, Spain.

[Kim and Noble, 2001] Kim, M. and Noble, B. (2001). Mobile network estimation. In *Proceedings of MobiCom'01*, pages 298–309, New York, NY, USA. ACM Press.

[Klensin, 2001] Klensin, J. (2001). RFC2821 - Simple Mail Transfer Protocol.

[Knuth, 1993] Knuth, D. E. (1993). *The Stanford GraphBase: A Platform for Combinatorial Computing.* Addison-Wesley.

[Kotz and Henderson, 2005] Kotz, D. and Henderson, T. (2005). CRAWDAD: A Community Resource for Archiving Wireless Data at Dartmouth. *IEEE Pervasive Computing*, 4(4):12–14.

[Kurkowski et al., 2005] Kurkowski, S., Camp, T., and Colagrosso, M. (2005). Manet simulation studies: the incredibles. *SIGMOBILE Mobile Computing and Communications Review*, 9(4):50–61.

[Le Boudec, 2006] Le Boudec, J.-Y. (2006). *Performance Evaluation of Computer and Communication Systems.* EPFL Lecture notes.

[Le Boudec and Vojnovic, 2005] Le Boudec, J.-Y. and Vojnovic, M. (2005). Perfect simulation and stationarity of a class of mobility models. In *Proceedings of IEEE INFOCOM'05*, pages 72 79.

[Lebrun et al., 2005] Lebrun, J., Chuah, C.-N., Ghosal, D., and Zhang, M. (2005). Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In *Proceedings of the IEEE Vehicular Technology Conference (VTC 2005-Spring)*, volume 4, pages 2289 2293.

[Legendre et al., 2006] Legendre, F., Borrel, V., Amorim, M. D. D., and Fdida, S. (2006). Reconsidering Microscopic Mobility Modeling for Self-Organizing Networks. *IEEE Network Magazine*, 20(6).

[Leguay et al., 2006] Leguay, J., Friedman, T., and Conan, V. (2006). Evaluating Mobility Pattern Space Routing for DTNs. In *Proceedings of INFOCOM'06*, Barcelona, Spain.

[Lelescu et al., 2006] Lelescu, D., Kozat, U. C., Jain, R., and Balakrishnan, M. (2006). Model t++:: an empirical joint space-time registration model. In *Proceedings of MobiHoc'06*, pages 61–72, New York, NY, USA. ACM Press.

[Li and Rus, 2000] Li, Q. and Rus, D. (2000). Sending messages to mobile users in disconnected ad-hoc wireless networks. In *MobiCom'00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 44–55, New York, NY, USA. ACM Press.

[Li et al., 2006] Li, Y., Lai, T. H., Liu, M. T., Sun, M.-T., and Yang, J. (2006). DTGR: Disruption-Tolerant Geographic Routing for Wireless Ad Hoc Networks. *Simulation*, 82(6):399–411.

[Lindgren et al., 2003] Lindgren, A., Doria, A., and Schelen, O. (2003). Probabilistic routing in intermittently connected networks. *Mobile Computing and Communications Review*, 7(3).

[Liu et al., 2004] Liu, T., Sadler, C. M., Zhang, P., and Martonosi, M. (2004). Implementing software on resource-constrained mobile sensors: experiences with Impala and ZebraNet. In *Proceedings of MobiSys'04*, pages 256–269, New York, NY, USA. ACM Press.

[Lynch, 1996] Lynch, N. A. (1996). *Distributed algorithms*. Morgan Kaufmann Publishers Inc.

[Maeda et al., 2005] Maeda, K., Sato, K., Konishi, K., Yamasaki, A., Uchiyama, A., Yamaguchi, H., Yasumotoy, K., and Higashino, T. (2005). Getting urban pedestrian flow from simple observation: Realistic mobility generation in wireless network simulation. In *Proceedings of MSWiM'05*, pages 151–158.

[Malkin, 1999] Malkin, G. (1999). RFC2453 – RIP Version 2.

[Mascolo and Musolesi, 2006] Mascolo, C. and Musolesi, M. (2006). SCAR: Context-aware Adaptive Routing in Delay Tolerant Mobile Sensor Networks. In *Proceedings of the Delay Tolerant Networks Symposium. ACM International Wireless Communications and Mobile Computing Conference (IWCMC'06)*. ACM Press.

[Mascolo et al., 2006] Mascolo, C., Musolesi, M., and Pastzor, B. (2006). Demo abstract: Data collection in delay tolerant mobile sensor networks using scar. In *Proceedings of the 4th ACM Conference on Embedded Networked Sensor Systems (SenSys 2006)*, Boulder, CO, USA. ACM Press.

[Mauve et al., 2001] Mauve, M., Widmer, J., and Hartenstein, H. (2001). A survey on position-based routing in mobile ad-hoc networks. *IEEE Network*, 15(6).

[McNett and Voelker, 2005] McNett, M. and Voelker, G. M. (2005). Access and Mobility of Wireless PDA User. *Mobile Computing Communications Review*, 9(2):40–55.

[Meinhold and Singpurwalla, 1983] Meinhold, R. and Singpurwalla, N. (1983). Understanding the Kalman Filter. In *The American Statistician.*

[Milgram, 1967] Milgram, S. (1967). The small world problem. *Psychology Today*, 2:60–67.

[Murthy and Garcia-Luna-Aceves, 1996] Murthy, S. and Garcia-Luna-Aceves, J. J. (1996). An efficient routing protocol for wireless networks. *ACM Mobile Networks and Applications*, pages 183–197.

[Musolesi et al., 2005] Musolesi, M., Hailes, S., and Mascolo, C. (2005). Adaptive routing for intermittently connected mobile ad hoc networks. In *Proceedings of WoWMoM'05. Taormina, Italy.* IEEE press.

[Musolesi and Mascolo, 2006a] Musolesi, M. and Mascolo, C. (2006a). A Community based Mobility Model for Ad Hoc Network Research. In *Proceedings of the 2nd ACM/SIGMOBILE International Workshop on Multi-hop Ad Hoc Networks: from theory to reality (REALMAN'06).* ACM Press.

[Musolesi and Mascolo, 2006b] Musolesi, M. and Mascolo, C. (2006b). Controlled Epidemic-style Dissemination Middleware for Mobile Ad Hoc Networks. In *Proceedings of the 3rd Annual International Conference on Mobile and Ubiquitous Systems: Networks and Services (MOBIQUITOUS'06).* ACM Press.

[Musolesi and Mascolo, 2006c] Musolesi, M. and Mascolo, C. (2006c). Evaluating context information predictability for autonomic communication. In *Proceedings of 2nd IEEE Workshop on Autonomic Communications and Computing (ACC'06). Co-located with 7th IEEE Int. Symp. WoWMoM'06*, Niagara Falls, NY. IEEE Computer Society Press.

[Musolesi et al., 2004] Musolesi, M., Mascolo, C., and Hailes, S. (2004). Adapting asynchronous messaging middleware to ad hoc networking. In *2nd ACM International Workshop on Middleware for Pervasive and Ad Hoc Computing (MPAC'04).* To appear in Journal of Personal and Ubiquitous Communication, Springer.

[Musolesi et al., 2006] Musolesi, M., Mascolo, C., and Hailes, S. (2006). EMMA: Epidemic Messaging Middleware for Ad hoc networks. *Personal Ubiquitous Computing*, 10(1):28–36.

[MySQL, 2007] MySQL (2007). MySQL Website. http://www.mysql.com.

[N. Sarafijanovic-Djukic and Grossglauser, 2006] N. Sarafijanovic-Djukic, M. P. and Grossglauser, M. (2006). Island Hopping: Efficient Mobility Assisted Forwarding in Partitioned Networks. In *Proceedings of IEEE SECON'06.*

[Newman, 2001] Newman, M. E. J. (2001). The Structure of Scientific Collaboration Networks. In *Proceedings of the National Academy of Science*, volume 98, pages 404–409.

[Newman, 2003] Newman, M. E. J. (2003). The structure and function of complex networks. *SIAM Review*, 19(1):1–42.

[Newman, 2005] Newman, M. E. J. (2005). Power laws, Pareto distributions and Zipf's law. *Contemporary Physics*, 46:323.

[Newman and Girvan, 2004] Newman, M. E. J. and Girvan, M. (2004). Finding and evaluating community structure in networks. *Physical Review E*, 69.

[Newman and Park, 2003] Newman, M. E. J. and Park, J. (2003). Why Social Networks are Different from Other Types of Networks. *Physical Review E*, 68.

[ns2, 2006] ns2 (2006). The Network Simulator (ns-2). http://www.isi.edu/nsnam/ns/.

[OPNET Technologies Inc., 2004] OPNET Technologies Inc. (2004). Opnet modeler. http://www.opnet.com/products/modeler/home.html.

[Park and Scott Corson, 1997] Park, V. D. and Scott Corson, M. (1997). A highly adaptive distributed algorithm for mobile wireless networks. *Proceedings of INFOCOM '97.*

[Partan et al., 2006] Partan, J., Kurose, J., and Levine, B. N. (2006). A Survey of Practical Issues in Underwater Networks. In *Proceedings of ACM International Workshop on UnderWater Networks (WUWNet'06)*, pages 17–24.

[Pei et al., 2000] Pei, G., Gerla, M., and Chen, T.-W. (2000). Fisheye state routing: a routing scheme for ad hoc wireless networks. In *Proceedings of International Conference of Communications 2000 (ICC'00).*

[Pentland et al., 2004] Pentland, A., Fletcher, R., and Hasson, A. (2004). DakNet: Rethinking Connectivity in Developing Regions. *IEEE Computer*, pages 4–9.

[Perkins, 2001] Perkins, C. E. (2001). *Ad Hoc Networking.* Addison-Wesley.

[Perkins and Bhagwat, 1994] Perkins, C. E. and Bhagwat, P. (1994). Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers. In *Proceedings of thr Conference on Communications Architecture, Protocols and Applications (SIGCOMM 94).*

[Perkins and Royer, 1999] Perkins, C. E. and Royer, E. M. (1999). Ad hoc On-demand Distance Vector routing. In *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications.*

[Poladian et al., 2004] Poladian, V., Garlan, D., Shaw, M., and Sousa, J. P. (2004). Dynamic configuration of resource-aware services. In *Proceedings of the 26th International Conference on Software Engineering (ICSE'04)*.

[Rapoport, 1957] Rapoport, A. (1957). Contribution to the theory of random and biased networks. *Bulletin of Mathematical Biophysics*, 19:257–277.

[Raverdy et al., 2006] Raverdy, P.-G., Riva, O., de La Chapelle, A., Chibout, R., and Issarny, V. (2006). Efficient context-aware service discovery in multi-protocol pervasive environments. In *Proceedings of the 7th International Conference on Mobile Data Management (MDM'06)*, Washington, DC, USA. IEEE Computer Society.

[Resta and Santi, 2006] Resta, G. and Santi, P. (2006). The QoS-RWP mobility and user behavior model for public area wireless networks. In *Proceedings of the 9th ACM international symposium on Modeling analysis and simulation of wireless and mobile systems (MSWiM'06)*, pages 375–384, New York, NY, USA. ACM Press.

[Royer and Toh, 1999] Royer, E. M. and Toh, C.-K. (1999). A review of current routing protocols for ad-hoc mobile wireless networks. *IEEE Personal Communications Magazine*.

[Russell and Norvig, 2002] Russell, S. J. and Norvig, P. (2002). *Artificial Intelligence: A Modern Approach*. Prentice Hall, second edition.

[Saha and Johnson, 2004] Saha, A. K. and Johnson, D. B. (2004). Modeling Mobility for Vehicular Ad Hoc Networks. In *Proceedings of VANET'04*, pages 91–92.

[Saltzer et al., 1984] Saltzer, J. H., Reed, D. P., and Clark, D. D. (1984). End-to-end arguments in system design. *ACM Transactions on Computer Systems*, 2(4):277–288.

[Sarshar and Chowdhury, 2003] Sarshar, N. and Chowdhury, R. (2003). Scale-free and stable structures in complex ad hoc networks. *Physical Review E 026101*.

[Sasson et al., 2003] Sasson, Y., Cavin, D., and Schiper, A. (2003). Probabilistic broadcast for flooding in wireless mobile ad hoc networks. In *IEEE Wireless Communications and Networking Conference (WCNC 2003)*.

[Satyanarayan et al., 1994] Satyanarayan, M., Noble, B., Kumar, P., and Price, M. (1994). Application-aware adaptation for mobile computing. *Proceedings of the 6th ACM SIGOPS European Workshop*.

[Scott, 2000] Scott, J. (2000). *Social Networks Analysis: A Handbook*. Sage Publications, London, United Kingdom, second edition.

[Seth et al., 2006] Seth, A., Kroeker, D., Zaharia, M., Guo, S., and Keshav, S. (2006). Low-cost communication for rural internet kiosks using mechanical backhaul. In *Proceedings of the 12th annual international conference on Mobile computing and networking (MobiCom'06)*, pages 334–345, New York, NY, USA. ACM Press.

[Shah et al., 2003] Shah, R. C., Roy, S., Jain, S., and Brunette, W. (2003). Data mules: modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the First IEEE Sensor Network Protocols and Applications*, pages 30–41.

[Sharma and Mazumdar, 2004] Sharma, G. and Mazumdar, R. R. (2004). Scaling laws for capacity and delay in wireless ad hoc networks with random mobility. In *IEEE International Conference on Communications (ICC'04)*, pages 3869– 3873.

[Small and Haas, 2003] Small, T. and Haas, Z. J. (2003). The shared wireless infostation model- a new ad hoc networking paradigm (or where is a whale, there is a way). In *Proceedings of MobiHoc'03.*

[Song et al., 2006] Song, L., Deshpande, U., Kozat, U., Kotz, D., and Jain, R. (2006). Predictability of WLAN mobility and its effects on bandwidth provisioning. In *Proceedings of IEEE INFOCOM'06.*

[Spyropoulos et al., 2007] Spyropoulos, T., Psounis, K., and Raghavendra, C. (2007). Spray and Focus: Efficient Mobility-Assisted Routing for Heterogeneous and Correlated Mobility. In *Proceedings of the International Workshop on Intermittently Connected Mobile Ad hoc Networks (ICMAN'07).*

[Spyropoulos et al., 2005] Spyropoulos, T., Psounis, K., and Raghavendra, C. S. (2005). Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *Proceeding of the 2005 ACM SIGCOMM Workshop on Delay Tolerant Networking(WDTN'05)*, pages 252–259, New York, NY, USA. ACM Press.

[Stauffer and Aharony, 1992] Stauffer, D. and Aharony, A. (1992). *Introduction to Percolation Theory.* Taylor and Francis.

[Su et al., 2004] Su, J., Chin, A., Popivanova, A., Goel, A., and de Lara, E. (2004). User mobility for opportunistic ad-hoc networking. In *Proceedings of the Sixth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA'04)*, pages 41–50, Washington, DC, USA. IEEE Computer Society.

[Su et al., 2006] Su, J., Scott, J., Hui, P., Upton, E., Lim, M. H., Diot, C., Crowcroft, J., Goel, A., and de Lara, E. (2006). Haggle: Clean-Slate Networking for Mobile Devices. Submitted for publication.

[Taha, 1996] Taha, H. A. (1996). *Operations Research: An Introduction.* Prentice Hall.

[Tang and Baker, 2000] Tang, D. and Baker, M. (2000). Analysis of a local-area wireless network. In *Proceedings of MOBICOM'00*, pages 1–10, New York, NY, USA. ACM Press.

[Tuduce and Gross, 2005] Tuduce, C. and Gross, T. (2005). A Mobility Model Based on WLAN Traces and its Validation. In *Proceedings of INFOCOM'05*, pages 19–24.

[Vahdat and Becker, 2000] Vahdat, A. and Becker, D. (2000). Epidemic Routing for Partially Connected Ad Hoc Networks. Technical Report CS-2000-06, Department of Computer Science, Duke University.

[Vargas, 2001] Vargas, A. (2001). The OMNeT++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference (ESM'01)*, Prague.

[Vasquez et al., 2003] Vasquez, A., Pastor-Satorras, R., and Vespignani, A. (2003). Large-scale topological and dynamical properties of the internet. *Physical Review E*, 67.

[Wang et al., 2005] Wang, Y., Jain, S., Martonosi, M., and Fall, K. (2005). Erasure-coding based routing for opportunistic networks. In *Proceedings of the 2005 ACM SIGCOMM Workshop on Delay Tolerant Networking (WDTN'05)*, pages 229–236, New York, NY, USA. ACM Press.

[Wang and Wu, 2006] Wang, Y. and Wu, H. (2006). DFT-MSN: The Delay/Fault-Tolerant Mobile Sensor Network for Pervasive Information Gathering. In *Proceedings of INFOCOM'06*, Barcelona, Spain.

[Watts, 1999] Watts, D. J. (1999). *Small Worlds The Dynamics of Networks between Order and Randomness.* Princeton Studies on Complexity. Princeton University Press.

[West and Harrison, 1997] West, M. and Harrison, J. (1997). *Bayesian Forecasting and Dynamic Models.* Springer-Verlag.

[Yoon et al., 2006] Yoon, J., Noble, B. D., Liu, M., and Kim, M. (2006). Building realistic mobility models from coarse-grained traces. In *Proceedings of MobiSys'06*, pages 177–190, New York, NY, USA. ACM Press.

[Zeng et al., 1998] Zeng, X., Bagrodia, R., and Gerla, M. (1998). GloMoSim: a library for parallel simulation of large-scale wireless networks. In *Proceedings of the 12th Workshop on Parallel and distributed simulation(PADS'98)*, pages 154–161, Washington, DC, USA. IEEE Computer Society.

[Zhang, 2006] Zhang, Z. (2006). Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges. *IEEE Communications Surveys and Tutorials*, 8(1).

[Zhao et al., 2004] Zhao, V., Ammar, M., and Zegura, E. (2004). A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *Proceedings of the 5th ACM International Symposiumm on Mobile Ad Hoc Networking and Computing (MobiHoc'04).*