Adaptive Routing for Intermittently Connected Mobile Ad Hoc Networks

Mirco Musolesi, Stephen Hailes, Cecilia Mascolo
Department of Computer Science, University College London
Gower Street, London WC1E 6BT, United Kingdom
{m.musolesi|s.hailes|c.mascolo}@cs.ucl.ac.uk

Abstract

The vast majority of mobile ad hoc networking research makes a very large assumption: that communication can only take place between nodes that are simultaneously accessible within in the same connected cloud (i.e., that communication is synchronous*). In reality, this assumption is likely to be a poor one, particularly for sparsely or irregularly populated environments. Moreover, asynchronous communication such as email, which is by far the preeminent form of networked person-to-person communication, has a natural fit to such partially-connected environments, but has been relatively little explored in the context of mobile ad-hoc networking. This is perhaps unsurprising, given the complexities involved. Indeed, the few approaches that have been described to date are simplistic or heavyweight, relying on brute force methods in order to achieve message delivery.*

In this paper, we present the Context-Aware Routing (CAR) algorithm. CAR is a novel approach to the provision of asynchronous communication in partially-connected mobile ad hoc networks, based on the intelligent placement of messages. We discuss the details of the algorithm, and then present simulation results demonstrating that it is possible for nodes to exploit context information in making local decisions that lead to good delivery ratios and latencies with small overheads.

1 Introduction

Since the earliest days of email, asynchronous communication has been the pre-eminent form of person-to-person communication; in comparison, relatively little Internet traffic is generated for synchronous personal communication, though the balance is expected to shift a little with the increasing deployment of VoIP. The reasons for the success of the asynchronous paradigm are clear: asynchronous communication works even when both parties are not simultaneously available; it is less intrusive than synchronous communication, since recipients can deal with messages at their convenience; and (most importantly for us) it is less sensitive to link failure than synchronous traffic, simply because the traffic is deemed to have higher tolerance to delay.

Ad hoc networks represent the purest form of decentralised systems and, therefore, they impose many challenges to cooperative communication. As a consequence, much ad hoc network research has focused on the investigation of fundamental algorithms for routing [1] on which almost everything else relies. However, in order to make the problem tractable, almost all research on routing algorithms makes the oversimplistic assumption that it is only meaningful to attempt to exchange messages within connected clouds of nodes, in other words, that all communication is synchronous in nature.

This assumption is overly constrained if one considers that there is a strong requirement for communication that is asynchronous in nature, as argued above. In such a case, the delay tolerant character of the traffic allows useful communication to still occur by using nodes moving between disconnected groups of nodes (clouds) to transport messages from one cloud to another. Thus, it is perfectly possible that 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.

In the absence of special information, the problem of predicting which nodes might make good carriers in ad hoc networks is a very challenging one. Likely future mobility patterns must be inferred from past mobility patterns, but this alone is inadequate; parameters such as remaining battery lifetime are also key in determining which potential carriers are most likely to result in successful delivery. In this paper, we consider what types of information are available to nodes in deciding on a carrier. We use this analysis in the design of a Context-aware Adaptive Routing algorithm (CAR), a general framework for the evaluation and prediction of context information, aimed at achieving efficient and timely delivery of messages. Using simulations, we explore the performance of the CAR algorithm with respect to epidemic routing [2] and flooding. Whilst, in the developed world, synchronous communication (in the form of phone 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 [3] or to assist populations in rural areas of India [4]. In the latter case, a number of villages each have their own local networking infrastructure, but there is no interconnection between them. 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 direct connection between sender and recipient, mail can still be delivered.

This paper is organized as follows: in Section 2 we discuss the most relevant aspects of asynchronous routing for mobile ad-hoc networks. Section 3 presents our approach. The details related to the evaluation of context information are discussed in Section 4. The description of the simulations carried out to evaluate CAR is provided in Section 5, together with an analysis of the results. In Section 6 we compare CAR with previous work in this area and Section 7 concludes the paper, outlining possible future research directions.

2 Asynchronous Communication in Mobile Ad Hoc Networks

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. However, as in the example of the bus delivering messages outlined in Section 1, it may be the case that sender and receiver are never in the same connected cloud, so this will not achieve the wanted results.

Only a small number of approaches have been proposed in the field of asynchronous communication for ad hoc networks [2, 5]. 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 is inappropriate, since sender and receiver may never meet.

An alternative, if inefficient, solution is to spread the messages to all hosts using a form of persistent flooding. In this approach, which is more properly known as *epidemic routing* [2], a host floods the message it wishes to send to all hosts within its connected cloud. Each carrier host buffers the message and if, as a result of movement, they come into contact with hosts that do not have a copy, they transfer it to them, making them new carriers, in an analogous was to the spread of disease. Eventually, the message will reach all nodes in the system, provided that movement patterns allow for this. Epidemic routing is a reasonable 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 any node from another, and the movement pattern of each node is individually random, the only choice about message placement is to place messages randomly or to place them everywhere, since there is no more intelligent basis for making a decision.

The aim of our protocol is to allow nodes to make intelligent local decisions about the choice of carriers for messages. These decisions are based on small amounts of information that are exchanged along with standard routing tables, and they are effected by using prediction techniques both to reduce the amount of information needs to be sent and to increase its utility. In the immediately following sections, we analyse the information gathering, prediction and exchange mechanisms before proceeding to an analysis of the performance of the protocol.

3 Context-aware Routing for Mobile Ad Hoc Networks

The Context-Aware Routing algorithm is built on the assumption that the only information a host has about its position is logical connectivity information. 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, and, indeed, we plan to examine its utility in the near future, it is currently unreasonable to assume the existence of GPS for all potential application domains for this technology. 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.

The delivery process depends on whether or not the recipient is present in the same cloud as the sender. If both are currently in the same connected portion of the network, the message is delivered using the underlying synchronous routing protocol to determine a forwarding path. In the remainder of this paper we assume that a proactive routing protocol is used (in our simulations we employed DSDV [6]). Reactive protocols require different approaches to optimisation that would simply confuse the presentation and so are deemed to be outside the scope of this particular work.

If a message cannot be delivered synchronously, 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 one or more of these hosts using the underlying synchronous mechanism.

Delivery probabilities are synthesized locally from context information such as the rate of change of connectivity of a host (i.e., the likelihood of it meeting other hosts) and its current energy level (i.e., the likelihood of it remaining alive to deliver the message). We define *context* as the set of attributes that describe the aspects of the system that can be used to optimize the process of message delivery. 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*). In this paper, 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 important, and represents a trade-off between storage overhead and likely performance. If the buffer overflows, messages will be lost from the system, since we assume the existence of a single replica.

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 1. 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 1. In this case, the host possessing the best delivery probability to host H_8 is 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). 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.

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

Let us suppose H_1 does not store any information about H_8. In this case, H_1 sends the message to the host in the cloud that has the highest mobility (or it keeps the message if it has the highest mobility). In other words, the message is stored by the host that in average meets the highest number of hosts and so that has the highest probability to get in reach with the recipient.

Finally, let us consider another case: let us suppose that H_8, while moving, meets another host H_9 with a delivery probability equal to 0.9. In this case, the message is transferred to the host with the higher delivery probability.

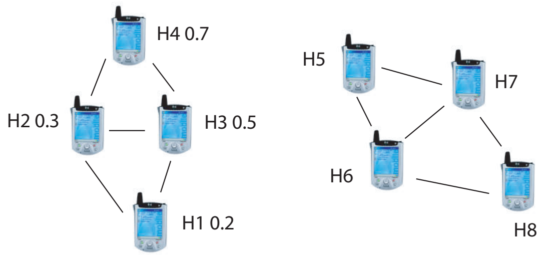

Figure 1. Two connected clouds, with associated delivery probabilities for message transmission between H_1 and H_8

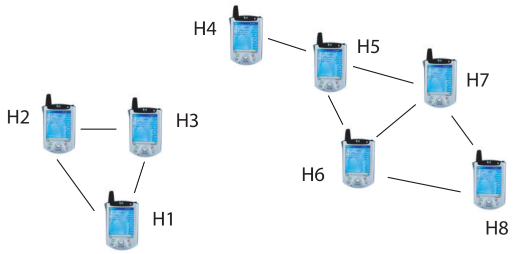

Figure 2. H_4, carrying the message, joins the second cloud.

4 Prediction and Evaluation of Context Information

The general problem from the point of view of the sender of a message is to find the host with the best delivery probability, as calculated using the predicted values of a range of context attributes. Instead of using the available context information as it is, CAR is optimized by using *predicted* future values for the context, so to have more realistic values. The process of prediction and evaluation of the context information can be summarized as follows.

- Each host calculates its delivery probabilities. This process is based on the *prediction* of the future values of the attributes describing the context (see Section 4.2) and on the *composition* of these estimated values using multi-attribute utility theory [7] (see Section 4.1). 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 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 it is carried out until it is possible to guarantee a certain accuracy. Morever, in the case of hosts within reach, the interval between update shipment is based on the possibility or not to make accurate predictions. In other words, hosts send updates only when the evolution of the mobile scenario follows a certain trend. This is done by evaluating the sampled values of context information.

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

4.1 Local evaluation of context information

There are several techniques that can be used to combine and evaluate the multiple dimensions of context in order to decide which nodes are the best candidates for carrying a particular message. The simplest is to allow application developers to define a static hierarchy among the predicted context attributes [8].

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 maximize its value. The optimization 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 [9]. However, in general, the definition of static priorities is inflexible. For more realistic situations, we expect to need to attempt simultaneous maximization of a range of different attributes, as opposed to using a predefined hierarchy of priorities.

4.1.1 Significance-based evaluation of context-aware information

The priority based technique just mentioned seems too simplistic because, in general, our decision problem involves multiple conflicting objectives [7]. For example, if we wish to determine which host has the best delivery probability, considering both the battery energy level and the rate of change of connectivity, it may happen that the host characterized by the highest mobility has scarce residual battery energy and vice versa. In general, maximization across all parameters will not be possible and, instead, we must trade off the achievement of one objective (i.e., the maximization of a single attribute) against others.

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. 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. We use the following definitions:

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

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

Definition 2 *The set of attributes* **Y** *is preferentially independent of the complementary set* **Z** *if and only if for some* **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* **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 1960 [10].

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 characterized 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 operation research community. More specifically, we use the *weights method* in order to find the host that has the highest probability of delivering the message.

Our aim is to maximize 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

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

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

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.

4.1.2 Autonomic adaptation of the utility evaluation function

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 nonlinearly 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 categorized as a typical autonomic mechanism [11]. 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:

a) Criticality of certain ranges of values, $a_{range_i}(x_i)$

b) Predictability of the context information, $a_{predictability_i}(x_i)$

c) Availability of the context information, $a_{availability_i}(x_i)$

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

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

Adaptive weights related to the possible ranges of values assumed by the attributes We can model the adaptive weights $a_{range}(x_i)$ as a function whose domain is $[0, 1]$. For example, with respect to the battery energy level (modeled 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 In general, it is possible to exploit different statistical attributes for the analysis of time series [12]. One could, for example, use the *autocorrelation function* to describe the degree of association between values of the time series at different lags. In short, this gives a measure of the predictability of the context information. Furthermore, there are clear guidelines for adapting the use of the autocorrelation function for non-stationary data with both trends and seasonal variations.

In building the autocorrelation function, we first need to consider the auto-covariance of the system: given a time series characterized by the mean μ_t at the time t, the *auto-covariance* $Cov(X_t, X_{t+k})$ of the time series $\{X_t\}$ at lag k is defined as follows

$$Cov(X_t, X_{t+k}) \equiv E[(X_t - \mu_t)(X_{t+k} - \mu_{t+k})] =$$

$$= \frac{\sum_{t=1}^{n-k} (x_t - \mu_t)(x_{t+k} - \mu_{t+k})}{n}$$

The *lag* represents the time difference (in terms of the number of samples) between the two instants being considered. The variance of the n samples of the time series can be expressed as follows

$$\sigma^2(X_t) \equiv \frac{\sum_{t=1}^{n} (x_t - \mu_t)^2}{n}$$

Therefore, we use the *autocorrelation coefficient* ρ_k, at lag k defined as follows

$$\rho_k \equiv \frac{Cov(X_t, X_{t+k})}{Var(X_t)}$$

that can also be expressed as

$$\rho_k = \frac{\sum_{t=1}^{n-k} (x_t - \mu_t)(x_{t+k} - \mu_{t+k})}{\sum_{t=1}^{n} (x_t - \mu_t)^2}$$

It is worth noting that is possible to prove that

$$0 \leq |\rho_k| \leq 1$$

The absolute value of ρ_k is exactly 1 for a perfect autocorrelation, whereas an autocorrelation coefficient close to zero (either positive or negative) indicates little or no correlation between two samples X_t and X_{t+k} (i.e., the impossibility of making predictions). In the case of a so-called random series, for a large number n of samples, the value of ρ_k is approximately equal to 0. We therefore determine parameter $a_{predictability_i}$ thus:

$$a_{predictability_i} = |\rho_k|$$

An interesting issue is the choice of the value of the lag k. It is possible that autocorrelation signals will drift slowly over time and, consequently, the value of k will also need to change to reflect this. However, we expect the underlying processes that determine the nature of the original signal to change slowly if at all.

Thus, in order to adapt the lag value to retain a strongly correlated signal, we adopt a very simple adaptive technique. At the initial instant t_0, k is set to 1. This is increased, up to a value of k_{MAX}, if the autocorrelation coefficient is below a given lower bound threshold $\rho_{strongCorrLB}$. The process wraps on reaching k_{MAX}, setting the value of k back to unity in order to ensure that the entire space is searched. If, on the other hand, the autocorrelation coefficient exceeds an upper bound threshold $\rho_{strongCorrUB}$, k is decreased until it reaches the value 1.

We can summarize these concepts using the following update equation for the lag k:

$$k(t+1) = \begin{cases} 1 & \text{if } t = t_0 \text{ or } k(t) = k_{MAX} \\ k(t) + 1 & \text{if } \rho(t) \leq \rho_{strongCorrLB}, k(t) < k_{MAX}, t \neq t_0 \\ k(t) - 1 & \text{if } \rho(t) > \rho_{strongCorrUB}, k(t) \geq 1, t \neq t_0 \\ k(t) & \text{otherwise} \end{cases}$$

Adaptive weights related to the availability of the 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 will be 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 $a_{availability_i}$.

Automatic adaptation of the refresh period of routing tables and context information In wired networks, routing table state update is often done on an unvarying regular basis as well as on a by-need basis. However, this approach is wasteful in mobile ad hoc environments. Thus, we consider how to adapt the rate of context information dissemination by noting that we already know that such information is predicted by recipients and that such predictions are likely to be most accurate when the signal on which they are based is most predictable. Thus, a possible function for the determination of refresh time is given by:

$$t(x_1, x_2, ..., x_n) = c \sum_{i=1}^{n} |\rho_{k_i}|$$

where c is a constant of proportionality.

There are several possible extensions of this model. For example, one might wish to take account of the absolute value of a parameter in determining update rates. Thus, for example, as battery energy levels decline, one might wish to update information increasingly less frequently despite the consequent unpredictability at the other end, in order to conserve remaining energy. If information at the recipients becomes totally outdated, then $a_{availability_i}$ will be set to zero for all our attributes and the result is that we will not be likely recipients of messages to transfer, which is in line with the behavior we would expect. Thus, we could replace the simple constant in the above equation with a generic function of values of individual attributes. Likewise, we could obtain a more refined model associating different weights with the autocorrelation coefficient for each attribute in a way similar to that applied previously for composing the utility functions for evaluating which host has the best message delivery probability.

4.2 Prediction of the context information attributes using Kalman filters

Kalman filter prediction techniques [13] 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 more realistic prediction of the evolution of the context of a host and to optimize the bandwidth use. As discussed above, the exchange of the routing tables that store the delivery probabilites is a potentially expensive process, and unnecessarily so where such information is relatively easily predictable.

If it is possible to predict future values of the attributes describing the context, it is possible to update the delivery probabilities stored in the routing tables, even if fresh information is unavailable. Fortunately, it is possible to express this prediction problem in the form of a state space model. We have a time series of observed values that represent context information. From this it is possible to 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 [14]. It is worth noting that 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 view of the fact that we use existing results, we do not present the mathematical aspects of the application of state space models theory and Kalman filter time series analysis in this paper; however, the interested reader can find these in [15].

The use of prediction is complicated by the fact that the information on which it relies for its accuracy travels across networks. In mobile settings, bit error rates are relatively high, and so the loss of messages is more probable than for wired settings. If the routing tables are exchanged only when significant, then its loss has a greater effect. The tradeoff between loss and the additional overhead needed for redundant transmission of context is a complex study in coping with uncertainty and is outside the scope of this paper.

5 Simulation and Results

We evaluated the CAR algorithm by using the OmNet++ discrete event simulator [16]. In order to obtain credible results and to test the peculiar characteristics of our protocol, it was also necessary for us to develop a new group mobility model, that will be presented in Section 5.2.

5.1 Description of the simulation

5.1.1 CAR Simulation

For reasons of space and in order to allow for fair comparison with existing research, we report results based on simulations that use only part of the full generality of the CAR algorithm. Thus, we simulated the CAR model 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 made the assumption that these factors have the same relevance, so assigned them the same weights in the evaluation of the overall utility (i.e., $w_i = 0.5$). Moreover, we also assumed that all the possible values in the range had the same importance (i.e, $a_{range_i}(x_i)$ = 1) and that the the values of attributes are always available during the simulation (i.e., $a_{availability_i}(x_i) = 1$).

The change rate of connectivity attribute is locally calculated by examining the percentage of a node's neighbors that have changed their connectivity status (connected to disconnected, or vice versa) between two instants. The colocation attribute measures the percentage of time that two hosts have been in reach. To calculate it, we periodically run a Kalman filtering process, assuming that the value is 1 if the host is currently in reach or 0 if not. Clearly, the resultant predicted values will be in the range $[0, 1]$ and they will directly express an estimation of the probability of being in reach of the host in the future.

We implemented a simplified version of the DSDV protocol [6] in order to simulate and test the synchronous delivery in connected portions of the network, as described in Section 3.

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, dist, bestHostId, delProb)
```

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. The overhead due to the addition of this field to the standard format of routing tables of DSDV is negligible. It is worth noting that all the autonomic mechanisms, such as the variable refresh period of routing tables, described previously, were implemented.

We also simulated flooding and the epidemic protocols in order to provide comparators for the performance of the CAR solution.

5.1.2 Flooding simulation

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.

5.1.3 Epidemic routing simulation

The implementation of the epidemic protocol follows the description presented in [2]. 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 neighbors (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 characterized by a unique message identifier and a hop count value; the latter determines the maximum number of possible exchanges of a message. Higher hop count values reduce the delivery latency, but, at the same time, increase the quantity of resources (memory, battery, bandwidth) consumed in this process. 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 the classic example of an asynchronous protocol and therefore provides the ideal comparator.

5.1.4 Simulation system parameters

We evaluated the performance of each protocol sending 100 messages with a simulation time equal to 300 seconds. The messages were sent after 40 seconds, in order to allow for the settling of initial routing table exchanges, and the intervals between each message were modeled as a Poisson process, with $\lambda = 5s^{-1}$, and the consequence that all messages are sent in about 20 seconds. The sender and receiver of each message are chosen randomly.

In the CAR simulation, each message has a field that is similar to a *time to live* value that is decreased each time that the message is transferred to another host (the initial value being 15). Moreover, in this case, we also introduced a *split horizon* mechanism to prevent messages from being retransmitted unnecessarily. The buffer for each node was set to 20 messages, unless otherwise specified. Table 1 summarizes the simulation parameters.

Table 1. Simulation parameters

Number of hosts	16/24/32
Simulation area	1 Km x 1 Km
Propagation model	free space
Antenna type	omnidirectional
Transmission range	200 m
Number of clouds	4
Cloud area	200 m x 200 m
Node speed	1-3 m/s
Cloud speed	1-2 m/s
Number of messages sent	100
Max number of hops	15
Message buffer size	10 to 100
Routing table size	20 entries
Max distance	15

The one key aspect of the simulation not yet addressed is that of the mobility model. Clearly, the random way-point mobility model, which is used extensively in such studies, largely for reasons of simplicity, does not accurately reflect human behaviour and annihilate the effect of the prediction since movement is entirely random. Consequently, we devised a new group-based mobility model [17]. This is presented briefly in the following section.

5.2 Mobility model

Mobility models that assume that individuals move independently of one another in random ways are unrealistic in terms of the deployment scenarios for ad hoc networks that are most commonly expounded. For example, on a battlefield, it would be indicative of a very troubled army if each soldier were to move randomly with respect to all others. Thus, we have developed a new model with a form of hierarchical clustering that better reflects the ways in which collections of people are structured at an organizational level and, consequently, the ways in which they move [17]. This model has been instantiated in a simple way for these experiments, and, as used here, is somewhat akin to those in [18, 19] (however, its potential is larger as explained in [17]). Thus, we introduce the concept of a collection of nodes, which has its own motion overlaid on a form of random motion within the cloud.

By parameterizing this model differently, we can represent different archetypes: for example, one would expect to use different parameters for an academic who spends her life traveling between home and the university, interacting with a very closed set of people, as opposed to a salesman who travels much more extensively and interacts less discriminatingly.

A host that belongs to a cloud moves inside it towards a goal (i.e, a point randomly chosen in the cloud space) using the standard random way-point model. When a host reaches a goal, it also implicitly reaches a decision point about whether to remain within the cloud, and, if leaving, to where it should go. Each of these decisions is taken by generating a random number and comparing it to a threshold (which is a parameter of the model). It is worth noting that clouds also move towards randomly chosen goals in the simulation space.

5.3 Simulation Configuration

50% of the hosts are initially placed randomly in a cloud, whereas the others are positioned randomly in the simulation area. Each cloud is defined using a squared area with a side length of 200 m. In other words, we randomly select the point $(minX, minY)$ that, together with the length of the side, defines the cloud area. For these simulations, there is only a single level of cloud.

Every host is characterized by two values, P_{escape}, indicating the probability of escaping from the current cloud, and $P_{escapeCloud}$ describing the probability of choosing a new goal in the space between clouds.

Each cloud moves with a random speed (with a value in the range 1-2 m/s); moreover, each host moves with a randomly generated different speed (with a value in the range 1-3 m/s). It is worth noting that the movement of a host is the result of the composition of these speeds.

In our simulation, the positions of all the hosts and clouds are updated every second. When a cloud reaches its goal, a new goal is chosen in the simulation space. When a host reaches its goal, a threshold probability $P_{escapeThreshold}$ is generated randomly (its range is clearly $[0, 1]$). If its P_{escape} is greater than $P_{escapeThreshold}$ the new goal is chosen outside the current cloud, else inside. If outside, we randomly generate $P_{escapeCloudThreshold}$ and compare it to $P_{escapeCloud}$ to determine whether or not the goal should be chosen in some other cloud or in the open space between clouds. For those hosts that are already outside a cloud, the choice of a new goal is done in an analogous way.

5.4 Analysis of results

In this subsection we will analyze the results of our simulations, comparing the performance of CAR with the flooding and epidemic protocols. We will discuss the variation of some performance indicators as functions dependent on the density of hosts (i.e., the number of the hosts in the simulation area) and the size of the buffers used to store messages in both the epidemic and CAR. In Figure 3, there is a comparison between the delivery ratios of the three protocols in each of three different scenarios (with 16, 24 and 32 hosts). In all cases, the number of messages that may coexist within a node's buffer is unconstrained.

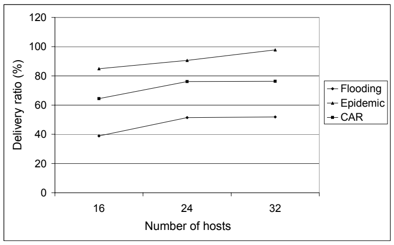

Figure 3. Delivery ratio vs population density

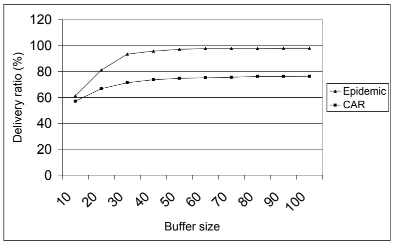

Figure 4. Delivery ratio vs buffer size

CAR achieves a performance between that of flooding and epidemic routing, as expected. Flooding suffers from the inability to deliver messages to recipients that are in other clouds when the messages are sent but is here simply as a comparator to demonstrate the numbers of messages being delivered that cannot be delivered directly, because the recipient is in a cloud different from the cloud of the sender. The epidemic protocol can be considered optimal in terms of delivery ratio, simply because each message is propagated to all accessible hosts, all of which have buffers large enough to hold it. In CAR, we have chosen to operate under the most stringent conditions: there is only ever a single copy of each message, which represents the worst case for this protocol. Clearly, it would be possible to trade off a small amount of intelligent replication (to improve the delivery ratio) against an increase in overhead.

The dependency of the delivery ratios on the buffer size is similar for all the protocols (see in Figure 4 the results for the 32 hosts scenario). Both of these demonstrate a substantial degradation of their performance as buffer size decreases; however, this phenomenon is more evident in the epidemic approach as a result of the degree of replication of messages.

Figure 5 is interesting because there are two competing effects at work for the epidemic protocol. When the buffer size is small, there is a high probability that messages will

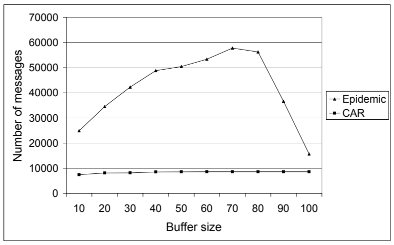

Figure 5. Number of messages vs buffer size

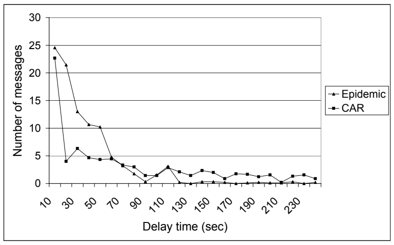

Figure 6. Average delay vs population density

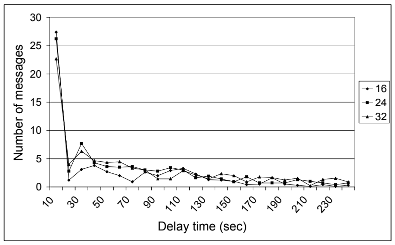

Figure 7. Average delay vs population density

be eliminated due to overflow, as discussed above. Consequently, the number of messages exchanged is also low. At the other end of the scale, as the buffer size increases to a point where it can accommodate all the messages in the system, there is no repeated exchange of messages, so the number is also low. In the middle of the range, however, the buffer size is insufficient to hold all messages and there is a cycle in which messages are eliminated by buffer overflow and then reinstated by other nodes, resulting in very high overhead. In the case of CAR, it is worth noting that the overhead in terms of the number of messages exchanged is more or less constant, regardless of buffer size, demonstrating its *scalability*. CAR will always be the limiting case for performance under this metric because it only creates a single copy of each message. Thus, even at the point where buffer size becomes effectively infinite, the epidemic protocol will necessarily exchange more messages than ours, simply as a result of the replication.

Figure 6 shows the distribution of the number of messages with respect to their delivery latency in the 32 hosts scenario. It is possible to observe that a proportion of the messages are delivered more or less immediately, since the recipients are in the same cloud as the sender. This phenomenon is more evident for the epidemic protocol since it allows for a rapid dissemination of the information in connected clouds. Another interesting comparison is showed in Figure 7: the distributions of the delivery latency in the case of different node densities are very similar. It is interesting to note that the performance of CAR in terms of delivery delay is rather insensitive to the population density, since it does not rely only on the probability of contact between hosts as the epidemic protocol.

6 Related Work and Discussion

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

Epidemic algorithms were first devised in the context of distributed database systems in an attempt to guarantee data consistency after disconnections [20]. 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 [2], described earlier, that forms the basis for much of the work in this field, applied this early approach to the field of asynchronous message delivery, but in a rather naive fashion.

Chen and Murphy refined the epidemic model, presenting the so-called Disconnected Transitive Communication paradigm [5]. Their approach is similar to ours, since it essentially argues for the use of utility functions, but it provides a general framework rather than a detailed instantiation, and so aspects related to the composition of calculated delivery probabilities are almost entirely missing.

In [21], Lindgren et al. propose a probabilistic routing approach to enable asynchronous communication among intermittently connected clouds of hosts. 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: in other words, 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 neighbor of H_B, with the probability that H_B becomes a neighbor of H_C. The calculation of the delivery probabilities is based, somewhat simplistically, on the period of time of co-location of two hosts, weighted by an aging factor that is used to decrease the overall probability with the increasing age of the information on which it was based.

In [22], Small and Haas describe a very interesting application of epidemic routing protocols to a problem of cost-effective data collection, using whales as message carriers.

Sasson et al. [23] studied 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. Another interesting epidemic model for mobile ad hoc networks is presented in [24]; in this paper the authors investigate the similarities between flooding-based approaches for the information dissemination in mobile ad-hoc networks and the epidemic spreading of diseases.

In [25], 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). This approach relies on routing mechanism presented in detail in [26], based on optimal or sub-optimal algorithms, according to the different knowledge about the topology of the networks and the sampled delivery delays, to compute the best end-to-end communication shortest path.

Zhao et al. in [27] 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.

With respect to the existing work in this research area, such as [26] and [27], we have introduced a general framework for the evolution and the prediction of the mobile context to provide efficient and effective communication mechanisms in mobile ad hoc networks. Moreover, we believe that it is possible to integrate our techniques with these approaches, since they address orthogonal aspects of the problem.

It is worth noting that 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.

7 Conclusions and Future Work

In this paper, we presented a novel approach to the challenge of asynchronous ad hoc routing. This is a problem that is deceptively easy to state but that requires the combination of results from many fields to address efficiently. Thus, we have designed a general and flexible framework for the evaluation of context information using probabilistic, statistical, autonomic and predictive techniques in order to optimize the consumption of the scarce resources of mobile devices whilst retaining good delivery performance. Previous solutions to the problem of asynchronous routing are either unoptimized (as in the case of the basic epidemic protocol) or are insufficiently specific to help in the construction of systems capable of dealing with the multidimensional nature of context (as in the Chen and Murphy's approach).

In order to assess our algorithm, a new mobility model [17], better reflecting the realities of human organization, was developed and used in simulations that give a feel for the relative performance of CAR relative to flooding and epidemic routing. The results demonstrated that, even without message replication, CAR performs respectably in terms of message delivery, with very much lower overheads than the alternatives.

In future, we will further explore the tradeoff between increasing delivery ratios via replication versus maintenance of low overhead. Moreover, we will further explore an acknowledgment mechanism in order to notify the sender about the correct delivery of messages (and to remove them from intermediate nodes), exploiting the same techniques as those used to deliver messages. Lastly, we intend further to investigate the application of mathematical models of social organization, most notably small world models, in assessing performance and in optimizing the reliability of the routing algorithm.

References

[1] C. E. Perkins, *Ad Hoc Networking*. Addison-Wesley, 2001.

[2] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Department of Computer Science, Duke University, Tech. Rep. CS-2000-06, 2000.

[3] A. Doria, M. Uden, and D. P. Pandey, "Providing connectivity to the Saami nomadic community," in *Proceedings of the Second International Conference on Open Collaborative Design for Sustainable Innovation*, December 2002.

[4] I. T. Union, "Connecting remote communities," *Documents of the World Summit on Information Society*, 2003, http://www.itu.int/osg/spu/wsis-themes.

[5] X. Chen and A. L. Murphy, "Enabling disconnected transitive communication in mobile ad hoc networks," in *Proceedings of the Workshop on Principles of Mobile Computing (POMC'01)*, August 2001, pp. 21–23.

[6] C. E. Perkins and P. Bhagwat, "Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers," in *Proceedings of thr Conference on Communications Architecture, Protocols and Applications (SIGCOMM 94)*, August 1994.

[7] R. L. Keeney and H. Raiffa, *Decisions with Multiple Objectives: Preference and Value Tradeoffs*. John Wiley and Sons, 1976.

[8] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Prentice Hall, 2002.

[9] H. A. Taha, *Operations Research: An Introduction*. Prentice Hall, 1996.

[10] G. Debreu, “Topological methods in cardinal utility theory,” in *Mathematical Methods in the Social Sciences*, K. J. Arrow, S. Karlin, and P. Suppes, Eds. Stanford University Press, 1959.

[11] D. F. Bantz, C. Bisdikian, D. Challener, J. P. Karidis, S. Mastrianni, A. Mohindra, D. G. Shea, and M. Vanover, “Autonomic personal computing,” *IBM Systems Journal*, vol. 42, no. 1, 2003.

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

[13] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Transactions of the ASME Journal of Basic Engineering*, March 1960.

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

[15] M. Musolesi, S. Hailes, and C. Mascolo, “Prediction of Context Information Using Kalman Filter Theory,” http://www.cs.ucl.ac.uk/staff/m.musolesi/papers/kalman.pdf.

[16] A. Varga, “The OMNeT++ discrete event simulation system,” in *Proceedings of the European Simulation Multiconference (ESM’2001)*, Prague, 2001.

[17] M. Musolesi, S. Hailes, and C. Mascolo, “An ad hoc mobility model founded on social network theory,” in *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*. ACM Press, October 2004.

[18] X. Hong, M. Gerla, G. Pei, and C.-C. Chiang, “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*, 1999, pp. 53–60.

[19] J. Carlos Cano, P. Manzoni, and M. Sanchez, “Evaluating the impact of group mobility on the performance of mobile ad hoc networks,” in *Proceedings of the 2004 IEEE International Conference on Communications (ICC 2004)*, 2004.

[20] A. Demers, D. Greene, C. Hauser, W. Irish, J. Larson, S. Shenker, H. Sturgis, D. Swinehart, and D. Terry, “Epidemic algorithms for replicated database maintenance,” *ACM SIGOPS Operating Systems Review*, vol. 22, no. 1, January 1988.

[21] A. Lindgren, A. Doria, and O. Schelen, “Probabilistic routing in intermittently connected networks,” *Mobile Computing and Communications Review*, vol. 7, no. 3, July 2003.

[22] T. Small and Z. J. Haas, “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*, June 2003.

[23] Y. Sasson, D. Cavin, and A. Schiper, “Probabilistic broadcast for flooding in wireless mobile ad hoc networks,” in *IEEE Wireless Communications and Networking Conference (WCNC 2003)*, March 2003.

[24] A. Khelil, C. Becker, and J. a. Tian, “An epidemic model for information diffusion in MANETs,” in *Proceeding of the the Fifth ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile System (MSWiM’02)*, September 2002.

[25] K. Fall, “A delay-tolerant network architecture for challenged internets,” in *Proceedings of SIGCOMM’03*, August 2004.

[26] J. Sushant, K. Fall, and R. Patra, “Routing in a delay tolerant network,” in *Proceedings of SIGCOMM’04*, August 2004.

[27] Z. Venrui, M. Ammar, and E. Zegura, “A message ferrying approach for data delivery in sparse mobile ad hoc networks,” in *Proceedings of the 5th ACM International Symposiunm on Mobile Ad Hoc Networking and Computing (MobiHoc’04)*, May 2004.